Governance and optimization models for autonomous and smart urban systems: a blockchain-enabled and learning-based approach

Politecnico di Bari

Department of Electrical and Information Engineering
Electrical and Information Engineering
Ph.D. Program
SSD: ING-INF/04 Systems and Control Engineering

Final Dissertation

Governance and Optimization models for Autonomous and Smart Urban Systems: A Blockchain-Enabled and Learning-Based Approach

by
Giuseppe Olivieri

Supervisor:

Prof.ssa Maria Pia Fanti

Coordinator of Ph.D. Program:
Prof. Mario Carpentieri

Cycle XXXVIII - November 1st, 2022 - October 31st, 2025

Politecnico di Bari

Department of Electrical and Information Engineering
Electrical and Information Engineering
Ph.D. Program
SSD: ING-INF/04 Systems and Control Engineering

Final Dissertation

Governance and Optimization models for Autonomous and Smart Urban Systems: A Blockchain-Enabled and Learning-Based Approach

by
Giuseppe Olivieri

Referees:

Prof. Mauro Franceschelli

Prof.ssa Maria Gabriella Xibilia

Supervisor:

Prof.ssa Maria Pia Fanti

Coordinator of Ph.D Program:

Prof. Mario Carpentieri

Cycle XXXVIII - November 1st, 2022 - October 31st, 2025

SCIENTIA OMNIA ILLUSTRAT

Acknowledgements

Il mio primo e più sentito ringraziamento va alla Professoressa Maria Pia Fanti, Direttrice del Laboratory of Control and Automation del Politecnico di Bari, che ha supervisionato questo lavoro di tesi e tutta la mia attività di ricerca. La sua guida è stata costante durante tutto il percorso del Dottorato, anni di forte cambiamento e non privi di difficoltà. Non ha mai mancato di spronarmi a proseguire, aiutandomi a mettere in fila gli obiettivi e a raggiungerli, passo dopo passo. Grazie al suo coordinamento ho potuto raggiungere traguardi che non credevo possibili; per questo e molto altro le sarò per sempre infinitamente grato. Un ringraziamento particolare va al Professor Agostino Marcello Mangini. Spero di avergli restituito qualche momento di gioia, così come lui ne ha donati a me. Il suo supporto e la sua competenza sono stati determinanti in questo percorso. Sono onorato di aver potuto affiancare il mio nome al loro.

Un pensiero riconoscente va a Gaetano e ai miei compagni di viaggio del PoliBa, tra cui Michele, Francesco e Ruotian, che mi hanno accompagnato in questo percorso sopportandomi e supportandomi, con pazienza e dedizione, confortandomi nei momenti di necessità e gioendo dei miei successi.

Abstract

Contemporary cyber-physical systems operating within urban infrastructures confront three fundamental challenges: establishing distributed trust mechanisms absent centralized authorities, coordinating heterogeneous autonomous agents under uncertainty, and optimizing resource allocation subject to competing stakeholder objectives. The reconciliation of mathematical optimality with operational constraints while maintaining cryptographically verifiable guarantees across heterogeneous ecosystems constitutes a critical research gap. This dissertation develops and validates a unified methodological framework integrating distributed ledger technologies with machine learning paradigms and mathematical optimization to address these challenges systematically across autonomous vehicle coordination, intelligent building energy management, and organizational resource allocation domains.

The proposed framework instantiates a three-layer architectural pattern that decomposes cyber-physical control into cryptographic trust establishment, intelligent decision synthesis, and domain-specific integration strata. The trust layer transforms local observations into globally verifiable intelligence through deterministic identification schemes. The intelligence layer implements domain-adapted optimization strategies comprising Deep Reinforcement Learning, Model Predictive Control augmented with Long Short-Term Memory networks, and Integer Linear Programming formulations. The integration layer provides semantic translation between mathematical abstractions and deployment-specific constraints through standardized interfaces maintaining backward compatibility with legacy systems.

The cryptographic foundation employs blockchain-based consensus mechanisms with dynamic threshold adaptation that modulates validation requirements according to event severity, temporal urgency, and proposer reputation. The mathematical formulation of a dynamic threshold enables rapid dissemination of critical infrastructure updates through economic incentive alignment. Smart contract architectures execute on Ethereum Virtual Machine compatible platforms, demonstrating gas-efficient operations through optimized storage patterns and event-based logging mechanisms that reduce on-chain footprint by maintaining only cryptographic commitments rather than complete datasets.

Experimental validation across three deployments substantiates the framework's efficacy and cross-domain transferability. The autonomous vehicle coordination system reduces training convergence time while demonstrating superior mean

rewards relative to monolithic baselines through hierarchical action space factorization across four urban zones. The building automation implementation achieves reduction in energy consumption while maintaining thermal comfort within regulatory bounds through blockchain-notarized K-means clustering for occupancy pattern classification integrated with lexicographic optimization. The organizational allocation system generates optimal solutions for competing minimax fairness and team cohesion objectives, enabling practitioners to navigate trade-offs through continuous parametrization of weighting coefficient.

This research establishes that distributed intelligence in cyber-physical systems emerges through systematic integration of cryptographic immutability providing mathematical certainty at the data layer, economic incentive structures aligning individual utility with collective optimum discovery, and algorithmic sophistication matching computational strategies to problem characteristics. The limitations identified include computational overhead of blockchain operations in resource-constrained environments, sample complexity requirements for Deep Reinforcement Learning convergence, and challenges in maintaining model accuracy when system dynamics evolve beyond training distributions. These constraints delineate boundaries for framework applicability while suggesting extensions through layer-two scaling solutions, transfer learning mechanisms, and robust optimization formulations that maintain performance despite modeling uncertainties. The validated methodological framework provides principled foundations for a trustworthy system design, demonstrating that reliable distributed coordination emerges from architectural coherence rather than technological innovation in isolation.

Contents

List of Figures

List of Tables

Abbreviations

AIM	Autonomous Intersection Management
API	Application Programming Interface
AV	Autonomous Vehicle
BCVTB	Building Controls Virtual Test Bed
CAV	Connected and Autonomous Vehicle
CBOR	Concise Binary Object Representation
CPS	Cyber-Physical System
CZ	Control Zone
DAG	Directed Acyclic Graphs
DAO	Decentralized Autonomous Organization
DEMS	District Energy Management System
DHI	Diffuse Horizontal Irradiance
DLT	Distributed Ledger Technology
DNI	Direct Normal Irradiance
DRL	Deep Reinforcement Learning
EIP	Ethereum Improvement Proposal
EOA	Externally owned account
EVM	Ethereum Virtual Machine
EPW	EnergyPlus Weather
ERL	EnergyPlus Runtime Language
GHI	Global Horizontal Irradiance
GPS	Global Positioning System

HR	Human Resources
HVAC	Heating, Ventilation, and Air-Conditioning
ILP	Integer Linear Programming
IoT	Internet of Things
ITS	Intelligent Transportation Systems
JCS	JSON Canonicalization Scheme
JSON	JavaScript Object Notation
LiDAR	Light Detection and Ranging
LSTM	Long Short-Term Memory
MDP	Markov Decision Process
ML	Machine Learning
MIP	Mixed Integer Programming
MPC	Model Predictive Control
MZ	Merging Zone
NUC	Next Unit of Computing
PID	Proportional Integral Derivative
PPO	Proximal Policy Optimization
PoS	Proof of Stake
RL	Reinforcement Learning
SC	Smart Contract
SHA-256	Secure Hash Algorithm 256
SNARKs	Succinct Non-interactive Arguments of Knowledge
STARKs	Scalable Transparent Arguments of Knowledge
SUMO	Simulation of Urban MObility
TraCI	Traffic Control Interface
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-Everything

List of Publications

[1] **G. Olivieri**, G. Volpe, A. M. Mangini, and M. Pia Fanti, "Enhancing intersection identification for autonomous vehicles: A hash-based approach," in *2024 10th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2024, pp. 700–705.

[2] F. Paparella, **G. Olivieri**, G. Volpe, A. M. Mangini, and M. P. Fanti, "A deep reinforcement learning approach for route planning of autonomous vehicles*," in *2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2024, pp. 2047–2052.

[3] **G. Olivieri**, G. Volpe, A. M. Mangini, and M. P. Fanti, "A user based hvac system management through blockchain technology and model predictive control," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 3621–3634, 2025.

[4] **G. Olivieri**, A. M. Mangini, and M. P. Fanti, "A blockchain framework for incentivized data sharing in autonomous vehicle networks," in *2025 11th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2025, accepted, in press.

[5] **G. Olivieri**, A. M. Mangini, and M. P. Fanti, "Optimizing personnel allocation: An integer linear programming problem for enhanced workplace efficiency," *Computers & Industrial Engineering*, vol. 212, p. 111724, 2026.

Part I

Introduction

1 Introduction

1.1 Objectives

The primary objective of this dissertation is to develop and validate a comprehensive methodological framework for designing trustworthy cyber-physical systems through the synergistic integration of distributed ledger technologies, machine learning paradigms, and optimization strategies. This framework addresses the inherent complexity of modern urban environments by establishing formal guarantees for security, efficiency, and deployability while maintaining computational tractability across heterogeneous system architectures.

The second objective focuses on establishing the mathematical foundations and computational primitives necessary for verifiable data exchange among distributed agents operating under Byzantine fault conditions. This encompasses the formulation of deterministic identification mechanisms through cryptographic hash functions with collision resistance properties, the design of adaptive consensus protocols featuring dynamic threshold adjustments based on contextual urgency metrics, and the implementation of game-theoretic incentive structures that achieve Nash equilibrium between individual utility maximization and collective truth discovery. The research investigates how blockchain architectures can provide immutable substrates for information dissemination while satisfying the stringent latency constraints characteristic of real-time cyber-physical applications, with particular emphasis on verification operations through Merkle tree structures.

A critical objective involves developing domain-specific optimization strategies that exploit the mathematical structure inherent in distinct urban subsystems. For autonomous vehicle coordination, this entails formulating modular deep reinforcement learning architectures based on proximal policy optimization that manage the NP-hard combinatorial complexity of urban routing while satisfying multi-objective constraints spanning temporal efficiency, safety margins quantified through time-to-collision metrics, and infrastructure utilization factors. In building automation contexts, the objective extends to synthesizing model predictive control schemes that employ Long Short-Term Memory networks for system identification while maintaining convex optimization formulations that guarantee global optimality within receding horizon frameworks. For organizational resource allocation, the research demonstrates that integer linear programming retains computational superiority when problem structure permits branch-and-bound algorithms to achieve polynomial-time solutions despite theoretical intractability.

Another fundamental objective centers on validating the transferability of proposed methodologies across disparate application domains through rigorous empirical analysis. This requires demonstrating that the three-layer architectural decomposition into cryptographic integrity, intelligent decision-making, and system integration components represents an invariant design pattern emerging from fundamental information-theoretic constraints rather than domain-specific contingencies. The research establishes that successful methodological transfer necessitates preserving mathematical invariants while adapting implementation details to domain-specific constraint structures, identifying the homomorphic mappings that enable cross-domain applicability while preserving correctness guarantees.

The subsequent objective addresses the engineering challenge of transitioning from theoretical contributions to operational deployment through the development of production-grade implementations that encapsulate computational complexity within accessible interfaces. This encompasses architecting RESTful APIs for human resource optimization systems, implementing WebSocket-based real-time dashboards for traffic management with sub-second latency requirements, and establishing Hardware-in-the-Loop co-simulation frameworks that interface with existing SCADA infrastructure through OPC UA protocols. The objective extends to comprehensive lifecycle management, including continuous integration pipelines for automated testing, telemetry collection for performance monitoring, and feedback loops that enable online learning from deployment experiences.

The final objective involves conducting statistically rigorous experimental validation across multiple case studies to quantify both individual component performance and emergent system-level properties. This requires designing experiments that control for confounding variables through randomized block designs, establishing statistical significance through non-parametric tests that avoid normality assumptions, and conducting sensitivity analysis to identify parameter regimes where different algorithmic choices dominate. The experimental campaigns span temporal scales from millisecond-level control loops to seasonal variations in building occupancy patterns, spatial scales from individual intersection management to metropolitan-area traffic coordination, and operational contexts from greenfield deployments to brownfield integration scenarios, ultimately establishing empirical bounds on performance metrics that validate theoretical predictions while revealing implementation trade-offs that inform practical deployment strategies.

1.2 Thesis structure

The structure of this dissertation reflects a deliberate progression from theoretical foundations through domain-specific applications to culminate in a unified analysis of transferability and generalization, organizing the research contributions into six interconnected parts that collectively establish a comprehensive framework for trustworthy cyber-physical systems.

The **first part** provides the introductory context and motivation for the research, establishing the fundamental challenges that arise when attempting to coordinate distributed agents in urban environments without centralized authority. Chapter 1 articulates the research objectives, delineates the scope of investigation, and presents the key contributions that distinguish this work from existing approaches in the intersection of blockchain technology, machine learning, and cyber-physical systems.

The **second part** establishes the theoretical foundations and methodological infrastructure upon which the subsequent contributions rest. Chapter 2 develops the cryptographic primitives and distributed ledger mechanisms that enable verifiable data exchange, progressing from fundamental concepts such as hash functions and Merkle trees to sophisticated blockchain architectures supporting smart contract execution. Chapter 3 presents the spectrum of optimization and learning paradigms employed throughout the dissertation, encompassing graph-theoretic models for discrete optimization, model predictive control augmented with LSTM-based system identification, and deep reinforcement learning for sequential decision-making under uncertainty. Chapter 4 synthesizes these elements into a unified methodological framework, introducing the three-layer architectural pattern that recurs throughout the applications while documenting the comprehensive simulation and development toolchain that enables reproducible experimentation across diverse domains.

The **third part** focuses on establishing identity and trust mechanisms within networked mobility systems, addressing the fundamental challenge of coordinating autonomous vehicles that lack pre-established trust relationships. Chapter 5 introduces a novel hash-based approach for deterministic intersection identification, demonstrating how cryptographic techniques can resolve spatial ambiguities that plague conventional location-based systems. Chapter 6 extends this foundation by developing a complete blockchain framework for incentivized data sharing, incorporating dynamic consensus thresholds and reputation mechanisms that encourage truthful reporting while maintaining system resilience against adversarial manipulation. Chapter 7 synthesizes these contributions to reveal the emergent design patterns, demonstrating how the convergence of deterministic identification

with economic incentivization creates self-reinforcing trust dynamics that transcend the capabilities of either mechanism in isolation.

The **fourth part** addresses control and optimization challenges in smart urban systems, showcasing how the established trust infrastructure enables sophisticated coordination strategies. Chapter 8 presents a modular deep reinforcement learning architecture for multi-objective routing in urban networks, demonstrating how decomposition strategies can manage the computational complexity inherent in city-scale optimization while preserving solution quality. Chapter 9 integrates blockchain-based user classification with model predictive control for building climate management, establishing how distributed ledger technologies can facilitate dynamic pricing mechanisms while LSTM networks capture building thermodynamics for anticipatory control that balances comfort and efficiency objectives.

The **fifth part** demonstrates the transferability of the developed methodologies to organizational contexts, validating that the proposed frameworks extend beyond purely technological systems. Chapter 10 provides a systematic analysis of how optimization paradigms, architectural patterns, and design principles manifest across the investigated domains, revealing the mathematical isomorphisms and structural parallels that enable successful methodological transfer. Chapter 11 presents a concrete application to office space optimization, demonstrating how integer linear programming techniques can address personnel allocation challenges while maintaining the same fundamental constraint structures and multi-objective trade-offs observed in vehicular and building automation contexts.

The **sixth part** concludes the dissertation by synthesizing the contributions and establishing their broader implications for cyber-physical system design.

1.3 Contributions

This thesis advances the state of knowledge in trustworthy cyber-physical systems through nine interconnected contributions that collectively establish a unified methodological framework transcending traditional domain boundaries.

The research introduces a **comprehensive architectural pattern** that systematically integrates distributed ledger technologies, machine learning paradigms, and optimization strategies across heterogeneous urban domains. This three-layer architecture decomposes systems into data integrity, intelligent decision-making, and system integration components, representing fundamental functional requirements that emerge consistently across diverse cyber-physical contexts rather than mere organizational convenience. The framework's generality derives from its mathematical foundations, wherein cryptographic primitives establish identity, optimization algorithms navigate trade-off spaces, and temporal dynamics govern adaptation mechanisms.

At the cryptographic layer, the thesis presents a **novel hash-based identification mechanism** for autonomous vehicle coordination that resolves spatial ambiguities inherent in GPS-based systems. Through the application of SHA-256 algorithms to canonicalized intersection descriptors and the construction of hierarchical Merkle trees, the method enables unambiguous spatial referencing even under degraded GPS conditions. This deterministic approach establishes a mathematical foundation for consensus protocols in vehicular networks, where agreement on spatial entities precedes coordination decisions.

Building upon this identification layer, the research develops a **sophisticated blockchain framework** incorporating dynamic consensus thresholds and reputation mechanisms. The smart contract architecture comprises FilterContract validation, VotingContract consensus, and TokenContract incentivization, wherein the dynamic threshold mechanism adapts validation requirements based on event severity, temporal urgency, and proposer reputation. This adaptation enables rapid dissemination of critical information while maintaining resistance to adversarial manipulation through game-theoretic equilibria.

The optimization contributions begin with a **modular Deep Reinforcement Learning architecture** that addresses the computational intractability of exhaustive training in urban routing contexts. By partitioning urban graphs into subsets and enabling parallel training of specialized agents, the decomposition strategy reduces training time from seven hours to three hours while demonstrating superior mean rewards compared to monolithic approaches. This modularity principle extends beyond computational efficiency to enable incremental system updates as urban infrastructure evolves.

The thesis pioneers the **convergence of distributed ledger technologies with Model Predictive Control** for building automation, wherein K-means clustering on blockchain-notarized consumption data creates dynamic pricing classes while LSTM networks capture thermodynamic relationships for anticipatory control. The lexicographic optimization framework prioritizes occupant comfort before energy minimization, achieving twenty percent reduction in consumption while maintaining thermal satisfaction. This integration demonstrates that trust mechanisms and control strategies operate synergistically rather than independently.

For discrete allocation problems, the research establishes a **mathematical formalization of personnel allocation** as an Integer Linear Programming problem with minimax fairness objectives and team cohesion constraints. The multi-objective formulation elegantly balances occupancy minimization with organizational unit preservation, providing practitioners with interpretable trade-offs through a weighting parameter. This contribution demonstrates that exact methods remain superior for discrete allocation problems with complete information, contrasting with the learning-based approaches necessary for continuous, partially observable domains.

Through systematic implementation across autonomous vehicles, building automation, and organizational management contexts, the thesis provides **empirical validation of cross-domain transferability**. The identification of mathematical isomorphisms, e.g., collision avoidance in vehicles mapping to separation constraints in personnel allocation, reveals fundamental structural patterns that guide methodological extension to novel domains. These patterns suggest that optimization principles transfer successfully when domain characteristics are carefully mapped to algorithmic requirements.

The practical impact manifests through **operational tools with user-accessible interfaces**, including web-based platforms, REST APIs, and visualization dashboards that abstract computational complexity while preserving system capabilities. Validation through stakeholder engagement across traffic engineers, facility managers, and human resource professionals demonstrates that sophisticated optimization becomes accessible without requiring technical expertise from end-users.

Finally, the research advances **theoretical understanding of distributed trust mechanisms** by demonstrating how deterministic identification at the cryptographic layer enables sophisticated consensus mechanisms at the application layer. Trust amplification emerges through systematic layering of verification mechanisms, wherein system reliability exceeds the sum of component reliabilities when mathematical certainty, economic incentives, and game-theoretic principles operate in concert.

The thesis acknowledges computational overhead of blockchain operations in resource-constrained environments, sample complexity requirements for deep reinforcement learning convergence, and challenges in maintaining model accuracy when system dynamics evolve beyond training distributions. These limitations are addressed through subnet deployment for reduced transaction costs, modular training strategies enabling incremental updates, and robust optimization formulations maintaining performance despite modeling uncertainties. While certain domains may require specialized adaptations beyond those explored herein, particularly real-time safety-critical systems demanding deterministic guarantees or environments with extreme adversarial dynamics necessitating stronger cryptographic primitives, the established methodological framework provides principled foundations for such extensions through clear integration points within the three-layer architecture.

Part II

Theoretical Foundations and Methods

2 Trustworthy Data and Decentralized Ledgers

The convergence of cyber-physical systems with decentralized architectures introduces fundamental challenges in establishing trust without central authorities, verifying data integrity across heterogeneous platforms, and coordinating actions among autonomous agents that may have conflicting incentives. Traditional approaches to data management, which rely on trusted intermediaries or centralized databases, become untenable when systems must operate across organizational boundaries, function in adversarial environments, or provide transparent audit trails for regulatory compliance. This chapter establishes the theoretical and practical foundations for constructing trustworthy data systems in decentralized settings, presenting a systematic progression from cryptographic primitives to distributed execution environments. We begin in Section 2.1 by formalizing the mathematical tools that enable tamper-evident data structures: cryptographic hash functions that provide collision-resistant fingerprinting, canonicalization schemes that ensure deterministic serialization across platforms, and Merkle trees that enable efficient membership proofs while maintaining compact commitments. These primitives serve as building blocks for the distributed systems examined in Section 2.2, where we analyze blockchain platforms as instantiations of replicated state machines that achieve consensus without central coordination. Through detailed examination of execution semantics, economic incentives, and scaling mechanisms, we establish how smart contracts can serve as trust anchors for cyber-physical systems, enabling verifiable computation and automated settlement while maintaining deterministic reproducibility essential for scientific evaluation.

2.1 Hashing, JSON canonicalization and Merkle trees

In cyber-physical systems where multiple autonomous agents must coordinate actions based on shared observations, the fundamental challenge extends beyond mere data exchange to establishing cryptographic guarantees about data integrity, authenticity, and temporal ordering. Consider an autonomous vehicle network where intersection states must be unambiguously identified across heterogeneous platforms, or a distributed HVAC control system where sensor readings from different manufacturers must be aggregated into verifiable commitments that can later anchor economic settlements. These scenarios demand primitives that transform arbitrary structured data into deterministic, fixed-length identifiers that remain stable across different implementations, programming languages, and execution environments. Furthermore, when such systems operate in adversarial or economically-incentivized contexts, e.g., when blockchain-based rewards depend on the correctness of reported measurements, the ability to prove membership in a committed dataset without revealing the entire collection becomes paramount for both privacy and efficiency. This section formalizes three foundational primitives that address these requirements: cryptographic hashing provides collision-resistant fingerprinting that enables content-addressed storage and tamper detection; JSON canonicalization eliminates representation ambiguity by enforcing a unique byte-level serialization for semantically equivalent data structures; and Merkle trees enable logarithmic-cost membership proofs while committing entire datasets to a single root digest that can be efficiently verified on resource-constrained platforms, including smart contracts with strict gas limits.

2.1.1 Cryptographic Hash Functions and Encoding Conventions

A cryptographic hash function serves as a digital fingerprint mechanism, transforming arbitrary input data into a fixed-length output that uniquely represents the original content while revealing nothing about its structure or composition. This primitive underpins virtually all modern cryptographic protocols, from digital signatures to blockchain consensus, by providing a deterministic yet unpredictable mapping that preserves integrity without preserving reversibility. In the context of cyber-physical systems and decentralized architectures, hash functions enable compact commitments to large datasets, efficient verification of data integrity, and stable content-based addressing that remains invariant across platforms and

implementations.

To formalize this transformation and establish precise security properties that our protocols will rely upon, we require a mathematical characterization of the hash function's behavior. At its core, a cryptographic hash function operates on binary representations of data, accepting inputs of arbitrary size while consistently producing outputs of predetermined length. This size-agnostic property proves essential for cyber-physical systems where sensor readings might range from single bytes to megabyte-scale images, yet all must be reduced to uniformly-sized identifiers for efficient indexing and verification. The mathematical abstraction captures this transformation as a function

$$H : \{0,1\}^* \to \{0,1\}^d \tag{2.1}$$

that maps bit strings of arbitrary length to digests of exactly d bits, where the notation $\{0,1\}^*$ denotes the set of all finite binary sequences and $\{0,1\}^d$ represents the set of binary strings of precisely d bits. Throughout this work, we adopt $d = 256$ to align with contemporary security margins and standard implementations [4, 5]. The choice of 256-bit output provides sufficient collision resistance against foreseeable computational advances while remaining efficient on modern architectures, particularly when leveraging hardware acceleration available in contemporary processors and cryptographic coprocessors.

For a hash function to be considered cryptographically secure in the context of our distributed protocols, it must exhibit three fundamental properties that collectively prevent adversarial manipulation [6]. The first property, **preimage resistance**, ensures that given only a hash output, no computationally bounded adversary can efficiently discover any input that produces that specific digest. This property protects the confidentiality of hashed data and enables secure password storage, among other applications. Formally, for any digest y selected uniformly from $\{0,1\}^d$, the probability that a polynomial-time adversary can find an x such that $H(x) = y$ must be negligible in the security parameter λ:

$$\Pr[\text{adversary finds } x : H(x) = y] \leq \text{negl}(\lambda). \tag{2.2}$$

The second property, **second-preimage resistance**, guarantees that even when an adversary knows one valid input-output pair, finding a different input with the same hash remains computationally infeasible. This property is crucial for preventing substitution attacks where an attacker might attempt to replace legitimate data with malicious content while preserving the cryptographic commitment. Given any input x, the probability of finding a distinct $x' \neq x$ where $H(x') = H(x)$ must remain negligible.

The third and strongest property, **collision resistance**, asserts that finding any two distinct inputs that hash to the same output should be computationally intractable, even when the adversary has complete freedom in choosing both inputs. This property underpins the security of digital signatures and Merkle tree constructions discussed in Section 2.1.3. While generic birthday attacks theoretically require approximately $2^{d/2}$ hash evaluations to find collisions [6], our choice of 256-bit outputs ensures that such attacks remain beyond practical computational reach for the foreseeable future [7].

Beyond these formal security properties, practical hash functions exhibit strong diffusion characteristics through the avalanche effect, whereby the slightest perturbation in the input cascades through the computation to produce radically different outputs that appear statistically independent. This sensitivity extends to modifications that remain imperceptible to human observation, such as trailing whitespace characters, non-printing Unicode symbols, or variations in encoding normalization, where semantically equivalent representations yield entirely divergent hash values. Such behavior ensures that hash functions serve as effective randomization primitives while simultaneously preventing information leakage through output correlation patterns, though it necessitates careful input canonicalization when deterministic identification across heterogeneous systems becomes paramount.

In the context of structured data processing, particularly when handling JSON objects or complex data structures, we must address the challenge of semantic equivalence versus byte-level representation. Two JSON objects may represent identical information while differing in their serialization due to key ordering, whitespace, or numeric formatting. To ensure deterministic hashing across implementations, we employ a canonicalization procedure $C(\cdot)$ that maps semantically equivalent values to identical byte sequences. The content digest of any value v is then defined as the composition $D(v) := H(C(v))$, where the canonicalization step, detailed in Section 2.1.2, guarantees platform-independent reproducibility.

When hash functions are employed within hierarchical structures such as Merkle trees, careful attention to domain separation becomes essential to prevent cross-protocol attacks and structural ambiguities. We adopt a systematic tagging scheme where different structural roles receive distinct single-octet prefixes, ensuring that leaf nodes and internal nodes occupy disjoint input spaces. Let $H : \{0,1\}^* \to \{0,1\}^{256}$ denote the cryptographic hash function employed (either SHA-256 or Keccak-256 as specified in Section 2.1.1). Leaf values are computed by prepending a null byte to the content digest:

$$h_{\text{leaf}}(v) := H(\texttt{0x00} \parallel D(v)), \tag{2.3}$$

where $D(v) = H(C_{\text{JCS}}(v))$ represents the content digest as previously defined. Internal nodes combining left and right child digests L and R are similarly tagged with a distinct prefix:

$$h_{\text{int}}(L, R) := H(\texttt{0x01} \parallel L \parallel R). \tag{2.4}$$

This tagged encoding scheme serves multiple security purposes simultaneously. It prevents type confusion attacks where an adversary might attempt to reinterpret internal nodes as leaves or vice versa, it ensures fixed-length inputs when combining digests, and it provides protection against length-extension vulnerabilities that historically affected certain hash constructions [8]. While modern sponge-based constructions such as Keccak-256, which underlies Ethereum's native hash function, are inherently immune to length extension, we maintain consistent encoding practices across all hash function families to ensure portability and defense in depth [9, 10].

The practical instantiation of these primitives depends on the deployment context. For off-chain computation and cross-platform interoperability, we standardize on SHA-256 given its ubiquitous support across programming languages, hardware security modules, and cryptographic libraries [11]. When on-chain verification within EVM-compatible environments is required, we adopt Keccak-256 to leverage the native `keccak256 opcode`, thereby minimizing gas consumption as discussed in Section 2.2.3. This dual-hash strategy, with explicit application-layer tagging to distinguish the two variants, optimizes for both computational efficiency and deployment flexibility while maintaining cryptographic integrity across heterogeneous environments.

2.1.2 JSON Canonicalization (JCS)

The ability to produce identical byte sequences from semantically equivalent data is fundamental to distributed systems that rely on cryptographic verification. JavaScript Object Notation (JSON), while ubiquitous in web-services and Application Programming Interfaces (APIs), allows multiple textual representations for the same abstract value: object members may appear in any order, insignificant whitespace can be inserted freely, and equal numbers may be written with different decimal spellings. In systems that rely on signatures, commitments, or Merkle roots, this variability is unacceptable because cryptographic operations must bind to one and only one byte sequence; otherwise the same data could hash to different digests depending on superficial formatting choices, breaking consensus mechanisms and invalidating proofs.

We adopt the JSON Canonicalization Scheme (JCS, RFC 8785), a deterministic transformation that maps any valid JSON value to a unique UTF-8 serialization [12]. The choice of UTF-8 encoding is critical for interoperability: it provides a universally accepted, byte-level representation that produces identical sequences across different programming languages, operating systems, and hardware architectures, eliminating platform-specific encoding variations that could compromise determinism. The scheme requires that objects do not contain duplicate member names and that their keys are ordered according to a total, reproducible ordering based on Unicode code points, with no locale or case-dependent rules. Arrays preserve their inherent order. Strings are emitted using minimal escaping and encoded in UTF-8 without applying Unicode normalization; characters that do not require escaping remain unescaped, while control characters, quotes, and backslashes are escaped as in standard JSON. Numbers are rendered in a minimal, unambiguous decimal form that preserves the exact binary value when converted back to IEEE 754 double precision [13]; this round-trip guarantee ensures that the same floating-point value always produces the same textual representation, regardless of the original decimal notation. Representations that are not valid JSON numbers in RFC 8259 (e.g., NaN or Infinity) are rejected during canonicalization. The result is a single, platform-independent byte string for each abstract JSON value.

Let $C_{\mathrm{JCS}}(\cdot)$ denote this canonicalization function. Consistently with Section 2.1.1, we define the content digest of a value v by hashing its canonical bytes, so that semantically equivalent values yield identical identifiers while any semantic modification triggers a different digest:

$$D(v) := H\big(C_{\mathrm{JCS}}(v)\big). \tag{2.5}$$

A concrete intuition helps clarify the transformation process. Consider the three JSON texts `{"b": 2, "a": 1.00}`, `{"a":1,"b":2}`, and `{\n "a":1,"b":2}`, which are semantically identical despite differences in whitespace, key order, and numeric spelling. JCS parses them to the same abstract value, orders the object keys by Unicode code points (placing `"a"` before `"b"`), renders the number `1.00` as `1` because this decimal representation is the shortest that converts back to the same binary value [14], removes superfluous whitespace, and emits the unique canonical UTF-8 serialization `{"a":1,"b":2}`. For strings, a value such as `"café"` remains unescaped and is encoded directly in UTF-8 as the byte sequence `0x63 0x61 0x66 0xC3 0xA9` (where é becomes the two-byte UTF-8 sequence), whereas a value containing a newline, e.g., `"a\nb"`, is rendered with the minimal escape sequence `"a\nb"`. By construction, the canonicalization does

not depend on parser quirks or runtime locales; the same input value always leads to the same byte sequence on all compliant implementations and platforms.

From a complexity standpoint, key sorting for an object with m members requires $O(m \log m)$ comparisons, which in practice is dominated by parsing and by the normalization of strings and numbers. Canonicalizing arrays is naturally streaming-friendly because order is preserved and elements can be emitted as they are parsed. For objects, JCS is not intrinsically streaming since all keys must be known before serialization; nevertheless, memory-bounded designs are feasible, e.g., by buffering only object keys and metadata before emitting the canonical order, while streaming array segments as usual. When strict streaming is a hard system requirement and both endpoints can agree on a binary format, Concise Binary Object Representation (CBOR) with deterministic encoding is a viable alternative [15]; in this thesis we retain JSON/JCS to maximize web-native interoperability and to align with widely deployed tooling.

2.1.3 Merkle Trees as Cryptographic Commitments

A fundamental challenge in distributed systems is efficiently proving that a specific piece of data belongs to a larger collection without transmitting the entire dataset. Merkle trees address this challenge by organizing data into a binary tree structure where each parent node contains the cryptographic hash of its children, ultimately producing a single root hash that serves as a compact fingerprint of the entire collection [16]. This construction enables a remarkable property: any element's membership can be verified by examining only a logarithmic number of hashes relative to the collection size, making verification feasible even for resource-constrained devices operating at the edge of cyber-physical systems.

To illustrate the practical value of this construction, consider applying it to a scenario in connected mobility where a vehicle needs to verify that a specific traffic update belongs to a certified dataset of thousands of updates. Rather than downloading and checking the entire dataset, the vehicle receives a short proof consisting of a path from the data item to the root, along with the necessary sibling hashes encountered along this path. By recomputing the hashes upward and comparing the final result with the known root, the vehicle achieves cryptographic certainty about the data's membership while consuming minimal bandwidth and computational resources. This same principle extends naturally to any cyber-physical system where edge devices must verify data authenticity without accessing complete datasets, e.g., IoT sensors validating configuration updates, smart building systems confirming energy pricing schedules, or distributed manufacturing nodes verifying production parameters.

The construction begins with a collection of values $v_1, \ldots, v_n$ that we wish to commit to. Each value is first canonicalized using JCS as described in Section 2.1.2, yielding the content digest $D(v_i) = H\big(C_{\mathrm{JCS}}(v_i)\big)$. To ensure deterministic tree construction independent of input ordering while preserving the ability to distinguish duplicate items, we impose a total order by sorting pairs $(D(v_i), i)$ lexicographically, first by digest and then by the original index i to break ties when multiple items share the same content. This approach maintains multiset semantics, e.g., two identical values at different positions remain distinguishable through their indices.

The tree structure employs explicit domain separation to prevent ambiguity between leaf and internal nodes. Each leaf is computed by prepending a distinctive tag to the content digest, while internal nodes combine their children's hashes with a different tag:

$$h_{\mathrm{leaf}}(v) = H(\texttt{0x00} \parallel D(v)), \tag{2.6}$$

$$h_{\mathrm{node}}(L, R) = H(\texttt{0x01} \parallel L \parallel R), \tag{2.7}$$

where L and R represent the left and right child digests respectively, each of fixed length 256 bits. This tagged structure ensures that the tree's topology is unambiguous and that no valid leaf can be misinterpreted as an internal node or vice versa, a property essential for security in systems where adversaries might attempt to forge proofs by exploiting structural ambiguities.

The tree is constructed bottom-up by pairing adjacent nodes at each level. When a level contains an odd number of nodes, the last unpaired node is promoted unchanged to the next level rather than being duplicated or paired with a synthetic value. This convention avoids introducing artificial redundancy and maintains a canonical tree shape for any given input set. The process continues until a single digest remains, which becomes the Merkle root r—a fixed-size commitment to the entire collection that changes unpredictably if any element is modified, added, or removed.

Verification of membership proceeds efficiently without reconstructing the entire tree. To prove that a value v_i is part of the committed collection, a prover supplies the value itself along with an authentication path consisting of the sibling hashes encountered from the leaf to the root, together with directional information (left or right) at each level. The verifier recomputes the leaf hash from the provided value, then iteratively combines it with the sibling hashes following the specified path directions, ultimately producing a candidate root. The proof is accepted if and only if this computed root matches the published commitment r. Both the proof size and verification complexity scale as $O(\log n)$, enabling practical verification even for collections containing millions of elements.

The security of this construction rests on the collision resistance of the underlying hash function. An adversary attempting to produce a fraudulent proof would need to find either a collision in H or a second preimage for one of the intermediate hashes, both of which are computationally infeasible for cryptographic hash functions [6, 17]. Furthermore, the deterministic ordering and explicit domain separation ensure that semantically different collections cannot produce the same root without violating the hash function's collision resistance, providing strong integrity guarantees for the committed data.

In our implementation, we standardize on SHA-256 for general-purpose deployments due to its widespread availability, hardware acceleration support, and extensive security analysis spanning decades. When on-chain verification is required on EVM-compatible platforms, we adopt Keccak-256 to align with the platform's native primitives and minimize gas costs. The choice between these hash functions is made explicit at the application layer through appropriate tagging, preventing any ambiguity when both variants coexist in a system. The concrete cost implications and optimization strategies for on-chain Merkle proof verification are examined in detail in Section 2.2.

2.2 Blockchain Platforms: Execution Semantics, Gas Economics, and Scaling)

The cryptographic primitives established in the previous section provide the mathematical foundation for data integrity and efficient verification, yet they alone cannot coordinate state transitions among mutually distrusting parties or enforce consistent execution of complex protocols across distributed systems. Blockchain platforms emerge as a systematic solution to this coordination problem, implementing replicated state machines that achieve *Byzantine fault tolerance* through a combination of cryptographic commitments, economic incentives, and deterministic execution environments. This section examines the architectural principles and operational mechanics of blockchain systems, with particular emphasis on platforms supporting programmable smart contracts that can encode arbitrary computational logic while maintaining consensus properties. We begin by establishing a taxonomy of distributed ledgers and their access models, clarifying the distinctions between blockchain and alternative DLT architectures while identifying the specific properties that make linear blockchains suitable for our experimental protocols. The analysis then proceeds to examine the Ethereum Virtual Machine as our reference execution environment, detailing how deterministic bytecode execution, atomic state transitions, and gas-based resource metering create a predictable computational substrate for decentralized applications. The economic dimensions of blockchain operation, particularly the gas cost model and its implications for protocol design, receive detailed treatment as these constraints fundamentally shape architectural decisions in resource-constrained environments. Finally, we survey contemporary scaling solutions including rollups and application-specific chains, analyzing how these architectures achieve higher throughput while preserving the execution semantics and security properties required for the cyber-physical system protocols developed in this dissertation.

2.2.1 Blockchain, Distributed Ledgers, and Access Taxonomy

Distributed Ledger Technologies (DLTs) are a family of replicated data structures designed to maintain a shared state across multiple nodes in the presence of faults or adversarial behavior [18]. At their core, DLTs implement state machine replication, a paradigm where multiple nodes maintain identical copies of a system's state and agree on the sequence of state transitions to apply, ensuring that each honest participant converges to the same global state despite network delays or Byzantine failures [19]. The concrete data structure and consensus mechanism that realize

this abstraction vary across systems.

Blockchains represent the most prominent subclass of DLTs, characterized by an append-only chain of blocks where each block cryptographically commits to its predecessor through hash pointers, forming an immutable historical record. Alternative DLT designs adopt different topological structures, most notably Directed Acyclic Graphs (DAGs), which organize transactions as vertices in a graph where edges represent causal dependencies, allowing multiple chains to coexist and merge rather than enforcing a single linear sequence [18, 20]. This structural difference enables DAG-based systems to process transactions in parallel, potentially increasing throughput at the cost of more complex finality rules and partial ordering semantics. In essence, all blockchains are DLTs, but not all DLTs are blockchains.

This dissertation focuses specifically on blockchain architectures because the protocols and measurements we develop rely on two fundamental properties that linear blockchains provide naturally. First, the total ordering of transactions, established by the underlying consensus protocol and embodied in the sequential block structure, simplifies both the formalization of on-chain governance mechanisms and the reproducible evaluation of execution costs and latencies, as every node processes transactions in the same deterministic sequence. Second, the hash-chained structure yields strong tamper-evidence through cryptographic commitments as detailed in Section 2.1.3, enabling succinct integrity anchoring for off-chain artifacts and providing auditable, verifiable experiment logs. While DAG-based ledgers offer valuable properties such as higher parallelism and potentially lower latency, their partial-order semantics and heterogeneous confirmation rules introduce complexities that are orthogonal to our evaluation goals and thus remain outside our scope.

Within the blockchain subclass, systems can be characterized along two orthogonal dimensions that determine their accessibility and governance model [21]. The first dimension concerns read visibility of the ledger state and transaction history. Public blockchains expose both current state and complete historical records to any observer without requiring authentication, thereby fostering independent verification, open auditability, and scientific reproducibility (these properties are central to our experimental methodology). In contrast, private blockchains restrict read access to authorized parties through cryptographic access controls or network-level restrictions, which can be advantageous in enterprise settings where data confidentiality is paramount but which inherently limit external auditability and independent verification.

The second dimension pertains to write permissions and the right to participate in consensus. Permissionless blockchains allow any participant to submit

transactions and, subject to the consensus protocol's rules such as proof-of-work or proof-of-stake requirements, to compete for the right to produce new blocks and extend the chain. Conversely, permissioned blockchains restrict these capabilities to a predetermined set of vetted identities operating under an explicit governance framework, typically managed through certificate authorities or multi-signature schemes. The four combinations arising from these two axes all exist in practice and serve different use cases. *Public* and *permissionless* systems, exemplified by Ethereum and Bitcoin, maximize openness and decentralization, forming the default substrate for our experiments due to their transparency and resistance to censorship. Public-permissioned deployments expose their state for audit purposes while confining block production rights to a controlled committee, balancing transparency with operational control. Private-permissioned solutions, common in consortium and enterprise deployments, prioritize regulatory compliance and data confidentiality while maintaining controlled membership. Private-permissionless configurations, while conceptually possible in specialized environments such as test networks, are uncommon in production systems and not pertinent to our evaluation goals.

We adopt public, permissionless platforms compatible with the Ethereum Virtual Machine (EVM) as our reference environment, as these systems provide the optimal combination of transparency, reproducibility, and decentralization required for our experiments. We occasionally contrast these with EVM-compatible subnets, which are sovereign blockchains that implement the EVM execution model but operate with independent validator sets and can configure custom parameters. These subnets trade the shared security guarantees of a global blockchain for greater configurability, enabling features such as deterministic fees or domain-specific permissioning models that can be valuable for controlled experiments while preserving the execution semantics required by our protocols.

2.2.2 EVM and Smart-Contract Execution Model

The execution environment that we considered in this dissertation, the EVM, has a computational engine that processes smart contract bytecode deterministically across all nodes in the network. The EVM operates as a stack-based virtual machine with 256-bit word size, meaning that it manipulates data through a last-in-first-out stack structure where each element occupies 256 bits, chosen to align with the cryptographic primitives commonly used in blockchain systems such as Keccak-256 hashes and elliptic curve signatures [22, 23]. This architecture eschews traditional register-based designs in favor of stack operations, simplifying the instruction set while ensuring that identical bytecode produces identical results

on any compliant implementation, a property essential for consensus among distributed nodes. The machine executes within a sandboxed environment on every validating node, ensuring that given an ordered transaction sequence and a fixed block environment, all honest nodes compute the same post-state and emit identical receipt logs, which proves essential for both the reproducibility of our experiments and the safety of the governance routines developed in subsequent chapters.

The EVM employs an account-based model that distinguishes between two fundamental types of accounts [24]. Externally Owned Accounts (EOAs) are controlled directly by users through cryptographic private keys and serve as the origin points for all transactions in the system, as only an entity possessing the corresponding private key can generate valid signatures to initiate state changes. Contract accounts, in contrast, are controlled by deployed bytecode and cannot initiate transactions independently but instead execute their code in response to being called by EOAs or other contracts. This distinction creates a clear security boundary where user intent, expressed through signed transactions from EOAs, triggers deterministic execution of contract logic that can manipulate state, transfer value, and invoke other contracts according to programmed rules.

Execution within the EVM follows strict atomicity and synchronicity principles that govern how state changes propagate through the system. When a contract function is invoked, whether directly from an EOA transaction or through an internal call from another contract, the execution proceeds synchronously within the current transaction context and must either complete successfully in its entirety or revert completely without any partial state modifications. This all-or-nothing semantic extends recursively through the entire call graph, meaning that if a deeply nested contract call fails, all state changes initiated by parent calls within the same transaction are rolled back, ensuring consistency even in complex multi-contract interactions. The atomic nature of transactions guarantees that the global state transitions from one consistent state to another without intermediate partially-applied states being visible to other transactions.

The EVM's isolation model strictly constrains how smart contracts interact with their environment, enforcing determinism and security through deliberate limitations. Contracts execute without access to system calls, network interfaces, or filesystem operations, operating instead through a carefully controlled set of opcodes that can only read and modify the blockchain's global state and emit event logs. Their persistent storage consists of a key-value mapping from 256-bit words to 256-bit words, providing durable state that survives between transactions, while transient memory exists only for the duration of a single transaction execution. The only mechanism for contracts to communicate information to off-chain observers

is through event logs, which are recorded immutably in transaction receipts and indexed for efficient querying but do not constitute part of the consensus state itself. This design ensures that contract execution remains deterministic and verifiable while providing sufficient expressiveness for complex decentralized applications.

Gas metering serves as the fundamental resource accounting mechanism that prevents infinite loops and ensures fair resource allocation across the network. Every operation in the EVM, from arithmetic operations to memory allocation to storage modifications, consumes a predetermined amount of gas that reflects its computational and storage burden on the network. Storage operations command particularly high gas costs, especially when writing to previously uninitialized storage slots, as these operations permanently expand the global state that all nodes must maintain. Memory expansion follows a quadratic cost model that grows super-linearly with peak usage, incentivizing efficient memory management, while operations that reduce state size, such as clearing storage slots, receive gas refunds to encourage state pruning. Detailed fee mechanics and their implications for protocol design are analyzed comprehensively in Subsection 2.2.3, but the key insight is that minimizing on-chain state footprint becomes a first-class design objective that fundamentally shapes architectural decisions.

Two architectural principles permeate the smart contract designs developed in this dissertation, both emerging from the constraints and capabilities of the EVM execution model. First, we adopt a commit-and-verify paradigm that maintains minimal on-chain footprints while cryptographically anchoring rich off-chain artifacts through the canonicalization pipelines and Merkle aggregation techniques introduced in Section 2.1.3. Contracts persist only succinct commitments in the form of single content hashes or Merkle roots, typically requiring just 32 bytes of storage, while complete data structures and their proofs remain off-chain until verification is required, at which point verifiers supply compact inclusion proofs that the EVM can validate efficiently. Second, we decompose contract systems into single-responsibility modules that communicate through explicit state transitions and structured event emissions, separating concerns such as data filtering, reputation management, voting mechanics, and settlement logic into distinct contracts. This modular boundary enforcement makes invariants explicit and verifiable while constraining persistent storage to safety-critical variables such as counters, reputation scores, governance flags, and commitment roots, whereas per-item details and individual vote records are externalized to event logs that provide auditability without storage overhead.

Temporal semantics in the EVM derive exclusively from the consensus layer's view of time, as smart contracts have no access to wall-clock time or external temporal sources. Block timestamps, provided by block producers and validated

through consensus rules, serve as the only temporal reference available on-chain, though their granularity and potential for minor manipulation within protocol bounds must be considered in mechanism design [25, 26]. Consequently, all time-dependent logic including deadline enforcement, voting periods, reputation decay, and settlement windows must be expressed in terms of block numbers or block timestamps rather than real-world time. Settlement routines are designed to be idempotent, meaning they can be safely executed multiple times without causing duplicate effects, which proves essential for handling the variable finality characteristics of different blockchain platforms and protecting against accidental or malicious replay attempts.

The portability of the EVM execution model across multiple blockchain platforms proves instrumental for our experimental methodology. Identical smart contract bytecode executes deterministically on Ethereum mainnet, Layer-2 rollup solutions that will be detailed in Subsection 2.2.4, the Avalanche C-Chain, and various Avalanche subnets configured with EVM-compatible runtimes [27]. While minor variations exist in gas schedules, available precompiled contracts, or block time parameters, these differences do not affect the functional determinism of contract logic and can be quantified through systematic profiling during evaluation. This cross-platform compatibility enables us to conduct experiments under different economic and performance conditions while maintaining consistent execution semantics, facilitating both cost-effective development on test networks and production deployment on mainnet infrastructure when appropriate.

2.2.3 Gas Economics and Cost Model

The execution of arbitrary computation on a decentralized network introduces fundamental challenges that do not arise in traditional computing environments. Without a central authority to regulate resource consumption, malicious actors could submit transactions containing infinite loops or computationally intensive operations that would permanently stall the network, rendering it unusable for legitimate participants. This vulnerability, rooted in the undecidability of the halting problem, necessitates a mechanism that bounds computational resources consumed by any single transaction while preventing denial-of-service attacks through transaction spam [28]. The Ethereum Virtual Machine addresses these challenges through a gas metering system that serves as both a resource accounting mechanism and an economic deterrent against network abuse.

Gas functions as an abstract unit of computation that meters every operation executed within the EVM, from simple arithmetic to complex state modifications. Each opcode in the EVM instruction set carries a predetermined gas cost that reflects

its computational complexity and its burden on network resources, particularly the permanent storage requirements that all nodes must maintain in perpetuity. Crucially, gas costs are borne exclusively by the entity initiating a state-changing transaction—the sender who signs and broadcasts the transaction to the network. This asymmetric cost model creates a fundamental distinction between two classes of operations: state-changing transactions that modify the blockchain's global state and must be included in blocks by miners or validators, thereby consuming network resources and requiring gas payment; and read-only queries that execute locally on individual nodes without consensus participation, incurring no gas costs to the caller. This distinction proves essential for protocol design, as it incentivizes architectures that minimize on-chain state mutations while enabling unlimited off-chain verification and querying.

The gas mechanism operates through a prepayment model that guarantees termination and prevents resource exhaustion. When submitting a transaction, the sender specifies a gas limit representing the maximum computational resources they are willing to purchase, along with a gas price indicating their willingness to pay per unit of gas consumed. The EVM interpreter tracks gas consumption incrementally as it processes each opcode, deducting the appropriate cost from the transaction's gas allowance. If the transaction completes successfully within the allocated gas budget, any unused gas is refunded to the sender; however, if the gas is exhausted before completion, the EVM immediately halts execution, reverts all state changes made within that transaction, yet still charges the sender for the gas consumed up to the point of failure. This atomic reversion ensures state consistency while the irrevocable gas charge prevents attackers from deliberately crafting failing transactions to waste network resources without cost.

The EVM meters state-changing execution in gas, an abstract unit accounting for opcode steps, memory expansion, persistent storage writes, and the size of calldata and logs. A transaction declares an upper gas limit; the interpreter consumes gas as execution progresses and either completes if the budget suffices or reverts on exhaustion. On fee-market networks, the effective payment is

$$\begin{aligned}\text{feePaid} \triangleq \text{gasUsed}\cdot \\ \min\big(\text{maxFeePerGas},\ \text{baseFee} + \text{maxPriorityFeePerGas}\big),\end{aligned} \tag{2.8}$$

with the base fee burned and the priority tip accruing to the block producer. This mechanism stabilizes congestion without altering opcode-level determinism [22, 29].

Call data is charged per byte, with different rates for zero and nonzero bytes, making compact binary encodings desirable. Memory expansion follows a

super-linear schedule, so algorithms should limit peak memory. Emitting logs incurs a fee per topic and per byte; although logs do not enlarge the Merkle state, they are charged at emission time.

Read-only execution is free to the caller only off the consensus path: the JSON-RPC `eth_call` primitive runs bytecode locally against a chosen state and pays no fees, whereas the same path within a transaction is fully metered. At block granularity, a bounded gas target regulates throughput via the base-fee adjustment. Execution is atomic: any revert discards state changes while still charging for gas consumed up to the failure point. Receipts expose gasUsed and structured logs uniformly across success and failure, enabling reproducible profiling.

These constraints inform the design choices adopted here without repeating the architectural discussion in Subsection 2.2.2. On-chain state is kept compact; voluminous artifacts are replaced with succinct commitments (single hashes or Merkle roots) whose canonicalization and verification are detailed in Section 2.1; per-item data travel as events rather than persistent storage. Aggregations that must be on chain are maintained as monotone counters to avoid write amplification, and settlement routines are idempotent to permit safe retries. In aggregate, these patterns bound SSTORE operations, keep calldata to a handful of 256-bit words, and limit memory footprints, yielding predictable fees on L1, rollups, and EVM-compatible subnets.

2.2.4 Scaling the EVM: Rollups, Validity Systems, and EVM-Compatible Subnets

The proliferation of EVM-compatible execution environments across diverse blockchain architectures creates a remarkable property for smart contract development: contracts written in Solidity, the predominant high-level language for EVM bytecode generation, compile to identical bytecode that executes deterministically across any compliant implementation. This portability transcends mere syntactic compatibility, as the deterministic compilation from Solidity source to EVM bytecode, combined with the standardized opcode semantics specified in the Ethereum Yellow Paper [22], ensures that a contract deployed with identical initialization parameters will exhibit functionally equivalent behavior whether executing on Ethereum mainnet, a Layer-2 rollup, an Avalanche subnet, or a local development environment. The practical implications for research and development prove substantial: protocols can be iteratively refined on low-cost test networks where transactions cost fractions of a cent, systematically validated on public testnets that mirror mainnet conditions without economic risk, and ultimately deployed to production environments with confidence that the extensively tested logic will

execute identically. This write-once, deploy-anywhere paradigm, reminiscent of Java's platform independence but achieved through cryptographic consensus rather than virtual machine abstraction, fundamentally shapes the experimental methodology employed throughout this dissertation, enabling cost-effective iteration cycles where complex multi-contract systems can be stress-tested across different economic and performance regimes before committing to more expensive and critical mainnet deployment.

The fundamental scalability constraints of blockchain systems arise from the requirement that every full node must process every transaction and maintain the complete global state, creating an inherent tension between decentralization, which demands low hardware requirements for node operators, and throughput, which would benefit from more powerful infrastructure. This trilemma has motivated the development of Layer-2 scaling solutions, a term that encompasses architectures which move computation and state transitions off the primary blockchain (Layer 1) while inheriting its security guarantees through cryptographic proofs or economic mechanisms [30,31]. Rather than requiring every node to execute every transaction, these systems perform execution in a more restricted environment and then provide the main chain with sufficient evidence to verify that the execution was correct, thereby achieving higher throughput without compromising the security properties of the underlying Layer-1 blockchain.

The predominant Layer-2 architecture for EVM compatibility consists of rollups, systems that execute transactions in an off-chain environment running a full EVM implementation and periodically submit compressed representations of state transitions back to Layer 1 [32]. Two fundamentally different security models have emerged for ensuring the validity of these off-chain computations, each making distinct trade-offs between complexity, cost, and finality characteristics.

Optimistic rollups adopt a fraud-proof paradigm where submitted state transitions are presumed valid unless challenged within a specified time window, typically one to seven days. During this challenge period, any observer can submit a fraud proof demonstrating that a state transition was computed incorrectly, triggering an on-chain verification process that either confirms the original submission or proves the fraud and reverts the invalid state transition. This optimistic assumption dramatically reduces computational overhead since proofs are only generated and verified when disputes arise, and the system can maintain near-perfect EVM compatibility since the fraud-proof mechanism can accommodate the full complexity of EVM execution. The primary drawback manifests in withdrawal latency, as users must wait for the challenge period to expire before Layer-1 finality is achieved and funds can be safely withdrawn to the main chain, though various liquidity mechanisms have emerged to mitigate this delay for typical users.

Validity rollups, often referred to as zero-knowledge rollups despite not all implementations requiring zero-knowledge properties, take the opposite approach by generating cryptographic proofs that mathematically demonstrate the correctness of every state transition [33]. These systems employ sophisticated cryptographic techniques, such as SNARKs (Succinct Non-interactive Arguments of Knowledge) or STARKs (Scalable Transparent Arguments of Knowledge), to produce proofs that can be verified on-chain in constant time regardless of the complexity of the underlying computation. The implementation of these proof systems for the EVM, termed zkEVM, represents a significant engineering achievement as it requires encoding the entire EVM execution semantics into arithmetic circuits or polynomial constraints that can be proven efficiently [34]. The term zkEVM itself encompasses a spectrum of approaches ranging from full bytecode-level compatibility, where existing Ethereum contracts can be deployed unchanged, to high-level language compatibility, where contracts must be written in specialized languages that compile to more proof-friendly representations.

A variant architecture known as validium extends the validity rollup concept by decoupling data availability from data validity [35]. While traditional rollups publish complete transaction data to Layer 1, ensuring that anyone can reconstruct the full state, validium systems store transaction data off-chain with a separate data availability provider and only submit validity proofs and state commitments to Layer 1. This approach dramatically reduces Layer-1 storage costs and enables even higher throughput, as the main chain only needs to verify succinct proofs rather than store full transaction data. The trade-off manifests in additional trust assumptions regarding data availability, as users must trust that the off-chain data will remain accessible for them to generate withdrawal proofs, though various mechanisms such as data availability committees or decentralized storage networks can mitigate this risk.

The economics of Layer-2 systems reflect their architectural differences and create distinct incentive structures for users and operators. Transaction fees in rollup systems decompose into two components that reflect the dual nature of the computation:

$$\begin{aligned} \text{fee}_{\text{L2}} = \; & \underbrace{\text{gasUsed}_{\text{L2}} \cdot \left(\text{baseFee}_{\text{L2}} + \text{priorityFee}_{\text{L2}}\right)}_{\text{execution on Layer 2}} \\ & + \underbrace{\text{bytes}_{\text{DA}} \cdot \text{price}_{\text{DA}\to\text{L1}}}_{\text{data availability on Layer 1}} . \end{aligned} \tag{2.9}$$

The execution component follows standard EVM gas accounting but at Layer-2

prices, which are typically orders of magnitude lower than Layer 1 due to the more centralized and efficient execution environment. The data availability component captures the cost of publishing transaction data or state commitments to Layer 1, where bytes_{DA} represents the compressed size of the transaction data and $\text{price}_{\text{DA}\to\text{L1}}$ reflects the current cost of Layer-1 block space. The introduction of EIP-4844 has created a separate fee market for blob data specifically designed for rollup data publication, substantially reducing this component of Layer-2 fees [36]. Validium architectures eliminate the data availability fee entirely by keeping data off Layer 1, though they may introduce alternative costs for off-chain data storage and availability guarantees.

Current rollup implementations typically employ centralized sequencers that order transactions and produce blocks, improving latency and user experience through instant soft confirmations while introducing temporary trust assumptions in the sequencer's behavior [37]. This centralization is mitigated through various mechanisms including forced transaction inclusion, where users can submit transactions directly to Layer 1 if the sequencer censors or becomes unavailable, and escape hatches that enable users to exit the rollup even in adversarial scenarios. The roadmap toward decentralized sequencing involves either rotating sequencer responsibilities among a permissioned set or implementing fair ordering protocols that prevent front-running and maximize extractable value capture, though these mechanisms remain in active development across different rollup implementations.

An orthogonal approach to EVM scaling emerges through application-specific blockchains, exemplified by the Avalanche platform's subnet architecture [27]. Avalanche itself operates as an independent Layer-1 blockchain with its C-Chain providing native EVM compatibility, achieving high throughput through its novel consensus protocol while maintaining full smart contract portability. Beyond this base layer, Avalanche introduces *subnets*, which are sovereign blockchains that can spawn from the primary network while implementing their own EVM-compatible execution environments with independent validator sets and economic parameters. This architectural pattern bears structural similarities to Ethereum's Layer-2 scaling approach, yet differs fundamentally in its security model: while Ethereum rollups inherit security from the base chain through cryptographic proofs or fraud challenges, Avalanche subnets derive their security entirely from their own validator set, creating a distinct trust boundary. This sovereignty enables subnets to implement custom gas tokens, modify fee schedules, enforce permissioned access, or optimize for specific application requirements without affecting the parent C-Chain.

The trade-off between sovereignty and shared security is explicit in this subnet model. While rollups benefit from the accumulated security of Ethereum's vast

validator network through cryptographic or economic bonds, subnets operate with complete autonomy, where security scales with the value staked by their specific validators rather than inheriting guarantees from the underlying platform. This independence provides a controlled experimental environment that proves valuable for testing novel mechanisms before deployment on public networks. Subnets can therefore serve as application-specific testing grounds where developers iterate on economic models and governance mechanisms without the constraints or costs associated with public mainnet deployment, though they must carefully consider the reduced security guarantees that come from operating outside a highly decentralized Layer-1 anchor.

The practical implications of these scaling approaches for the protocols developed in this dissertation center on three key considerations. First, the identical EVM execution semantics across Layer 1, Layer-2 rollups, and EVM-compatible subnets ensure that our smart contract logic remains portable and deterministic regardless of the deployment target, though gas optimization strategies must account for the different cost structures. Second, the varying finality models, from immediate soft confirmation to delayed hard finality, necessitate careful design of settlement mechanisms that remain safe under different confirmation assumptions, motivating the idempotent settlement routines discussed in Subsection 2.2.2. Third, the dramatic cost differences between platforms, potentially spanning three orders of magnitude from mainnet to subnets, enable iterative development workflows where contracts are tested extensively on low-cost environments before production deployment, while the experimental methodology must account for these cost variations when evaluating economic mechanisms.

3 Optimization and Learning for CPS

3.1 Graph-Theoretic Models and Integer Programming Paradigms

Graph-theoretic representations provide the fundamental abstraction for modeling discrete optimization problems that arise in cyber-physical systems, where entities and their relationships naturally map to nodes and edges. In urban contexts, road networks become directed graphs with intersections as vertices and street segments as edges, while in organizational settings, compatibility constraints between personnel and resources translate into bipartite matching structures. This representational framework enables the systematic encoding of spatial, temporal, and relational constraints that govern real-world systems, transforming complex operational requirements into mathematically tractable formulations.

Integer Linear Programming (ILP) and its generalization, Mixed-Integer Programming (MIP), constitute the predominant paradigm for solving combinatorial optimization problems on these graph structures. The restriction to integer-valued decision variables captures the inherently discrete nature of many allocation and routing decisions: a vehicle either traverses a specific road segment or does not, a person is assigned to exactly one office, a resource is allocated in whole units rather than fractions. While the integrality constraints render these problems NP-hard in general, modern solvers leverage sophisticated branch-and-bound algorithms, cutting plane methods, and preprocessing techniques to solve practical instances with thousands of variables and constraints within acceptable time bounds.

The strength of the ILP/MIP framework lies not merely in its computational tractability but in its expressiveness. Linear constraints naturally encode capacity limitations, precedence relationships, mutual exclusion requirements, and conservation laws that govern physical systems. The objective function, being a linear combination of decision variables, can represent total costs, distances, delays, or any additive performance metric. This linearity assumption, while restrictive, often suffices for practical applications and enables the exploitation of duality theory, sensitivity analysis, and decomposition techniques that would be intractable for general nonlinear formulations.

Three solution paradigms emerge as particularly relevant for cyber-physical

applications, each addressing distinct aspects of multi-stakeholder decision-making under uncertainty and competing objectives. The first paradigm employs **minimax objectives** to ensure fairness and robustness under worst-case scenarios. Rather than optimizing average performance, which might mask severe imbalances or leave certain entities severely disadvantaged, the minimax formulation explicitly minimizes the maximum cost, delay, or resource consumption experienced by any participant. This egalitarian approach proves essential when system acceptability depends on avoiding extreme outcomes, such as preventing any single room from becoming overcrowded or ensuring that no vehicle experiences excessive delays.

The mathematical formulation of minimax objectives introduces an auxiliary variable that bounds the individual costs from above, transforming the nonlinear max operator into a set of linear constraints. Specifically, if c_i represents the cost incurred by entity i, the minimax objective $\min \max_i c_i$ becomes $\min z$ subject to $z \geq c_i$ for all i, where z is the auxiliary variable. This linearization preserves optimality while maintaining compatibility with standard ILP solvers, though it increases the problem dimensionality by one variable and adds as many constraints as there are entities to consider.

The second paradigm, **lexicographic optimization**, addresses scenarios where objectives exhibit strict priority orderings that cannot be captured by weighted-sum scalarization. In multi-objective settings, stakeholders often express preferences not as numerical trade-offs but as hierarchical priorities: safety must be guaranteed before efficiency is considered, regulatory compliance takes precedence over cost minimization, or user comfort is optimized only after capacity constraints are satisfied. Lexicographic optimization operationalizes these priorities by solving a sequence of optimization problems, where each successive problem incorporates the optimal value of higher-priority objectives as constraints.

The lexicographic approach proceeds iteratively: first, the highest-priority objective is optimized to obtain its optimal value; then, this value is fixed as a constraint while optimizing the second-priority objective, and the process continues through all objectives in order. This methodology guarantees Pareto optimality while respecting the prescribed priority structure, though it requires solving multiple optimization problems and may yield solutions that are arbitrarily poor in lower-priority objectives if the constraint hierarchy is too restrictive. The computational burden can be mitigated by allowing small tolerances in the constraint enforcement, effectively trading strict lexicographic ordering for improved lower-priority performance within acceptable bounds.

The third paradigm introduces **feasibility-switching mechanisms** that gracefully handle overconstrained problems by automatically relaxing requirements when no feasible solution exists under the original specification. Real-world

systems often operate under aspirational targets that may become unattainable due to unexpected disruptions, resource limitations, or conflicting requirements. Rather than declaring the problem infeasible and providing no solution, feasibility-switching techniques employ a hierarchy of constraint sets, attempting to satisfy the most stringent requirements first and progressively relaxing them until a feasible solution emerges.

The implementation of feasibility switching typically involves binary indicator variables that activate or deactivate constraint sets, coupled with big-M formulations that render constraints ineffective when their corresponding indicators are zero. Consider a capacity constraint requiring room occupancy below eighty percent: if this proves infeasible, the system automatically switches to a fallback constraint allowing full occupancy. The switching logic is encoded through complementary constraints that ensure exactly one constraint set is active, while the objective function may include penalty terms that discourage unnecessary relaxation. This approach guarantees that some solution is always returned while maintaining transparency about which requirements were relaxed.

These three paradigms (minimax fairness, lexicographic prioritization, and feasibility switching) are not mutually exclusive but rather complementary techniques that can be combined within a single optimization framework. A lexicographic formulation might use minimax objectives at certain priority levels, while feasibility switching ensures that each lexicographic stage yields some solution even if ideal targets prove unattainable. The resulting models exhibit both robustness to uncertainty and adaptability to varying operational contexts, essential properties for deployment in dynamic urban environments.

The computational complexity of ILP/MIP problems necessitates careful consideration of solution quality versus computation time trade-offs. While small instances may be solved to proven optimality using exact algorithms, larger problems often require approximation techniques or heuristic methods. Valid inequalities constitute a powerful class of cutting planes that strengthen the linear programming relaxation by eliminating fractional solutions from the feasible region while preserving all integer feasible points, thereby significantly reducing solution times through more aggressive pruning of the branch-and-bound search tree [38, 39]. Problem-specific decomposition techniques, such as Dantzig-Wolfe decomposition for problems with block structure [40] or Benders decomposition for problems with complicating variables, enable the solution of problems that would be intractable if approached monolithically [41].

Performance guarantees for polynomial-time approximation algorithms depend critically on problem structure, revealing a fundamental dichotomy in computational complexity that governs practical solvability. For certain graph problems,

such as minimum vertex cover or maximum matching, constant-factor approximations exist, enabling solutions within provable bounds of optimality through efficient algorithms. Conversely, the general integer programming problem exhibits a more pessimistic complexity-theoretic landscape, where the existence of efficient approximation schemes would imply the unexpected resolution of foundational questions in theoretical computer science regarding the relationship between deterministic and nondeterministic polynomial-time computation. This theoretical barrier manifests practically as an inherent trade-off: as we seek approximations with tighter optimality guarantees, the required computational effort grows prohibitively, often matching or exceeding that of exact solution methods. The identification of special cases where efficient algorithms exist—such as totally unimodular constraint matrices that guarantee integer vertices in the linear programming relaxation, or network flow problems that admit polynomial (time exact solutions through specialized algorithms) guides the modeling process toward formulations that exploit these structural properties. This awareness of computational boundaries shapes not merely the choice of solution algorithm but the very formulation of the optimization model itself, encouraging reformulations that transform intractable problems into sequences of tractable subproblems or relaxations that preserve essential solution characteristics while enabling efficient computation.

The transition from theoretical formulation to practical implementation requires careful attention to numerical stability, solver selection, and parameter tuning. Modern commercial solvers like CPLEX (IBM ILOG CPLEX Optimizer; IBM) and Gurobi (Gurobi Optimizer; Gurobi Optimization, LLC) incorporate decades of algorithmic improvements and can exploit problem structure through automatic reformulation, symmetry detection, and parallel processing. Open-source alternatives such as CBC (COIN-OR Branch-and-Cut; COIN-OR Foundation) and SCIP (Solving Constraint Integer Programs; developed at the Zuse Institute Berlin) provide comparable functionality for many applications while enabling deeper customization. The choice of solver, along with decisions about preprocessing, cutting plane generation, and branching strategies, can affect solution times by orders of magnitude for the same mathematical model.

3.2 Model Predictive Control with LSTM-Based System Identification

Model Predictive Control represents a sophisticated control methodology that distinguishes itself from classical feedback approaches through its explicit incorporation of future behavior predictions into current control decisions. At its core, MPC operates by solving an optimization problem at each control instant, considering a finite prediction horizon while respecting system constraints and performance objectives. This receding-horizon philosophy, where the optimization window continuously shifts forward in time, enables the controller to anticipate future system evolution and preemptively adjust control actions, rather than merely reacting to observed deviations.

The fundamental challenge in deploying MPC for complex cyber-physical systems lies in obtaining accurate predictive models that capture the intricate dynamics of real-world processes. Traditional physics-based modeling approaches, while providing interpretable representations grounded in first principles, often struggle with nonlinear phenomena, time-varying parameters, and the presence of unmeasured disturbances. Building thermal dynamics, for instance, involve complex heat transfer mechanisms, occupancy patterns, and weather-dependent behaviors that resist precise analytical characterization. Similarly, traffic flow exhibits emergent properties arising from individual driver decisions, infrastructure constraints, and stochastic events that challenge conventional modeling paradigms.

Long Short-Term Memory (LSTM) networks emerged from the recognition that standard recurrent neural architectures suffer from vanishing and exploding gradient problems when learning long-term temporal dependencies [42]. The LSTM architecture introduces a sophisticated gating mechanism (comprising input, forget, and output gates) that selectively retains, updates, and exposes information across time steps. This controlled information flow through the cell state enables LSTMs to maintain relevant historical context over extended sequences, making them particularly suited for modeling systems where current behavior depends on events from the distant past. The gates themselves are learned during training, allowing the network to automatically discover which temporal patterns and dependencies are most relevant for accurate prediction.

The synergy between MPC and LSTM architectures arises from their complementary strengths. While MPC provides a principled optimization framework for constraint-aware decision-making, it requires accurate multi-step-ahead predictions to compute effective control trajectories [43]. LSTMs, having demonstrated exceptional capability in sequence-to-sequence learning tasks, naturally fulfill this

requirement by learning complex temporal mappings from historical data. The learned LSTM model serves as the prediction engine within the MPC formulation, transforming the control problem from one requiring explicit system identification to one leveraging data-driven function approximation.

Training an LSTM for integration with MPC involves several methodological considerations that extend beyond standard supervised learning practices. The prediction model must not only minimize one-step-ahead forecast errors but also maintain accuracy over the entire prediction horizon employed by the MPC controller. This requirement motivates the use of multi-step-ahead loss functions during training, where the network is penalized for prediction errors at multiple future time steps. *Teacher forcing*, a training technique where ground-truth values are fed as inputs during the learning phase, accelerates convergence but can create a mismatch between training and deployment conditions [44, 45]. To address this exposure bias, scheduled sampling gradually transitions from teacher forcing to using the model's own predictions as inputs, improving robustness to prediction errors that accumulate during multi-step forecasting.

The temporal structure of the training data significantly influences model performance. Time series from cyber-physical systems often exhibit multiple periodicities (daily, weekly, and seasonal patterns) alongside trend components and irregular variations. The training dataset must be sufficiently rich to capture these diverse temporal scales while avoiding overfitting to specific historical episodes. Data augmentation techniques, such as adding controlled noise or generating synthetic trajectories through simulation, can enhance model generalization, particularly when historical data is limited or does not fully explore the operational envelope.

Validation of LSTM-based predictive models for MPC applications requires assessment beyond conventional machine learning metrics. While mean squared error and other point-wise accuracy measures provide useful diagnostics, the ultimate validation criterion is closed-loop control performance. This necessitates evaluating how prediction errors propagate through the MPC optimization and affect tracking accuracy, constraint satisfaction, and control effort. Monte Carlo simulations with randomized initial conditions and disturbance sequences reveal the robustness of the integrated system, while hardware-in-the-loop testing provides final validation before deployment.

The computational architecture for real-time implementation presents additional design considerations. The LSTM inference must execute within the sampling period of the control system, potentially on edge devices with limited computational resources. Model compression techniques, including pruning, quantization, and knowledge distillation, can reduce the computational footprint while preserving

prediction accuracy. The MPC optimization itself benefits from warm-starting strategies, where the solution from the previous time step initializes the current optimization, reducing iteration counts and ensuring timely completion.

The integration of learning-based components into safety-critical control systems raises important questions about reliability and interpretability. Unlike physics-based models whose behavior can be analyzed through established control-theoretic tools, LSTM predictions may exhibit unexpected behaviors outside the training distribution. Techniques for uncertainty quantification, such as ensemble methods or Bayesian neural networks, provide confidence bounds on predictions that can be incorporated into robust MPC formulations. These approaches enable the controller to adopt conservative strategies when prediction uncertainty is high, trading performance for safety.

The described architecture, combining the systematic optimization of MPC with the adaptive modeling capacity of LSTMs, establishes a flexible framework for controlling complex cyber-physical systems. The methodology adapts to diverse application domains (from building climate control to traffic management—by learning domain-specific dynamics from data while maintaining the constraint-handling and multi) objective optimization capabilities essential for practical deployment.

3.3 Deep Reinforcement Learning for Sequential Decision-Making

Reinforcement Learning (RL) constitutes a computational paradigm wherein agents learn optimal behavioral policies through iterative interactions with an environment, receiving scalar feedback signals that encode the desirability of state-action pairs. Unlike supervised learning, which relies on labeled examples of correct input-output mappings, and unsupervised learning, which discovers latent structure in unlabeled data, reinforcement learning addresses the fundamental problem of learning what actions to take, in which situations, to maximize a notion of cumulative reward over time. The theoretical foundations of RL rest upon the framework of Markov Decision Processes (MDPs), which provide a mathematically tractable formalization of sequential decision-making under uncertainty, while modern deep reinforcement learning (DRL) extends these classical methods by leveraging neural networks as function approximators, enabling the treatment of high-dimensional state and action spaces that characterize real-world cyber-physical systems.

The canonical MDP formulation comprises a five-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$ denotes the state space encompassing all possible configurations of the environment, $\mathcal{A}$ represents the action space available to the agent, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ defines the state transition probability function encoding the environment's dynamics, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ specifies the immediate reward function, and $\gamma \in [0, 1)$ serves as the discount factor that controls the trade-off between immediate and future rewards [46, 47].

The Markov property, which asserts that future states depend only on the current state and action rather than the entire history, enables tractable solution methods while remaining sufficiently expressive to model a broad class of sequential decision problems. An agent's behavior is characterized by a policy $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$, mapping states to probability distributions over actions, with the objective of finding an optimal policy π^* that maximizes the expected discounted cumulative reward, formalized as the value function or its action-value counterpart.

Among the constellation of algorithms developed for solving MDPs in the reinforcement learning context, policy gradient methods have emerged as particularly effective for continuous control problems and scenarios with large or continuous action spaces. These methods directly optimize parameterized policies through gradient ascent on expected returns, circumventing the need for explicit value function estimation that characterizes value-based approaches such as Q-learning [48]. The theoretical elegance of policy gradient approaches, first formalized through

the policy gradient theorem, establishes that the gradient of expected returns can be estimated through Monte Carlo sampling without requiring knowledge of the environment dynamics, a property that proves invaluable when dealing with complex cyber-physical systems where accurate models are difficult to obtain. However, the vanilla policy gradient formulation suffers from high variance in gradient estimates and sensitivity to step size selection, where excessively large updates can catastrophically degrade previously learned behaviors while conservative steps result in prohibitively slow convergence. These limitations motivated the development of natural policy gradient methods that incorporate Fisher information to achieve invariance to policy parameterization, though at substantial computational cost, and subsequently led to trust region policy optimization approaches that explicitly constrain the magnitude of policy updates through KL-divergence penalties between successive policies. Proximal Policy Optimization (PPO), emerging from this evolutionary trajectory as an ingenious simplification of trust region methods, achieves comparable performance through a clipped surrogate objective that implicitly enforces trust region constraints without requiring complex second-order optimization or conjugate gradient procedures [49]. The clipped objective function, which takes the minimum between the standard importance-weighted advantage estimate and a pessimistically clipped version, elegantly prevents both excessively large policy improvements that might destabilize training and the exploitation of advantage estimation errors that plague importance sampling methods. This balance between theoretical rigor and computational practicality has established PPO as a de facto standard for practical applications, particularly in domains where sample collection is expensive or where training robustness across diverse initial conditions is paramount, characteristics that precisely match the requirements of urban cyber-physical systems where real-world deployment demands both sample efficiency and reliable convergence.

The core innovation of PPO lies in its treatment of the policy optimization problem through a surrogate objective that approximates the true policy gradient while incorporating safeguards against excessive updates. Rather than directly maximizing the expected return, PPO optimizes a clipped probability ratio that effectively limits the magnitude of policy changes between successive iterations, preventing the algorithm from taking overly aggressive steps that might lead to performance degradation. The clipping mechanism operates by computing the ratio between the new and old policy probabilities for each state-action pair, then constraining this ratio to lie within a specified range, typically parameterized by a hyperparameter (ϵ). This approach elegantly sidesteps the computational complexity of second-order methods while retaining their stability benefits, making PPO particularly well-suited for problems where the policy must be refined

over extended training periods without manual intervention or hyperparameter scheduling.

A critical consideration in applying reinforcement learning to real-world problems involves the design of *reward functions* that accurately capture the desired behavior while remaining learnable through exploration. Reward shaping, the practice of augmenting or modifying the natural reward signal to accelerate learning or guide exploration, represents both an opportunity and a challenge in practical deployments. While carefully designed shaped rewards can dramatically improve sample efficiency and convergence speed, poorly conceived modifications may introduce unintended biases or lead to suboptimal policies that exploit the shaped signal rather than solving the true underlying task. The principle of potential-based reward shaping provides theoretical guarantees that certain forms of reward augmentation preserve optimal policies, allowing domain knowledge to be incorporated without fundamentally altering the problem structure. This theoretical foundation enables practitioners to encode prior knowledge about desirable intermediate behaviors, safety constraints, or efficiency objectives while maintaining convergence to globally optimal solutions.

The challenge of multi-objective optimization frequently arises in cyber-physical systems, where multiple, potentially conflicting performance criteria must be balanced. Scalarization techniques transform vector-valued reward signals into scalar quantities suitable for standard RL algorithms, with linear scalarization representing the simplest approach through weighted combination of individual objectives. More sophisticated methods, including lexicographic ordering and Pareto-based approaches [50, 51], provide alternative frameworks for handling objective trade-offs, each with distinct implications for the resulting policies and computational requirements [52, 53]. The choice of scalarization scheme profoundly influences the learned behavior, determining not only the final performance across different metrics but also the exploration dynamics during training and the interpretability of the resulting policies.

Curriculum learning strategies address the sample complexity challenge by structuring the learning process as a progression through increasingly difficult tasks, mirroring pedagogical principles from human education [54]. Rather than immediately confronting the agent with the full complexity of the target domain, curriculum learning begins with simplified scenarios that establish fundamental skills, gradually introducing additional challenges as competence develops. This approach can dramatically accelerate convergence, particularly in domains with sparse rewards or complex hierarchical structure, by ensuring that the agent maintains a reasonable success rate throughout training. The curriculum design itself becomes a critical consideration, requiring careful sequencing of

task difficulty, appropriate metrics for assessing readiness for progression, and mechanisms for preventing catastrophic forgetting of earlier skills.

Evaluation of reinforcement learning systems demands careful consideration of multiple performance dimensions beyond simple reward accumulation. Sample efficiency, measuring the amount of environment interaction required to achieve a given performance level, assumes particular importance in domains where data collection is expensive or time-consuming. Generalization capability, assessed through performance on previously unseen scenarios or parameter configurations, determines the practical applicability of learned policies beyond the training distribution. Robustness to distribution shift, quantified through systematic perturbation of environment dynamics or observation noise, characterizes the reliability of deployed systems under real-world variability. These evaluation criteria, together with computational requirements and interpretability considerations, inform the selection and refinement of algorithms for specific application domains.

3.4 User Segmentation through Clustering and Cohesion Metrics

Unsupervised clustering constitutes a fundamental pillar in the analysis of heterogeneous cyber-physical systems, where the inherent diversity of users, devices, and operational contexts precludes simple, uniform treatment. The challenge of partitioning complex, multi-dimensional populations into meaningful subgroups extends far beyond the mechanical application of clustering algorithms, encompassing critical decisions about feature representation, distance metrics, cluster validation, and the operationalization of results in distributed environments. The apparent simplicity of clustering (grouping similar entities together) belies the profound technical challenges that arise when these techniques must operate on mixed-type data, adapt to temporal dynamics, provide verifiable results in trustless settings, and yield actionable insights for real-time control systems.

The K-*means algorithm*, despite its conceptual elegance and decades-long history, remains remarkably relevant for modern cyber-physical applications due to its computational efficiency and amenability to distributed implementation [55, 56]. However, deploying K-means effectively in urban systems requires addressing multiple interconnected challenges that span theoretical, algorithmic, and systems-level considerations. The non-convex optimization landscape leads to sensitivity to initialization and local minimum, necessitating careful analysis of convergence properties and initialization strategies. The assumption of spherical clusters with equal variance often conflicts with the natural structure of real-world data, requiring sophisticated validation metrics that can detect and quantify violations of these assumptions. The presence of mixed continuous and categorical features in cyber-physical datasets invalidates the standard Euclidean distance, demanding alternative metrics that respect the heterogeneous nature of the feature space. Furthermore, the dynamic nature of urban systems introduces temporal considerations that static clustering cannot address, while the decentralized architecture of blockchain-based systems imposes stringent requirements on how clustering models are stored, transmitted, and verified across trust boundaries.

The introduction of the normalized mean squared distance metric (described as y_{nmsd}in in Chapter 9) addresses a specific but critical gap in cluster evaluation: the need for scale-invariant, interpretable measures of cluster cohesion that remain meaningful across different feature spaces, dataset sizes, and application domains. Traditional metrics such as inertia or sum of squared errors are heavily influenced by the absolute scale of features and the number of observations, making cross-domain comparisons problematic. The normalization strategy must account for

both the intrinsic variability of the data and the expected behavior under null models, requiring careful mathematical formulation and empirical validation.

This section provides a comprehensive treatment of these multifaceted challenges, establishing the theoretical foundations necessary for the practical deployment of clustering in the subsequent chapters.

3.4.1 K-means Algorithm and Convergence Properties

The K-means algorithm partitions a dataset of n observations into k clusters by iteratively refining cluster assignments and centroid positions to minimize the within-cluster sum of squared distances. Given a set of feature vectors $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ where $\mathbf{x}_i \in \mathbb{R}^d$, the algorithm seeks to find k cluster centroids $C = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k\}$ and corresponding cluster assignments that minimize the total inertia, defined as the sum of squared Euclidean distances between each point and its assigned centroid.

The iterative procedure alternates between two phases: the assignment step, where each observation is allocated to the cluster whose centroid is nearest according to the Euclidean metric, and the update step, where each centroid is recomputed as the mean of all observations currently assigned to that cluster. This alternating optimization guarantees monotonic decrease of the objective function, though convergence to a global minimum is not assured due to the non-convex nature of the optimization landscape. The quality of the final clustering depends critically on the initial centroid positions, motivating various initialization strategies that balance computational cost with solution quality.

Standard initialization methods include random selection from the dataset, which offers simplicity but high variance in solution quality, and the K-means++ algorithm, which selects initial centroids through a probabilistic scheme that favors spatial dispersion [57]. The latter approach provides theoretical guarantees on the expected approximation ratio relative to the optimal clustering, though at increased computational cost during initialization. In practice, multiple runs with different random seeds are often employed, selecting the solution with minimum final inertia to mitigate the impact of poor local minima.

3.4.2 Normalized Mean Squared Distance Metric

While the standard within-cluster sum of squares provides a natural optimization target, comparing cluster quality across different feature scales, dimensionalities, or dataset sizes requires normalized metrics. The normalized mean squared distance,

denoted y_{nmsd}, addresses these limitations by accounting for both the inherent variance in the feature space and the expected distances under random clustering.

For a given clustering configuration with k clusters, let S_i denote the set of observations assigned to cluster i with centroid $\mathbf{c}_i$. The normalized mean squared distance for cluster i is computed by first calculating the mean squared distance from the centroid, then normalizing by a reference scale that captures the characteristic spread of the data. This reference scale can be derived from the total variance of the dataset, the average inter-cluster distance, or domain-specific thresholds that reflect meaningful distinctions in the application context.

The aggregate cohesion metric y_{nmsd} combines the individual cluster contributions through a weighted average, where weights may reflect cluster sizes, importance factors derived from domain knowledge, or uniform weighting for balanced assessment. Values approaching zero indicate tight, well-separated clusters, while larger values suggest either poor cluster definition or inherent overlap in the feature space. This normalization enables meaningful comparisons across different clustering runs, parameter settings, or even different datasets within the same application domain.

3.4.3 Optimal Cluster Number Selection

Determining the appropriate number of clusters remains a fundamental challenge in unsupervised learning, as the true number of natural groupings is rarely known a priori. The elbow method provides a heuristic approach by plotting the total within-cluster sum of squares against the number of clusters and identifying the point where the rate of decrease sharply diminishes. This inflection point, resembling an elbow in the curve, suggests a balance between model complexity and explanatory power.

However, the elbow point is often ambiguous in real datasets, particularly when clusters exhibit varying densities or non-spherical shapes. The silhouette coefficient offers a complementary perspective by quantifying how similar each observation is to its assigned cluster compared to neighboring clusters. For each observation, the silhouette value ranges from -1 to 1, where positive values indicate appropriate assignment and negative values suggest potential misclassification. The average silhouette coefficient across all observations provides a global measure of clustering quality that peaks at the optimal number of clusters.

Additional validation indices, such as the Calinski-Harabasz [58] score and the Davies-Bouldin [59] index, provide alternative perspectives on cluster validity by considering different aspects of cluster separation and compactness. Cross-validation approaches can also be adapted to clustering by holding out portions of

the data and assessing the stability of cluster assignments or the predictive power of the learned centroids on unseen observations.

3.4.4 Mixed-Type Features and Distance Metrics

Cyber-physical systems often generate heterogeneous data combining continuous measurements, categorical attributes, and discrete counts, necessitating careful consideration of distance metrics. The standard Euclidean distance assumes all features are continuous and equally scaled, which rarely holds in practice. Feature standardization through z-score normalization or min-max scaling addresses scale disparities but does not handle categorical variables.

For mixed-type data, Gower's distance provides a unified framework by computing partial distances for each feature type and combining them through weighted averaging [60]. Continuous features contribute squared differences after appropriate scaling, binary features contribute simple matching coefficients, and categorical features contribute indicator functions for equality. The weights can be adjusted to reflect the relative importance of different feature types or to account for missing values through available-case analysis.

Alternative approaches include encoding categorical variables as binary indicators through one-hot encoding, though this can lead to high-dimensional sparse representations that complicate distance calculations and increase computational requirements. Embedding methods that map categorical values to continuous representations learned from data offer a middle ground, preserving semantic relationships while maintaining computational tractability.

3.4.5 Distributed Storage and Verification

In decentralized systems where clustering results must be shared and verified across multiple nodes, efficient representation and validation of cluster models become critical concerns. Storing complete cluster memberships for large datasets would be prohibitive, both in terms of storage requirements and verification complexity. Instead, compact representations focusing on cluster centroids and summary statistics enable efficient validation while preserving privacy.

Each cluster can be characterized by its centroid coordinates, the number of assigned observations, and dispersion metrics such as the covariance matrix or radius containing a specified percentage of members. These parameters, typically requiring only $O(kd)$ storage for k clusters in d dimensions, can be serialized into standardized formats and committed to distributed ledgers through cryptographic hashes. Subsequent classification of new observations requires only the centroids

and distance calculations, enabling efficient on-chain verification without exposing individual data points.

The validation process involves computing distances from a query point to all stored centroids and assigning membership based on the nearest neighbor rule. For probabilistic assignments, the distances can be transformed into membership probabilities through softmax normalization or Gaussian mixture model interpretations. Into a blockchain environment, smart contracts can implement these calculations directly when the dimensionality is modest, or can verify proof of correct computation when the calculations are performed off-chain, balancing computational cost with trust requirements.

3.4.6 Temporal Dynamics and Incremental Updates

Static clustering assumes that the underlying population characteristics remain constant, yet cyber-physical systems often exhibit temporal variations in user behavior, environmental conditions, or system states. Incremental clustering algorithms address this challenge by updating cluster assignments and centroids as new observations arrive, avoiding complete reclustering while adapting to distribution shifts.

Online K-means variants process observations sequentially, updating the nearest centroid through exponentially weighted moving averages that balance stability with responsiveness. The learning rate parameter controls this trade-off, with smaller values preserving historical patterns and larger values rapidly adapting to new data. Adaptive learning rates that decrease over time or increase in response to detected distribution shifts provide additional flexibility.

Change detection mechanisms monitor clustering quality metrics over time, triggering complete reclustering when the incremental updates can no longer adequately represent the current data distribution. Statistical tests comparing current observations to historical cluster profiles, tracking the fraction of observations falling outside established cluster boundaries, or monitoring the stability of cluster assignments across sliding windows provide signals for when comprehensive model updates are necessary. These mechanisms ensure that the clustering model remains representative while avoiding unnecessary computational overhead from frequent complete reclustering.

3.5 Algorithmic Decision Framework for Cyber-Physical Systems

The taxonomy establishes a systematic mapping from fundamental problem characteristics of state-space dimensionality, temporal coupling, stochasticity, and multi-agent interactions to their corresponding algorithmic families, thereby providing clear criteria for selecting among exact optimization through ILP/MIP formulations, predictive control via MPC augmented with learned dynamics models, or adaptive policies acquired through reinforcement learning paradigms. Each method occupies a distinct position in the complexity-performance spectrum: ILP/MIP provides global optimality guarantees for discrete problems with moderate dimensionality but requires complete problem specification and scales poorly with instance size; MPC+LSTM balances model-based prediction with receding-horizon optimization, excelling in systems with measurable dynamics and explicit constraints but demanding accurate system identification; DRL offers model-free adaptation to complex, partially observable environments at the cost of sample efficiency and interpretability.

Beyond these core methodologies, auxiliary techniques complement the decision layer when specific data-processing needs arise. Unsupervised clustering via K-means, for instance, segments heterogeneous user populations into homogeneous groups, enabling personalized control strategies without explicit labeling. This is a capability particularly valuable in building automation contexts where occupant preferences vary significantly but ground-truth categories are unavailable. The normalized mean squared distance metric, expressed as y_{nmsd}, provides a scale-invariant measure of cluster cohesion, facilitating cross-domain comparisons and on-chain validation through compact centroid representations. The technical details of such clustering approaches, including mixed-type distance metrics and distributed storage mechanisms, are deferred to their specific application contexts in subsequent chapters.

The toolkit operationalizes these algorithmic choices through standardized interfaces, benchmark suites, and evaluation protocols that ensure reproducibility across diverse application domains. Reference implementations employ consistent design patterns that encompass modular architectures, explicit state management, and comprehensive logging capabilities, thereby facilitating both standalone experimentation and integration with the blockchain-based trust layer. Performance metrics span multiple dimensions, from computational efficiency metrics such as runtime, memory footprint, and gas costs for on-chain components to solution quality indicators including optimality gaps, constraint violations, and convergence

rates, while robustness assessment encompasses sensitivity to initialization, generalization across problem instances, and performance degradation under adversarial inputs. This multi-criteria evaluation framework acknowledges that no single algorithm dominates across all cyber-physical contexts; instead, the appropriate choice depends on the specific trade-offs between solution quality, computational resources, real-time constraints, and verification requirements that characterize each application domain.

4 Methodological Architecture and Experimental Protocols

The systematic investigation of cyber-physical systems necessitates a rigorous methodological framework that transcends the boundaries of individual application domains while maintaining the specificity required for meaningful empirical validation. This chapter establishes the foundational infrastructure upon which the experimental contributions of this dissertation rest, delineating the computational tools, architectural patterns, and validation protocols that enable reproducible and transferable research outcomes across the diverse spectrum of applications spanning from autonomous mobility to intelligent building management.

The methodological architecture presented herein emerges from the recognition that the complexity inherent in modern cyber-physical systems demands not merely sophisticated algorithms or advanced technologies, but rather a holistic integration framework that orchestrates multiple computational paradigms within a unified conceptual structure. The development of such a framework requires careful consideration of the tensions between generality and specificity, between theoretical elegance and practical constraints, and between computational optimality and real-time feasibility.

The organization of this chapter reflects the hierarchical nature of the methodological framework itself, proceeding from high-level architectural patterns through specific technical implementations to conclude with rigorous validation protocols. Section 4.1 introduces the three-layer architectural pattern that recurs throughout the dissertation, establishing the conceptual foundation for systematic decomposition of complex cyber-physical control problems. Section 4.2 provides comprehensive documentation of the simulation and development infrastructure, encompassing traffic modeling through SUMO, building energy simulation via EnergyPlus, blockchain development using Hardhat, and machine learning implementations leveraging state-of-the-art frameworks. Section 4.3 establishes the statistical validation framework and performance metrics that ensure rigorous empirical assessment. Section 4.4 addresses implementation challenges and their mitigation strategies, while Section 4.5 synthesizes these elements into a unified experimental protocol that guides the empirical investigations presented in subsequent chapters.

4.1 Unified Methodological Framework

The systematic investigation of cyber-physical systems demands a methodological architecture that transcends individual application boundaries while preserving the specificity required for empirical validation. The framework presented herein emerges from the recognition that complex distributed systems, regardless of their specific domain, exhibit recurring patterns in their fundamental challenges: establishing data integrity in untrusted environments, making intelligent decisions under uncertainty, and bridging the gap between computational intelligence and physical actuation.

The theoretical foundations and computational primitives delineated thus far converge toward the construction of an integrated methodological scaffold that transcends isolated technical contributions. The subsequent exposition synthesizes these disparate elements into cohesive architectural patterns and operational protocols that constitute the methodological substrate upon which the specific research contributions are constructed. This synthesis transforms abstract computational concepts and isolated technical mechanisms into actionable frameworks that demonstrate consistent efficacy across the investigated cyber-physical domains, establishing not merely a collection of techniques but rather a unified methodological philosophy that guides the systematic investigation of complex distributed systems.

4.1.1 The Three-Layer Architectural Pattern

The decomposition of cyber-physical control problems into three distinct yet interconnected layers represents a fundamental design principle that emerges consistently across the investigated domains. This stratification, far from being an arbitrary organizational choice, reflects the natural separation of concerns inherent in systems that must simultaneously ensure data trustworthiness, compute optimal decisions, and execute actions in physical environments. Each layer encapsulates specific functionalities while maintaining well-defined interfaces that facilitate both vertical integration within a single system and horizontal composition across multiple domains.

Layer 1: Data Integrity and Trust Mechanisms. The foundational layer establishes the cryptographic and distributed consensus infrastructure necessary for operating in environments where data sources cannot be implicitly trusted. At its core, this layer employs one-way cryptographic hash functions to create tamper-evident fingerprints of system states and events, with the SHA-256 algorithm providing the requisite collision resistance and avalanche properties. The

construction of Merkle trees atop individual hashes enables efficient verification of data subset membership with logarithmic complexity, creating hierarchical commitment structures that scale to large datasets while maintaining constant-size root proofs.

The integration of blockchain technology within this layer transcends simple data storage, establishing temporal ordering through cryptographically-linked blocks and enabling programmable trust through smart contracts. The notarization workflow transforms ephemeral system observations into immutable records, while consensus mechanisms with dynamic thresholds adapt validation requirements based on contextual factors. The mathematical foundation for these adaptive thresholds incorporates severity metrics normalized to the interval $[0, 1]$ and temporal decay functions that prevent indefinite postponement of decisions while maintaining robustness against premature acceptance.

JSON canonicalization addresses the critical challenge of ensuring deterministic serialization across heterogeneous implementations, establishing a bijective mapping between logical data structures and their canonical representations. This standardization proves essential when multiple independent nodes must reach consensus on data equality without centralized coordination, as even semantically equivalent JSON objects might produce different hash values without proper canonicalization.

Layer 2: Intelligent Decision Making. The intermediate layer encompasses the algorithmic intelligence that transforms verified observations into optimal or near-optimal action policies. The computational approaches employed at this layer span multiple paradigms, from model-free reinforcement learning that discovers policies through environmental interaction to model-based predictive control that exploits explicit system dynamics for anticipatory optimization.

Deep Reinforcement Learning architectures, particularly those employing policy gradient methods with function approximation, enable adaptive behavior in stochastic environments where traditional control approaches fail due to dimensionality or uncertainty. The modular decomposition of these learning systems, wherein specialized agents handle tractable subproblems before policy composition, addresses the computational intractability of monolithic approaches while preserving convergence guarantees under appropriate assumptions.

Model Predictive Control formulations introduce receding-horizon optimization that naturally handles state and input constraints while providing robustness through feedback. The integration of Long Short-Term Memory networks for system identification enables these controllers to capture complex temporal dependencies and non-linear dynamics that characterize real-world cyber-physical systems. The lexicographic optimization framework employed for multi-objective scenarios

provides a principled approach to handling competing goals, solving a sequence of constrained problems that maintain higher-priority objective values while optimizing lower-priority criteria.

For discrete resource allocation problems, Integer Linear Programming provides globally optimal solutions through systematic exploration of the feasible region, with modern branch-and-cut algorithms achieving practical solve times for problems of moderate scale. The unsupervised learning components, particularly clustering algorithms for behavioral categorization, operate without labeled training data, discovering latent patterns that inform both real-time decisions and long-term system adaptation.

Layer 3: System Actuation and Validation. The uppermost layer manages the critical transition from computational decisions to physical actions, implementing the interfaces and protocols necessary for real-world deployment. This layer encompasses not merely the mechanical execution of computed policies but also the validation infrastructure that ensures actions achieve their intended effects while maintaining system safety and stability.

The actuation mechanisms employ domain-specific interfaces that abstract hardware heterogeneity while providing deterministic execution guarantees. In vehicular applications, this manifests through traffic control protocols that manipulate vehicle trajectories and signal timings, while building automation scenarios require integration with HVAC actuators and occupancy sensors. The abstraction provided at this layer enables the same decision-making components to control diverse physical systems through standardized command interfaces.

Smart contract interactions at this layer extend beyond simple state updates to encompass complex workflows involving conditional execution, event emission, and cross-contract calls. The implementation of token economics creates incentive alignment between individual participants and system-wide objectives, with reputation scores evolving according to contribution quality and consistency. These economic mechanisms, grounded in game-theoretic principles, ensure that rational participants find cooperation more profitable than defection, thereby maintaining system integrity without centralized enforcement.

The validation framework operates continuously, comparing predicted outcomes against observed reality and triggering corrective actions when deviations exceed acceptable thresholds. This closed-loop architecture ensures that the system maintains responsiveness to environmental changes and modeling errors, with performance metrics flowing back to influence future decision-making and potentially trigger model retraining or architectural adaptation.

4.1.2 Cross-Domain Applicability and Methodological Transfer

The power of the three-layer architecture lies not in its application to any single domain but in its demonstrated transferability across radically different cyber-physical contexts. This transferability emerges from careful abstraction of domain-independent computational patterns coupled with well-defined specialization points that accommodate context-specific requirements without compromising architectural integrity.

The transformation from autonomous vehicle coordination to intelligent building management exemplifies this methodological transfer. In mobility applications, cryptographic hashes identify unique intersections, consensus mechanisms validate traffic events, and reinforcement learning optimizes routing decisions. The same architectural skeleton, when applied to building systems, employs hashes for consumption profile identification, consensus for data notarization, and predictive control for HVAC optimization. The fundamental computational patterns remain invariant while their instantiation adapts to domain-specific constraints and objectives.

The abstraction process identifies computational primitives that transcend application boundaries. Cryptographic commitments, whether applied to intersection topology or energy consumption patterns, follow identical mathematical principles. Optimization algorithms modify their objective functions and constraint sets according to domain requirements while preserving fundamental solution strategies. The consensus layer adapts its thresholds and timing parameters but maintains consistent voting semantics and Byzantine fault tolerance properties.

Specialization occurs through three primary mechanisms: parameter configuration, interface adaptation, and objective redefinition. Parameter configuration adjusts numerical constants such as consensus thresholds, learning rates, and prediction horizons to match domain-specific dynamics. Interface adaptation maps abstract commands to concrete actuators, translating high-level decisions into domain-appropriate actions. Objective redefinition reformulates optimization goals while preserving the mathematical structure of the solution approach, enabling algorithm reuse across diverse applications.

The systematic documentation of successful transfers and failed attempts establishes empirical boundaries for methodological applicability. Not all techniques transfer with equal success; the effectiveness of specific approaches depends on structural similarities between source and target domains. Temporal dynamics, state space characteristics, and constraint patterns influence transfer success rates, with closely related domains exhibiting higher transfer efficiency than disparate applications. This accumulated knowledge guides future transfer attempts, sug-

gesting appropriate adaptation strategies and identifying potential pitfalls before implementation.

The architectural pattern thus provides both a descriptive framework for understanding existing cyber-physical systems and a prescriptive methodology for developing new applications. By maintaining consistent structure while permitting domain-specific customization, the framework achieves an optimal balance between standardization and flexibility, establishing a foundation for systematic advancement of cyber-physical system engineering across diverse application domains.

4.2 Simulation and Development Toolchain

The computational infrastructure underpinning the research presented in this dissertation encompasses a comprehensive suite of simulation environments and development frameworks, each meticulously selected to address the unique requirements of cyber-physical systems modeling across multiple application domains. The integration of these heterogeneous platforms necessitates a sophisticated orchestration methodology that preserves the fidelity of domain-specific dynamics while enabling cross-layer communication and control architectures.

4.2.1 Traffic and Urban Mobility Simulation Environment

The selection of simulation infrastructure for urban mobility studies necessitates careful consideration of the trade-off between microscopic fidelity and computational tractability. The methodology adopted employs the Simulation of Urban Mobility (SUMO) platform, chosen after systematic evaluation against alternative frameworks including VISSIM, MATSim, and Paramics [61]. The decisive factors favoring SUMO encompass its open-source architecture enabling deep customization, the availability of programmatic runtime control through the Traffic Control Interface (TraCI), and demonstrated scalability to city-scale networks exceeding 10^5 edges without prohibitive computational overhead [62].

The methodological requirement for real-time intervention capabilities during simulation execution eliminates purely analytical or mesoscopic approaches that sacrifice individual vehicle controllability for computational efficiency. The TraCI protocol provides synchronous state observation and control injection at configurable time steps, typically set at 1Hz for strategic planning applications or up to 10Hz for safety-critical scenarios. This temporal granularity proves sufficient for validating learning-based controllers while maintaining deterministic reproducibility essential for algorithm comparison.

Geographic representativeness is achieved through OpenStreetMap integration, which provides topologically consistent road networks while preserving essential traffic management infrastructure. The automated import pipeline maintains lane-level detail and intersection geometry, though manual validation remains necessary for complex junctions where heuristic inference may produce suboptimal lane connectivity. The selection of test scenarios spanning from grid-like American cities to organic European medieval centers ensures that developed methodologies generalize beyond specific urban morphologies, with particular attention to edge cases such as the "Swindon Magic Roundabout" that challenge conventional intersection modeling paradigms.

4.2.2 Building Energy and Thermodynamic Simulation

EnergyPlus represents the state-of-the-art in whole-building energy simulation, implementing fundamental heat transfer equations and thermodynamic principles to model the complex interactions between building envelope systems, HVAC equipment, and environmental conditions. The simulation engine employs a modular architecture wherein building components are represented as interconnected objects that exchange thermal energy through conduction, convection, and radiation mechanisms, with the heat balance method operating at user-defined time steps to solve the coupled differential equations governing zone air temperature evolution, surface heat fluxes, and moisture transport while simultaneously accounting for internal heat gains from occupants, lighting, and equipment [63].

The mathematical foundation of EnergyPlus rests upon the zone heat balance equation, which captures the fundamental thermodynamic principle that the rate of change of internal energy equals the sum of all heat transfer mechanisms:

$$C_z \frac{dT_z}{dt} = \sum_i Q_{surf,i} + Q_{inf} + Q_{vent} + Q_{int} + Q_{sys} \tag{4.1}$$

where C_z represents the zone thermal capacitance, T_z denotes the zone air temperature, $Q_{surf,i}$ captures convective heat transfer from surface i, while Q_{inf} and Q_{vent} account for infiltration and ventilation loads respectively, Q_{int} represents internal gains, and Q_{sys} denotes the HVAC system heat addition or removal rate [64]. This formulation enables the accurate prediction of transient thermal responses to time-varying boundary conditions and control inputs, providing the essential state evolution model upon which predictive control algorithms can operate.

The EnergyPlus Weather (EPW) file format encapsulates comprehensive meteorological data through standardized fields including dry-bulb temperature, Direct Normal Irradiance (DNI), Diffuse Horizontal Irradiance (DHI), Global Horizontal Irradiance (GHI), wind speed and direction, atmospheric pressure, and sky cover observations, with wet-bulb temperature notably absent from the standard field definitions but derivable through auxiliary processing when humidity data permits. These datasets, typically derived from ground-based weather stations or reanalysis products, undergo quality assurance procedures through the Weather Converter utility to identify and correct anomalous values while maintaining statistical consistency with long-term climate patterns, ensuring that interpolation schemes for sub-hourly time steps preserve the integrity of the original hourly observations. The temporal resolution of EPW files, standardized at hourly intervals, provides sufficient granularity for annual energy simulations while the simulation engine's internal interpolation algorithms generate smooth

transitions between discrete meteorological observations, thereby preventing numerical instabilities in the heat balance calculations.

The canonical five-zone building model serves as a representative testbed for HVAC control strategy development, incorporating architectural features commonly encountered in commercial and institutional facilities while maintaining computational tractability for iterative optimization procedures. This single-story rectangular structure, measuring 30.5×15.2 meters with a total conditioned floor area of 463.6 square meters, comprises four perimeter zones and one interior zone, each equipped with independent thermostat control and configured to reflect the asymmetric thermal loads arising from solar exposure variations throughout the diurnal cycle. The construction assembly specifications incorporate multi-layer wall compositions with explicitly defined thermal resistance values conforming to ASHRAE 90.1 standards, double-pane windows characterized by solar heat gain coefficients that account for both transmitted and absorbed radiation components [65], and a roof structure integrating insulation layers whose thermal properties reflect contemporary energy code requirements while the HVAC system configuration, consisting of a centralized air handling unit with variable air volume distribution terminals, an electric chiller providing chilled water for cooling coils, and zone-level reheat capabilities implemented through hot water or electric resistance elements, represents the prevalent mechanical system architecture deployed in modern commercial buildings.

The integration pathway between EnergyPlus and external Model Predictive Control algorithms leverages either the Building Controls Virtual Test Bed (BCVTB) middleware for legacy implementations or direct Python bindings through the EnergyPlus Runtime Language (ERL) interface for contemporary deployments, with both approaches facilitating bidirectional data exchange at each simulation time step. This co-simulation framework orchestrates the exchange of sensor measurements including zone temperatures, humidity ratios, and occupancy states from the building model to the control algorithm, while actuator commands comprising setpoint trajectories, equipment staging decisions, and damper positions flow from the optimizer to the simulation engine, with synchronization mechanisms ensuring that thermodynamic state progression occurs only after control calculations complete. The temporal coupling between the plant model and the control algorithm maintains causality constraints inherent in predictive control formulations, enabling the MPC optimizer to exploit forecasted disturbances and anticipated occupancy patterns while respecting the fundamental thermodynamic constraints governing building thermal dynamics, ultimately facilitating the evaluation of advanced control strategies that minimize energy consumption subject to comfort bounds expressed as time-varying constraints on zone temperature and

humidity trajectories.

4.2.3 Blockchain Development Infrastructure

The implementation of decentralized trust mechanisms and smart contract architectures necessitates a comprehensive development infrastructure capable of supporting both rapid prototyping and production-grade deployment. The Hardhat framework emerges as the cornerstone of our blockchain development methodology, providing an integrated environment for smart contract compilation, testing, and deployment. This framework facilitates the creation of sophisticated testing suites through its built-in network emulation capabilities, enabling the simulation of complex multi-agent interactions without incurring actual transaction costs. The framework's architecture supports deterministic testing through controlled block mining and timestamp manipulation, essential features for validating time-dependent consensus mechanisms and dynamic threshold adjustments.

Central to the blockchain infrastructure is the Ethereum Virtual Machine (EVM) emulation environment, which provides a sandboxed execution context for smart contract development. The EVM's stack-based architecture and deterministic execution model ensure that contract behavior remains consistent across different deployment targets, from local development networks to public mainnets. This compatibility layer proves particularly valuable when considering the portability of developed solutions across multiple blockchain platforms, as the EVM specification has become a de facto standard adopted by numerous blockchain networks beyond Ethereum itself.

The development of modular smart contracts in Solidity requires careful consideration of established programming patterns that balance functionality with gas efficiency. The adopted architectural approach emphasizes contract composability through well-defined interfaces, enabling the separation of concerns between data storage, business logic, and access control mechanisms. This modular design philosophy manifests through the implementation of upgradeable proxy patterns, which allow for contract logic updates while preserving state continuity, and the factory pattern, which enables the dynamic instantiation of contract instances with standardized configurations. Furthermore, the integration of library contracts for common functionalities reduces code duplication and deployment costs while maintaining security through bytecode verification.

The selection of Avalanche subnets as a production deployment target reflects a strategic consideration of scalability, cost-effectiveness, and customizability requirements. Avalanche's subnet architecture enables the creation of application-specific blockchains that inherit the security properties of the primary network while

allowing for customized virtual machines and fee structures. This architectural flexibility proves essential when optimizing for high-throughput scenarios where traditional public blockchains might impose prohibitive transaction costs or latency constraints. The subnet's ability to define custom gas economics and validator requirements enables fine-tuning of the network parameters to match specific application demands while maintaining the decentralized trust guarantees inherent to blockchain systems.

Gas optimization strategies permeate every aspect of the smart contract development process, from algorithmic design choices to low-level implementation details. The analysis of gas consumption patterns reveals that storage operations constitute the primary cost driver, necessitating careful data structure selection and the strategic use of memory versus storage variables. Advanced optimization techniques, such as packed struct encoding and the utilization of events for data logging rather than storage, significantly reduce operational costs without compromising functionality. The implementation of gas-efficient algorithms for consensus mechanisms, particularly those involving iterative voting processes, requires careful consideration of loop bounds and early termination conditions to prevent excessive gas consumption during peak network congestion periods.

4.2.4 Machine Learning and Optimization Stack

The computational infrastructure for intelligent decision-making components reflects a deliberate selection process prioritizing algorithmic flexibility, distributed scalability, and production stability over cutting-edge experimental frameworks. The reinforcement learning infrastructure centers on Ray RLlib, selected for its native support of distributed training and standardized algorithm implementations that facilitate reproducible benchmarking [66]. Alternative frameworks such as Stable Baselines3 and ACME were evaluated but ultimately rejected due to limited multi-agent support and insufficient production hardening, respectively [67].

For time-series modeling and system identification tasks, the TensorFlow ecosystem provides the necessary balance between low-level control for custom layer implementations and high-level abstractions for standard architectures. The critical requirement for exporting trained models to edge devices with limited computational resources eliminates frameworks lacking comprehensive deployment toolchains. The integration with TensorFlow Lite enables model quantization and hardware acceleration on embedded platforms, essential for real-time control applications where cloud-based inference introduces unacceptable latency.

Discrete optimization problems leverage commercial-grade solvers, specifically CPLEX for integer programming and Gurobi as a fallback option, recognizing that

open-source alternatives such as CBC and GLPK exhibit performance degradation exceeding an order of magnitude on realistically-sized problems [68]. The investment in commercial solver licenses is justified by the guaranteed optimality and solution certificates they provide, critical for safety-critical applications where heuristic solutions are unacceptable. For continuous nonlinear optimization where global optimality cannot be guaranteed, the Sequential Least Squares Programming implementation in SciPy offers robust convergence properties with minimal dependencies, though specialized solvers such as IPOPT are employed when problem structure permits exploitation of sparsity patterns.

The selection criteria explicitly prioritize mature, well-documented tools over experimental frameworks, acknowledging that methodological contributions should remain independent of implementation artifacts. This conservative approach ensures that reported improvements reflect genuine algorithmic advances rather than benefiting from implementation-specific optimizations that may not generalize across platforms.

4.2.5 Edge Computing and Distributed Test Infrastructure

The validation of distributed cyber-physical architectures necessitates a heterogeneous computational infrastructure that transcends the limitations of purely simulated environments, establishing a continuum from resource-constrained edge devices to high-performance computing clusters that faithfully reproduces the computational asymmetries inherent in production deployments. The methodological imperative for hardware-in-the-loop validation emerges from the recognition that algorithmic performance metrics obtained through idealized simulations frequently exhibit significant degradation when confronted with the realities of network latency, memory constraints, and processor heterogeneity that characterize distributed cyber-physical systems.

The foundational tier of the test infrastructure comprises a cluster of Raspberry Pi 4 Model B units, depicted in Fig. 4.1, each equipped with quad-core ARM Cortex-A72 processors and 8GB LPDDR4 memory, interconnected through a dedicated gigabit Ethernet switch to isolate experimental traffic from institutional network fluctuations. To facilitate mounting and repeatable cable routing, we also employed custom 3D-printed supports. This configuration deliberately mirrors the computational capabilities of automotive electronic control units and smart building controllers, where ARM-based architectures dominate due to their favorable performance-per-watt characteristics. The deployment of containerized applications through *Docker Swarm orchestration* enables rapid reconfiguration between experimental scenarios while maintaining reproducible execution environments,

with resource constraints artificially imposed through cgroups to emulate devices with even more limited capabilities, thereby stress-testing algorithmic robustness under severe computational limitations [69].

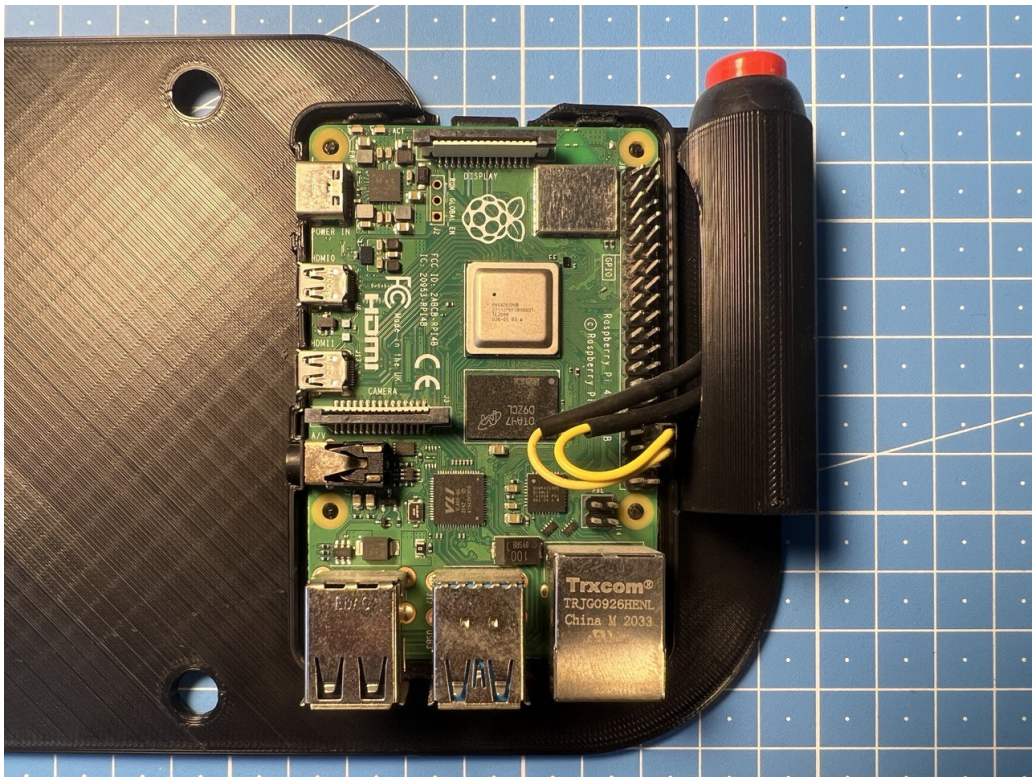


Figure 4.1: *Experimental hardware setup with a Raspberry Pi 4 Model B mounted on a custom 3-D-printed bracket*

The intermediate computational tier leverages Intel Next Unit of Computing (NUC) platforms, specifically the configuration featuring 11th generation Core i7 processors with integrated Iris Xe graphics, representing the class of edge servers increasingly deployed in intelligent transportation infrastructure and building automation systems. These platforms bridge the gap between embedded devices and cloud resources, executing computationally intensive tasks such as LSTM inference for thermodynamic prediction and local optimization routines that would overwhelm resource-constrained endpoints while maintaining sub-second response latencies critical for real-time control applications. The heterogeneous instruction set architectures between ARM and x86 platforms necessitate careful attention to numerical stability and floating-point consistency, particularly when distributed

consensus algorithms depend on deterministic computation across heterogeneous nodes.

The elasticity requirements for large-scale experiments, particularly those involving multi-agent reinforcement learning with thousands of concurrent episodes, necessitate dynamic provisioning of cloud-based resources through infrastructure-as-a-service platforms. Virtual machine instances ranging from general-purpose configurations for parameter sweep studies to GPU-accelerated instances for neural network training are orchestrated through Terraform scripts that ensure reproducible infrastructure deployment while maintaining cost optimization through spot instance utilization when experimental deadlines permit interruption tolerance. The geographic distribution of compute resources across multiple availability zones enables the evaluation of consensus algorithms under realistic wide-area network conditions, where packet loss rates and latency distributions reflect the challenges of coordinating cyber-physical systems across metropolitan or regional scales.

For computationally intensive optimization workloads that exceed the capabilities of virtualized infrastructure, dedicated bare-metal servers equipped with dual Intel Xeon Platinum 8280 processors providing 56 physical cores and 768GB of ECC memory enable the solution of large-scale integer programming problems and exhaustive hyperparameter searches that would require prohibitive execution times on commodity hardware. These high-performance computing resources, accessed through institutional clusters or commercial providers such as IBM Cloud Bare Metal Servers, prove essential when validating scalability claims through systematic weak and strong scaling studies that quantify the relationship between problem size and computational requirements.

The network topology interconnecting these heterogeneous computational resources deliberately introduces controlled variability through software-defined networking configurations that emulate the communication characteristics of deployed cyber-physical systems, including variable bandwidth constraints representative of cellular backhaul links, asymmetric routing paths that mirror the reality of internet transit arrangements, and artificial packet loss injection to stress-test the robustness of distributed algorithms. The implementation of network emulation through tc (traffic control) and netem (network emulation) utilities enables fine-grained control over latency distributions, including the long-tail characteristics observed in congested urban wireless networks, while maintaining experimental reproducibility through deterministic pseudo-random number generation.

The orchestration of experiments across this heterogeneous infrastructure leverages Ansible playbooks for configuration management and Prometheus for performance monitoring, establishing a comprehensive observability framework

that captures system-level metrics including CPU utilization, memory pressure, network throughput, and application-specific key performance indicators. The aggregation of telemetry data through Grafana dashboards enables real-time visualization of experimental progress while time-series databases preserve detailed performance traces for post-hoc analysis, facilitating the identification of performance bottlenecks and the validation of theoretical complexity bounds through empirical measurement.

This multi-tiered infrastructure strategy acknowledges that no single computational platform adequately represents the diversity of deployment environments encountered in cyber-physical systems, necessitating a portfolio approach that validates algorithmic contributions across the entire spectrum from severely resource-constrained edge devices to effectively unlimited cloud resources.

4.3 Evaluation Protocol and Metrics

The rigorous assessment of cyber-physical systems across heterogeneous domains necessitates a comprehensive evaluation framework that transcends traditional single-metric approaches. This section establishes the theoretical foundations for performance evaluation, baseline comparison, and statistical validation that underpin the experimental methodology adopted throughout this dissertation. The framework addresses the inherent complexity of evaluating systems that operate simultaneously in physical, computational, and distributed ledger domains, each characterized by distinct performance indicators and validation requirements.

4.3.1 Domain-Specific Performance Metrics

The quantitative assessment of cyber-physical systems operating across heterogeneous domains necessitates the formulation of a unified metric framework that preserves domain-specific semantics while enabling meaningful cross-domain comparisons. This framework rests upon the mathematical foundation of normalized metric spaces, wherein domain-specific measurements undergo systematic transformation to enable commensurable evaluation across disparate operational contexts.

The composite performance function for any given domain d assumes the general form $\mathcal{P}_d = \sum_{i=1}^{n} \omega_i \cdot \hat{m}_i$, where $\hat{m}_i \in [0, 1]$ represents the normalized value of metric m_i, and the weight vector $\boldsymbol{\omega} = [\omega_1, \ldots, \omega_n]^T$ satisfies the convexity constraints $\sum_{i=1}^{n} \omega_i = 1$ and $\omega_i \geq 0$ for all i. The normalization function $\hat{m}_i = \phi_i(m_i)$ maps raw measurements to the unit interval through domain-specific transformations that preserve ordering relationships while accounting for the inherent scale and distribution characteristics of each metric [70].

In autonomous mobility applications, the metric space encompasses geometric path efficiency, kinematic smoothness, and safety-oriented indicators. The path efficiency metric incorporates not merely Euclidean distance but also the energetic cost associated with acceleration and deceleration cycles, formalized through the integral $\mathcal{E}_{\text{path}} = \int_0^T \left(\alpha |a(t)|^2 + \beta v(t)^2 \right) dt$, where $a(t)$ and $v(t)$ denote instantaneous acceleration and velocity respectively, with weighting parameters α and β calibrated to vehicle-specific dynamics. The turn severity metric quantifies directional changes through a composite function accounting for both angular displacement and the differential risk associated with crossing opposing traffic flows, particularly relevant in jurisdictions where left turns (or their mirror equivalent) introduce asymmetric hazards.

Building energy management systems require fundamentally different evaluation criteria that capture the inherent tension between energy efficiency and occupant satisfaction. The performance assessment employs a bi-objective formulation wherein energy consumption minimization operates under time-varying comfort constraints expressed as $C(t) = \exp\left(-\lambda \cdot |T_z(t) - T_{\text{pref}}(t)|^2\right)$, where $T_z(t)$ represents the zone temperature, $T_{\text{pref}}(t)$ denotes the occupant preference profile, and λ controls the sensitivity to deviations. The introduction of consumption class membership functions $\mu_c(x)$ enables the quantification of behavioral transitions, where x represents the feature vector characterizing consumption patterns and $c \in \{1, \ldots, K\}$ indexes predefined consumption archetypes.

Distributed ledger systems introduce metrics that capture both computational efficiency and trust dynamics. The consensus achievement time τ_c must be evaluated not in isolation but relative to the information criticality factor $\kappa \in [0, 1]$, yielding the urgency-adjusted latency metric $\mathcal{L}_{\text{adj}} = \tau_c \cdot (1+\kappa)^{-1}$. Gas consumption in smart contract execution requires decomposition into deployment overhead G_{deploy} and operational costs G_{op}, with the amortized cost per transaction computed as $\bar{G} = G_{\text{deploy}}/N_{\text{expected}} + \mathbb{E}[G_{\text{op}}]$, where N_{expected} represents the anticipated transaction volume over the contract lifetime.

The establishment of cross-domain comparison capability requires the projection of domain-specific metrics onto a common evaluation manifold through dimensionality-preserving transformations. The normalized metric tensor $\mathbf{M} \in \mathbb{R}^{D \times K}$ encodes performance across D domains and K standardized criteria, with each element $M_{d,k}$ representing the projection of domain d metrics onto criterion k through learned mapping functions. The singular value decomposition of $\mathbf{M}$ reveals the principal performance dimensions that account for variance across domains, enabling the identification of methodological contributions that transcend specific application contexts. This mathematical framework ensures that performance improvements in one domain can be rigorously compared against achievements in another, despite the absence of directly comparable raw metrics.

4.3.2 Baseline Establishment and Comparative Analysis

The establishment of appropriate baselines constitutes a critical component of experimental validation, providing reference points against which proposed methodologies can be rigorously evaluated. In the context of learning-based approaches, the comparison between exhaustive and modular training paradigms reveals fundamental trade-offs between computational complexity and solution quality. Exhaustive approaches, which consider all possible state-action pairs in reinforcement learning contexts, provide theoretical optimal solutions but suffer from

exponential scaling with problem dimensionality. Modular decomposition strategies, conversely, sacrifice global optimality guarantees in exchange for tractable computation and improved generalization capabilities.

The juxtaposition of traditional control methodologies with predictive control schemes illuminates the advantages of anticipatory decision-making in dynamic environments. Traditional proportional-integral-derivative (PID) controllers, while robust and well-understood, operate purely on feedback mechanisms without explicit consideration of future states. Model Predictive Control (MPC) frameworks, by contrast, optimize control actions over finite prediction horizons, enabling proactive response to anticipated disturbances. The performance gap between these approaches becomes particularly pronounced in systems with significant time delays or complex multi-variable interactions, where the predictive capability of MPC yields substantial improvements in both transient response and steady-state error.

Architectural comparisons between centralized and decentralized systems reveal fundamental differences in scalability, fault tolerance, and coordination overhead. Centralized architectures, exemplified by traditional client-server models, offer simplified coordination and global optimization potential but introduce single points of failure and scalability bottlenecks. Decentralized approaches, particularly those leveraging blockchain consensus mechanisms, distribute both computation and decision-making authority, enhancing system resilience at the potential cost of increased communication overhead and consensus latency. The quantification of these trade-offs requires careful consideration of multiple performance dimensions, including throughput, latency, fault tolerance, and resource utilization.

Performance gap quantification necessitates the development of rigorous comparative methodologies that account for both statistical significance and practical relevance. The establishment of confidence intervals through repeated experimentation ensures that observed differences reflect genuine performance variations rather than stochastic fluctuations. Moreover, the analysis must consider the computational resources required to achieve reported performance levels, as superior results obtained through disproportionate resource allocation may lack practical applicability. The concept of Pareto efficiency provides a valuable framework for evaluating multi-objective trade-offs, identifying solutions that cannot be improved in one dimension without degradation in another [50, 71].

4.3.3 Statistical Validation Framework

The establishment of statistical rigor in cyber-physical system evaluation demands a comprehensive validation framework that addresses the unique challenges posed

by high-dimensional state spaces, temporal dependencies, and multiple hypothesis testing scenarios. This framework extends beyond conventional statistical testing to encompass power analysis for experimental design, appropriate corrections for multiple comparisons, and specialized techniques for time-series validation in non-stationary environments.

The determination of adequate sample sizes through power analysis requires careful consideration of the effect sizes meaningful within each application domain. For a desired statistical power of $1 - \beta$ (conventionally 0.8) and significance level α (typically 0.05), the minimum sample size for detecting an effect of magnitude δ follows from the non-central distribution of the test statistic. In the context of two-sample comparisons with pooled variance σ^2, the required sample size per group satisfies $n \geq 2\sigma^2(z_{\alpha/2} + z_\beta)^2/\delta^2$, where $z_{\alpha/2}$ and z_β denote the corresponding quantiles of the standard normal distribution [72]. This classical formulation requires adaptation for cyber-physical systems where observations exhibit temporal or spatial correlations, necessitating the incorporation of effective sample size adjustments through autocorrelation-based deflation factors.

Monte Carlo simulation protocols provide the computational foundation for exploring system behavior across the operational parameter space. The sampling strategy must balance computational tractability with comprehensive coverage, employing variance reduction techniques such as stratified sampling or Latin hypercube designs when exhaustive exploration proves infeasible [73, 74]. The selection of batch size requires careful tuning to ensure approximate independence between batch means while maintaining sufficient samples per batch for reliable variance estimation.

The multiple comparison problem emerges prominently when evaluating numerous configurations or testing multiple hypotheses simultaneously. While the Bonferroni correction provides strong control of the family-wise error rate through the adjusted significance threshold $\alpha' = \alpha/m$ for m comparisons, its conservative nature often proves excessively stringent for exploratory analyses. The *False Discovery Rate control* proposed by Benjamini and Hochberg offers a more nuanced approach, ordering the m p-values as $p_{(1)} \leq \ldots \leq p_{(m)}$ and rejecting hypotheses $1, \ldots, k^*$ where $k^* = \max\{k : p_{(k)} \leq k\alpha/m\}$. This procedure maintains the expected proportion of false discoveries below α while preserving greater statistical power for detecting true effects [75, 76].

Time-series validation in cyber-physical systems confronts the challenge of preserving temporal dependencies while ensuring unbiased performance estimation. The walk-forward validation scheme maintains temporal causality by training on data up to time t and evaluating on the subsequent period $[t+1, t+h]$, progressively advancing the training window. This approach naturally accommodates concept

drift and non-stationary dynamics inherent in real-world deployments. The block bootstrap methodology provides an alternative framework for uncertainty quantification, resampling contiguous blocks of length l to preserve local temporal structure while enabling the construction of confidence intervals through percentile methods.

The selection of appropriate test statistics requires alignment with the distributional characteristics of the observed data. When normality assumptions prove untenable, as frequently occurs with heavy-tailed distributions in network latency measurements or multimodal distributions in user behavior patterns, nonparametric alternatives provide robust inference. The Wilcoxon signed-rank test for paired comparisons [77] and the Mann-Whitney U test for independent samples [78] offer distribution-free alternatives that maintain validity under weaker assumptions. For multivariate comparisons, permutation tests provide exact p-values through exhaustive or Monte Carlo enumeration of the permutation distribution, requiring only the exchangeability assumption under the null hypothesis.

The quantification of uncertainty in complex cyber-physical systems extends beyond point estimates to encompass prediction intervals and confidence regions that acknowledge multiple sources of variability. The bootstrap percentile method constructs confidence intervals by resampling the empirical distribution, while parametric approaches based on asymptotic theory provide computationally efficient alternatives when distributional assumptions can be justified. The propagation of uncertainty through composed systems requires careful attention to dependency structures, employing techniques from sensitivity analysis to quantify how input uncertainties manifest in output variability.

4.4 Implementation Challenges and Solutions

4.4.1 Scalability and Computational Complexity

The implementation of cyber-physical systems that integrate autonomous decision-making, distributed consensus mechanisms, and real-time optimization presents fundamental scalability challenges that must be addressed through systematic architectural design. The computational complexity inherent in exhaustive optimization approaches becomes prohibitive when dealing with large-scale urban environments or extensive sensor networks, necessitating the adoption of modular decomposition strategies that partition the problem space into manageable subproblems while preserving global optimality guarantees.

The *modular decomposition approach* enables the transformation of monolithic optimization problems into sets of smaller, tractable subproblems that can be solved independently or with limited interdependencies. This architectural pattern reduces the computational burden from exponential to polynomial complexity in many practical scenarios, though careful attention must be paid to the interfaces between modules to ensure that the decomposition does not introduce suboptimality or coordination failures. The theoretical foundation for such decomposition rests on the principle of separability in the objective function and the identification of weakly coupled constraints that can be relaxed without significantly affecting the solution quality.

Parallel training architectures become essential when deploying machine learning models across distributed systems, particularly in reinforcement learning scenarios where multiple agents must explore vast state-action spaces. The parallelization strategy must balance the trade-off between exploration efficiency and computational resource utilization, ensuring that parallel agents maintain sufficient diversity in their exploration patterns while avoiding redundant computation. The synchronization overhead in parallel architectures introduces additional complexity that must be carefully managed through asynchronous update schemes or periodic synchronization barriers that minimize communication latency while maintaining convergence properties.

The deployment of blockchain-based consensus mechanisms introduces unique scalability considerations related to transaction throughput, block propagation latency, and state storage requirements. The adoption of subnet architectures or layer-two scaling solutions provides a pathway to achieving the necessary transaction rates for real-time applications while maintaining the security guarantees of the underlying blockchain. However, this architectural choice introduces additional complexity in terms of cross-subnet communication and state consistency,

requiring careful protocol design to ensure that the benefits of horizontal scaling are not negated by increased coordination overhead.

Computational resource optimization in distributed cyber-physical systems requires a holistic approach that considers not only the raw computational requirements but also the communication bandwidth, storage capacity, and energy constraints of the participating nodes. The heterogeneity of computational resources across the network necessitates adaptive algorithms that can dynamically adjust their computational intensity based on available resources, potentially degrading gracefully when resources are constrained while maintaining essential safety and correctness properties.

The fundamental trade-off between accuracy and computational efficiency manifests differently across various system components. In optimization contexts, this trade-off often involves choosing between exact and approximate solution methods, where heuristic approaches can provide near-optimal solutions with significantly reduced computational requirements. In machine learning applications, model complexity must be balanced against inference latency, particularly in real-time control scenarios where decision delays can have safety implications. The selection of appropriate trade-off points requires careful empirical evaluation and often domain-specific knowledge about the relative importance of solution quality versus computational efficiency.

4.4.2 Data Quality and System Integrity

The preservation of system integrity in distributed cyber-physical architectures necessitates sophisticated mechanisms for distinguishing between legitimate observations, transient errors, and deliberate misinformation. The fundamental challenge arises from the inherent uncertainty in sensor measurements compounded by the possibility of both unintentional failures and adversarial manipulation, requiring a multi-layered approach that balances robustness against false positives with protection against coordinated attacks.

The theoretical foundation for data quality assurance in distributed systems draws from the intersection of fault-tolerant computing and game-theoretic reputation models. In environments where multiple autonomous agents contribute observations, the distinction between Byzantine failures and benign errors becomes critical for system design. Classical Byzantine fault tolerance results establish that achieving consensus in the presence of f Byzantine nodes requires at least $3f + 1$ total participants [79], though these bounds assume worst-case adversarial behavior that may be overly conservative for practical cyber-physical systems where physical constraints and redundant sensing modalities provide additional

validation mechanisms.

Reputation-based filtering emerges as a probabilistic framework for assessing data source reliability through the lens of repeated game theory. Let each agent v maintain a reputation score $\text{rep}(v) \in \mathbb{R}_{\geq 0}$, initialized at a baseline value and evolving according to the observed accuracy of contributed data. The evolution of reputation follows a Markovian process where future trustworthiness assessments depend on the accumulated history of past interactions, creating evolutionary stable strategies that incentivize truthful reporting while accommodating occasional errors inherent in physical measurement systems [80].

A critical design consideration in reputation systems involves the calibration of penalty mechanisms to distinguish between systematic failures and transient errors. The double-hit penalty policy exemplifies this principle through its two-stage warning system: an initial detection of erroneous data triggers a binary flag $f(v) \in \{0, 1\}$ without immediate reputation degradation, with actual penalties applied only upon consecutive failures [81–83]. This approach acknowledges the stochastic nature of sensor systems operating in uncertain environments, where electromagnetic interference, temporary occlusions, or communication disruptions may cause isolated errors without indicating systematic unreliability. The mathematical formulation ensures that the probability of false penalization decreases exponentially with the tolerance threshold, while the detection probability for persistent malicious behavior remains high.

Hash-based duplicate detection provides an efficient mechanism for identifying redundant submissions in distributed sensor networks, leveraging the collision-resistance properties of cryptographic hash functions. The challenge lies in establishing canonical representations of sensor observations that remain invariant under semantically insignificant transformations while detecting meaningful variations. JSON Canonicalization Scheme (JCS) addresses this requirement by defining deterministic serialization rules that ensure identical logical content produces identical hash values regardless of the originating system's internal representation, as discussed in Section 2.1.2. The resulting hash values serve as unique identifiers for detecting duplicate submissions across temporal and spatial dimensions, preventing both inadvertent redundancy and deliberate replay attacks.

The propagation of uncertainty through multi-stage processing pipelines requires careful tracking of confidence bounds to maintain meaningful quality assessments. When sensor measurements with known error distributions undergo transformations through state estimation, data fusion, and decision algorithms, the resulting uncertainty must be quantified to inform downstream consumers about data reliability. Bayesian frameworks provide principled approaches for uncertainty propagation, though computational constraints often necessitate approximations

through techniques such as unscented transforms or Monte Carlo sampling.

The integration of physical and cyber validation mechanisms creates defense-in-depth against sophisticated attacks that might compromise individual detection methods. Physical invariants, such as conservation laws or kinematic constraints, provide domain-specific validation criteria that complement cryptographic integrity checks. For instance, vehicle trajectories must satisfy acceleration and turning radius limits imposed by physical dynamics, enabling the detection of spoofed location reports that violate these constraints. Similarly, thermodynamic systems must respect energy conservation principles, allowing the identification of manipulated sensor readings that would imply physically impossible heat transfers.

The temporal correlation of errors provides additional discriminatory power between random failures and coordinated attacks. Legitimate sensor errors typically exhibit statistical independence across time and space, while adversarial manipulations often display correlated patterns reflecting the attacker's objectives. Time-series analysis techniques, particularly those based on autoregressive models or hidden Markov processes, enable the detection of anomalous correlation structures that deviate from expected error distributions. The establishment of baseline error models through historical data analysis provides reference distributions against which current observations can be compared, with significant deviations triggering enhanced scrutiny or temporary exclusion from consensus processes.

The economic dimension of data quality assurance introduces mechanism design considerations wherein the cost of providing high-quality data must be balanced against the rewards for participation. The alignment of individual incentives with collective data quality objectives requires careful calibration of reward structures, penalty mechanisms, and participation thresholds. Token-based economies provide flexible frameworks for implementing such incentive structures, where the value of contributions reflects both immediate data utility and long-term reputation effects, creating sustainable equilibria that promote honest participation while remaining robust against exploitation attempts.

4.4.3 Real-Time Constraints and Responsiveness

The deployment of cyber-physical systems across diverse application domains necessitates careful consideration of temporal constraints and system responsiveness. In Model Predictive Control architectures, the selection of an appropriate prediction horizon represents a fundamental trade-off between computational tractability and control optimality. The horizon length S must be sufficiently extensive to capture the system dynamics while remaining computationally feasible

for real-time execution within the sampling period. This temporal decomposition becomes particularly critical when integrating learning-based components, where the inference latency of neural network models must be accommodated within the control loop timing constraints.

Dynamic threshold adjustment mechanisms introduce additional complexity in distributed consensus protocols. The mathematical formulation of time-varying acceptance criteria requires careful balancing between responsiveness to urgent events and robustness against spurious or malicious inputs. The incorporation of temporal decay functions, e.g., logarithmic time penalties of the form $\ln(1 + \Delta t)$, ensures that pending decisions naturally progress toward resolution while maintaining system stability. Such mechanisms must be designed to prevent both premature acceptance of unvalidated information and excessive delays in critical decision-making processes.

The establishment of immediate decision criteria serves as a crucial safety mechanism in time-critical scenarios. These deterministic rules, triggered when specific conditions are satisfied, bypass the standard consensus waiting period to enable rapid system response. The mathematical formulation of such criteria must account for the inherent trade-off between false positive and false negative rates, particularly in safety-critical applications where the cost of incorrect decisions may be asymmetric. The implementation of these immediate triggers requires careful consideration of race conditions and potential exploitation vectors, necessitating formal verification of the decision logic under all possible system states.

Latency optimization in distributed architectures demands a holistic approach encompassing network communication, computation, and storage operations. The decomposition of complex optimization problems into parallelizable subproblems enables significant latency reduction through concurrent execution. However, this parallelization introduces synchronization challenges and potential consistency issues that must be carefully managed through appropriate coordination mechanisms. The selection of communication protocols and data serialization formats directly impacts the achievable latency bounds, with binary protocols offering superior performance characteristics compared to text-based alternatives in bandwidth-constrained environments.

Time-critical system guarantees require formal analysis of worst-case execution times and probabilistic characterization of typical performance. The integration of multiple heterogeneous components, each with distinct timing characteristics, necessitates compositional verification techniques to establish end-to-end latency bounds. Statistical approaches based on extreme value theory provide probabilistic guarantees for systems where deterministic bounds are either infeasible or overly conservative. The establishment of graceful degradation mechanisms ensures

that system functionality is maintained even when ideal timing constraints cannot be satisfied, with progressive quality reduction preferred over complete system failure.

4.4.4 Blockchain Computational Overhead and Real-Time Implications

The integration of blockchain technology into cyber-physical control architectures introduces computational overhead that warrants careful analysis with respect to real-time operational requirements. The additional complexity manifests through three primary mechanisms: transaction construction and signing operations, network propagation latency, and consensus finalization delays, each contributing distinct temporal components to the overall system response time.

Transaction construction encompasses the serialization of application data into blockchain-compatible formats, the computation of cryptographic signatures using elliptic curve operations, and the preparation of transaction metadata including nonce management and gas estimation. On contemporary hardware, these operations typically require between 10 and 50 milliseconds per transaction, a latency that proves negligible for building automation applications operating on minute-scale control horizons but potentially significant for vehicular coordination scenarios demanding sub-second response times. The adoption of hardware security modules or dedicated cryptographic accelerators can reduce signing latency to sub-millisecond levels when throughput requirements justify the additional infrastructure investment.

Network propagation and consensus finalization introduce latencies fundamentally bounded by the underlying blockchain platform's block time and finality characteristics. Public Ethereum mainnet exhibits block times of approximately 12 seconds with probabilistic finality requiring multiple confirmations, rendering it unsuitable for time-critical control loops. However, the architectural flexibility enabled by EVM-compatible platforms, as discussed in Section 2.2, permits deployment on networks with substantially reduced latency profiles. Avalanche subnets achieve sub-second finality through the Snowman consensus protocol, while Layer-2 rollup solutions provide instant soft confirmations suitable for optimistic real-time operation with eventual cryptographic settlement.

The critical insight emerging from this analysis is that blockchain integration need not impose real-time constraints when architectural decisions appropriately separate time-critical control paths from trust-establishment mechanisms. In the HVAC control system presented in Chapter 9, the MPC optimization loop executes locally at minute-scale intervals without blockchain involvement, while

consumption data notarization and payment settlement occur asynchronously at billing-period boundaries where multi-second latencies prove entirely acceptable. Similarly, the vehicular data sharing framework of Chapter 6 employs blockchain for incentive distribution and reputation management rather than real-time collision avoidance, with time-critical coordination occurring through direct vehicle-to-vehicle communication channels that bypass blockchain latency entirely. This architectural pattern of decoupling real-time control from blockchain-mediated trust establishment enables the benefits of cryptographic verification and economic incentivization without compromising the temporal responsiveness essential for cyber-physical system safety and performance.

4.4.5 Interoperability and Platform Independence

The achievement of genuine platform independence in distributed cyber-physical systems requires careful abstraction of platform-specific details while maintaining operational efficiency. The Ethereum Virtual Machine (EVM) compatibility layer represents a significant advancement in blockchain interoperability, enabling smart contract portability across diverse blockchain implementations. This abstraction, however, introduces computational overhead that must be carefully characterized and optimized for production deployments. The bytecode representation provides a platform-agnostic execution model, yet the underlying consensus mechanisms and economic models may vary significantly across compatible platforms, necessitating adaptive deployment strategies.

Standardized data format adoption constitutes a cornerstone of system interoperability. The JSON has emerged as a de facto standard for structured data exchange, offering human readability and broad programming language support. However, the inherent ambiguity in JSON representation, particularly regarding numerical precision and object key ordering, necessitates additional canonicalization procedures to ensure deterministic processing. The JSON Canonicalization Scheme addresses these concerns through strict ordering rules and normalized representations, enabling reproducible cryptographic hashing of structured data. For domain-specific applications, specialized formats such as EnergyPlus Weather files provide standardized representations of environmental data, facilitating model portability across simulation platforms.

Application Programming Interface design patterns significantly influence system extensibility and third-party integration capabilities. The adoption of RESTful architectural principles provides stateless, resource-oriented interfaces that align naturally with distributed system architectures. The versioning strategy for APIs must balance backward compatibility with the need for protocol evolution,

often requiring the simultaneous support of multiple API versions during transition periods. The specification of clear interface contracts through formal schemas, e.g., OpenAPI specifications, enables automated client generation and reduces integration errors arising from interface misunderstandings.

Cross-platform deployment strategies must account for heterogeneous execution environments ranging from resource-constrained embedded systems to cloud-based computational clusters. The containerization of application components through technologies such as Docker provides environmental isolation and dependency management, yet introduces additional resource overhead that may be prohibitive in certain deployment contexts. The selection of appropriate middleware layers and communication protocols must consider the specific constraints of each target platform while maintaining functional equivalence across deployments. Network topology variations, particularly in vehicular ad-hoc networks, require adaptive routing strategies that can accommodate dynamic connectivity patterns without compromising system functionality.

Legacy system integration presents unique challenges arising from technological obsolescence and architectural mismatches. The implementation of adapter patterns and protocol translation layers enables bidirectional communication between modern distributed systems and existing infrastructure. However, the semantic gap between legacy data models and contemporary representations often requires sophisticated mapping logic that must be carefully validated to prevent information loss or misinterpretation. The gradual migration strategy, wherein legacy components are progressively replaced while maintaining system operation, demands careful orchestration of parallel systems and controlled switchover mechanisms to minimize disruption to ongoing operations.

4.5 Unified Experimental Protocol

The research methodology adopted throughout this dissertation establishes a comprehensive experimental framework that ensures consistency, reproducibility, and transferability across diverse cyber-physical system domains. This unified protocol emerges from the necessity to bridge theoretical contributions with practical implementations while maintaining scientific rigor and facilitating independent validation of results.

4.5.1 Standardized Experimental Workflow

The experimental methodology follows a structured five-phase approach, depicted in Fig. 4.2, wherein each phase builds upon the outcomes of its predecessors while maintaining sufficient modularity to accommodate domain-specific requirements. This systematic progression ensures that the transition from theoretical formulation to practical validation proceeds through well-defined stages, each characterized by specific objectives, deliverables, and validation criteria, with embedded quality checkpoints that determine progression or iteration requirements.

The initial phase of data acquisition and curation establishes the empirical foundation upon which subsequent analyses rest. This phase encompasses the collection of domain-specific datasets through simulation environments or real-world deployments, followed by systematic preprocessing operations including temporal alignment, synchronization, and quality assessment metrics. The preprocessing pipeline incorporates standardized transformations that preserve essential statistical properties while ensuring compatibility with downstream learning algorithms. Particular attention is devoted to maintaining temporal coherence in time-series data and spatial relationships in network-structured information, as these characteristics prove fundamental for cyber-physical system applications. The data quality assessment checkpoint determines whether the curated datasets satisfy minimum requirements for model development or necessitate recollection efforts.

The second phase addresses model development through architecture selection and hyperparameter optimization, wherein machine learning components undergo systematic optimization to achieve desired performance characteristics. This phase employs cross-validation strategies to prevent overfitting while ensuring generalization capabilities across diverse operational scenarios. The optimization process extends beyond simple parameter tuning to encompass comprehensive hyperparameter search through grid search or Bayesian optimization techniques, with careful documentation of the search space and selection criteria. The training methodology incorporates convergence monitoring mechanisms and regularization

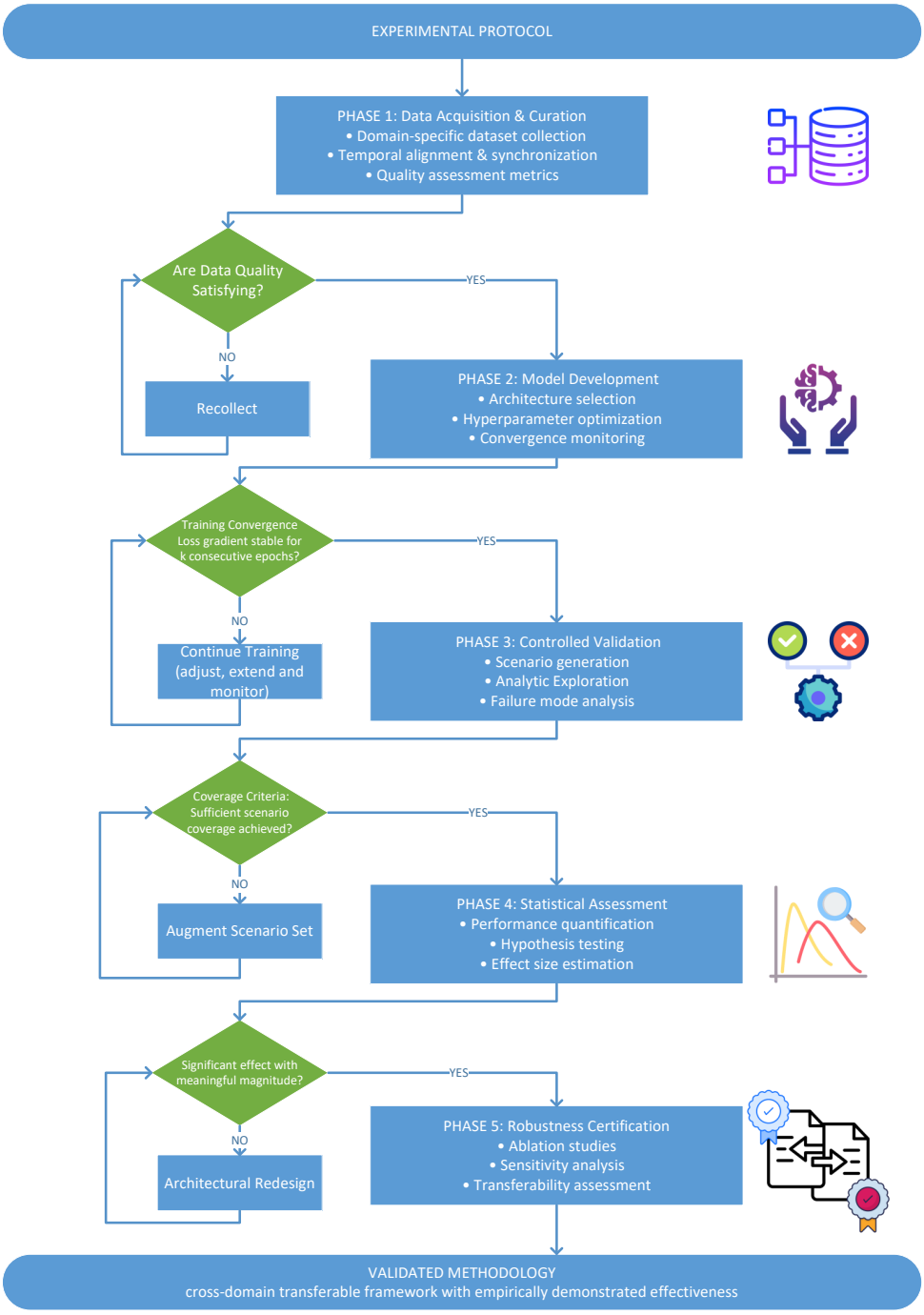


Figure 4.2: *Five-phase experimental protocol with iterative quality control checkpoints*

techniques to balance model complexity with predictive accuracy, while maintaining computational tractability for real-time deployment scenarios. Training convergence criteria establish whether the model achieves stable performance for k consecutive epochs before proceeding to validation.

Controlled validation constitutes the third phase, leveraging scenario generation and analytic exploration to assess system behavior under controlled conditions before deployment. This phase conducts comprehensive failure mode analysis through stochastic sampling methodologies (e.g., Monte Carlo techniques) to explore the operational envelope of proposed solutions, systematically varying environmental parameters, initial conditions, and disturbance patterns to characterize robustness and failure modes. The simulation framework maintains strict separation between training and validation scenarios to ensure unbiased performance assessment, while providing sufficient coverage of edge cases and adversarial conditions that might compromise system integrity. The coverage criteria checkpoint ensures sufficient scenario exploration has been achieved, potentially requiring augmentation of the scenario set when coverage remains incomplete.

The fourth phase focuses on statistical assessment through performance quantification and hypothesis testing that capture both domain-specific objectives and cross-cutting concerns such as computational efficiency and scalability. The evaluation protocol employs multiple complementary metrics including effect size estimation to provide holistic assessment of system performance, avoiding the pitfalls of single-metric optimization that might obscure important trade-offs. Statistical significance testing accompanies all performance comparisons, with appropriate corrections for multiple hypothesis testing when evaluating across numerous experimental conditions. The significance threshold determines whether results demonstrate meaningful improvements warranting progression or necessitate architectural redesign.

Robustness certification in the fifth phase systematically examines ablation studies, sensitivity analysis, and transferability assessment to establish operational guarantees. This phase employs controlled experiments that isolate specific architectural elements or algorithmic choices, quantifying their impact on overall system performance. The sensitivity analysis explores the parameter space through systematic perturbation studies, identifying critical parameters that significantly influence system behavior and establishing operational boundaries within which performance guarantees can be maintained. The transferability assessment validates cross-domain applicability, ensuring that the methodology achieves its intended generalization objectives.

The iterative refinement process that encompasses all phases acknowledges that

experimental insights frequently necessitate revisiting earlier design decisions or methodological choices, as explicitly captured by the feedback loops in the experimental protocol. This iterative approach maintains a formal feedback mechanism whereby performance deficiencies identified during evaluation trigger targeted modifications to data preprocessing, model architecture, or validation procedures. Each iteration generates documented improvements and lessons learned, contributing to a cumulative knowledge base that informs future experimental campaigns and ultimately produces a validated methodology demonstrating cross-domain transferable framework with empirically demonstrated effectiveness.

4.5.2 Reproducibility and Open Science Guidelines

The commitment to reproducible research manifests through systematic documentation practices and standardized development workflows that facilitate independent verification of experimental results. The adoption of version control systems extends beyond source code management to encompass configuration files, experimental scripts, and documentation, ensuring complete traceability of the research evolution.

Code repository organization follows established best practices with clear separation between data processing pipelines, model implementations, evaluation scripts, and visualization utilities. Each repository maintains comprehensive README documentation that guides users through installation procedures, dependency management, and execution workflows. The modular code architecture facilitates component reuse while maintaining clear interfaces between functional units, enabling researchers to adapt specific elements for their applications without requiring wholesale system reimplementation.

Parameter documentation standards ensure that all experimental configurations remain fully specified and reproducible. Configuration files employ human-readable formats such as YAML or JSON, with extensive inline comments explaining parameter semantics and acceptable value ranges. The documentation explicitly captures not only the final parameter selections but also the rationale behind these choices, including failed experiments and parameter combinations that proved ineffective, thereby preventing repetition of unsuccessful approaches.

Environment containerization through technologies such as Docker ensures consistent execution environments across different computational platforms, eliminating the notorious "works on my machine" problem that plagues computational research. Container definitions specify exact versions of all dependencies, from operating system libraries to domain-specific simulation tools, creating reproducible computational environments that persist independently of underlying infrastructure

changes. The containerization strategy extends to include data processing pipelines and evaluation frameworks, ensuring that the entire experimental workflow remains portable and reproducible.

Dataset versioning and availability protocols ensure that experimental data remains accessible for validation and extension of research findings. The versioning system tracks not only raw data but also preprocessed datasets and intermediate results, maintaining clear provenance chains that link final outcomes to original data sources. When data privacy or proprietary concerns preclude full data release, the documentation provides synthetic data generators or anonymization procedures that preserve essential statistical properties while protecting sensitive information.

Computational resource requirements receive explicit documentation to enable accurate reproduction of experimental timelines and facilitate resource planning for validation studies. This documentation encompasses hardware specifications including processor architectures, memory configurations, and accelerator availability, along with execution time measurements for critical experimental phases. The resource documentation distinguishes between minimum requirements for functional execution and recommended configurations for practical experimental timelines, acknowledging that computational constraints often influence methodological choices in resource-limited settings.

4.5.3 Transferability Assessment Protocol

The systematic evaluation of methodological transferability across domains establishes the broader applicability of proposed approaches beyond their initial application contexts. This assessment protocol recognizes that successful transfer requires not merely technical compatibility but also careful consideration of domain-specific constraints, objectives, and evaluation criteria.

Domain adaptation methodologies address the challenge of applying models trained in one context to related but distinct application scenarios. The adaptation process encompasses both parameter-level adjustments, where existing models undergo fine-tuning for new domains, and architectural modifications that accommodate domain-specific requirements while preserving core algorithmic innovations. The protocol systematically documents successful and unsuccessful transfer attempts, identifying critical factors that facilitate or impede cross-domain application.

Parameter transfer strategies explore the extent to which optimized parameters from one domain provide effective initialization points for related applications. This analysis distinguishes between parameters that encode domain-invariant patterns, which transfer successfully across contexts, and those that capture domain-specific

characteristics requiring recalibration. The transfer protocol includes systematic ablation studies that quantify the benefit of transferred parameters compared to random initialization, establishing empirical foundations for transfer learning approaches.

Generalization metrics definition extends beyond traditional performance measures to capture the degree of successful knowledge transfer across domains. These metrics quantify not only predictive accuracy but also computational efficiency gains, convergence acceleration, and stability improvements achieved through transfer learning. The metric framework acknowledges that successful transfer might manifest differently across domains, with some applications prioritizing accuracy while others emphasize computational efficiency or robustness guarantees.

Cross-domain validation procedures establish rigorous protocols for assessing transferability claims through systematic experimentation across multiple target domains. These procedures mandate that validation datasets remain strictly independent of any data used during initial model development or transfer adaptation, preventing optimistic bias in transferability assessments. The validation protocol includes stress testing under domain-shift conditions that deliberately violate assumptions from the source domain, characterizing graceful degradation patterns and identifying failure modes that might compromise transferred models.

The accumulation of lessons learned and best practices through systematic documentation of transfer experiences creates a knowledge base that guides future transferability efforts.

Part III

Identity and Trust in Networked Mobility

5 Enhancing Intersection Identification for Autonomous Vehicles: A Hash-Based Approach

5.1 Introduction

The rapid advancement and deployment of Autonomous Vehicles (AVs) need to develop efficient strategies for uniquely identifying intersections [84]. The complexity of road networks, particularly in urban macro centers and especially in intersection areas, represents a challenge for current map matching algorithms. Furthermore, in the context of intersection management and Autonomous Intersection Management (AIM), the problem arises of potential conflicts between AVs that may erroneously agree on the same intersection, being instead at different intersections, for example, overlapping.

GPS alone is insufficient for robust AV navigation in large cities. Indeed, urban structures like tall buildings and tunnels degrade GPS accuracy through occlusion and multipath interference, producing localization errors exceeding 20 meters (over four times the 5 meter precision in ideal conditions). Such significant discrepancies between actual and estimated positions compromise autonomous driving safety and GPS outages from tunnels and underground streets also disrupt urban operations [85].

In recent years, the rapid development of AVs has led to a growing research interest in algorithms and related technologies. Chang and Shih [86] proposed an Intersection Location Service and Gao et al. [87] studied autonomous car vision systems, focusing on automatically detecting and classifying road contexts and traffic intersections under different weather conditions. Connected Autonomous Vehicles (CAVs) intersection management is outlined in a comprehensive survey by Khayatian et al. [88], which discusses key facets of real-world intersection management strategy development, including intersection management interfaces. However, despite the attention to details of the road context, these studies do not provide a specific solution for the unique identification of intersections. In the Jabbar et al. review [89], blockchain is viewed mainly as an enabler for secure vehicular communication, decentralized data management, payments, and other Internet of Vehicles applications. Although spatial hashing techniques are

effective for collision detection and may have great applicability for intersection identification [90], there is still ample space for further research in this area.

To fill this gap, we propose a novel approach that employs blockchain-inspired hash algorithms to identify intersections uniquely. In particular, this technique applies a hash algorithm to the intersection data, producing a unique fingerprint of the intersection itself. The proposed approach is inspired by blockchain systems, where similar hash generation is used to create Transaction Hash and Block Hash. The utilization of the proposed methodology is twofold: first, it allows the creation of a unique intersection footprint, enabling us to recall or uniquely identify the considered intersection in various contexts involving AVs. Secondly, it is conceivable that, at the end of their routes, AVs could potentially notarize on the blockchain, through a specific transaction or within a particular smart contract, the decisions made at every intersection, thereby unambiguously identifying them. This approach could trigger dedicated actions or smart contracts within the blockchain, such as imposing penalties for crossing into a restricted traffic zone, as a possibility for future application.

The remaining parts of the chapter are organized as follows. Section 5.2 describes the problem and Section 5.3 proposes the hash-based solution. In addition, Section 5.4 proposes a decentralized approach for hash generation and depicts the procedure for the AVs. Eventually, Section 5.5 describes and discusses a case study and Section 5.6 draws the conclusions.

5.2 Problem Description

In the related literature, the Autonomous Intersection Management (AIM) can be faced in centralized or decentralized approaches. In centralized AIM, vehicles communicate with an intersection manager to reserve conflict-free trajectories through the intersection space-time [91]. Overlapping adjacent intersections create difficulties in delineating clear communication zone boundaries. Vehicles may interact with multiple intersection managers simultaneously or experience confusion determining which manager to contact [92]. Centralized approaches to combining nearby intersections into a single coordination zone have been explored but face drawbacks like single points of failure, infrastructure constraints, and limited scalability [93]. The decentralized AIM has advantages in robustness and flexibility, but ambiguities in intersection boundaries can lead to coordination and interpretation issues between vehicles.

The introduction of lane-free AIM schemes by Li et al. [94], where batch-processing frameworks leverage, points out the necessity of having a unique identifier for each intersection. In such dynamic environments where CAVs adjust their velocities and paths continuously and cooperatively within an intersection, the absence of a unique identifier can escalate the complexity of the batch coordination process. It might give rise to scenarios where batches from different intersections incorrectly intertwine due to the need for precise intersection delineations, leading to safety hazards and inefficiencies. As proposed in this chapter, establishing a unique intersection hash would facilitate seamless batch processing by clearly defining the bounds of each intersection, thereby ensuring that the cooperative trajectory planning within each batch is confined to the correct intersection space.

Despite the challenges identified with both centralized and decentralized AIM systems, there has been a promising approach leveraged by DRL proposed by Isele et al. [95] to address the safety and efficiency in unsignaled intersections. Furthermore, integrating a unique intersection identification system, as proposed in this chapter, could work synergistically with DRL approaches, offering a more data-driven solution to navigate through intersections efficiently. Establishing a unique fingerprint for each intersection makes channeling more data into the DRL systems possible. This approach can simplify the learning processes and facilitate the application of the consensus protocols in real-world scenarios at intersections. Then, we face the problem of accurately identifying intersections and their associated communication zones in efficient AIM systems.

Code 1 Block Data

```
{   "hash": "000000000000000001
    ca3f1c683b050242df5842a20692515793b7c051b2b0f",
   "height": 571119,
   [...] "merkleroot": "9f217be3a492ebbab53fb207092204f1c
   b8530ae1144038848e8db044862ea53",
   "tx": [
      "a654a7f137b660d4d36973084c34d152e1e1970ecee8753057
   c4936427171cf1",
      [...],
      "6a747d2b4c98bc7a8e510e6b50abd036553eeefe66c86633bf 5
   ec7c444947e0b",
      [...],
      "be3728bb8cdcb0957fc6c600a094f34f521f09d46b2c450009
   bc65cbebda1975"
   ],
   "nonce": 334061627,
   "difficulty": 6393023717201.863,
   "chainwork": "0000000000000000000000000000000000000000000000 05
   c7966f5de524c504466f30",
   "nTx": 2100,
   "previousblockhash": "00000000000000000027703744d8d48 4
   a7fda73a16cd88af92fab3ea6f9d297" }
```

5.3 Hash-Based Solution

5.3.1 Inspirations from the Blockchain: The Role of Hash Algorithms

Satoshi Nakamoto, in 2009, already made extensive use of hash algorithms to uniquely identify transactions within his blockchain and subsequently to identify each block uniquely [96]. Hash algorithms are cryptographic functions that input data of any size and output a fixed-length string. This string is essentially a unique "fingerprint" of the original data. A key feature of hash algorithms is that even a small change to the input data produces an entirely different hash. This makes hash algorithms extremely useful for verifying data integrity [97].

The Code 1 shows block 571119's raw code of the Bitcoin blockchain. Note that all transactions included within this specific block are reported as a hash, where `a654a7...7171cf1` is the hash of the first one and `be3728...bda1975` is the hash of the last transaction included in the block. Along with the transactions in the block, a series of collateral information is also inserted into the block,

including the previous block's hash. With this method, the Bitcoin blockchain uniquely identifies blocks and individual transactions.

In cryptographic theory, it is recognized that alterations to a given piece of information invariably result in a corresponding change to its hash. A slight modification in the input, even a change as minor as flipping a single bit, leads to a significant alteration in the output hash, a phenomenon commonly known as the "avalanche effect". This property is critical in numerous applications of hash functions, including digital signatures, integrity checks, and password security, among others. This sensitivity to input data contributes to the robustness of hash functions in detecting tampering or corruption in the original information [98].

5.3.2 Proposed Solution

There is a consensus among researchers that GPS alone does not provide sufficient accuracy for autonomous vehicle navigation [99, 100].

Pal et al. [101] proposed recording the locations of traffic signs in a database and then using the database to complement GPS and sensor data in autonomous vehicles. Inspired by this approach, we build a *JavaScript Object Notation* (JSON) object that could be replicated deterministically and straightforwardly. Since a JSON object is a collection of key-value pairs, where the "value" can be a number, a string, a Boolean, an array, or even another JSON object, we choose to include in the JSON the attributes depicted in Table 5.1 to identify an intersection uniquely. This allows us to nest three different JSON objects within the identifying data of our intersection (*location*, *toponymy* and *parameters*); these data are based on the vehicle's Navigation System.

A critical issue is determining the level of accuracy required for GPS coordinates since excessive precision could generate different hashes. In this approach, excessive precision by the GPS device could be a problem since two vehicles within the same smaller roundabout, even just being at two different points, could produce two different hashes. To address this problem, it must be taken into account that within GPS coordinates, one degree without decimal places has an approximation of about 111 km [102], three decimal places offer an accuracy of 111 meters and four decimal places offer an accuracy of 11 meters. Since GPS is not the only tool for the univocal identification of the intersection, it was chosen to approximate it up to three digits. The same approach has been used for the altitude.

Utilizing the attributes of the intersection to create a JSON object, the decision to process it through the SHA256 algorithm was determined after carefully considering the prevalent SHA-256, part of the SHA-2 family, over SHA-3 alternatives. Despite the potentially higher security of SHA-3 options, SHA-256

Table 5.1: *Attributes of the Intersection Data JSON Object*

Key	**Type**	**Description**
location	Object	An object containing the geographical coordinates of the intersection.
├ latitude	Number	The latitude of the intersection, in decimal degrees.
├ longitude	Number	The longitude of the intersection, in decimal degrees.
└ altitude	Number	The altitude of the intersection, in meters above sea level.
toponymy	Object	An object containing the names associated with the intersection.
├ streetNames	Array of Strings	The streets names designating the intersection provided by the navigation system..
└ language	String	The ISO 639-1 code of the language set in the Navigation System
parameters	Object	An object containing details describing the characteristics of the intersection.
├ streetCount	Number	The number of streets intersecting at the intersection.
└ trafficLights	Boolean	A boolean indicating the presence of traffic lights at the intersection.

was favored due to its demonstrated stability, uniqueness of the results with the same inputs and uniform and rapid processing capabilities unaffected by input size variations [103].

Algorithm 1 specifies the JSON objects that are defined using the dictionary data structure in Python, which closely aligns with the conventions of JSON. Here, we choose to produce the hashes of the single JSON-child objects to use them separately.

Algorithm 1 Hashing JSON Object Children

procedure HASHJSONOBJECTS(obj)
  $child1, child2, child3 \leftarrow ExtractChildren(obj)$
    ▷ Extract the three JSON child objects
  for $i \in \{1, 2, 3\}$ **do**
    $jsonString_i \leftarrow SortAndStringify(child_i)$
      ▷ Convert the i-th JSON object to a sorted string
    $encodedString_i \leftarrow Encode(jsonString_i)$
      ▷ Encode the i-th JSON string
    $hash_i \leftarrow InitializeHash("SHA\text{-}256")$
      ▷ Initialize a new SHA-256 hash object for the i-th child
    $Update(hash_i, encodedString_i)$
      ▷ Update the i-th hash with the encoded JSON string
    $hashedJson_i \leftarrow Finalize(hash_i)$
      ▷ Convert the i-th hash to a hexadecimal string
  return $hashedJson_1, hashedJson_2, hashedJson_3$

Following the execution of Algorithm 1, each JSON child object is individually processed through a series of transformations to obtain their SHA-256 hash representations eventually.

Firstly, the *SortAndStringify* function ensures a consistent, ordered serialization of the dictionary entries, a crucial aspect given that the SHA-256 hashing algorithm operates sequentially on the input data; different ordering would result in distinct hashes. This function can also include other normalization parameters, such as avoiding capital letters or replacing toponymic abbreviations. Downstream of the actual transformation of the information into hashes, finally the SHA-256 hash objects are converted to their hexadecimal string representations facilitating easy readability and further analytical processing. Through this structured approach, the algorithm ensures the consistent and reliable hashing of JSON object children, readying them for subsequent use or analysis.

To obtain the unique identifier of the entire intersection, the path to follow is

not the creation of the hash of the JSON as a whole object but rather the result of the hash of the hashes of the child JSON, making a *Merkle Tree*. A Merkle Tree is a hash-based data structure that generalizes the hash list. It is a tree structure in which each leaf node is a hash of a block of data, and each non-leaf node is a hash of its children [104].

Particularly when dealing with complex JSON objects that encompass multiple keys such as "location", "toponymy", and "parameters", a Merkle Tree allows granular integrity checks. Each key-value pair can be hashed individually to form leaf nodes in the tree, enabling rapid and resource-efficient verification of sub-components of the data. This hierarchical structure facilitates pinpointing discrepancies down to specific keys, obviating the need to traverse or verify the entire dataset. At that point, the AV can check whether the hashes of the smaller objects match or not and where the discrepancies are. By doing this, we establish that the threshold to be respected is that 2/3 of the child hashes must match.

5.4 Intersection Identification for Autonomous Vehicles

This section provides a tool for mitigating the problems that can occur when autonomous vehicles have to cross an intersection.

In the related literature the intersection is divided into two zones: the Control Zone (CZ), where the vehicles communicate to find an agreement about the order by which they get through the intersection, and the Merging Zone (MZ), where the collisions can occur [105, 106].

The generation of a hash that identifies the intersection allows the vehicles to unequivocally identify the intersection before entering the CZ to guarantee that the autonomous vehicles communicate with the vehicles involved in the same intersection.

To avoid single points of failure within the system and to increase scalability and flexibility, we adopted a decentralized approach, so the vehicle can calculate the hashes of the intersections when planning the route, verifying them and proceeding when all the vehicles approaching the intersection agree on the identifier of the crossing they are facing.

Algorithm 2 reports the steps the autonomous vehicles can perform. As the vehicle approaches an intersection, it uses the `ApproachControlZone (intersection)` function to enter the designated control zone. Within this zone, the vehicle utilizes the `CollectDataAndRecheckHash(zone)` function to harvest real-time data and recompute the hash accordingly. This fetches the necessary data and validates the hash, thus ensuring a secure traversal through the intersection. A consensus mechanism is instituted, wherein the newly obtained hash is exchanged with other vehicles in the vicinity using the `ExchangeHash(hash, hashes)` function. If a consensus isn't achieved, then a secondary level of consensus is sought by exchanging child hashes, which have a more granular data management. This strategy needs at least a 2/3 agreement via the `ExchangeChildHashes(childHashes)` function. If consensus remains elusive and if our approach has been layered "on top" of another type of consensus, it is possible to revert the procedure to the original consensus mechanism (or to the standard rules of the road) through the `FallbackToOriginalConsensus()` function, ensuring navigation continuity while maintaining safety.

Algorithm 2 Procedure of Autonomous Vehicle

```
1:  procedure AutonomousVehicleProcedure
2:      route ← DefineRoute()                                  ▷ Define route
3:      path ← CalculatePath(route)                            ▷ Get path
4:      intersections ← IdentifyIntersections(path)
5:                                                             ▷ Find intersections
6:      hashes ← Calc.IntersectionsHashes(intersections)
7:                                                             ▷ Get hashes
8:      StartJourney(path)
9:      for each intersection in intersections do
10:         zone ← ApproachControlZone(intersection)
11:         data, hash ← CollectDataAndRecheckHash(zone)
12:                                                            ▷ Get data and recheck hash
13:         consensus ← ExchangeHash(hash, hashes)
14:         if not consensus then
15:             childHashes ← GetChildHashes(data)
16:             consensus ← ExchangeChildHashes(childHashes)
17:                                                            ▷ 2/3 consensus
18:         if not consensus then
19:             FallbackToOriginalConsensus()
20:     ReachDestination()
21:     UploadToBlockchain(hashes)                             ▷ Upload data
```

5.5 Case study

5.5.1 Case study description

In this section, we show an application of the proposed solution to a real complex case study: the *Magic Roundabout* in Swindon, England formed by an unconventional multi-roundabout configuration as illustrated in Fig. 5.1. With five satellite roundabouts encircling a central hub, identifying the correct pathway through this complex traffic circle is non-trivial compared to traditional single-ring layouts.

Several factors contribute to the difficulty autonomous vehicles face when approaching this particular roundabout:

- The unusual arrangement of alternating clockwise and counter-clockwise flows across multiple intersections introduces ambiguity in determining the appropriate yielding behavior. Unlike single roundabouts where the right-of-way is clear, the crisscrossing of traffic directions at the Magic Roundabout leads to confusion.
- Occlusions from infrastructure and other vehicles can obstruct sensor visibility. With multiple intersections in close proximity, obstructions are more likely to prevent early detection of crossing traffic flows.
- The complexity makes a priory mapping of valid trajectories difficult. Predefined driving routes are less effective since the environment is highly dynamic.
- Close proximity of entry and exit points creates uncertainty in intersection identification. Determining which specific roundabout an autonomous vehicle should target is obscured by near parallel roads.

5.5.2 Proposed Solution

The considered Magic Roundabout is divided into a set of sections as illustrated in Fig. 5.2, and each section represents a roundabout and is denoted with a different color. We focus on two roundabouts, named Blue and Pink. These roundabouts are characterized by unique features, ensuring their unequivocal identification. In particular, this precise identification is attainable even under conditions where the GPS signal's accuracy, traditionally considered the primary source of location information, is compromised. This resilience is due to the inclusion of additional data stored securely within the individual JSON objects

Figure 5.1: *Map view of the "Magic Roundabout" in Swindon, England (UK) [1]*

corresponding to each roundabout. Essentially, each roundabout is seen as an independent entity possessing its unique data.

After obtaining the unique hash derived from the JSON objects, the vehicles initiate a data-sharing protocol with the other vehicles circulating within the *Magic Roundabout*. This protocol provides these vehicles with access to accurate, real-time data regarding the behavior and movement of vehicles within the smaller roundabouts.

For instance, a vehicle traveling on the A 4259 and wishing to continue on it, will approach this junction by first crossing the Pink roundabout, then the Cyan roundabout and finally the Blue roundabout. Then, the vehicle communicates its movements to all the present vehicles, identifying precisely which sections of the macro roundabout it will face at each time interval. This allows it to negotiate the intersection simultaneously with vehicles that do not need the smaller roundabouts it uses in the same time interval.

In the following, we describe the distinguishing features of the JSON objects associated with the two roundabouts Blue and Pink that are determined by Code 2 and Code 3, respectively.

Figure 5.2: *Control Zones intersections on Swindon Magic Roundabout.svg by Hk kng [2]*

Code 2 Data of the Blue roundabout in 5.2 in JSON format.

```
{"location": {"latitude": 51.562, "longitude": -1.771, "
    altitude": 102},
  "toponymy": {"streetNames": ["County Road", "Magic
    Roundabout", "Private Road", "A 4259"], "Language": "en"},
  "parameters": {"streetCount": 4, "trafficLights": false}}
```

Code 3 Data of the Pink roundabout in 5.2 in JSON format.

```
{"location": {"latitude": 51.562, "longitude": -1.771, "
    altitude": 101},
  "toponymy": {"streetNames": ["A 4259", "Magic Roundabout", "
    Queen's Drive"], "Language": "en"},
  "parameters": {"streetCount": 3, "trafficLights": false}}
```

According to Algorithm 1, the SHA-256 hash of the JSON object representing the Blue and the Pink roundabout are represented in Table 5.2 where, for brevity, only the first hash is reported.

Drawing parallels with operational protocols in the context of blockchain systems, our approach can be compared to the methodology employed within a Bitcoin block. In such a system, each transaction is represented not by its

specific details but through a unique hash that encapsulates all the requisite information. Likewise, according to the Merkle Tree structure, a unique identifier could be assigned to the entire roundabout, which is a composite hash computed by aggregating the hashes of all the smaller roundabouts (Blue, Cyan, Pink, etc.).

Table 5.2: *Computed SHA-256 Hashes for the Child Objects in the Blue and Pink JSON Representations*

JSON Object	JSON-child	SHA-256 Hash
Blue	location	`7b01cba7acb9b65b4b726289d66 64e19be31e243beba38491de16d d16f610ec6`
	toponymy	`275b8a67be38...cf3c09784e1b`
	parameters	`05ad4e76a622...252ad6a4d1b7`
Pink	location	`a3d51bacbb32...17e2733b96b7`
	toponymy	`607a69ded6d6...168bab948a4c`
	parameters	`062bf46d1d72...cf5db03f0d16`

Following the construction of the Merkle Tree, the hash of the Blue roundabout is `82fb8b6622ae104f323ed77d7a35ff120362f7d8f257a7822f185e67b460f2c8` and the hash of the Pink one is `cce096295cfe607badb2a37c6a34c9584e78f5720ba678b17ff2e66df3726502`.

5.5.3 Discussion of the results and critical issues analysis

The proposed approach allows us to divide a very complex intersection with a huge CZ into smaller and more manageable intersections and CZs. Although these new CZs overlap, each one is distinctly identified, thereby accelerating the consensus-reaching process for approaching vehicles.

The considered case study has an overall circumference of approximately 100 meters. Therefore, given how the system has been devised, every single argument of the JSON object does not need to be unique, given that the uniqueness of the object is given by the sum of the elements it contains and not by the single data. So, even if we choose to cut the GPS coordinates at 3 digits, having two small roundabouts utterly identical regarding GPS coordinates, the identification hashes will still be unique.

An additional problem that can occur is the nomenclature of the streets. For example, having two systems in different languages could produce slightly different

input data; e.g., it could happen if a vehicle with the Latin alphabet generated the unique identifiers of Greek road junctions, which local cars identify with the Greek alphabet. The "toponymy" child hash match could be deprioritized to resolve this issue so that other data are preferred. It should also be noted that the number of child objects within the JSON can be enriched or modified depending on particular needs.

Utilizing unique intersection identification can enhance application scenarios significantly, enabling the navigation of autonomous vehicles with increased precision. Furthermore, remaining in the blockchain realm, our solution can be used to notarize the route expressed as an array of intersection hashes recorded by the autonomous vehicle throughout a journey or a day. This information, being uniquely notarized on the blockchain, would also allow the resolution of disputes, eliminating data uncertainty.

5.6 Conclusions

This chapter leveraged features inspired by blockchain technology, specifically employing hash algorithms, to make a significant advancement in the unambiguous identification of road intersections, a critical aspect in the autonomous vehicle (AV) field. With this approach, intersections can now be distinguished unequivocally based on their unique characteristics, thereby enhancing resilience against inaccuracies inherent in GPS technology. We described a methodology to assign a unique identifier to each intersection. Moreover, the delineation of CZs facilitates swift consensus among vehicles, even amidst visual barriers. The efficacy of this approach is evidenced in the proposed case study, showing the practical applicability of the proposed solution. In future research, we plan to add a quantitative and qualitative analysis in order to show the effectiveness of the proposed method and to use Artificial Intelligence to draw a basic pattern of the intersections and hash the derived image composed only of uniquely raw geometric shapes.

6 A Blockchain Framework for Incentivized Data Sharing in Autonomous Vehicle Networks

6.1 Introduction

Autonomous vehicles (AVs) generate vast, high-resolution datasets through advanced sensor networks (including LiDAR, camera arrays, and radar systems), capturing everything from evolving road conditions to near-instantaneous traffic updates. Although these data are essential for enhancing road safety, optimizing route planning, and guiding infrastructure development, most information remains locked within brand-specific platforms. As a result, AVs from different manufacturers do not benefit from each other's real-time observations, and public authorities, service providers, and end users miss out on crucial insights for city-wide traffic management, road maintenance, and urban planning. Additionally, modern proprietary navigation platforms typically operate under centralized, closed-source paradigms, offering no direct economic incentives to users for sharing high-precision data and severely limiting data ownership transparency within the provider's ecosystem.

Recently, blockchain-based architectures have emerged as promising approaches for addressing information sharing, interoperability, and trust issues in Intelligent Transportation Systems (ITS) [107]. Rajkumar et al. [108] proposed vehicular data architectures leveraging trusted execution environments, and Cui et al. [109] introduced consortium blockchains for secure V2V communications. Both works, however, primarily focus on privacy/security without considering economic incentives or reputation-driven consensus mechanisms, which are essential factors pursued in our approach. Moreover, discrete event system techniques have also been considered for enhancing cyber-attack detection, security, and fault diagnosis in decentralized AV data-sharing systems [110, 111].

Recent systematic reviews, such as those by Sarwatt et al. [112], Alherimi et al. [113], Kim and Vong [114], and Vairam et al. [115], catalog diverse blockchain applications spanning automotive, IoT, and ITS domains. These contributions generally overlook barriers associated with cross-manufacturer collaboration, open participation, and economically-driven data-sharing ecosystems, which constitute core challenges our work explicitly tackles.

Additionally, Yang et al. [116] and Rasool et al. [117] explored Decentralized Autonomous Organizations (DAOs) as governance frameworks, enabling decentralized decision-making processes. Nevertheless, these contributions do not directly support dynamic sensor-data monetization or cross-brand interoperability. Similarly, Qin et al. [118] introduced a multi-blockchain ("TriBoDeS") architecture suited to hazard dissemination in vehicular environments. Despite the innovation, the tri-blockchain configuration brings significant operational complexity, impeding applicability to realistic AV scenarios involving heterogeneous manufacturers.

Motivated by these limitations, this chapter proposes a lightweight, decentralized framework for real-time AV data exchange, integrating modular smart contracts, dynamic consensus threshold mechanisms, vehicle reputation-based incentivization, and blockchain tokenomics. Unlike existing solutions, our approach uniquely combines cross-brand interoperability, real-time validation, economic incentivization, and high scalability, ultimately fostering open, collaborative vehicular data marketplaces. Verified contributions directly translate into incentives (tokens) redeemable as practical benefits (charging discounts, parking fees, toll reductions), thereby aligning individual vehicle incentives with collective intelligent mobility objectives.

The remainder of this chapter proceeds as follows: Section 6.2 describes the problem, and Section 6.3 outlines the System Architecture. Section 6.4 mathematically outlines the dynamic threshold of the considered system, Section 6.5 presents the blockchain simulation and Section 6.6 draws the conclusions.

6.2 Problem Description

Autonomous vehicles can collect both static information, such as road infrastructure characteristics and classification, and dynamic phenomena, such as unexpected traffic incidents, temporary road closures, or routine maintenance work. Despite the critical importance of these insights for real-time decision-making and broader traffic management, the data collected often remains confined within proprietary ecosystems belonging to individual manufacturers.

The isolation of data within brand-specific platforms severely limits interoperability and stifles the potential benefits of a more comprehensive, cross-manufacturer data repository. For instance, consider a scenario in which a vehicle navigates what was initially marked as a secondary, paved route in its onboard navigation system yet encounters an unpaved, rough terrain in reality. The AV's sensors detect this mismatch, generating valuable information about the true state of the roadway. However, because this information remains locked within a single manufacturer's proprietary system, other AVs and stakeholders do not benefit from these real-time observations.

A key challenge in addressing these limitations lies in establishing a secure, trust-based mechanism that incentivizes the seamless exchange of vehicular data across multiple stakeholders, specifically addressing several key elements. Firstly, ensuring robust data provenance and integrity is imperative to guarantee confidence in the origin and authenticity of shared information and effectively mitigate risks associated with falsification or unauthorized tampering. Secondly, owner privacy must be meticulously preserved, especially in contexts where vehicular datasets encompass sensitive or personally identifiable information, necessitating advanced anonymization and privacy-preserving strategies. Thirdly, a carefully structured monetization and incentive system is essential to motivate extensive and willing participation in data-sharing initiatives. Finally, the capability to support real-time processing and scalability is vital, enabling rapid, near-instantaneous transactions within distributed environments and adequately managing the substantial throughput demands typical of contemporary sensor networks deployed in autonomous vehicles.

In light of these considerations, the need for a distributed data framework becomes evident. By leveraging blockchain-based architectures, smart contracts, and advanced cryptographic techniques, AV data can be securely exchanged and monetized across brand and platform boundaries.

6.3 System Architecture and Modular Smart Contracts

The system architecture, depicted in Fig. 6.1, enables autonomous vehicles from different manufacturers to detect, broadcast, and validate critical environmental changes through a public, permissionless blockchain infrastructure. The core components and interactions are summarized in the following subsections.

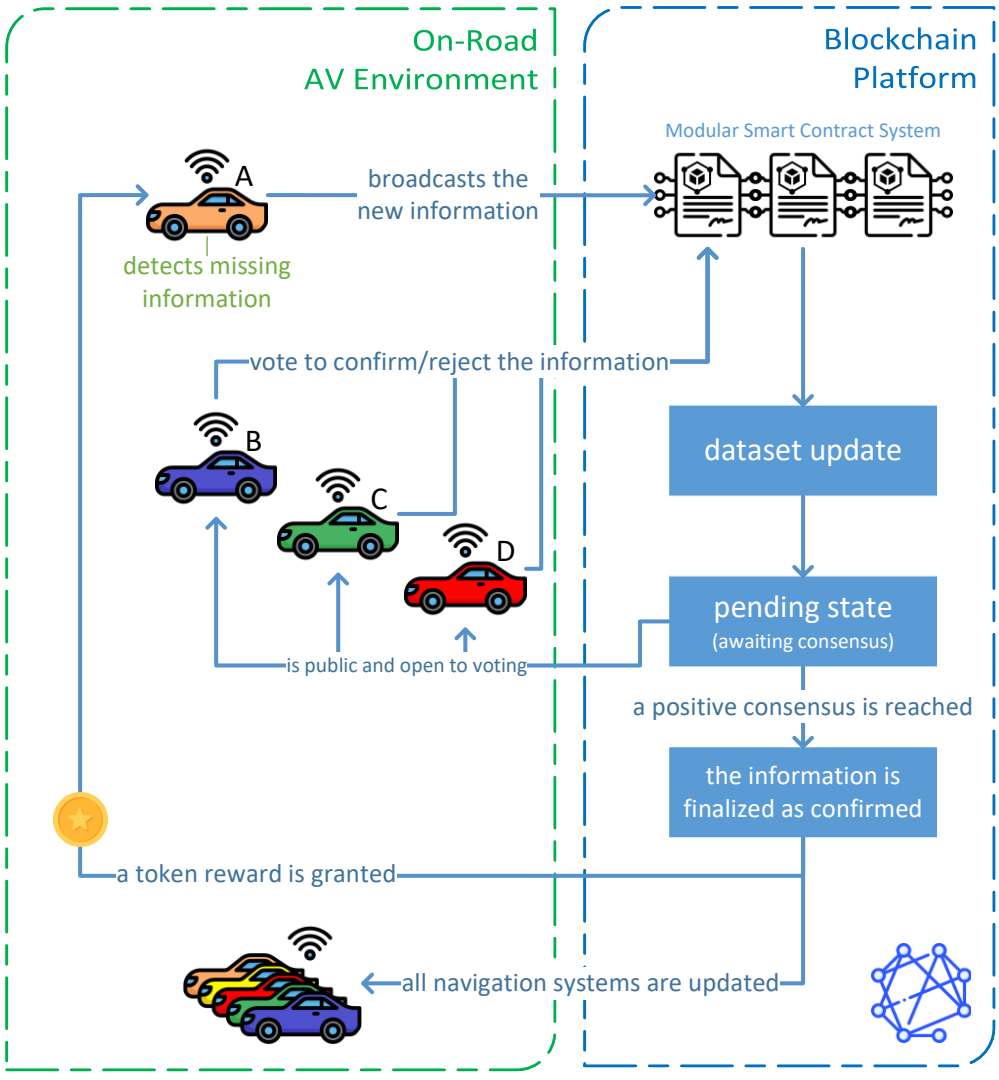


Figure 6.1: *System architecture [3]*

6.3.1 Overview of the Proposed Workflow

Step 1: Detection of Discrepancy. Vehicle **A** (shown in orange in Fig. 6.1) identifies a mismatch between the real-world state of an infrastructure element (e.g., a newly unpaved road or a temporary closure) and the existing information in its onboard navigation system. This discrepancy can arise from *ad hoc* changes (e.g., ongoing construction) or more permanent infrastructural modifications.

Step 2: Broadcast of New Information. Once Vehicle **A** detects this discrepancy, it encodes the updated information (e.g., the exact location, time, and nature of the change) in a structured JSON object. Our earlier research [119] indicates that implementing a hash-based system for uniquely identifying intersections is feasible. This JSON payload is then submitted to the *smart contract system* (further elaborated in Algorithm 3), triggering a blockchain transaction. This transaction records key metadata such as a timestamp, vehicle pseudonym (to preserve privacy), and event category.

Step 3: Pending State Creation. Upon receiving the transaction, the blockchain-based smart contract appends the new data to a global dataset in a *pending state*. At this stage, the reported change has no immediate effect on navigation systems; rather, it awaits consensus from other vehicles.

Step 4: Crowd-Sourced Validation. As vehicles **B**, **C**, and **D** traverse or otherwise sense the same region, they submit *confirmation* or *rejection* transactions to the smart contract. These vehicles also rely on their own sensor inputs to verify the reported information. Multiple confirmation transactions strengthen the claim, while multiple rejection transactions may overturn it.

Step 5: Consensus Threshold. Once the threshold discussed in Section 6.4 is reached, the pending state is resolved. The threshold is dynamically adjusted based on road type, event severity, and vehicle reputation. If consensus deems the newly reported data to be accurate, it is marked as *confirmed*. Otherwise, the entry is *rejected*.

Step 6: Data Finalization and Reward. In the event of a successful confirmation, the updated information is promoted to the *structured data layer* and disseminated to all vehicles' navigation systems. This ensures that other AVs immediately benefit from the new information, improving their route planning and operational safety. Vehicle **A** –the original proposer of the valid data– is awarded a token as an incentive.

6.3.2 Modular Smart Contracts System

The proposed framework achieves flexibility and scalability by deploying three closely integrated smart contracts, each with a distinct but interlocking responsibility. As detailed in Algorithm 3, the *FilterContract* first receives every new data submission transmitted by a vehicle, performing checks on the JSON-encoded payload and verifying that no identical *pending* entry exists for the same event. When a new submission is found to be unique, the contract relays it to the *VotingContract* with *pending* status;

Once a proposal is marked as *pending*, the *VotingContract* takes over to manage crowd-sourced validation. Each vehicle with relevant sensor data can cast a `CONFIRM` or `REJECT` vote. The contract keeps a running tally of both counts and, after each vote, recalculates a dynamic acceptance threshold according to the methodology presented in Section 6.4. This threshold, denoted $T_{\text{dyn}}(i)$, varies based on different factors such as the severity of the event, the criticality of the road segment, the elapsed time since the proposal was submitted, and the reputation of the proposer. In this way, the *VotingContract* can either finalize an outcome before the voting window concludes –triggered by immediate confirmation or rejection criteria– or wait until a predefined time limit or a minimum vote count is reached. Once a decision has been reached, the result is irrevocably recorded on-chain, ensuring transparency and accountability in the consensus process. The final module, the *TokenContract*, governs the system's incentive layer. Suppose the *VotingContract* signals that a proposal has been confirmed. In that case, the TokenContract mints one or more tokens in the name of the original proposer, thereby acknowledging a valid and beneficial data contribution to the network. Note that all reputation updates and flag management, incorporating the double-hit logic (discussed further below in Section 6.4.3) for handling consecutive negative proposals, are performed within the *VotingContract*, which decouples these operations from token management. The *TokenContract* further handles an array of token lifecycle operations, including transfer, balance tracking, and burn logic. This modular design, wherein the tokenomics are confined to a dedicated contract, allows for tailored governance and dynamic adjustments to reward policies without disrupting the consensus or filtering processes.

Together, these three smart contracts operate in unison to secure real-time data, validate its authenticity, and incentivize honest reporting. By cleanly separating the *filtering*, *consensus*, and *reward* functionalities, the architecture remains flexible to evolving requirements while preserving core trust assumptions.

Algorithm 3 Workflow for Distributed AV Data Sharing via Modular Smart Contracts

Step 1: FilterContract - Initial Validation and Duplicate Checking
Receive *txData* (e.g., infrastructure mismatch) and v_{prop} (vehicle pseudonym).
Perform basic checks on *txData* (required fields in JSON format).
Verify whether an identical *pending* entry already exists for the same issue.
if no matching entry is found **then**
    Forward *txData* to `VotingContract` with *pending* status.
else
    Convert this submission into a `CONFIRM` vote for the existing *pending* entry in `VotingContract`.
    Do not create a new pending record.
Step 2: VotingContract - Consensus Mechanism
Initialize vote counts for each *pending* proposal: *yesVotes* = 0, *noVotes* = 0.
for each vote received from vehicles **do**
    if vote is `CONFIRM` **then**
        Increment *yesVotes*.
    else if vote is `REJECT` **then**
        Increment *noVotes*.
    Dynamic recalculation of $T_{\text{dyn}}(i)$ *and immediate check*
    Check if *yesVotes* and *noVotes* trigger immediate approval or rejection.
At the end of the voting window or upon reaching the minimum required number of votes
Check final consensus according to Section 6.4.
if proposal is approved **then**
    Forward the approval result to `TokenContract`.
else
    Mark the proposal as rejected.
Emit an event indicating the outcome (`"CONFIRMED"` or `"REJECTED"`).
Update *rep(v)* and *f(v)* in accordance with Section 6.4.3.
Step 3: TokenContract - Token Management
if proposal is approved **then**
    Mint a reward token to the proposer v_{prop}.
    Store the validated data in the *confirmed* repository.
Handle all token lifecycle operations (e.g., emission, balance tracking, burn logic).

6.3.3 Blockchain Platform

In order to maximize transparency, immutability, and cross-brand participation, the proposed system leverages a blockchain **public**, one in which every transaction on the ledger is openly readable, enabling stakeholders to audit data flows and verify event integrity, and **permissionless**, meaning that any entity may participate in block validation or production (in accordance with the network's consensus rules).

Since vehicles remain pseudonymized via cryptographic addresses, privacy concerns are mitigated while retaining the benefits of a trustless, decentralized architecture. This permissionless approach fosters an open ecosystem where entities from multiple automotive manufacturers can seamlessly contribute to and benefit from the shared dataset by obviating the need for a central authority.

Although the proposed system already integrates a reputation model and dynamic thresholding, additional mechanisms could further refine both the consensus process and the reward structure. Under such a *dynamically weighted* scheme, submissions from reputable entities could reach an accelerated consensus on critical events while mitigating the influence of malicious actors. Similarly, the token issuance could be tied to contextual factors such as the severity or criticality of the event, effectively rewarding contributors based on the *quality* or *importance* of their reports.

Furthermore, practical considerations for real-time, large-scale deployments underscore the need to minimize transaction costs and ensure network scalability. One viable solution is to adopt a high-throughput platform or a specialized *subnet* on public permissionless blockchains like Avalanche [27], or to employ a layer-two scaling solution on Ethereum [120]. By leveraging existing validator infrastructures, the system can dramatically reduce per-transaction fees and latencies, thereby enhancing both cost-effectiveness and responsiveness. Such an approach also preserves the trustless, decentralized nature of the network, which is essential for an open, multi-vendor ecosystem of autonomous vehicles.

6.3.4 Real-World Applications of the Reward Mechanism

The token-based reward mechanism introduced in our framework offers practical utilities beyond data contribution incentives, encouraging continuous engagement and fostering an ecosystem of mutual value creation among vehicles, users, and service providers. Specifically, tokens can be practically redeemed in multiple key scenarios within the autonomous vehicle ecosystem, such as:

- **Premium Data Access:** Vehicles holding tokens can utilize them to reduce

or waive fees for accessing specialized or high-value data within the system, thus incentivizing continued participation.

- **Charging Discounts:** Electric and hybrid vehicles may redeem tokens to obtain discounts at charging stations, thereby reducing operational expenditures.
- **Parking Fee Management:** Integration of tokens into urban parking systems allows users to conveniently pay parking fees or receive price reductions.
- **Toll Payment Reduction:** Toll authorities can offer preferential toll rates or facilitate smoother billing transactions through token adoption, rewarding vehicles that actively contribute reliable road-condition data.

These practical applications, which can be later implemented, establish a continuous cycle of economic incentives, fostering widespread adoption by aligning individual benefits with collective improvements in road safety and efficiency.

6.4 Mathematical Formulation for Determining the Dynamic Threshold

This section proposes a mathematical formulation for a dynamic threshold to efficiently manage the second smart contract described in Section 6.3.2 and Algorithm 3. The goal is to parameterize the various factors involved to allow urgent information to be accepted and transmitted to navigation systems in a timely manner. At the same time, it is crucial to prevent system manipulation by entities that repeatedly provide incorrect or fraudulent information, disfavoring them in the long term while accounting for possible detection and transmission errors.

A reputation system was introduced for vehicles transmitting new information to the system; this parameter will play a role in modeling the dynamic acceptance threshold. However, we have chosen not to have the reputation affect the reaching of consensus (so individual vehicles will not have different weights when voting on whether a piece of information is correct or not). This ensures that the system is as distributed as possible and avoids single points of failure.

6.4.1 Dynamic Acceptance Threshold: Formulation and Notation

Parameters and variables employed throughout this section are summarized in Table 6.1.

The dynamic acceptance threshold $T_{\text{dyn}}(i)$ for proposal i is defined as:

$$\begin{aligned} T_{\text{dyn}}(i) \;=\; \text{clamp}\Big[\; & \tau_{\text{base}} \;+\; \alpha\,\text{sev}(i) \;+\; \beta\,\text{road}(i) \\ & -\; \gamma\,\ln\!\big(1+\Delta t_i\big) \;-\; \lambda\,\text{rep}\big(v_{\text{prop}}(i)\big),\; T_{\min},\; T_{\max}\Big] \end{aligned} \tag{6.1}$$

where $\text{clamp}[x, a, b] = \min\big(\max(x,\, a),\, b\big)$ ensures that x is limited between a and b, preventing the threshold from becoming negative or excessively high and the parameters τ_{base}, α, β, γ, and λ are weighting coefficients that determine the influence of each factor on the dynamic threshold.

In this initial application scenario, we opted to initially set all coefficients equal to 1 ($\alpha = \beta = \gamma = \lambda = \tau_{\text{base}} = 1$), deferring a more detailed parameter tuning to subsequent analyses. This allows us to assess the behavior of the normalized variables before applying different weights and to lay the groundwork for future multi-objective optimization as part of a simulation. Retaining the coefficients in

the formula provides flexibility for potential adjustments based on empirical data or changing requirements.

Variables $\text{sev}(i)$ and $\text{road}(i)$ are normalized within the range $[0, 1]$ for consistency. The term Δt_i is measured in terms of the *number of blocks* emitted by the network, as it is the basic unit within the blockchain. Its scaling will depend on the selected blockchain platform.

The reputation $\text{rep}(v)$ is updated according to the rules specified in Section 6.4.3. The global thresholds $T_{\min}$ and $T_{\max}$ ensure that $T_{\text{dyn}}(i)$ stays within acceptable bounds. With this formulation, if the proposer's reputation is very high and/or a significant amount of time has passed (large Δt_i), the threshold lowers but never goes below $T_{\min}$. Conversely, if the event's severity or the road segment's criticality is very high, the threshold cannot rise indefinitely due to $T_{\max}$. Defining $T_{\min}$ and $T_{\max}$ based on preliminary tests or simulations to reflect real voting conditions and vehicle participation is advisable.

6.4.2 Voting Rules and Final Decision

Let $\text{Y}(i) \in \mathbb{N}_{\geq 0}$ and $\text{N}(i) \in \mathbb{N}_{\geq 0}$ respectively denote the cumulative counts of approval (`CONFIRM`) and rejection (`REJECT`) votes for a pending proposal i. Let $T_{\text{dyn}}(i)$ represent the dynamic validation threshold for proposal i, calculated according to the formula presented in Sec. 6.4.1. Additionally, let $k_{\min}$ represent the minimum number of votes necessary and timeWindow the maximum allowed voting duration.

The decision-making follows four phases:

1. **Immediate Confirmation Criterion:** A submitted data item is immediately accepted and finalized as **confirmed** if:

$$\text{Y}(i) \geq T_{\text{dyn}}(i) \quad \text{and} \quad \text{Y}(i) > \text{N}(i). \tag{6.2}$$

2. **Immediate Rejection Criterion:** Conversely, a proposal is immediately marked as **rejected** if either of the following conditions holds:

$$\text{N}(i) \geq T_{\text{dyn}}(i) \quad \text{or} \quad \text{N}(i) > \text{Y}(i). \tag{6.3}$$

3. **Final Resolution at Voting Closure:** If neither immediate confirmation nor immediate rejection criteria have been triggered by the end of the voting window (i.e., upon completion of time timeWindow or upon reaching the

minimum vote requirement $k_{\min}$), the final state of the proposal is decided according to the majority criterion:

$$\text{status}(i) = \begin{cases} \textbf{confirmed}, & \text{Y}(i) > \text{N}(i), \\ \textbf{rejected}, & \text{Y}(i) \leq \text{N}(i). \end{cases} \tag{6.4}$$

4. **Resolving Simultaneous Threshold Exceedance:** In scenarios where both approvals votes $\text{Y}(i)$ and rejection votes $\text{N}(i)$ concurrently cross the dynamic threshold $T_{\text{dyn}}(i)$ within the same time interval (e.g., within the same block of blockchain transactions), we deterministically resolve the ambiguity by evaluating the vote difference:

$$\text{status}(i) = \begin{cases} \textbf{confirmed} & \text{if } \text{Y}(i) - \text{N}(i) > 0, \\ \textbf{rejected}, & \text{otherwise.} \end{cases} \tag{6.5}$$

This deterministic voting scheme effectively prevents ambiguity and mitigates potential manipulation or exploitation of the voting system. The combination of dynamic thresholding, minimum vote counts, and defined timing windows guarantees reliable and timely decision-making, thereby enhancing overall data accuracy and consistency across automotive platforms.

6.4.3 Vehicle Reputation and Incentive Mechanism

A simple reputation model inspired by repeated-game frameworks is implemented to incentivize truthful data submission and build long-term vehicle collaboration [83]. Each vehicle $v \in \mathcal{V}$ maintains a reputation score $\text{rep}(v)$, initialized at a positive default value $\text{rep}_{\text{default}}$.

Our system adopts a *double-hit penalty* policy to balance fairness and robustness to errors. Specifically, when a vehicle proposes information that is subsequently rejected through the consensus procedure, it does not suffer an immediate penalty, rather receiving a one-time binary warning flag $f(v)$. Only upon submitting two consecutive rejected proposals does the vehicle incur a reputation penalty. Conversely, if previously decreased, a successfully confirmed contribution resets the flagged status and increases the vehicle's reputation score.

These reputation adjustments have a twofold benefit: on the one hand, they discourage malicious and careless behavior while avoiding excessively penalizing occasional sensor inaccuracies or unavoidable false positives. This straightforward

yet effective logic creates incentives for sustained collaborative participation by rewarding accurate, timely inputs and gently discouraging unreliable submissions. In our implementation, the reputation updating logic remains encapsulated within the VotingContract (Section 6.3.2), thus clearly separating incentive management from foundational blockchain transactions.

Table 6.1: *Summary of Symbols and Notation*

Symbol	Description
$\mathcal{V} = \{v_1, \ldots, v_N\}$	Set of vehicles involved.
$v_{\text{prop}}(i) \in \mathcal{V}$	Vehicle (proposer) that submits proposal i.
$\text{sev}(i) \in [0, 1]$	Severity of the event reported by proposal i.
$\text{road}(i) \in [0, 1]$	Criticality of the road segment for proposal i.
$\Delta t_i \geq 0$	Time elapsed since the initial submission of proposal i, i.e., $\Delta t_i = \text{timestamp}_{\text{current}} - \text{timestamp}^{(i)}_{\text{start}}$.
$\text{rep}(v) \in \mathbb{R}_{\geq 0}$	Reputation score of vehicle v.
$f(v) \in \{0, 1\}$	Alert flag: $f(v) = 1$ if vehicle v is flagged in an alert state due to a recently rejected proposal
$\tau_{\text{base}} \geq 0$	Base threshold for approving a proposal with minimum severity and criticality.
λ	Weighting coefficient for the proposer's reputation term $\text{rep}(v_{\text{prop}}(i))$.
α	Weighting coefficient for the event severity term $\text{sev}(i)$.
β	Weighting coefficient for the road criticality term $\text{road}(i)$.
γ	Weighting coefficient for the elapsed time term $\ln(1 + \Delta t_i)$, ensuring diminishing returns.
$T_{\text{dyn}}(i)$	Dynamic acceptance threshold for proposal i
$T_{\text{min}} \geq 0, \quad T_{\text{max}} \geq T_{\text{min}}$	Global minimum and maximum bounds for the dynamic threshold $T_{\text{dyn}}(i)$.
$\text{timeWindow} > 0$	Maximum time window within which a proposal must be confirmed or rejected.
$k_{\text{min}} \geq 1$	Minimum number of votes required to reach a decision.
$\text{Y}(i) \in \mathbb{N}_{\geq 0}$	Number of `CONFIRM/YES` votes for proposal i.
$\text{N}(i) \in \mathbb{N}_{\geq 0}$	Number of `REJECT/NO` votes for proposal i.

6.5 Blockchain Simulation and Experimental Validation

To preliminarily validate the proposed blockchain architecture, the modular smart contract system outlined in Algorithm 3 was implemented and tested, including the *FilterContract*, *VotingContract*, and *TokenContract*. Experiments were conducted using the Hardhat Ethereum development environment, deploying Solidity-based smart contracts on an Ethereum Virtual Machine (EVM) emulator. All simulations were executed under a modest computer environment hosted by Ubuntu Linux 20.04 LTS system, powered by an Intel Core i3 processor and 8 GB of RAM. Additionally, to thoroughly emulate multiple interacting vehicles, the Hardhat environment was configured to feature an increased number of default accounts to carry out an extensive simulation. The simulation involved scripting in JavaScript to automate testing, thus ensuring repeatability and reproducibility. Notably, deploying the simulation on an EVM-based emulator allowed the immediate transferability of validated smart contracts to any compatible Ethereum-like blockchain network, such as an Ethereum public testnet or Avalanche Fuji public testnet.

Specifically, our simulations successfully demonstrated on-chain dynamic threshold updates in accordance with the mathematical formulation defined in Section 6.4. As illustrated in the Hardhat simulation output in Fig. 6.2, test `I` verifies the correct deployment of all smart contracts and confirms that the initial reward token balance for the proposer account `0xf[...]226` is zero. Test `II` focuses on duplicate submission handling: the same submission –e.g., an identical JSON payload– is sent from a different proposer account. In this scenario, the smart contract identifies the duplicate event, but (since the submission originates from a distinct account) it automatically counts this as an affirmative vote for the initial proposal. Test `III` evaluates the dynamic acceptance threshold, confirming that it properly decreases over time as expected, and test `IV` demonstrates the dynamic threshold's sensitivity to the proposer's reputation. Here, the same proposer submits two proposals characterized by different parameters, each exerting the same weight on the dynamic threshold, both of which are rejected. Notably, the threshold value for both proposals remains unchanged, as the flag mechanism described in Section 6.4.3 is activated after the first rejection. However, the proposer's reputation score is decreased upon the second consecutive rejection. As a result, when the same proposer submits a third proposal, the initial dynamic threshold increases from 14 to 15 votes. Following the acceptance of this third proposal, the reward token balance for the proposer is incremented by one unit, and the proposer's reputation is restored to its default value. The code behind these

```
DYNAMIC THRESHOLD VOTING AND REPUTATION MECHANISM SIMULATION
  I) Deployment and Proposal Initiation
✓ Proposer: 0xf[...]266
✓ Initial token balance: 0.0
     ✓ New proposal should be stored on-chain with status Pending
   II) Duplicate Submission Handling
✓ Duplicate submission processed as vote from account1. yes: 1
     ✓ Duplicate submission from a different account should count
 as an affirmative vote
  III) Dynamic Threshold Behavior
Dynamic Threshold Test: Proposal submitted. Proposal ID: 0x6[...]
Initial dynamic threshold (norm.): 14
Dynamic threshold after 6 hours (norm.): 5
     ✓ Dynamic threshold should decrease gradually with time
  IV) Proposals Lifecycle and Reputation Update
First Proposal *Initial* Dynamic Threshold: 14 votes
     ✓ The First Proposal should be rejected
Second Proposal *Initial* Dynamic Threshold: 14 votes
Proposer Reputation After Second Proposal: 0.5
     ✓ The Second Proposal should be rejected to trigger a reputa
tion penalty
Third Proposal *Initial* Dynamic Threshold: 15 votes
✓ Proposer token balance increased by: 1.0 tokens
Proposer Reputation After Third Proposal: 1.0
     ✓ The Third Proposal should be confirmed and reward the prop
oser
```

Figure 6.2: *Excerpt from the Hardhat simulation logs*

tests is publicly available in the GitHub repository [121]. All experimentations and validations confirmed the practical feasibility and functionality of the designed system architecture.

6.6 Conclusions

This chapter presents a novel decentralized, blockchain-based framework designed to facilitate the secure and trustworthy exchange of sensor data among autonomous vehicles produced by diverse manufacturers. By employing a carefully designed modular smart contract architecture (comprising distinct filtering, consensus voting, and reward modules) long-standing challenges related to interoperability, validation, and incentivization in Intelligent Transportation Systems are effectively addressed. Specifically, our proposed approach leverages smart contracts, reputation-based voting mechanisms, and blockchain-based tokenization for economically incentivizing authenticated real-time data exchanges, thus significantly enhancing cross-brand cooperation.

The key of the proposed contribution is the *dynamic threshold model*, formulated to support real-time decision-making processes. The model incorporates multiple parameters, including event severity, road segment criticality, elapsed time, and submitter reputation, thereby enabling agile responses to critical infrastructural changes while mitigating adversarial input from malicious entities. A simulation using Hardhat, a widely adopted smart contract testing platform, was conducted to evaluate the approach's effectiveness and practical performance rigorously.

Simulation results validate the efficacy and robustness of our proposed framework. Specifically, the implemented voting mechanism efficiently adapted the consensus conditions to varying contextual parameters, accurately adjusting acceptance thresholds. Furthermore, the smart contracts adequately managed proposal processing, validation, and reward token issuing, underscoring the system's operational feasibility and potential scalability.

Future work will implement realistic traffic simulations and optimize threshold coefficients through adaptive multi-objective optimization approaches.

7 From Identity to Incentives: Design Patterns for Trust in Mobility CPS

The architectural patterns emerging from the integration of cryptographic identification mechanisms with blockchain-based incentivization reveal fundamental principles for establishing trust in distributed cyber-physical systems that operate without central coordination. The works presented in Chapters 5 and 6 transcends their individual technical contributions, crystallizing into a compositional framework where mathematical certainty at the data layer enables economic coordination at the application layer, ultimately yielding emergent trust properties that exceed the sum of constituent guarantees. This architectural evolution establishes a methodological template applicable across diverse cyber-physical domains where multiple stakeholders must coordinate despite potentially misaligned interests and the absence of pre-established trust relationships.

The theoretical contribution extends beyond mechanical integration of existing technologies, elucidating instead the fundamental principles governing the transformation of local observations into globally trusted intelligence through systematic layering of cryptographic, consensus, and game-theoretic mechanisms. The architecture that emerges from this analysis provides a structured approach to decomposing complex trust requirements into tractable subproblems, each addressed at the appropriate abstraction level while maintaining compositional guarantees that propagate throughout the system. This chapter examines how the convergence of deterministic identification with economic incentivization creates a self-reinforcing trust dynamic, analyzes the complete data lifecycle from detection through dissemination, synthesizes these patterns into a generalizable architectural framework, and demonstrates the transferability of these design patterns to broader cyber-physical system contexts beyond the mobility domain.

7.1 Architectural Convergence: From Deterministic Identification to Distributed Trust

7.1.1 The Foundational Role of Deterministic Identification

The transformation of physical infrastructure characteristics into cryptographic commitments establishes the fundamental substrate upon which distributed trust mechanisms operate, creating an invariant namespace that transcends the ambiguities inherent in conventional spatial referencing systems. This deterministic transformation, achieved through the systematic application of cryptographic hashing to canonicalized data representations, provides the mathematical certainty required for consensus mechanisms to function effectively across heterogeneous vehicular networks where participants lack pre-established trust relationships. The significance of this architectural choice extends beyond mere technical elegance, fundamentally reshaping how distributed systems reason about spatial identity and enabling consensus protocols that would otherwise founder on referential ambiguity.

The hierarchical structuring through Merkle tree constructions introduces a critical capability for granular verification that proves instrumental when environmental conditions, sensor limitations, or temporal variations prevent complete data agreement among observing vehicles. This compositional approach acknowledges the fundamental tension between the desire for complete consensus and the practical realities of distributed sensing, where legitimate observational differences must be distinguished from adversarial manipulation. The mathematical properties of the chosen cryptographic primitives, particularly their collision resistance and one-way characteristics, provide the foundation upon which higher-level protocols establish trust guarantees that propagate throughout the entire system architecture.

7.1.2 From Local Hashing to Global Consensus

The architectural evolution from locally computed identifiers to globally validated consensus represents a fundamental phase transition in system capabilities, bridging the gap between individual vehicle observations and collective network intelligence through blockchain-mediated coordination mechanisms. The integration of hash-based identification with smart contract validation creates a unified framework where spatial references achieve both local precision and global verifiability, enabling multiple vehicles to collaboratively establish ground truth without recourse to centralized authorities. This convergence manifests through the natural alignment

between Merkle tree structures and blockchain validation requirements, where cryptographic commitments generated at the edge seamlessly integrate with consensus protocols executing within the distributed ledger.

The modular smart contract architecture leverages deterministic identification to implement sophisticated deduplication and aggregation mechanisms that would be computationally infeasible with probabilistic location references. When multiple vehicles independently observe identical infrastructure states, their submissions converge to the same cryptographic fingerprint despite originating from different observers, enabling the system to aggregate corroborating evidence while maintaining precise spatial attribution. This architectural synergy between deterministic identification and blockchain consensus creates emergent properties that exceed the capabilities of either component in isolation, establishing a foundation for trusted information dissemination across heterogeneous vehicular networks.

7.1.3 Trust Amplification through Layered Verification

The progression from simple threshold validation to sophisticated reputation-modulated consensus exemplifies how architectural layering amplifies trust guarantees through the systematic composition of verification mechanisms operating at different abstraction levels. The evolution from static two-thirds thresholds to dynamic acceptance criteria that incorporate temporal urgency, event criticality, and participant reputation demonstrates how simple cryptographic primitives at the foundation enable complex trust dynamics at higher architectural layers. This amplification effect emerges through the careful orchestration of complementary mechanisms: cryptographic commitments provide mathematical certainty, hierarchical verification enables partial consensus despite incomplete agreement, and economic incentives align individual behavior with collective truth discovery.

The reputation system's integration with the underlying deterministic identification framework creates a positive feedback loop where consistent accuracy in reporting verifiable events accumulates long-term value that transcends individual transactions. This architectural pattern, wherein cryptographic certainty at the data layer enables sophisticated trust mechanisms at the application layer, establishes a generalizable template for constructing resilient cyber-physical systems. The synthesis of these layered verification mechanisms creates a trust amplification effect whereby system reliability exceeds the sum of component reliabilities, transforming uncertain local observations into globally trusted intelligence through the systematic application of mathematical, economic, and game-theoretic principles that operate in concert rather than isolation.

7.2 Data Lifecycle: From Detection to Dissemination

The transformation of raw vehicular sensor observations into trusted, network-wide intelligence unfolds through six distinct yet interdependent phases that collectively ensure information integrity while maintaining system responsiveness. This lifecycle (encompassing detection, cryptographic fingerprinting, proposal submission, distributed validation, economic incentivization, and network-wide dissemination) represents a carefully orchestrated pipeline wherein each phase contributes specific guarantees while building upon properties established by its predecessors. The architectural pattern that emerges from this sequential processing creates a compositional framework where trust accumulates progressively, transforming uncertain local observations into globally accepted knowledge through the systematic application of cryptographic, game-theoretic, and distributed systems principles.

7.2.1 Detection and Canonicalization Phase

The initiation of the data lifecycle occurs at the precise moment when vehicular sensor arrays detect discrepancies between observed physical infrastructure and their internally maintained navigational baselines, triggering a capture protocol that must balance comprehensiveness with computational efficiency. The multimodal sensor fusion process, integrating LiDAR point clouds with visual imagery and radar signatures, produces a high-dimensional representation that undergoes dimensionality reduction through feature extraction algorithms specifically tuned to preserve those characteristics most salient for intersection identification. This reduction process, far from being merely a compression technique, embodies domain-specific knowledge about which infrastructure attributes exhibit temporal stability versus those subject to frequent variation, thereby ensuring that the resulting representation captures the essence of the intersection while remaining robust to transient environmental conditions.

The transformation of this feature-rich representation into a canonicalized form represents a critical juncture where implementation-specific variations must yield to protocol-level standardization. The canonicalization pipeline operates through a sequence of normalization operations that systematically eliminate sources of representational ambiguity: coordinate systems undergo transformation to a universal reference frame, numerical values are rounded to protocol-specified precision levels, and optional fields are either populated with default values or excluded according to deterministic rules. This process culminates in the generation of a byte sequence that serves as input to the cryptographic hashing operation, producing the deterministic fingerprint that will identify this specific

infrastructure state throughout all subsequent processing phases.

The temporal dynamics of the detection phase introduce subtle complexities that influence the entire downstream lifecycle. Vehicles traversing the same intersection within short temporal windows may observe slightly different configurations due to traffic signal state changes, temporary obstructions, or varying illumination conditions affecting sensor performance. The system addresses this inherent variability through a two-tier detection strategy: immediate anomalies that suggest safety-critical infrastructure changes trigger expedited processing paths, while gradual deviations accumulate evidence over multiple observations before initiating the proposal phase. This temporal aggregation mechanism reduces the computational and economic overhead of processing transient variations while maintaining responsiveness to genuine infrastructure modifications.

7.2.2 Proposal and Consensus Mechanisms

The transition from local detection to network-wide proposal marks the evolution from individual observation to collective validation, wherein the detecting vehicle must package its findings into a format that enables meaningful evaluation by other network participants. The proposal transaction encompasses not merely the computed hash value but a comprehensive metadata structure that provides context for validation: the precise timestamp of observation enables other vehicles to account for temporal evolution, the classified severity level influences the urgency of validation efforts, and the proposer's pseudonymous identifier allows the reputation system to modulate trust levels appropriately. This metadata packaging represents a delicate balance between providing sufficient information for informed validation and maintaining privacy guarantees that prevent the tracking of individual vehicle movements.

The smart contract infrastructure orchestrates a sophisticated state machine that governs the proposal's evolution from submission through final determination. Upon receipt, a dedicated smart contract performs structural validation and deduplication, leveraging the deterministic hash values to identify when multiple vehicles have independently detected the same infrastructure modification. This deduplication mechanism operates through a temporal sliding window that aggregates proposals referring to identical hashes, treating them as corroborating evidence rather than independent events, thereby strengthening the statistical confidence in the observation while reducing blockchain storage requirements. The aggregation process must account for the possibility that identical hashes might represent different temporal observations of a recurring phenomenon, necessitating careful examination of timestamp distributions to distinguish between simultaneous

corroboration and sequential re-observation.

The consensus mechanism implements a sophisticated voting protocol that extends beyond simple majority rule to incorporate contextual factors that influence the reliability and urgency of different proposals. Rather than applying uniform validation criteria, the system modulates its acceptance thresholds based on multiple dimensions of context: infrastructure elements along emergency evacuation routes receive expedited validation to ensure rapid dissemination of critical safety information, while cosmetic modifications to low-traffic areas may undergo extended validation periods to accumulate higher statistical confidence. The voting process itself operates through cryptographically signed attestations that bind each validator's assessment to their identity, creating an auditable trail that enables post-hoc analysis of validation patterns and the identification of systematic biases or adversarial behavior. The accumulation of votes proceeds asynchronously, with the smart contract continuously evaluating whether the accumulated evidence has crossed the dynamically determined threshold for acceptance or rejection, enabling rapid convergence for unambiguous cases while allowing extended deliberation for contested observations.

7.2.3 Incentivization and Network Effects

The crystallization of validated proposals into economic rewards represents a phase transition wherein abstract contributions to collective knowledge transform into tangible value within the mobility ecosystem, creating cascading effects that extend far beyond the immediate participants. The temporal dynamics of reward distribution exhibit critical influence on system behavior: immediate token minting upon validation creates strong short-term incentives for rapid proposal submission, yet the introduction of vesting schedules or graduated release mechanisms could encourage longer-term engagement and reduce the likelihood of hit-and-run participation patterns. This temporal structuring of incentives operates in concert with the reputation system to create a multi-dimensional value proposition where immediate economic gains complement long-term reputational capital accumulation.

The emergence of secondary markets for validated information introduces complex dynamics that fundamentally alter the economics of participation. When third-party navigation services, insurance companies, or urban planning authorities express willingness to purchase verified infrastructure updates, the token ecosystem evolves from a closed-loop reward system to an open marketplace where information quality directly translates to economic value. This marketization process creates price discovery mechanisms that reveal the true economic value of different

categories of infrastructure information: safety-critical updates command premium valuations, while routine traffic pattern observations may trade at commodity prices. The stratification of information value creates specialization incentives, potentially leading to the emergence of professional validators who develop expertise in specific geographic regions or infrastructure types, thereby improving overall system quality through division of labor.

The network effects that materialize within this incentivized ecosystem exhibit non-linear growth patterns characterized by tipping points and phase transitions. Initial participation may remain modest until the validated information density reaches a critical threshold where the navigation improvements become perceptible to average users, triggering exponential adoption as word-of-mouth propagation amplifies the perceived benefits. This growth trajectory, however, must contend with potential negative network effects that emerge at scale: as participant numbers increase, the probability of conflicting observations rises, potentially creating validation gridlock where consensus becomes increasingly difficult to achieve. The system must therefore implement adaptive mechanisms that scale consensus requirements with network size while maintaining decision latency within acceptable bounds, potentially through hierarchical validation structures or probabilistic sampling techniques that achieve statistical confidence without requiring universal participation.

The sustainability of the incentive ecosystem depends critically on maintaining equilibrium between token emission through rewards and token consumption through service redemption. Excessive emission without corresponding demand creates inflationary pressure that erodes participation incentives, while insufficient rewards relative to redemption opportunities may create deflationary spirals that concentrate tokens among early participants and discourage new entrants. The system must therefore implement dynamic adjustment mechanisms, potentially governed by algorithmic monetary policy encoded in smart contracts, that modulate reward rates based on participation levels, validation accuracy, and token velocity metrics. These adjustments must occur gradually enough to maintain predictability for participants while remaining responsive to changing market conditions and network growth patterns.

7.2.4 Dissemination and Integration Patterns

The propagation of validated infrastructure updates through the heterogeneous landscape of navigation systems, vehicle control platforms, and urban management infrastructure necessitates sophisticated translation and adaptation mechanisms that preserve semantic integrity while accommodating diverse technical requirements.

The initial broadcast phase leverages the inherent properties of blockchain systems where confirmed transactions become visible to all network observers, yet this passive visibility must transform into active integration through purpose-built middleware that monitors blockchain events, extracts relevant updates, and reformats them according to recipient-specific schemas. This transformation pipeline must maintain cryptographic proofs of validation throughout the translation process, enabling downstream consumers to independently verify the authenticity and consensus status of received updates without requiring direct blockchain interaction.

The temporal coherence of disseminated information presents particular challenges when dealing with systems operating at different update frequencies and exhibiting varying tolerance for staleness. High-frequency trading algorithms optimizing delivery routes may require sub-second update latency, while municipal planning systems might aggregate weekly infrastructure changes for batch processing. The dissemination architecture must therefore implement multi-resolution temporal views that enable consumers to subscribe to update streams matching their specific temporal requirements: real-time event streams for safety-critical applications, aggregated summaries for planning purposes, and historical archives for trend analysis. This temporal multiplexing requires sophisticated caching and aggregation mechanisms that maintain consistency across different temporal resolutions while minimizing redundant processing and storage overhead.

The integration patterns that emerge from successful deployments reveal a consistent evolution from peripheral augmentation toward core system integration. Initial adoptions typically manifest as overlay systems that supplement existing navigation databases with blockchain-validated updates, maintaining clear separation between traditional trusted sources and crowd-sourced information. As confidence in the validation mechanism grows through empirical verification of accuracy, the integration deepens: validated updates begin influencing routing algorithms, then inform predictive models, and eventually achieve parity with traditional data sources in terms of trust and authority. This evolutionary trajectory necessitates careful versioning and rollback mechanisms that enable graceful degradation when validated updates prove erroneous, maintaining system resilience while encouraging experimentation and adoption.

The establishment of feedback mechanisms that close the loop between information consumers and producers creates opportunities for continuous refinement of both data quality and dissemination efficiency. When navigation systems encounter discrepancies between validated updates and ground truth, their reports flow back through the system as new proposals or challenges to existing information, creating a self-correcting dynamic that improves accuracy over time. These feedback

loops operate across multiple timescales: immediate corrections for critical errors, daily reconciliation for minor discrepancies, and long-term pattern analysis that identifies systematic biases or recurring failure modes. The aggregation of feedback across multiple consumers provides statistical power to distinguish between isolated anomalies and genuine validation failures, enabling the system to adapt its validation parameters and improve its resilience against both innocent errors and adversarial manipulation. Furthermore, the patterns observed in feedback data inform the evolution of the canonicalization and validation protocols themselves, creating a meta-learning dynamic where the system progressively refines its ability to distinguish signal from noise in the complex, ever-changing landscape of urban infrastructure.

7.3 Architectural Synthesis: The Three-Layer Trust Framework

The convergence of cryptographic identification mechanisms and blockchain-based incentivization strategies crystallizes into a stratified architectural framework that transcends the immediate application domain of autonomous vehicle navigation. This three-layer architecture emerges as a natural consequence of separating concerns between cryptographic guarantees, economic coordination, and user-facing functionality, establishing a generalizable pattern for constructing trustworthy cyber-physical systems where multiple stakeholders coordinate without centralized authority.

7.3.1 Layer 1: Cryptographic Identity and Immutability

The foundational stratum establishes the cryptographic substrate upon which all trust guarantees ultimately rest, transforming physical-world observations into mathematically verifiable commitments that resist both temporal degradation and adversarial manipulation. Within this layer, the deterministic transformation of intersection descriptors into fingerprints through cryptographic hashing creates an invariant namespace where spatial references achieve mathematical precision, while the hierarchical organization through Merkle structures enables granular verification capabilities essential for partial consensus scenarios. The blockchain's append-only ledger extends these guarantees temporally, creating a distributed state machine where the computational cost of historical revisionism grows exponentially with confirmation depth.

The critical contribution of this layer extends beyond mere data integrity to establish what constitutes a universal source of truth accessible to all network participants regardless of their computational capabilities or trust relationships. The canonicalization protocols ensure that heterogeneous implementations achieve consensus not through administrative fiat but through deterministic algorithms that produce identical outputs given identical inputs, thereby eliminating the coordination ambiguities that plague systems relying on natural language descriptions or probabilistic identifiers. This foundational certainty propagates upward through the architecture, enabling higher layers to reason about data authenticity without reimplementing verification logic.

7.3.2 Layer 2: Distributed Validation and Economic Alignment

The intermediate layer transforms cryptographically secured data into economically validated information through sophisticated orchestration of consensus mechanisms and incentive structures that align individual profit maximization with collective truth discovery. Rather than imposing uniform validation criteria, the smart contract infrastructure implements context-aware consensus that modulates acceptance thresholds based on event criticality, temporal urgency, and participant reputation, creating a responsive system that adapts to varying operational requirements while maintaining resistance to manipulation.

The economic mechanisms operating at this layer establish a repeated game structure where truthful reporting emerges as the dominant strategy through careful calibration of rewards and penalties that account for both immediate payoffs and long-term reputational consequences. The separation between reputation tracking and token management enables independent evolution of social and economic incentives, preventing plutocratic capture while still differentiating between consistently reliable contributors and sporadic participants. This layer effectively mediates between the mathematical certainty of cryptographic commitments and the probabilistic nature of real-world observations, creating a bridge where Byzantine fault tolerance meets practical sensor uncertainties.

The validation orchestration extends beyond simple voting to encompass sophisticated state management that tracks proposal lifecycles, aggregates corroborating evidence, and manages the temporal dynamics of consensus formation. The smart contract architecture provides the computational substrate for these complex interactions while maintaining transparency and auditability, ensuring that validation decisions can be retrospectively examined and systematically improved based on empirical outcomes.

7.3.3 Layer 3: Application Integration and User Experience

The uppermost layer mediates between the blockchain-based trust infrastructure and the diverse ecosystem of applications that consume validated mobility data, abstracting away protocol complexity while preserving cryptographic guarantees. Through carefully designed APIs and middleware services, traditional navigation systems integrate with the trust framework without requiring architectural overhauls, enabling incremental adoption paths that respect existing technology investments while progressively enhancing data reliability.

The token redemption mechanisms implemented at this layer establish tangible value propositions that sustain long-term engagement, creating closed-loop eco-

nomic cycles where earned tokens translate into operational benefits that directly enhance participant welfare. These redemption pathways extend beyond simple monetary exchanges to encompass priority services, computational resources, and information access privileges that create multi-dimensional value beyond immediate economic gains. The careful design of these economic loops, with appropriate controls on emission and consumption rates, maintains system sustainability while fostering network effects that amplify participation value as the ecosystem grows.

The dissemination protocols operating at this layer acknowledge the fundamental tension between immediacy and accuracy, implementing probabilistic propagation schemes that enable risk-adjusted incorporation of validated updates based on specific operational requirements. High-confidence information spreads rapidly through priority channels, while contested observations undergo extended validation before integration, allowing individual systems to calibrate their trust-latency trade-offs according to their specific safety requirements and operational constraints. This graduated approach to information integration enables the framework to simultaneously serve applications with diverse temporal requirements and risk tolerances, from real-time collision avoidance systems demanding immediate updates to urban planning tools prioritizing comprehensive accuracy over response speed.

7.4 Generalization to Cyber-Physical Systems

The architectural patterns developed throughout this chapter exhibit fundamental properties that transcend the specific context of vehicular networks, offering a methodological template for establishing trust in diverse cyber-physical domains where distributed entities must coordinate without central authorities. Progressing from cryptographic identity through distributed validation to application integration, the methodology proposed addresses challenges inherent to any system requiring verifiable data exchange among heterogeneous participants with potentially misaligned incentives.

7.4.1 Transferable Design Patterns

The compositional architecture presented herein manifests applicability across multiple cyber-physical domains through three core abstractions that remain invariant despite contextual variations. First, the deterministic transformation of physical-world states into cryptographic commitments provides unambiguous referencing capabilities essential in any domain requiring precise identification—from smart grid prosumer contributions to industrial IoT production stages. Second, the separation between reputation accumulation and economic rewards enables flexible adaptation to domains with varying trust dynamics, whether rapid-turnover emergency response systems or long-term infrastructure monitoring networks. Third, the modular smart contract design, decomposing complex workflows into filtering, validation, and incentivization components, facilitates domain-specific customization while preserving the overall trust guarantees.

Smart grid applications exemplify this transferability through direct architectural correspondence: distributed energy resources require deterministic identification analogous to intersection fingerprinting, prosumer energy contributions demand validation mechanisms similar to vehicular data proposals, and grid stability concerns map naturally to our severity-based threshold modulation. The hierarchical validation enabled by Merkle structures proves particularly valuable when aggregating measurements from heterogeneous smart meters with varying precision and reliability characteristics. Similarly, supply chain management systems benefit from the same cryptographic audit trails, where production milestones generate verifiable commitments that accumulate into comprehensive provenance records, while the dynamic consensus thresholds naturally accommodate varying criticality levels from pharmaceutical cold chains to consumer goods logistics.

The double-hit penalty mechanism, balancing tolerance for honest errors against adversarial deterrence, generalizes to any cyber-physical system where

sensor inaccuracies coexist with potential manipulation attempts. Environmental monitoring networks, for instance, must distinguish between calibration drift in low-cost sensors and deliberate misreporting of pollution levels, a challenge structurally identical to our vehicular scenario. The economic incentivization layer similarly translates across domains: citizen science initiatives benefit from token rewards for validated observations, while industrial predictive maintenance systems could implement internal markets where equipment operators earn credits for accurate failure predictions, redeemable for priority maintenance scheduling or resource allocation.

7.4.2 Scalability Considerations and Trade-offs

The application of our framework to diverse cyber-physical domains reveals fundamental tensions between architectural choices that manifest differently across deployment scales and operational constraints. The granularity of consensus—exemplified by our two-thirds threshold for partial validation—must adapt to domain-specific reliability requirements: medical device networks might demand near-unanimous consensus given patient safety implications, while crowd-sourced weather observations could operate with simple majority validation given their non-critical nature. This calibration extends beyond static threshold adjustment to encompass the temporal dynamics of consensus formation, where real-time control systems cannot accommodate the extended voting periods acceptable in urban planning applications.

The choice between on-chain data storage and hash-only notarization becomes particularly acute when considering domains with vastly different data generation rates. While vehicular networks produce manageable volumes of intersection observations, continuous industrial process monitoring or high-frequency trading systems would rapidly saturate any blockchain's capacity if attempting full on-chain storage. Layer-two solutions, such as state channels for high-frequency bilateral exchanges or optimistic rollups for batched validation, become essential architectural components rather than optional optimizations. The economic implications compound this technical challenge: domains with thin profit margins cannot sustain the transaction fees associated with frequent on-chain operations, necessitating careful analysis of which commitments truly require blockchain immutability versus those manageable through traditional databases with periodic blockchain checkpointing.

Cross-domain interoperability introduces additional scalability challenges absent from single-domain deployments. When smart city systems attempt to correlate vehicular data with energy consumption patterns and air quality mea-

surements, the heterogeneous nature of these data streams—varying in frequency, format, and validation requirements—demands sophisticated middleware that can maintain cryptographic guarantees while performing necessary transformations. The reputation portability across domains presents another complex trade-off: while unified reputation scores enable participants to leverage their credibility across multiple systems, domain-specific expertise may be lost in aggregation, potentially allowing actors with strong reputation in one domain to make unreliable contributions in another where their expertise does not transfer.

7.4.3 Architectural Requirements for Domain Adaptation

The successful instantiation of our framework in new cyber-physical domains necessitates careful consideration of domain-specific characteristics that influence architectural choices while preserving core trust guarantees. The identification mechanism must accommodate the inherent identifiability of domain entities: while road intersections possess stable physical characteristics amenable to deterministic hashing, phenomena such as air quality or electromagnetic interference exhibit continuous spatial variation requiring alternative approaches, potentially combining geographic tessellation with temporal aggregation to create identifiable observation units. The consensus mechanism similarly requires domain-aware configuration, accounting for factors such as the natural update frequency of observed phenomena, the availability and distribution of validators, and the consequences of false positives versus false negatives in validation decisions.

The incentive structure must align with domain-specific value creation and capture dynamics. Domains with clear beneficiaries of improved data quality, such as insurance companies utilizing verified driver behavior data, can sustain market-based token economies where consumers purchase information from producers. Conversely, public goods domains such as disaster response may require subsidized incentive mechanisms funded through governmental or philanthropic sources, fundamentally altering the economic sustainability model. The temporal dynamics of value realization also vary significantly: immediate benefits in navigation optimization contrast with long-term value in climate monitoring, necessitating different approaches to token vesting and reputation accumulation that maintain participant engagement across these varying timescales.

The integration requirements at the application layer depend critically on existing infrastructure maturity and regulatory constraints within target domains. Healthcare applications must accommodate HIPAA compliance while maintaining cryptographic verifiability, potentially requiring zero-knowledge proofs to validate data properties without revealing protected information. Critical infrastructure

domains may mandate permissioned blockchain deployments for regulatory compliance, sacrificing some decentralization benefits for auditability and accountability. These adaptations, while necessary for practical deployment, must be carefully designed to preserve the fundamental trust properties that motivate blockchain adoption, avoiding architectural compromises that inadvertently reintroduce the centralization vulnerabilities the framework seeks to eliminate.

Part IV

Control and Optimization in Smart Urban Systems

8 A Deep Reinforcement Learning Approach for Route Planning of Autonomous Vehicles

8.1 Introduction

With the evolution of the urban autonomous driving, the choice of personalized and optimal routes is emerging as a crucial element in the context of roadway planning. The customized routes is based on the dynamic adaptation of pathways in response to urban context variables, such as traffic and user preferences. Optimizing routes and travel times is a way to reduce congestion, decrease gas emissions and contribute to global efforts to mitigate climate change [122], [123].

In the recent years, the research and innovation areas focus on Cooperative, Connected and Automated Mobility topics and services for managing and controlling autonomous driving systems. In particular, the related literature is large so that only the most relevant studies are cited here. For instance, some novel contributions enlighten that the most modern technologies based on Information and communications technologies, such as Connected and Cooperative Services, Artificial Intelligence and Big Data allow to connect users, vehicles and infrastructures in an intelligent, efficient, safe and sustainable manner [124–126].

In the route planning, Ma et al. [127] propose a graph convolutional network-based multi-objective meta-deep Q-learning method to efficiently learn a dynamically changing signalized traffic network and automatically explore eco-routes based on drivers' preferences for travel time and fuel consumption.

Sharma et al. [128] propose a Graph Neural Network based approach for real-time estimation of traffic speed in sustainable smart cities, which has important implications for route planning applications. The authors develop a novel Spatio-Temporal Gated Graph Attention Network model that captures both spatial dependencies and temporal dynamics within the road network graph structure.

Efficiently matching trip requests and available drivers are central operational problems, as pointed out by Qin et al. [129]. The authors propose a Reinforcement Learning (RL) based approach to tackle the challenge of finding an optimal delayed matching policy in a complex ride-hailing environment, where the efficiency of matching can be substantially improved by adaptively adjusting the matching time

interval.

In a comprehensive systematic literature review, Teusch et al. [130] provide an in-depth exploration of the diverse applications of Machine Learning (ML) techniques in Shared Mobility Systems (SMS), offering invaluable insights for service providers seeking to optimize their decision-making processes and enhance daily operations. The review highlights the transformative potential of ML as a powerful methodological solution to tackle specific management challenges that are pivotal for the efficient and effective functioning of SMS.

Amarnath et al. [131] propose a novel ML-based approach to enhance transportation services through route-based user segmentation and clustering. The method harnesses real-time location data and users' route preferences to dynamically group travelers sharing common routes, facilitating efficient communication and information sharing during transit.

Moreover, some works are oriented to optimize only specific features of a scenario or finding the shortest path. For instance, Chen et al. [132] propose a generic bi-criteria optimum path-finding framework based on deep reinforcement learning (DRL). Simulations are performed to verify the effectiveness of the proposed approach, where two criteria (e.g., solar radiation and crime risk) are modeled based on the real-world data in downtown New York.

Indeed, DRL is a promising solution especially in the domains of driving policy, predictive perception, path and motion planning, and low level controller design. Reinforcement learning is still an active and emerging area in real-world autonomous driving applications. Although there are a few successful commercial applications, there is very little literature or large-scale public datasets available. Open issues will include: validating the performance of RL based systems, the simulation-reality gap, sample efficiency, designing good reward functions, incorporating safety into decision making RL systems for autonomous agents.

The development of explicitly multi-agent DRL approaches to the autonomous driving problem is an important future challenge that has not received a lot of attention [133]. Some recent studies use DRL to train a driving agent by using Light Detection and Ranging (Lidar), and a camera sensor. E.g., a study by Shafique et al. [134] uses Lidar for path tracking and a combination of Lidar and a camera sensor for obstacle avoidance. On the contrary, Durgabhavani et al. [135] proposes an adaptive path planning approach of Autonomous Vehicle (AV) established multi-light trained RL, aimed to enhance the fuel cost as well as AV comfort. However, a path planning that seeks to limit safety problems for an AV during its journey, such as avoiding left/right turns or using priority lanes, has not been proposed.

This chapter aims to address these gaps and considers the basic problem of

designing suitable routes in the cities for AVs by focusing on multi objectives that are the length of the routes, the requirements of minimizing the turns during the travel and the selection of dedicated lanes. The problem is solved by applying a modular DRL model based on the training of agents associated with the AVs. Since, considering all the possible routes of the cities for completing the training is a very time consuming and complex task, we address such issue by proposing a modular DRL architecture based on the division of the city in a set of zones. A set of agents is trained and each agent is associated to routes starting from a zone and ending to a different one. The AV will select the route that exhibits the best multi-objective strategy enlightened by the best value of the reward. The modular approach allows us to train the agents in parallel and in limited urban areas.

The proposed modular DRL based strategy is applied to the city center of Bari, a town of Southern Italy and the agents are trained in a simulation environment. Some results show the advantages of the proposed methodology. Naturally, if the city of study changes a new agent training is needed to take in account different characteristic as traffic patterns and road network.

The rest of the chapter is organized as follows. Section II presents the formulation of the problem and Section III introduces the used deep reinforcement learning model. Moreover, Section IV describes the case study and the obtained results. Finally, Section VI draws conclusions and future works.

8.2 Problem Formulation

Let us consider an AV that has to travel in the city: the problem is to find the optimal path p starting from a generic point of the city to a final point with the aim of minimizing the length of the route, minimizing the number of right and left turns and using the lanes dedicated to the AVs.

The road network is modeled as a graph $G(J, E)$ where the set of nodes $J = \{j | j = 1, ..., n\}$ is the set of junctions or intersections and the set of edges $E = \{e_1, e_2, ...e_m\}$ denotes the set of the city streets. In particular, it holds that an edge $e_i = (j, r) \in E$ if there exists a street starting from $j \in J$ and ending to $r \in J$. Moreover, we consider a set of edges $E_p \subset E$ that are priority edges, i.e., they correspond to streets dedicated to the AVs. We consider the problem of determining the route that connects a pair of edges $c = (e_s, e_f)$ where e_s is the starting edge and e_f is the ending edge. A possible route connecting the pair $c = (e_s, e_f)$ is a path p represented by a sequence of edges $p = (e_s, ...e_j, .., e_f)$. The problem to be solved for a given pair $c = (e_s, e_f)$, is determining the optimum path exhibiting the minimum path length, the minimum number of right/left turns crossed and the maximum number of priority traveled routes.

8.3 Deep Reinforcement Learning Model

The Markov decision processes (MDPs) are considered the standard when formalising sequential decision making problems involving a single RL agent. A MDP is defined as a five-tuple

$$< S, A, T, R, \gamma >$$

where S is the set of states, A is the set of actions which could change the status, T is the transition function, which is the probability of the state change under the certain action, R is the reward function, and γ is known as the discount factor, which models the importance of the future and immediate rewards.

We model the route planner problem as a MDP, where the agents follow a policy $\pi\,(a|s)$ in a predetermined environment. More specifically, at each time step t, given the current state $s(t) \in S$, each agent chooses an action $a(t) \in \mathcal{A}$, according to the current policy, transits to the next state $s(t+1)$ and finally receives a reward $r(t) \in \mathbb{R}$. The agent purpose is to maximize the expectation of the return over time that is called the discounted cumulative reward and is defined as $G(s(t)) = \sum_{h=0}^{\infty} \gamma^h r(t+h+1)$, where $\gamma \in [0, 1]$ is the discount factor.

For a given couple of source and destination edges $c = (e_s, e_f)$, in the proposed system an RL agent is associated with the AV that chooses the best action to obtain the minimum distance, with the minimum number of right/left turns and the maximum number of priority edges.

8.3.1 State Space and Action Space

The state of the agent $s(t) \in S$ at time t is the following:

$$s(t) = [x(t), \phi_v(t), e_(t)]$$

where $x(t) = (Lat(t), Lon(t))$ denotes the latitude $Lat(t)$ and the longitude $Lon(t)$ of the position of the AV at time t respectively, $\phi_v(t)$ represents the heading angle of the AV (i.e., the angle of the AV, going clockwise with 0 at the 12'o clock position) and $e(t) \in E$ is the edge occupied by the AV at time t.

When an AV arrives to a node $j \in J$ at time t, it can change the edge and can decide among three possible actions: $A = \{Gostraight, Turnright, Turnleft\}$.

8.3.2 Multi-objective Reward function

We define a Multi-objective Reward function composed by a reward for shortest path, by left and right turns and finally a reward to use a priority path for the AV.

8.3.2.1 Shortest path

To describe a reward that show how the agent moves towards the destination edge, we define the distance $d(t)$ between the current edge at time t, and the destination edge e_f. In case of the current distance $d(t)$ is less than the distance $d(t-1)$ at time $t-1$, then it means that the AV moves towards destination, instead if $d(t)$ is higher that the distance $d(t-1)$, then the agent moves far. The reward function is defined as follows:

$$r_{shortest}(t) = \begin{cases} 1 & \text{if } d(t) < d(t-1) \\ -0.5 & \text{otherwise.} \end{cases} \tag{8.1}$$

8.3.2.2 Avoiding left/right turns

If the modulus of the difference between the angle at time t $\phi(t)$ and the angle $\phi(t+1)$ at time $t+1$ is greater than a fixed angle ϕ_{fixed}, then we suppose that the AV turns. In such a case the agent is penalized:

$$r_{TlTr}(t) = \begin{cases} -1 & \text{if } |\phi(t) - \phi(t-1)| \geq \phi_{fixed} \\ 0 & \text{otherwise.} \end{cases} \tag{8.2}$$

8.3.2.3 Priority lanes

For considering the priority lanes, it is enough to define a positive reward if at the step t, the current edge $e(t) \in E_p$:

$$r_{priority}(t) = \begin{cases} 1 & \text{if } e(t) \in E_p \\ 0 & \text{otherwise.} \end{cases} \tag{8.3}$$

8.3.2.4 Multi-objective reward

A multi-objective reward function r_{tot} at each time t is formulated as follows:

$$r_{tot}(t) = \alpha r_{shortest} + \beta r_{TlTr}(t) + \gamma r_{priority}(t) \tag{8.4}$$

where α, β and γ are the weights assigned to each reward component.

8.3.3 The Modular DRL Architecture

The standard training of the DRL model should consider all possible pairs of (e_s, e_f) so as to cover the considered map. Such exhaustive approach is depicted in Fig. 8.1. However, when the cardinality of set E is high, i.e. , when it is necessary to deal with big maps with thousand of edges, this approach becomes infeasible due to the very long training times and high variance of the resulting neural network.

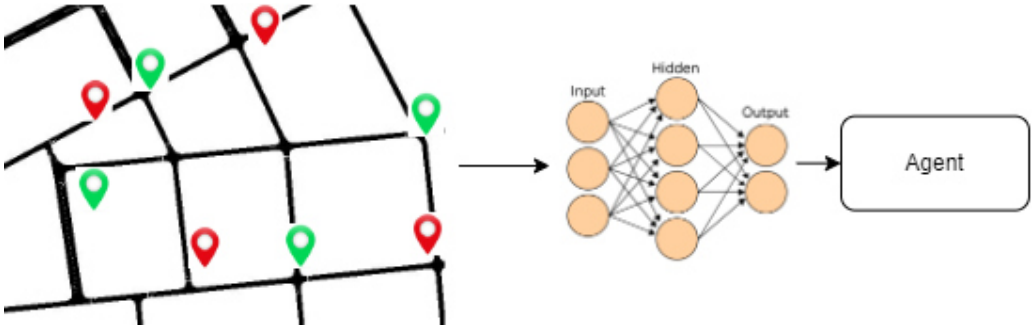


Figure 8.1: *DRL training considering all possible pairs of* (e_s, e_f)

In this chapter, we propose a modular architecture approach in which the set E is partitioned in N subsets such that $E = E_1 \cup E_2 \cup ... E_N$ and $E_i \cap E_j = \emptyset \quad \forall i, j = 1, 2, \ldots, N$ and $i \neq j$. Then, as Fig. 8.2 shows, we train N agents in a parallel fashion, each with a randomly selected pair $c_i = (e_s, e_f)$, where $e_s \in E_i$ and $e_f \in E_j \quad \forall i, j = 1, \ldots, N$.

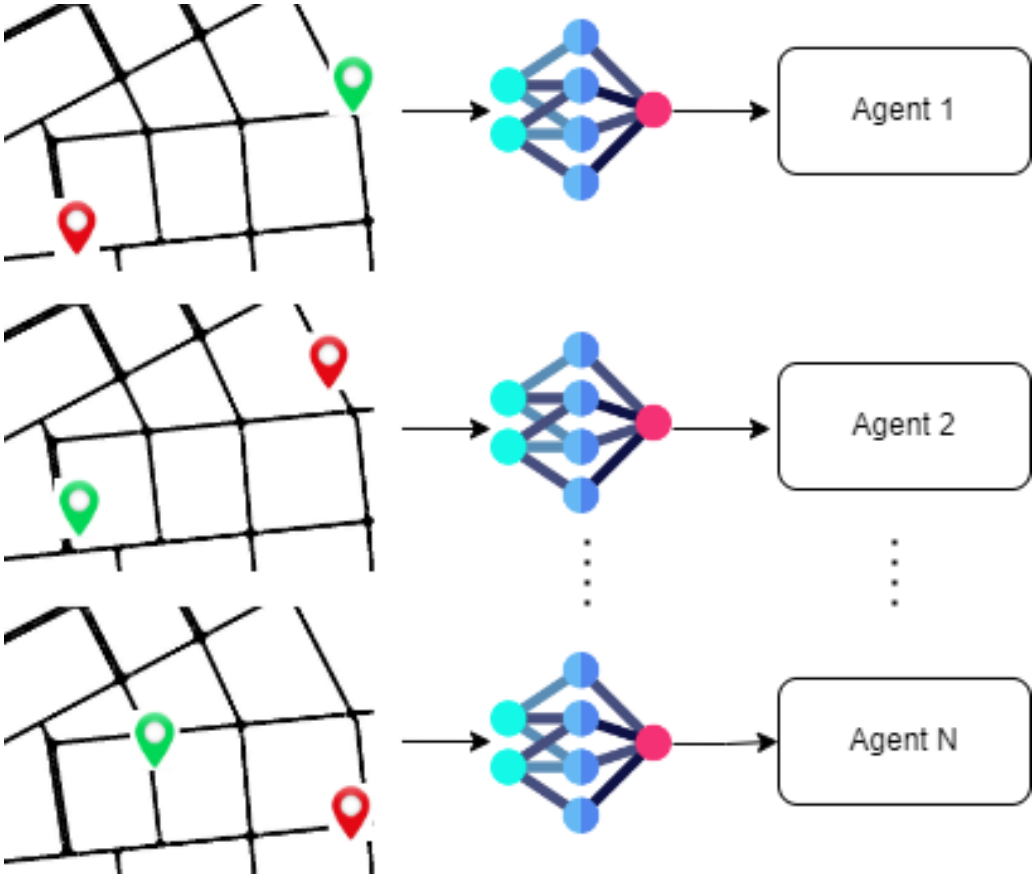


Figure 8.2: *Modular DRL training*

At the end of the training, we obtain N agents that can be used at runtime to calculate N suboptimal paths for a specific pair of $c_i = (e_s, e_f)$ requested by an AV. Each agent determines a suboptimal path p_i^k that is associated with a reward r_k, with $k = 1, ..., N$. At this point the proposed strategy chooses the path p_i^{max} to which is associated the highest reward r_i^{max}. Hence, the obtained optimal solution the path p_i^{max} is returned to the AV. The modular DRL-based route planner scheme is depicted in Fig. 8.3. Note that the agents do not communicate with each other since it is the system manger that select the best route.

In contrast to the exhaustive approach, the advantage of the proposed modular strategy is that the main problem is split in N smaller sub-problems that cover all different zones of the map and can be trained in parallel to reduce complexity of the resulting neural networks and reducing training time. The number N of the considered subsets should be chosen according to the size of the map.

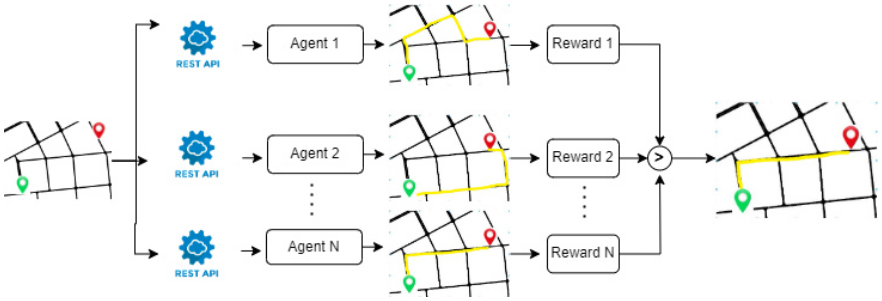


Figure 8.3: *Modular DRL-based route planner.*

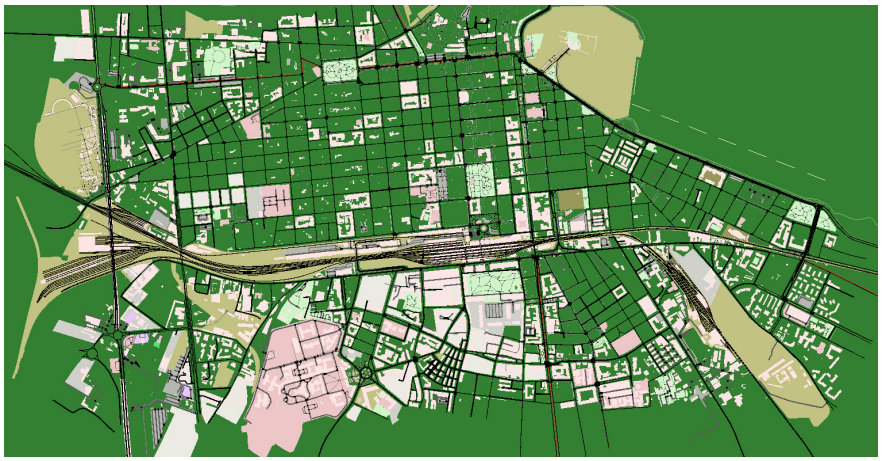


Figure 8.4: *Map of city centre of Bari, Italy.*

8.4 Case Study

In this section, we implement the proposed modular DRL-based strategy to calculate optimal routes for AVs operating in the city center of Bari (Italy).

8.4.1 SUMO Simulation Environment

We consider a map of the city centre of Bari extracted from *OpenStreetMap*. We import the map in *SUMO* [136] Simulator obtaining a graph network of 3426 edges. Each edge comes with several spatial information such as road segment type, allowed maximum speed limits, the number of lanes and the number of traffic signs. The map is depicted in Fig. 8.4.

The edge set E of the graph modeling the city center is partitioned in $N = 4$ subsets E_i with $i = 1, ..., 4$.. We use a standard passenger car to emulate an AV and we place random traffic on the map to make the simulation more realistic.

The modular DRL training is implemented in *Python* language using the *RLLib* [137] libraries. The *Python* script is continuously connected to the *SUMO* environment by the *TraCi* interface. After the training, we use a *REST-Api* that exposes the trained agents and allows the AV to dispatch a route planning request.

We run all the simulations by a personal computer equipped with *Intel(R) Core(TM) i7-8565U* CPU and 32 GB of RAM.

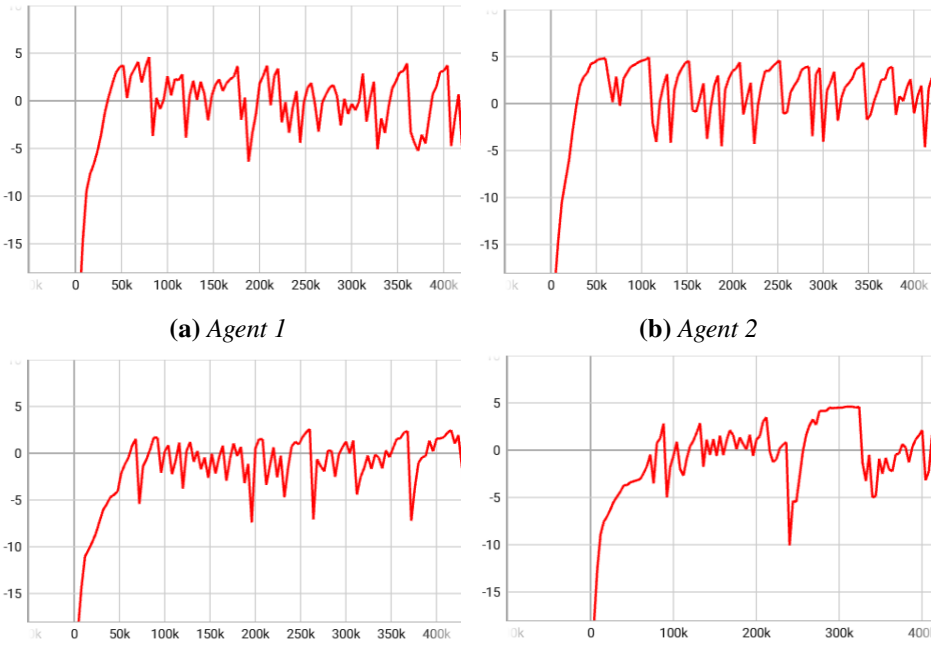
8.4.2 Modular DRL Training Results

A *PPO-Clip* DRL algorithm provided by *RLLib* is used to train the agents. In particular, the discount factor is set to 0.99, the clip parameter to 0.2 and learning rate to 2.5e-4.

Four random pairs of source and destination edges are selected from subsets E_1,E_2,E_3 and E_4. For each pair, $p_i = (e_s, e_f)$ the points e_s and e_f are chosen so that the Euclidean distance between the two edges is at least 500 meters.

After each simulation step t, the AV chooses and executes one of the available actions. Then, we use the *TraCi* interface to collect the current position $x(t)$ of the AV on the map, the heading angle $\phi_v(t)$ and the current edge $e(t)$. The collected values are used to calculate the reward $r_{tot}(t)$. In this work, we set the weights $\alpha = \beta = \gamma = 1$.

Each episode terminates when the AV arrives at its destination or if an error occurs in the simulation environment. The results of the training are depicted in Fig. 8.5. In detail, Fig. 8.5a-8.5d show the mean reward of each episode for the



(a) *Agent 1* **(b)** *Agent 2*

(c) *Agent 3* **(d)** *Agent 4*

Figure 8.5: *DRL episode mean reward with modular strategy.*

four agents. We observe that all the agents converge to a mean value of 0 in almost 400k episodes after three hours of training.

We perform the same training using the exhaustive approach described in Section 8.3.3 to compare the results. In contrast to the proposed method, in this case we use only a single agent and we choose a random pair of edges $c = (e_s, e_f)$ for each training episode. As Fig. 8.6 shows, the exhaustive approach converges to a value of -5 after 400k episodes and seven hour of training.

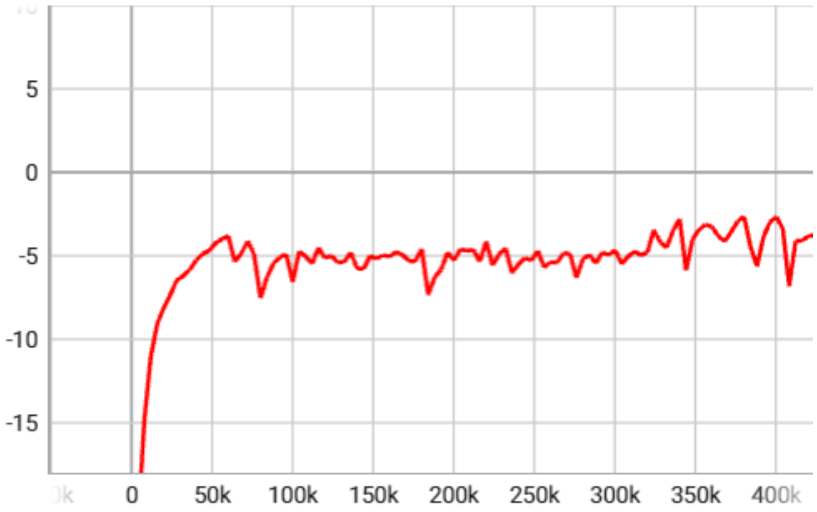
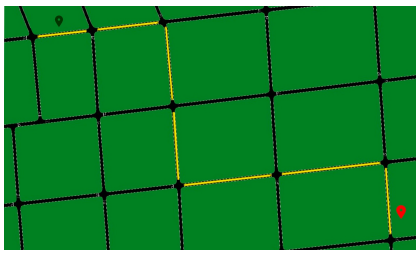
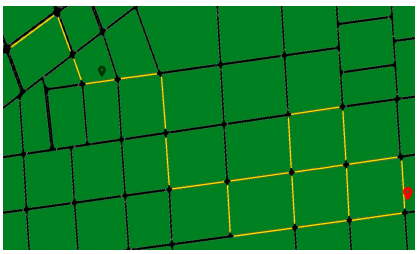


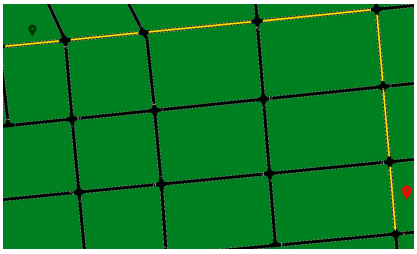
Figure 8.6: *DRL episode mean reward with exhaustive approach.*



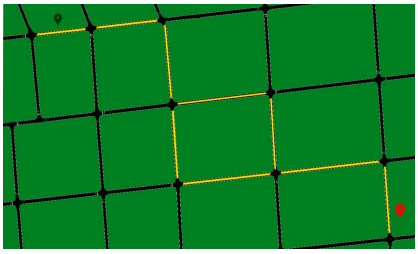
(a) *Agent 1 (Reward = -9)*



(b) *Agent 2 (Reward = -49)*



(c) *Agent 3 (Reward = -6)*



(d) *Agent 4 (Reward = -12)*

Figure 8.7: *Paths and rewards returned by the Modular DRL Route Planner.*

Since the mean reward obtained by the four agents is greater than the one obtained by the exhaustive approach, it is evident that the modular DRL strategy is better in terms of performances. Moreover, due to the less complexity of the

sub-problems compared to the main problem, the training time is also much less than the traditional approach.

8.4.3 Modular DRL Route Planner Example

In this subsection, we show an example of a route planning request dispatched by an AV and processed by the agents trained in Section 8.4.2 and the architecture described in Fig. 8.3.

More in detail, the AV chooses two edges e_s and e_f as origin and destination edge to form the pair $c = (e_s, e_f)$ and sends the request by the *REST-Api* interface to agents 1, 2, 3 and 4 and collects the paths and the associated rewards.

In Fig. 8.7, the paths returned by the agents are shown. We observe that each agent determined a different path for the trip from e_s to e_f. The four returned paths p_1, p_2, p_3, p_4 received the following rewards: of $r_1 = -9$, $r_2 = -49$,$r_3 = -6$ and $r_4 = -12$.

The path calculated by Agent 1 is shown in Fig. 8.7a and is the shortest path. However, it does not pass by the priority edges that are marked in red in Fig. 8.4 and does not contribute to the third component of the multi-objective reward defined in Section 8.3.2. On the contrary, Fig. 8.7b shows that the path calculated by Agent 2 passes by the priority edges but it is too long and exhibits many turns. Moreover, the path returned by Agent 4 is similar to the one of Agent 1.

Finally, the path calculated by Agent 3 passes by the priority edge, includes only one turn and has a total length slightly higher than the one calculated by Agent 1. Moreover, it passes by the priority edges defined on the map and contributes to all the components of the multi-objective proposed reward. As expected, this path received the highest reward (-6) and is selected as the optimal path by the AV.

8.5 Conclusion

The approach used in this chapter highlights the effectiveness of the modular Deep Reinforcement Learning (DRL)approach in urban routing optimization. By integrating modular DRL algorithms, we are able to dynamically adapt to changing urban traffic conditions and user preferences.

Training the model on data extracted from the open-source platform OpenStreetMap, using diverse urban scenarios diversified with the SUMO simulator, allowed us to optimize routes considering factors such as user preferences, costs, and utility.

These findings support the idea that combining artificial intelligence with urban planning can lead to smarter and more responsive transportation systems in the smart cities of the future.

The future research will integrate the proposed modular DRL approach with other technologies such as vehicle-to-everything (V2X) communication in order to enable more efficient and safer coordination of urban traffic. Moreover, the weights assigned to each reward component and the the rules for the map division will be optimized.

9 A User Based HVAC System Management Through Blockchain Technology and Model Predictive Control

9.1 Introduction

The quest for energy efficiency and consumption control is fundamental from an environmental perspective. Fifty percent of building energy consumption is used in Heating, Ventilation, and Air-Conditioning (HVAC) systems [138]. Hence, effective HVAC control techniques for energy consumption minimization and thermal comfort guarantee have attracted the attention of researchers. In order to simultaneously maximize comfort and minimize energy consumption, it is essential to manage the building network (district) in a way that optimally balances real-time energy usage.

While existing regulations have initiated some changes, traditional penalty-based systems often fail to encourage full user compliance. Emerging researchers advocate for integrating blockchain technology, like proposing a novel system that dynamically rewards or penalizes users based on their real-time energy consumption. This approach promises enhanced security and privacy, addressing key concerns in energy management [139]. The concept of bestowing complimentary energy credits to domestic end-users as a means to mitigate the demands during periods of peak loads, leveraging a direct load control mechanism envisaged by Erdinc et al. [140], has also undergone scrutiny. Furthermore, Shi et al. work [141] elucidates the efficacy of incentive-based demand response in eliciting shifts in consumer energy usage, providing a refined methodology for assessing load profiles and attenuating peak loads through economic inducements.

In the context of decentralized energy management, it is crucial to consider the market dynamics. Mnatsakanyan et al. [142] propose an innovative electricity market structure emphasizing individual pricing mechanisms for fairer demand response benefits distribution. Brahmia et al. [143] highlight the challenges of electricity price forecasting in multi-microgrid systems, advocating for advanced predictive models. Additionally, using reinforcement learning, Biemann et al. [144]

explore real-time pricing optimization in data center HVAC control, underscoring the need for responsive management solutions in fluctuating market conditions.

Advanced control and prediction algorithms, such as Long Short-Term Memory (LSTM) networks, have been utilized effectively for time-series prediction in energy management, capturing temporal dependencies for accurate real-time predictions and enabling more efficient energy resource utilization [145]. In parallel, Model Predictive Control (MPC) has proven pivotal for the dynamic optimization of complex systems like multi-zone buildings, which is critical in balancing energy efficiency with user comfort [146]. Integrating energy management algorithms with blockchains for secure communication is a substantial advancement. However, there are still notable gaps, particularly in tracking financial transactions and enhancing user comprehension of their energy usage patterns.

This chapter proposes an HVAC control system connected to the District Energy Management System (DEMS) and devoted to improving the HVAC consumptions' user management. In detail, the DEMS receives the electricity invoice from the energy supplier at the end of the billing cycle and other data concerning weather conditions, calendar days, and law limitations. Such data are notarized via a dedicated transaction on a blockchain platform.

Moreover, at the end of the billing period, the blockchain classifies the users into Consumption Classes, and rewards or penalties are assigned to each class. Hence, the blockchain decides all the user payments based on a virtuous classification performed by a K-means clustering algorithm proposed in a previous paper by the author of this dissertation [147].

At this point, the users can decide to modify their behavior to pass to a less expensive class. To this aim, a Model Predictive Control (MPC) strategy is locally applied by the user to determine the thermostat set-points of the HVAC system and the intervals in which the system HVAC must be switched off or switched on. A lexicographic optimization allows the user to minimize energy consumption by guaranteeing the user's comfort. In addition, an LSTM-based method is presented to determine the building's thermodynamic model and HVAC energy consumption. Then, the MPC approach solves the lexicographic optimization problem by using the building thermodynamic model and the HVAC energy consumption description obtained by the LSTM pre-trained network.

The new contribution of the study presented in this chapter is twofold.

First, the proposed HVAC control system is connected with the DEMS pricing and classifies users in the consumption classes using a K-means clustering algorithm. The approach's novelty also lies in the use of the blockchain platform, which guarantees accountability, transparency, and data notarization.

Second, we propose a methodology that the users can implement to pass to

a more virtuous class. Even if applying the MPC approaches and the LSTM-based methods are familiar in the HVAC control, we formulate and solve a novel lexicographic minimization problem. The solution can be applied by the users to suitably manage the HVAC system and reach the desired *Consumption Class* by satisfying their comfort.

The system architecture incentivizes users to shift towards more virtuous energy practices, ensuring that energy suppliers do not bear the consequences of inefficient consumption. By implementing a penalty-reward system rooted in blockchain technology, we advocate for a self-regulating user community where sustainable actions are incentivized and wasteful habits are discouraged.

The remaining structure of the chapter is as follows: Section 9.2 presents the literature review about the role of blockchain technology, Machine Learning (ML), MPC and LSTM methods within Smart Grids (SG) and DEMS. Moreover, Sections 9.3 and 9.4 describe the proposed HVAC Control System and the blockchain platform architecture, respectively. In addition, Section 9.5 formulates the district user clustering and the class follower problem. Section 9.6 designs the control system based on the MPC strategy and the LSTM thermodynamic model. Finally, Section 9.7 discusses the case study and Section 9.8 draws the conclusions.

9.2 Literature Review

9.2.1 Blockchain in Smart Grid and District Energy Management

With its distributed ledger architecture, blockchain technology is pivotal in enhancing the reliability and security of SG and district energy systems. Each transaction within this system is meticulously cataloged in structured blocks, linked sequentially to form a sequential or chain-like structure. Each block in the blockchain is securely linked to its predecessor by incorporating the output of the Secure Hash Algorithm 256 (SHA-256), commonly referred to as the *hash*, of the previous block, which is included in the structure of the current block [148]. The slightest variation would produce a completely different control number (**H**), also known as a hash number. The strength of storing data in this way lies in the mathematical certainty of the correctness of the data stored. The same data processed with the same hashing function will return the same number **H** in a deterministic way.

In SG, blockchain can be employed to transparently manage energy distribution, recording transactions from energy production to consumption. This ensures integrity in the trade of renewable energy certificates and facilitates real-time billing for consumers. This system could be applied to a database to certify data integrity. However, the innovation brought by blockchain lies in the fact that blocks include the **H** of the previous block within the input information [149].

In the context of SG security and privacy, various studies have highlighted the challenges and opportunities arising from emerging technologies, such as ML and blockchain [150, 151]. Haji Mirzaee [150] explored security and privacy challenges in SGs, including vulnerabilities and potential attacks in evolving power networks, emphasizing the need for additional research into security and privacy mechanisms. Furthermore, the authors discussed the growing use of ML algorithms in SG components for attack detection and threat analysis, highlighting the susceptibility of ML systems to adversarial attacks.

As previously mentioned, blockchain, a distributed technology that, thanks to its structure, enhances system redundancy and resilience to failures and cyberattacks, has emerged as a promising application within the SG (Smart Grid) paradigm [151]. In the pursuit of building Smart Cities, a smart district model has been designed, leveraging new technologies and efficient energy management systems integrated into an Internet of Things (IoT) and blockchain platform [152]. Christidis and Devetsikiotis [153] delved into the integration of blockchains and smart contracts with IoT, illustrating how these technologies could foster a marketplace of services

between devices and automate multi-step processes in a cryptographically verifiable manner. In a similar vein, Benedict et al. [154] introduced an IoT blockchain solution, i.e., an IoT-enabled blockchain for air quality monitoring systems in smart cities. Implementing *chaincodes* for air quality monitoring systems, the proposed architecture addressed prevailing security and performance challenges associated with IoT cloud solutions.

Moreover, blockchain has been applied in the SG for cybersecurity [155]. Kosba et al. [156] presented a decentralized smart contract system that ensures transactional privacy in decentralized cryptocurrencies, enabling programmers to write private smart contracts without implementing cryptography directly, as the compiler automatically generates an efficient cryptographic protocol. Li et al. [157] provided a quantitative and qualitative review of blockchain research from 2015 to 2021, identifying six research hotspots and five research frontiers to offer a comprehensive view of recent trends in the field. Malla et al. [158] conducted a state-of-the-art review on the status, challenges, and future directions of blockchain technology in power systems, discussing interfaces and possibilities that can ensure trust, security, and transparency, facilitating a decentralized power system and power market. The Hyperledger Fabric, a modular and extensible open-source blockchain system, is a promising solution for supply chain management, allowing customization for specific use cases and trust models without relying on a native cryptocurrency [159]. Although it is important to acknowledge the inherent limitations associated with its nature as a closed and non-public platform, the Hyperledger platform enables a secure and efficient way to track and manage transactions and assets throughout the entire supply chain.

9.2.2 Optimization Models for Energy Management

In recent years, ML algorithms have been applied to energy consumption prediction in smart buildings, examining the performance of Support Vector Regression, Artificial Neural Networks, and Random Forest algorithms. Wu and Chu identified in their study *Random Forest* as the best-performing algorithm and investigated the impact of sampling strategy on prediction accuracy. They discovered that increasing sampling density in high variance data enhanced prediction results, which can be employed to optimize ML algorithms for building energy consumption prediction, ultimately contributing to energy conservation, environmental protection, and smart city development [160].

Another study by Roccotelli et al. addressed the energy management issue in cooperative microgrids within a smart energy district [161]. It proposed an optimization model that aims to maximize the use of energy purchased at

the day-ahead market, minimizes the need for expensive real-time energy, and optimizes the integration of renewable energy sources, energy storage systems, and electric vehicle batteries. In order to tackle the uncertainties of key parameters, the proposed optimization model was solved using two approaches: one deterministic and one stochastic.

Muralidar et al. [162] emphasized the need for integrating blockchain and ML technologies in energy management systems to enhance efficiency, reduce costs, and support the implementation of renewable technologies in smart buildings. In their review, the energy management systems are at the center of monitoring and controlling energy needs in industrial buildings, underlining the necessity for such systems to address energy use efficiency improvement, energy cost reduction, and renewable energy technology implementation to cater to local energy loads in structures with distributed resources. Rajith et al. [163] pioneered the development of a real-time optimized HVAC control system using IoT, which was built upon an IoT framework that collected thermal parameters from sensors and user feedback information for real-time processing in a distributed cloud environment. Incorporating optimization techniques, demand response, and predictive models in their system led to a 20%–40% reduction in energy consumption while maintaining user thermal comfort.

9.2.3 Model Predictive Control for Building Climate Management

The utility of MPC in sustainable building management is increasingly recognized, especially when integrated with data-driven methodologies. Chen et al. [164] introduce a Data-Driven Robust MPC framework, which tackles the prevalent issue of weather forecast uncertainty. This work aligns with the thrust of our proposal to enhance climate control strategies in buildings. By incorporating ML techniques for constructing uncertainty sets, paper [164] establishes a foundation upon which our research builds, particularly in developing a tailored predictive model that accounts for the unique climatic and architectural characteristics of our focus buildings.

On the topic of learning-based approaches, the work of Eini and Abdelwahed [165] is noteworthy for its integration of Artificial Neural Networks (ANNs) with the MPC framework, resulting in notable energy savings and improved occupant comfort. Their method offers a proof of concept that resonates with our proposal's objective to optimize energy management while maintaining thermal comfort. Our research seeks to bridge the gap identified in their study by extending the learning-based control scheme to a wider range of building types and climatic

conditions, ensuring broader applicability and scalability of the proposed solutions.

Recent research has explored the optimization of building energy management and indoor thermal comfort. Homod et al. [166] presented a hybrid model highlighting the importance of non-temperature factors in HVAC system references. The potential of MPC in HVAC systems has been underscored by multiple studies, emphasizing its utility in ensuring energy efficiency and thermal comfort. However, from a pricing perspective, the limitations on the operational time-frames of HVAC systems imposed by some nations are not optimal solutions, as they risk leading to exceeding energy peaks [167].

While these studies lay the groundwork for innovative climate control through MPC, our research intends to expand on these methodologies. We aim to explore the intersection of advanced control algorithms and emerging technologies, such as the IoT and edge computing, to enable more responsive and adaptive building management systems. By leveraging the strengths of the existing models and identifying areas for improvement, our proposal aspires to contribute to the evolution of smart building energy systems that are both efficient and responsive to the occupants' needs.

9.2.4 Long Short-Term Memory Networks in HVAC Systems

LSTM networks are special kinds of Recurrent Neural Networks (RNNs) that are gaining attention in the field of HVAC systems for their ability to model and predict time series data with long-term dependencies. LSTMs are particularly well-suited for HVAC load prediction because of their capability to remember information for long periods, which is essential for capturing the dynamics of energy consumption in buildings.

Friansa et al. [168] compared LSTM and bi-directional LSTM models for the prediction of HVAC electricity load based on daily datasets. Their findings showed that Bi-LSTM models yielded higher accuracy, with a Mean Absolute Percentage Error of 15.35%, suggesting that LSTMs could significantly improve the prediction accuracy for HVAC electricity load management.

In a similar way, Wang et al. [169] proposed a Distributed Fusion LSTM model to forecast temperature and relative humidity in smart buildings. Their model, which utilizes distributed data-fusion technology, outperformed other forecasting methods, including Support Vector Regression and classical LSTM, highlighting the efficacy of LSTMs in predicting key environmental variables that affect HVAC performance.

Alden et al. [170] explored the use of LSTM networks to separate HVAC energy use from total residential load, which can be pivotal for enhancing energy

management systems in smart homes. They developed LSTM encoder-decoder models using future weather data, which proved to be effective in providing accurate day-ahead HVAC energy forecasts, thus facilitating more efficient energy management.

9.3 HVAC Control System

In this section, we propose an HVAC control system at the DEMS level to improve the user management of HVAC consumption. The system architecture is described in Fig. 9.1, which points out the main system components.

In detail, the DEMS receives the electricity invoice from the energy supplier at the end of the billing cycle and other data concerning weather conditions, calendar days, and law limitations. Such data are notarized via a dedicated transaction on the blockchain platform. On the other side, the IoT devices transmit HVAC states to the blockchain, such as the temperature at which the thermostat has been set.

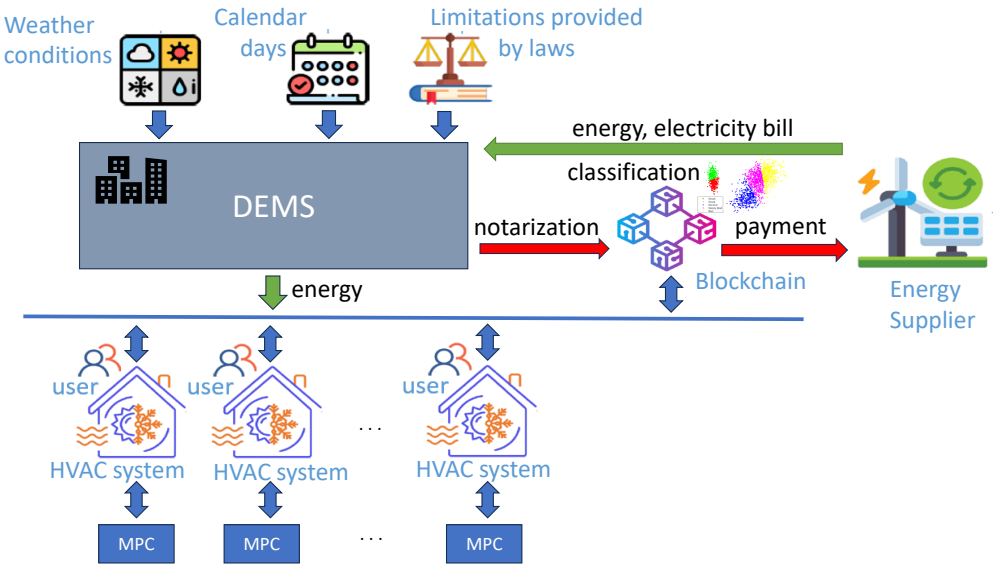


Figure 9.1: *System Architecture*

The start-up and closing times of the billing cycle are managed in a distributed manner through the use of a Smart Contract. The blockchain also notarises the data collected into the Smart Contract with a time-stamp through the same code of the blockchain platform, which includes the transaction containing the information within a time-stamped block.

At the end of the billing period, the blockchain classifies the users in n Consumption Classes and a reward or penalty is assigned to each class. Hence, the blockchain decides all the user payments based on a virtuous classification performed by a K-means clustering algorithm proposed in [147]. In particular, the

users are classified into consumption classes by a K-means clustering algorithm, and each class is characterized by a multiplier coefficient that increases (penalty) or decreases (reward) the user energy costs. Such coefficients are notarized on the blockchain, which provides payment at the end of a billing cycle.

Obviously, a user belonging to a virtuous class contributes less to the payment of an electricity invoice than a user belonging to a less virtuous class due to rewards and penalties. For this reason, a user can be encouraged to move from a less virtuous class to a more virtuous one. The strategy to enable a user to change their behavior to pass a less expensive class is implemented by an MPC approach.

In particular, the MPC uses a predictive model to forecast future performance and determines a control action to meet predefined constraints and objectives over a designated horizon.

The significance of MPC in smart DEMS lies in its predictive power and adaptability. It allows the system to pre-emptively adjust HVAC settings in response to user behavior and external factors, ensuring that the energy consumption is simultaneously efficient and economical. The proposed MPC scheme provides the thermostat set-points of the HVAC system in the different zones of the building to allow the user to reach a less expensive consumption class.

9.4 Blockchain Architecture

This section describes the blockchain platform and the design of the related Smart Contracts.

9.4.1 Blockchain Platform

It is worth recalling that reading information on the blockchain has no cost; on the contrary, writing can be expensive. This leads to the first critical issue to be addressed, which is the choice regarding the blockchain platform to be implemented in order to prevent the user from incurring costs that exceed what is necessary.

On the client side, the system is required to record a substantial amount of data on the blockchain, as the consumption data needs to be notarized over time. To address this challenge, several solutions exist: one local approach involves processing the data locally, regularly generating hashes of the data, and then notarizing only these hashes on the blockchain. In this way, the control over the data would not be direct but would still guarantee high reliability. This is because the optimization tool would acquire the data via the Application Programming Interface (API) directly from the server, then submit them to the same hashing algorithm (deterministic), and eventually compare the hash calculated with the notarized one in the blockchain. If data inconsistency occurs, the system assumes manipulation and applies the maximum possible coefficient. This hybrid solution involves a high-risk factor, given the poor resilience of the system to errors. A single variation of data in the period under consideration would heavily penalize the user, and it cannot be assumed that all of these errors are attributable to system manipulations.

Conversely, systems that employ on-chain data storage demonstrate enhanced resilience, as exemplified by the subsequent proposed solutions. One alternative is a blockchain that is compatible with the Ethereum Virtual Machine (EVM), which benefits from reduced costs when compared to notarization fees associated with the main Ethereum Blockchain.

Alternatively, when full on-chain data writing and optimization execution are required, a more innovative approach is preferable. This is exemplified by an *Avalanche subnet*. To date, the Avalanche Blockchain offers a solution with lower access costs and more immediate usability, largely due to its governance policy managed by a Proof of Stake (PoS) consensus algorithm. In PoS blockchains, part of the nodes that contribute to archiving the blockchain's history can also write to it, composing the new blocks. The requirement is to lock (stake) some native cryptocurrencies so that any malicious actors can be penalized by eroding the

staked capital (with different methodologies depending on the parameters of the consensus algorithm implemented).

The role of these specific nodes on a PoS blockchain is called "*validator*". The Avalanche Blockchain's validators can write contextually to the main Blockchain and also to different blockchains called *subnets*, and each subnet can be programmed individually [27]. An evolved use case envisages that a particularly advanced blockchain application such as the one being studied here can be implemented on a proprietary *subnet*. This solution cuts transaction costs, bringing them to a minimum. In order to have the subnet working smoothly, the developed blockchain application must provide incentives for the validators to attract them to validate the subnet.

Furthermore, while the blockchain itself is not anonymous, it offers a high degree of pseudonymity by allowing transactions and interactions through addresses that are not directly linked to the users' identities. This ensures verifiability and integrity without compromising privacy. Moreover, the advanced customizability of Avalanche's subnets permits the system to be tailored to adhere to specific legal and regulatory requirements. Adjusting the parameters of a subnet can ensure compliance with data protection laws, such as the GDPR in the European Union, thereby managing permissions and safeguarding sensitive data. This flexible configuration underscores our commitment to ethical and legal responsibility, balancing technological innovation with due diligence in a dynamic regulatory landscape.

It should also be remembered that *Avalanche* blockchain supports EVM and adopting the EVM allows for the seamless migration and verification of smart contracts' logic and state, thus reinforcing the advisability of developing on an EVM-based platform for enhanced flexibility and interoperability [171]. In the proposed scheme, we deploy a diptych Smart Contracts system to record the connected user's data and to manage the payment of the invoices.

9.4.2 Smart Contracts Design

Smart contracts, a revolutionary feature introduced with the advent of the Ethereum Blockchain, automate the execution of agreements, effectively preventing the need for traditional intermediaries. These contracts are considered "smart" because they can self-execute and self-enforce contract terms embedded in the blockchain's immutable ledger. Leveraging Ethereum's decentralized architecture, smart contracts encode obligations and conditions in code, thus creating a trustless environment where transactions, operations, or agreements are automatically executed once certain conditions are met [172].

In the presented system architecture, the DEMS sends and receives information and data through the blockchain platform that is connected with the building HVAC systems and the energy supplier (see Fig. 9.2). Utilizing smart contracts, the DEMS initiates the process by securely sending data to the blockchain, a step we refer to as "notarization". This process ensures that all energy and electricity bill-related data are immutable and verifiable, fostering trust and transparency in the energy trading market.

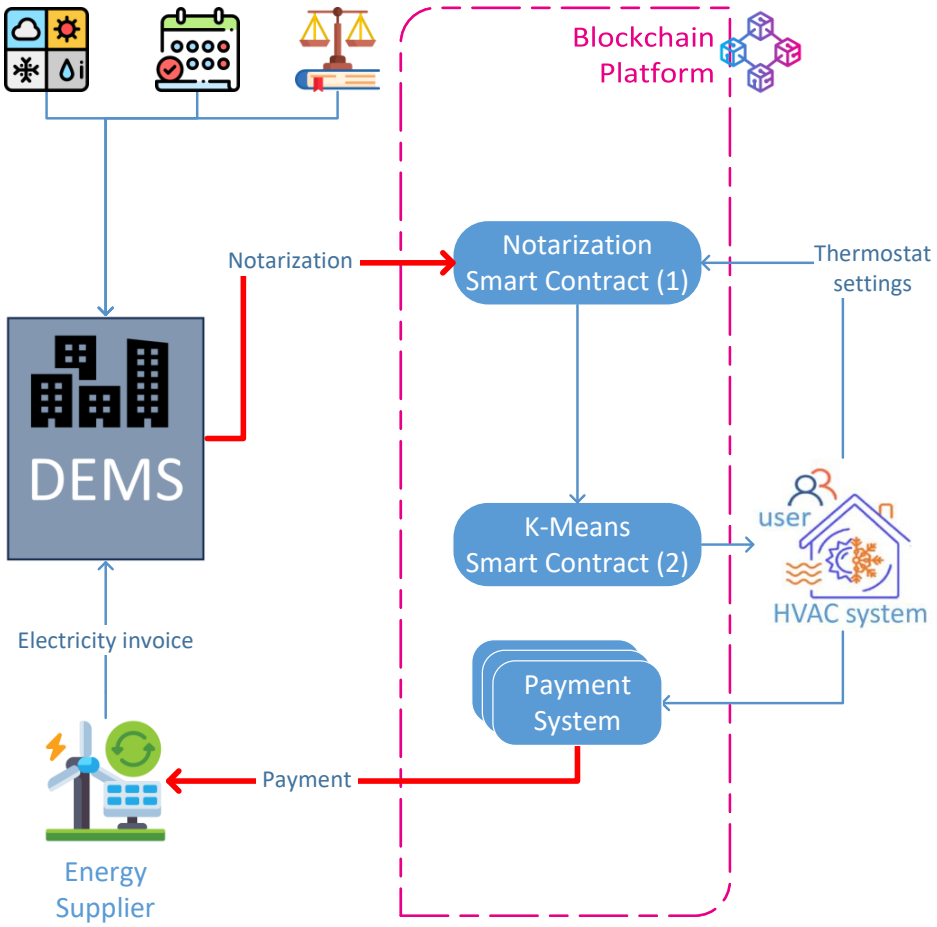


Figure 9.2: *The blockchain platform connected with the DEMS of Fig. 9.1*

The notarization process started by the DEMS is performed by Smart Contract

4 that is designed to handle the complexity and variety of data generated by the DEMS. The contract acts as a notary, storing the relevant information transmitted by each dedicated transaction, thus ensuring data integrity and non-repudiation. This step lays the groundwork for accurate classification and subsequent billing.

The pseudo-code for the Smart Contract 4, named *Data Notarization*, is described in the following.

Code 4 Smart Contract 1: Data Notarization

```
contract Data Notarization {
    struct UserData {
        uint256 indoorTemp;
        uint256 outdoorTemp;
        uint256 seasonalTemp;
        uint256 lawTemp;
        uint256 activeIntv;
        uint256 totalIntv;
    }

    mapping(address => UserData)
      public userData;

    // Store user's data into the contract
    function store(
      address _user,
      UserData memory _data
    ) public {
        userData[_user] = _data;
    }
}
```

A structure called *UserData* is defined to encapsulate all the key metrics needed for each user. These metrics include both indoor and outdoor temperatures, seasonal averages, and the number of intervals during which the HVAC system is operational. The contract incorporates a function, denominated `store`, which enables the secure archival of data corresponding to each user on the blockchain. This step furnishes a reliable and immutable ledger that can be subsequently accessed for analytical endeavors.

Upon the notarization of energy data, Smart Contract 5, named *Kmeans Clustering*, determines the payment classes, the incentives and the penalties, also in collaboration with the energy supplier. The smart contract encodes the payment mechanism, ensuring that once the classification is complete and bills are calculated, payments are autonomously disbursed to the energy suppliers' accounts.

The automated nature of these transactions reduces the latency and potential errors associated with manual processing, offering a streamlined and efficient payment process. This contract is initialized with the address of the *DataNotarization* Smart Contract, enabling it to fetch the necessary data for clustering. Specifically, it retrieves the notarized data from the blockchain to perform the clustering algorithm.

Successively, it invokes a K-means clustering algorithm `kmeans` that has three main tasks: 1) retrieving the user data stored in Smart Contract *DataNotarization*, 2) executing the K-means algorithm using such data, 3) allocating users to their respective clusters, 4) calculate the pricing through an internal function that can use a particular pricing model and 5) distribute these results to the users' accounts.

Utilizing separate smart contracts for data notarization and the K-means algorithm presents distinct advantages and disadvantages. On the positive side, separating these functionalities clearly delineates responsibilities, simplifying system management and future extensibility. Furthermore, this separation allows for easier scalability as each contract can be optimized for its specific task. The separation also offers the benefit of code reusability, particularly for the data notarization contract, which could be employed in various other contexts or projects. Conversely, the system's overall complexity could increase due to managing multiple contracts. Additionally, interacting between multiple contracts (when writing) may incur extra transaction costs in terms of gas. However, this concern is mitigated by the current implementation using an EVM-compatible environment with negligible gas costs for such operations.

Code 5 Smart Contract 2: K-Means

```
contract KmeansClustering {
  Data Notarization dataNotarization;

  // Initialize with DataNotarization
  // contract address constructor
  constructor(
    address _dataNotarizationAddr
  ) {
    dataNotarization =
    DataNotarization(_dataNotarizationAddr);
  }

  // Execute K-means algorithm and set
     prices
  function kmeans() public {
      // 1. Fetch data from
      //    DataNotarization contract
      // 2. Perform K-means clustering
      // 3. Assign users to clusters
      // 4. Calculate pricing based on
      //    clustering (internal fn)
      // 5. Distribute the pricing to
      //    accounts
  }
  // Calculate pricing
  function calculatePricing()
  internal {
      // it can use a pricing model
  }
}
```

9.5 District User Clustering and Class Follower Problem

In this section, the preliminary *District User Clustering* scheme based on *k-means* algorithm is formally described and the subsequent *Class Follower Problem* is introduced.

9.5.1 K-means District User Clustering

Let us consider the set of m users $\mathcal{U} = \{u_i | i = 1, 2, .., m\}$ in the district. A *Consumption Class* $c_j \in C$, with $C = \{c_j | j = 1, ..., n\}$, is assigned to each user $u_i \in \mathcal{U}$. A *Consumption Class* c_j basically represents the user's consumption profile in a certain time frame $h \in N$, where N is the set of natural numbers, and a billing period is made of a certain number of time frames. As suggested in [147], the profile can be driven by the level of compliance of the user to the law and to the best practices in terms of energy saving and environment preservation.

To the purpose of class assignment, for each considered time frame, the set of feature vectors $\mathcal{V}(h) = \{\mathbf{v}_1(h), \mathbf{v}_2(h), ..., \mathbf{v}_i(h), \ldots, \mathbf{v}_m(h)\}$ is defined, where $\mathbf{v}_i(h)$ is the feature vector associated with the user u_i at time frame h. The components of $\mathbf{v}_i(h)$ are four average metrics collected by the sensors operated by i-th user over h and safely stored in the Smart Contract 1.

In this work, the following metrics for $\mathbf{v}_i(h)$ are defined:

- the average indoor temperature $\overline{T_{ih}}$ of user u_i;
- the average outdoor temperature $\overline{Te_h}$;
- the average seasonal outdoor temperature $\overline{Ts}$;
- the seasonal indoor temperature threshold enforced by the law $\overline{Tl}$;
- the number of time intervals H_{ih} in which the air conditioning system is powered on for user u_i;
- the total number of time intervals H_{tot} in h.

Now, with the defined data, the feature vector of user u_i is formally defined as $\mathbf{v}_i(h) = [\alpha_{1i}(h), \alpha_{2i}(h), \alpha_{3i}(h), \alpha_{4i}(h)]^T$ with:

- $\alpha_{1i}(h) = \overline{T_{ih}}/\overline{Te_h}$;
- $\alpha_{2i}(h) = \overline{T_{ih}}/\overline{Ts}$;

- $\alpha_{3i}(h) = \overline{T_{ih}}/\overline{Tl}$;
- $\alpha_{4i}(h) = H_{ih}/H_{tot}$.

In particular, $\alpha_{1i}(h), \alpha_{2i}(h), \alpha_{3i}(h)$ represent the average indoor temperature measured at time h compared to the average outdoor temperature, the seasonal average temperature and the threshold enforced by the law, respectively. In addition, $\alpha_{4i}(h)$ considers the level of operation of the air conditioning system by comparing the total number of hours of operation to the total number of hours H_{tot} in time frame h.

Now, the *k-means* clustering algorithm is applied to the set $\mathcal{V}(h)$ and each vector $\mathbf{v}_i(h)$ is assigned to a class $c_j \in C$. We denote by $\mathbf{k}_j(h)$ with $j = 1, ..., n$ the centroid of class c_j, i.e., a vector with the same dimensions of $\mathbf{v}_i(h)$ representing the center of cluster c_j at time frame h. The class assigned to the i-th user at time frame h is denoted as $P_i(h) \in C$, where $P_i(h)$ is the class with the least Euclidean distance between $\mathbf{v}_i(h)$ and the centroid $\mathbf{k}_j(h)$ for $j = 1, ..., n$.

For the purpose of applying discounts or penalties to each user, the centroids can be calculated by averaging the values of all time frames belonging to a specific billing period. Moreover, the classes in the set C are ordered on the basis of their centroid in an appropriate way, starting from the most virtuous class (c_1) to the least virtuous one (c_n).

Example. Figure 9.3 shows the clustering performed by k-means with $n = 5$ classes and $m = 2000$ users. At the end of the procedure, based on collected values, users are partitioned into five different classes ranked from *Small* (c_1), meaning a small power consumption, to *Bad* (c_5), in which the users with high power consumption and high relative difference between indoor and outdoor and seasonal temperatures are placed. In order to visualize the partitions in only two dimensions, the Principal Component Analysis (PCA) [173] is applied to the classified user set by projecting the data on the first two principal components (PC1 and PC2 in Fig. 9.3).

9.5.2 Class Follower Problem

Let us assume that user $u_i \in \mathcal{U}$ has been assigned to class c_k for the h time frame and at time h the current billing period ends. This user wants to be assigned to a different and better class c_j for time frame $h + 1$, where $k \neq j$. Now, the *Class Follower Problem* (CFP) for each generic user is defined as follows:

$$\min_{\mathbf{v}(h+1) \in \mathcal{V}(h+1)} \|\mathbf{k}_j(h) - \mathbf{v}(h+1)\|. \tag{9.1}$$

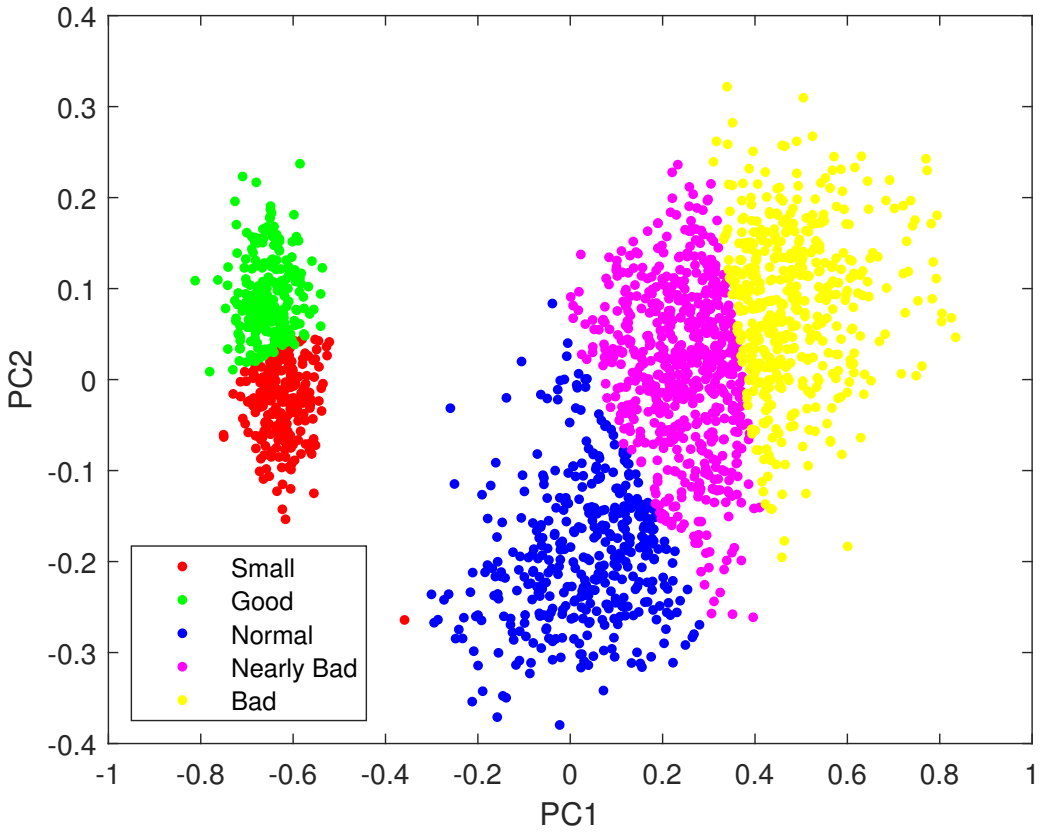


Figure 9.3: *K-means clustering*

Note that for the sake of simplicity, the index i denoting the i-th user is omitted in the following.

The aim of the objective function (9.1) is selecting vector $\mathbf{v}(h+1) \in \mathcal{V}(h+1)$ that exhibits the minimum Euclidean distance between vector $\mathbf{v}(h+1)$ and the centroid $\mathbf{k}_j(h)$ of the desired c_j class as calculated in previous time frame h. In that respect, it is evident that the global minimum of (9.1) is reached when $\mathbf{v}(h+1) = \mathbf{k}_j(h)$.

In order to minimize (9.1), it is necessary to determine the indoor temperatures and the HVAC actuators values during the time frame $h+1$. To this aim the time frame $h+1$ is divided in S time steps $s = 1, .., S$ and the following decision variables are defined:

$$T_{h+1}(s) \in \mathcal{R}^+ \text{ for } s = 1, 2, ..., S; \tag{9.2a}$$

$$H_{h+1}(s) \in \{0, 1\}, \text{for } s = 1, 2, ..., S; \tag{9.2b}$$

where $T_{h+1}(s)$ is the indoor average temperature collected at time step s and $H_{h+1}(s) = 1$ means that the HVAC of user u_i is powered on during the time step s, otherwise $H_{h+1}(s) = 0$ means that it is powered off.

Moreover, the feature vector elements at at time $h+1$ for each user $u_i \in \mathcal{U}$ are computed by the following variables:

$$\alpha_1(h+1) = \sum_{s=1}^{S} \frac{T_{h+1}(s)}{Te_{h+1}(s)} \tag{9.3a}$$

$$\alpha_2(h+1) = \frac{1}{S}\frac{1}{\overline{Ts}} \sum_{s=1}^{S} T_{h+1}(s) \tag{9.3b}$$

$$\alpha_3(h+1) = \frac{1}{S}\frac{1}{\overline{T_l}} \sum_{s=1}^{S} T_{h+1}(s) \tag{9.3c}$$

$$\alpha_4(h+1) = \frac{1}{S} \sum_{s=1}^{S} H_{h+1}(s). \tag{9.3d}$$

Now, the CFP (9.1) can be rewritten as follows:

$$\min \sum_{k=1}^{4} (k_j(h)_k - \alpha_k(h+1))^2 \tag{9.4a}$$

$$\text{s.t.} \tag{9.4b}$$

$$H_{h+1}(s) \in \{0, 1\} \; \forall s = 1, ..., S \tag{9.4c}$$

$$T_{h+1}(s) \in \mathcal{R}^+ \; \forall s = 1, ..., S. \tag{9.4d}$$

We assume that the average outdoor temperature $Te_{h+1}(s)$ at each time step s is determined by a suitable stochastic function.

To solve the CFP (9.4), we need to predict the values of the indoor temperature $T_{h+1}(s)$ at each time step s. However, the indoor temperature is related to intrinsic features of the building, such as the wall and floor materials and the number, position and size of the windows. Nevertheless, when an HVAC system is present, the indoor temperature is mainly driven by the thermostat set-point. We denote by $y_{h+1}(s) \in [0, 1]$ the real value of the set-point at time step s in time frame $h + 1$, so that we can write:

$$T_{h+1}(s) = L(y_{h+1}(s)) \; s = 1, ..., S$$

where $L(y_{h+1}(s))$ denotes the building thermodynamic model and determines the indoor temperature $T_{h+1}(s)$ at time step s corresponding to the thermostat set point $y_{h+1}(s) \in [0, 1]$. Hence, the decision variables of CFP (9.4) are the thermostat set-point values $y_{h+1}(s)$ for $s = 1, ..., S$.

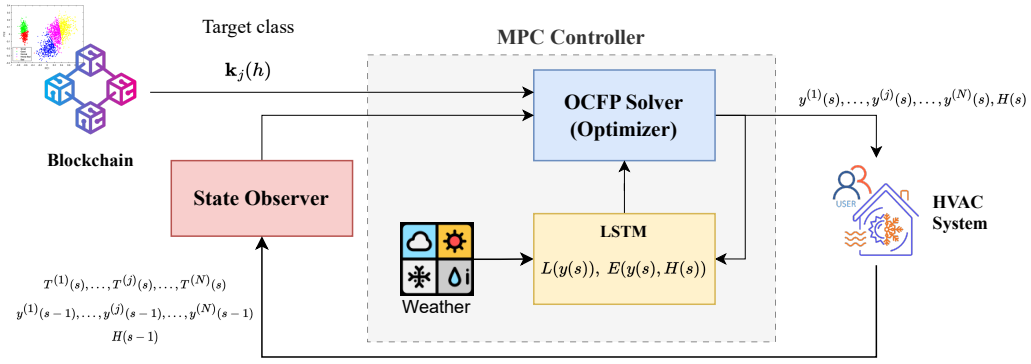


Figure 9.4: *MPC Architecture*

9.5.3 Energy Consumption Optimization

Since CFP (9.4) may have more optimal solutions, we have the possibility of considering a second objective function, related to the energy consumption denoted by $E(y_{h+1}(s), H_{h+1}(s))$, which is function of the set points $y_{h+1}(s)$ and the ON/OFF positions of the HVAC system at time steps $s = 1, ..., S$. Then, the cumulative energy consumption over the time frame $h + 1$ is the following:

$$\sum_{s=1}^{S} E(y_{h+1}(s), H_{h+1}(s)). \tag{9.5}$$

Since equation (9.5) is not straightforward to ascertain, its value will be computed through an approximation in a subsequent part of this work, facilitated by the introduction of an LSTM. Now, the following lexicographic minimization problem is formulated:

$$\begin{aligned} \text{lex min} \sum_{k=1}^{4} (k_j(h)_k - \alpha_k(h+1))^2,\ \sum_{s=1}^{S} E(y_{h+1}(s), H_{h+1}(s)) \\ \text{s.t.} \\ y_{h+1}(s) \in [0,1]\ \forall s = 1, ..., S \\ H y_{h+1}(s) \in \{0,1\}\ \forall s = 1, ..., S. \end{aligned} \tag{9.6}$$

The lexicographic optimization consists of subdividing a multi-objective problem into a set of single-task optimizations that are solved in series according to their priority order [174]: the optimization with the highest priority is solved first and, then, the successive one is addressed with an additional constraint which aims at guaranteeing the optimality of the higher priority cost function. In this work, problem (9.6) is denoted as *Energy Consumption Optimization CFP* (OCFP).

9.6 Control System Design and Simulation

In this section, a data-driven MPC strategy to solve the OCFP in real-time is introduced. Since the proposed approach is iterated over each time frame h, for the sake of simplicity, the suffix $h + 1$ is omitted in the relevant notations.

9.6.1 System Architecture

The architecture of the proposed system for a specific building is depicted in Fig. 9.4. In more detail, the system is made of three main blocks: (i) the Plant, (ii) the State Observer and (iii) the MPC Controller. The Plant is represented by the user building and the associated HVAC system. The State Observer is basically constituted by a set of sensors installed inside and outside the building to monitor indoor and outdoor temperatures. The components of the State Observer are detailed in the subsequent Algorithm 4. The MPC Controller is composed of three sub-blocks: the outdoor temperature predictor, the LSTM-based plant thermodynamic model that predicts indoor temperatures and energy consumption based on historical thermostat set-points and HVAC operational times, and the Optimizer that solves the OCFP problem (9.6) over time frame $h + 1$.

More specifically, the Optimizer, at each time step s, first takes the current Plant state and the target class $\mathbf{k}_j(h)$ to follow as a reference, then it determines a set of feasible inputs that minimizes the distance to $\mathbf{k}_j(h)$ and, finally, it selects the solution with the least energy consumption. At the end of the optimization procedure, the updated inputs are applied to the Plant by adjusting thermostat set-points and switching *ON* or *OFF* the HVAC system.

9.6.2 LSTM-based Plant Thermodynamic Model

The indoor temperature variation and the HVAC energy consumption are influenced not only by the inputs, such as the thermostat set-points, but also by several intrinsic features of the building and the HVAC system. Hence, finding an analytical solution is not always feasible. Moreover, both the temperature and the consumption depend not only on the current input but also on the past inputs applied to the HVAC system.

In this section, we propose an LSTM-based method to determine function $L(y(s))$ for $s = 1, ..., S$ and the energy consumption function $E(y(s), H(s))$ for $s = 1, ..., S$. LSTM is a variant of typical RNNs and can avoid the vanishing gradient problem existing in regular RNNs.

A data-driven solution is proposed to approximate both L and E functions using LSTM. The calculation formulas of LSTM cell units from the input to the output are obtained as shown in Table 9.1 [42, 175] [176], where i, f, $\tilde{c}$ and o, represent an input gate, forget gate, candidate vector and output gate, respectively, $\mathbf{W}_i$, $\mathbf{W}_f$, $\mathbf{W}_c$ and $\mathbf{W}_o$ denote weight matrices, $\mathbf{b}_i$, $\mathbf{b}_f$, $\mathbf{b}_c$ and $\mathbf{b}_o$ represent bias vectors, $\sigma(\cdot)$ and $\tanh(\cdot)$ denote the sigmoid and hyperbolic tangent functions, respectively, $\odot$ denotes a dotwise product, and c represents a cell state.

Table 9.1: *LSTM Equations:*

Input gate controller:	$i_s = \sigma(\mathbf{W}_i[h_{s-1}, \hat{x}_s] + \mathbf{b}_i)$
Forget gate controller:	$f_s = \sigma(\mathbf{W}_f[h_{s-1}, \hat{x}_s] + \mathbf{b}_f)$
Candidate vector:	$\tilde{c}_s = \tanh(\mathbf{W}_c[h_{s-1}, \hat{x}_s] + \mathbf{b}_c)$
Cell Memory:	$c_s = f_t \odot c_{s-1} + i_t \odot \tilde{c}_s$
Output gate controller:	$o_s = \sigma(\mathbf{W}_o[h_{s-1}, \hat{x}_s] + \mathbf{b}_o)$
Output:	$h_s = o_s \odot \tanh(c_s)$

As shown in Fig. 9.5, the cell unit structure of an LSTM network consists of three gates: (i) a forget gate, (ii) an input gate, and (iii) an output gate. The forget gate determines what information of the past cell state is to be forgotten. The input gate is used to control what information of the input at the current time is to be added to the cell state. Finally, the output gate is used to determine what information of the cell state at the current time is to be used as output.

In order to train the LSTM, sample data need to be collected for a certain time from temperature sensors inside and outside the building, as well as the associated thermostat set-points and HVAC operating states. Although in OCFP problem (9.6) only indoor average temperature is considered, a building is composed of a set of rooms, each one is equipped with a thermostat and indoor sensor. The sample input and output vectors collected for LSTM training at time step s are defined as follows:

$$\mathbf{X}(s) = \left[y^{(1)}, ..., y^{(j)}, ..., y^{(N)}, Te, H(s)\right]^T \tag{9.7}$$

$$\mathbf{T}(s) = \left[T^{(1)}, ..., T^{(j)}, ..., T^{(N)}, E\right]^T \tag{9.8}$$

where N is the number of rooms in the building, $y^{(j)}$ is the set-point of thermostat

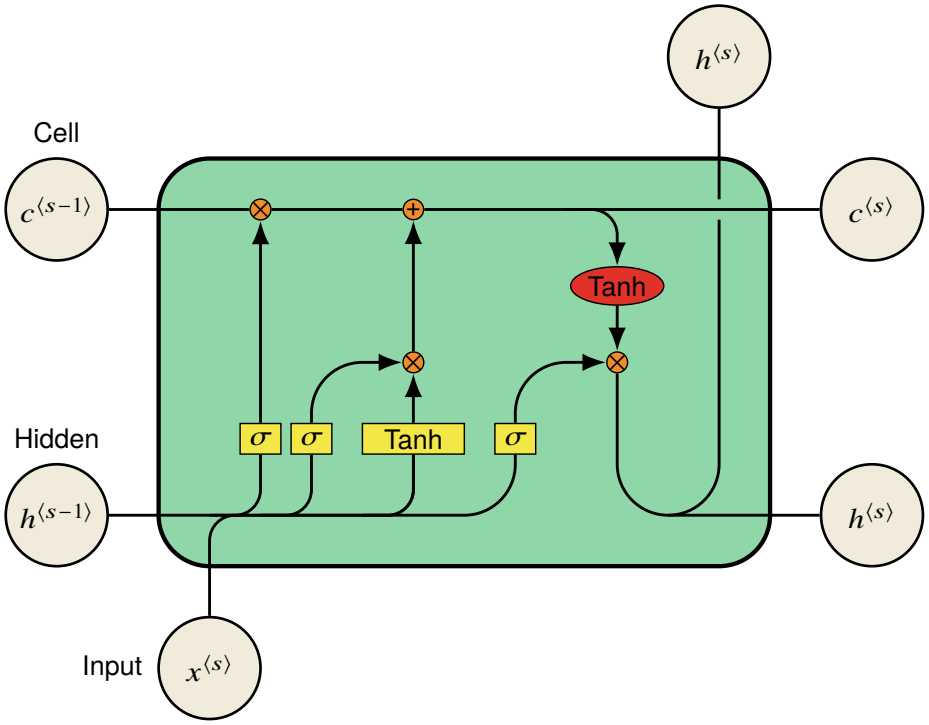


Figure 9.5: *LSTM cell architecture*

j, $T^{(j)}$ is the environment temperature of room j, $H(s) \in \{0, 1\}$ is the HVAC operating state and E is the cumulative energy consumption measured in Watts.

As it is shown in Fig. 9.6, if S consecutive time steps are considered for LSTM training, the final structure of the network is made of a series of S interconnected unit cells and the final value $\mathbf{T}(S)$ is the output prediction of indoor temperatures and energy consumption after S time steps. The mean indoor temperature value $\overline{T(S)}$ at time step S can then be easily calculated by averaging the individual rooms' predicted temperatures.

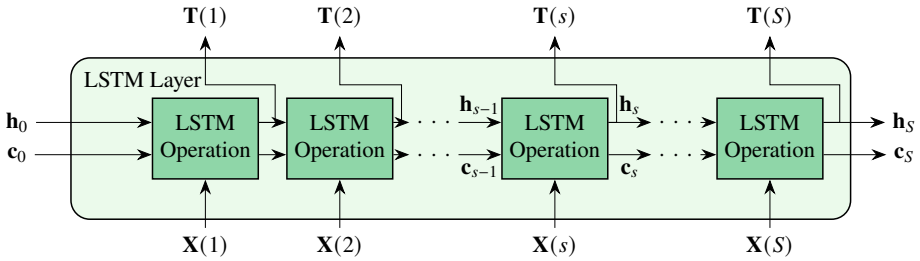


Figure 9.6: *LSTM final structure*

9.6.3 Considerations on Physics-Informed Neural Networks

While the data-driven LSTM architecture described above demonstrates effective predictive capabilities for building thermodynamics, an alternative paradigm worthy of consideration involves Physics-Informed Neural Networks (PINNs), which incorporate governing physical laws directly into the learning process through modified loss functions that penalize violations of known differential equations. A Physics-Informed LSTM variant would embed the fundamental heat balance equation presented in Equation 4.1 as a soft constraint during training, potentially reducing the data requirements for model convergence while ensuring that predictions remain physically plausible even when extrapolating beyond the training distribution.

The potential advantages of such an approach for HVAC thermal prediction include improved generalization to unseen operating conditions, since the network cannot learn spurious correlations that violate thermodynamic principles, alongside reduced sample complexity arising from the regularization effect of physical constraints that effectively inject domain knowledge into the learning process. Furthermore, physics-informed architectures typically exhibit enhanced interpretability, as the learned representations must align with physically meaningful quantities such as thermal conductances and capacitances rather than abstract latent features.

However, several considerations motivated the adoption of purely data-driven LSTM in the present work. The formulation of appropriate physics-informed loss terms for multi-zone buildings with complex HVAC interactions requires detailed knowledge of system parameters that may not be readily available in retrofit scenarios where the proposed methodology finds primary application. The computational overhead associated with evaluating differential equation residuals at each training iteration increases training time substantially, potentially by an order of magnitude, which proves problematic when frequent model updates are required to track seasonal variations in building behavior. Additionally, the strict enforcement of physical constraints may inadvertently limit the model's capacity to capture phenomena not explicitly represented in the simplified governing equations, such as occupant-induced thermal disturbances or equipment degradation effects that manifest as apparent violations of idealized thermodynamic relationships. The purely data-driven approach thus offers pragmatic advantages in deployment flexibility while achieving prediction accuracy sufficient for the MPC application, though future implementations targeting buildings with well-characterized thermal properties may benefit from physics-informed formulations that leverage this additional structural knowledge.

9.6.4 Model Predictive Control Scheme

The real-time control of the HVAC system is implemented by the MPC approach. In detail, at each time step s, the MPC solves the OCFP (9.6), in which the functions $L(y(s))$ and $E(y(s), H(s))$ are approximated by the LSTM pre-trained network. The time horizon of the MPC scheme is equal to S time steps, i.e., the duration of the time frame $h + 1$.

Algorithm 4 Proposed MPC Scheme

1: Set reference as $\mathbf{k}_j(h)$
2: Collect:
- Indoor temperatures:
- $T^{(1)}(s), ..., T^{(j)}(s), ..., T^{(N)}(s)$
- Thermostat set-points:
- $y^{(1)}(s-1), ..., y^{(j)}(s-1), ..., y^{(N)}(s-1)$
- HVAC state:
- $H(s-1)$

3: Average indoor temperatures
4: Perform lexicographic optimization (9.6) over S steps:
5: **for** each time step s **do**
6: Compute candidate input vector $\hat{\mathbf{X}}(s)$
7: Select input vector with least energy consumption
8: Apply thermostat set-points
9: $y^{(1)}(s), ..., y^{(j)}(s), ..., y^{(N)}(s)$ and $H(s)$
10: to the HVAC system

In Algorithm 4, the proposed MPC scheme is shown. At step 1, the system takes the followed class $\mathbf{k}_j(h)$ as reference. At step 2, the State Observer collects current indoor temperatures: $T^{(1)}(s), ..., T^{(j)}(s), ..., T^{(N)}(s)$, last applied room thermostat set-points $y^{(1)}(s-1), ..., y^{(j)}(s-1), ..., y^{(N)}(s-1)$, as well as last HVAC operating state $H(s-1)$.

At step 4, the Optimizer pre-processes the data by averaging the indoor temperatures and performs the lexicographic optimization (9.6) over the next S time steps by minimizing the two cost functions. The non-linear problem can be solved by iterative methods, such as Sequential Least SQuares Programming (SLQP) [177]. Moreover, for each time step s, the Optimizer calculates a candidate optimal input vector $\hat{\mathbf{X}}(s)$ for the LSTM model as follows:

$$\hat{\mathbf{X}}(s) = \Big[y^{(1)}(s+i), ..., y^{(j)}(s+i), ..., y^{(N)}(s+i), \\ H(s+i), ..., y^{(N)}(s+S-1), H(s+S-1)\Big]^T \tag{9.9}$$

for $i = 0, 1, 2, ..., S-1$.

Here, the outdoor temperature $Te(s)$, required by the LSTM network as part of the input, is not considered as a decision variable as it is assumed as predicted by a stochastic function and, therefore, is not included in (9.9).

The input vector spans from time step s to $s + S - 1$ and allows the Optimizer to fit to the MPC time horizon. At the end of the optimization procedure, the input vector with the least cumulative energy consumption over the considered future time horizon is selected as optimal solution and the first set of thermostat set-points: $y^{(1)}(s), ..., y^{(j)}(s), ..., y^{(N)}(s)$, as well as the operating state $H(s)$ are applied to the HVAC system.

The whole control scheme reiterates for time step $s + 1$ up to the end of the billing period $h + 1$.

9.7 Case Study

In this section, the results of the numerical simulations performed to validate the proposed approach are described and discussed. The well-known building energy simulation *EnergyPlus* is used with a standard 5-zone building to collect data and simulate the plant in the MPC scheme.

9.7.1 The Studied Building

We use a single-floor rectangular building of $30.5m \times 15.2m$ made of five different air-conditioned zones with individual thermostats, named Zone 1 to Zone 5. The building is located in the city of Bari, in Italy. There are windows on all four exterior facades and glass doors on the south and north facades. The air conditioning system is made of centralized HVAC equipment with an electric chiller, air-cooled condenser, and a return plenum. The total floor area is $463.6m^2$.

The building is shown in Fig. 9.7 and is included in the standard *EnergyPlus* package as *5ZoneAirCooled.idf* file.

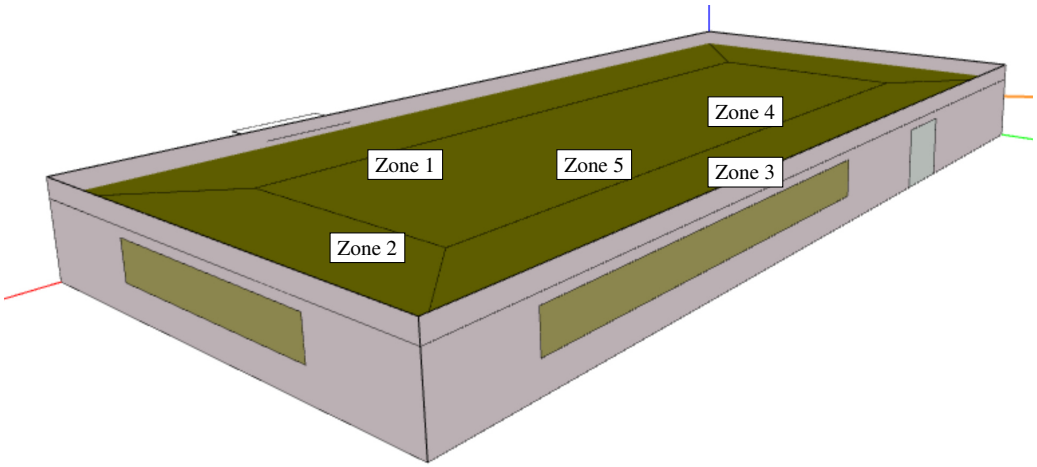


Figure 9.7: *5-zones air-conditioned building*

9.7.2 The Used Dataset

The dataset used to train the LSTM network is collected by running multiple Monte Carlo simulations with *EnergyPlus* and the 5-zones building file. To get

Table 9.2: *Samples from the dataset for July 7*

H	Zone 1 Thermostat	Zone 2 Thermostat	Zone 3 Thermostat	Zone 4 Thermostat	Zone 5 Thermostat	System Status	Air Temp. Zone 1	Air Temp. Zone 2	Air Temp. Zone 3	Air Temp. Zone 4	Air Temp. Zone 5	Outdoor Temperature	Chiller Electr. Rate
1	21.3	24.4	19.9	20.1	21.7	ON	22.2	19.2	20.6	21.5	20.6	22.0	2649.8
2	19.9	19.8	21.1	21.7	21.9	ON	21.3	21.7	19.9	20.1	21.4	21.6	2555.9
3	24.3	20.6	23.5	22.7	19.5	OFF	20.1	19.9	21.1	20.9	21.2	21.2	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
24	24.9	22.2	22.6	24.3	21.4	OFF	21.9	21.8	22.0	22.2	22.1	19.2	0

the outdoor temperature throughout the simulations, we use a weather file in *EnergyPlus Weather File* (EPW) format containing one year of historical weather data of Bari city, in Italy.

We set the time step s to one hour and we draw each thermostat set-point value, for each sample, from a uniform distribution between 19°C and 25°C. Similarly, we set the HVAC operating state to *ON* or *OFF* from a random distribution between 0 and 1, by rounding each sampled value to the nearest integer.

We collect 43,800 one-hour samples over five consecutive yearly simulations. Each sample includes the individual thermostat set-points at time step s and the HVAC operating state as the input $\mathbf{X}(s)$, and the individual indoor temperatures and the energy consumption in Watts as the output $\mathbf{T}(s)$. Table 9.2 shows some sample rows of the dataset.

9.7.3 The LSTM Training

We use the collected dataset to train the LSTM network and predict the thermodynamic behavior of the building. We present the samples to the training procedure in groups of $S = 6$ hourly time steps.

More specifically, each input sample $\mathbf{X}(s)$ is grouped with the previous $\mathbf{X}(s-1), \mathbf{X}(s-2), ...\mathbf{X}(s-5)$ samples and is paired with the output sample $\mathbf{T}(s)$, in order for the resulting LSTM network to be able to predict the indoor temperatures and energy consumption after six consecutive thermostats set-points and six HVAC operating state settings.

We run the training on a server equipped with a 14 cores *Intel Core i7* CPU and 32GB RAM. *Python Keras* library and *TensorFlow* are used to implement the LSTM network with *Adam* optimizer. The dataset is normalized and split into 80% of values for training and 20% for test. *Mean Square Error* (MSE) metric is adopted to evaluate the performance of the training procedure.

In Fig. 9.8, the results after 60 training epochs are shown. At the end of the procedure, the MSE for training and testing converges to 5e−3, allowing good predictions for the MPC Controller.

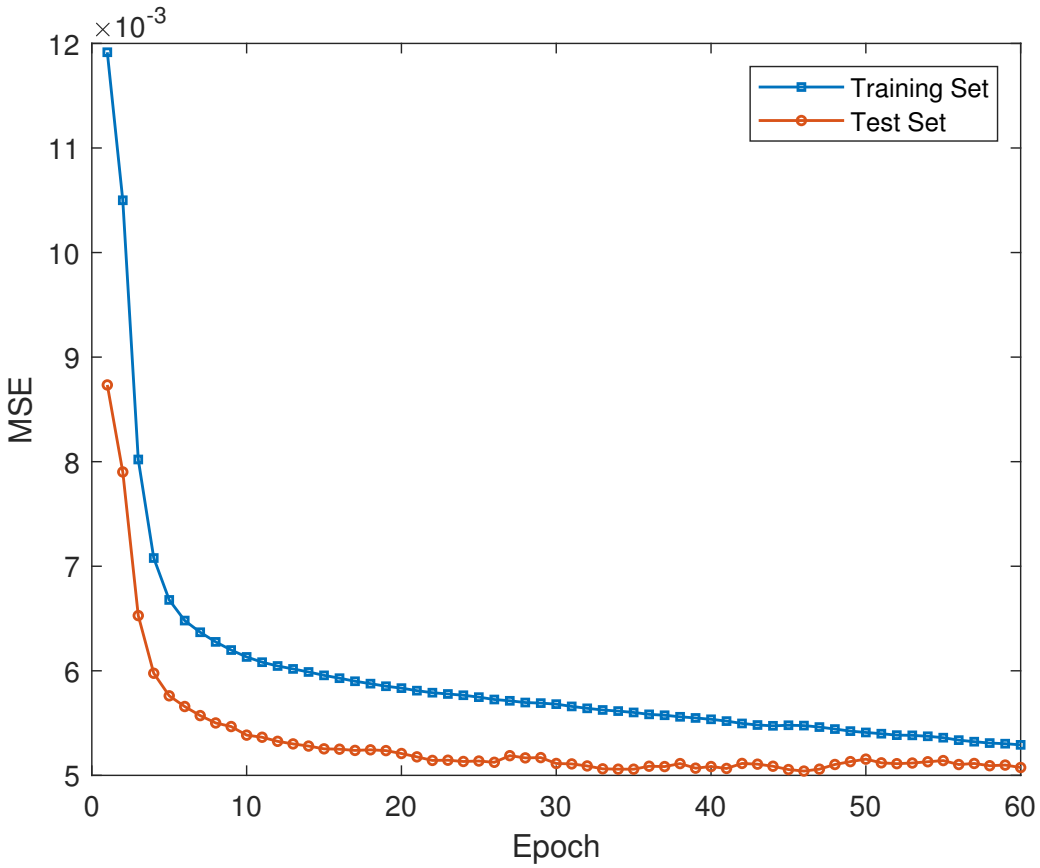


Figure 9.8: *LSTM Training*

9.7.4 The OCFP Solution

In order to assess the performance of the proposed approach, we consider for the case study a district $\mathcal{U}$ of $m = 2000$ users located in Bari (Italy) for a billing period of one week in July. We set $\overline{T_s} = 25°C$, which is the average temperature in July and $\overline{T_l} = 26°C$, that we assume as the minimum temperature prescribed by the law for public offices and institutions to save energy.

We assume that at the end of the current billing period, at time frame h, the user set $\mathcal{U}$ has been partitioned in $n = 5$ behavior classes by the *k-means* algorithm described in Section 9.5.1. Table 9.3 shows the classes resulting from applying the *k-means* clustering algorithm.

Table 9.3: *Partioned Users Set*

	Class	No.	Temperature [$^{\circ}C$]		Daily Usage [Hrs]
			Indoor	Outdoor	
1	Small	501	27.00	27.00	7.01
2	Good	317	26.00	27.00	10.76
3	Normal	291	22.05	27.50	12.48
4	Nearly Bad	516	21.53	28.00	13.71
5	Bad	375	21.17	27.87	14.10
	Total	2000			

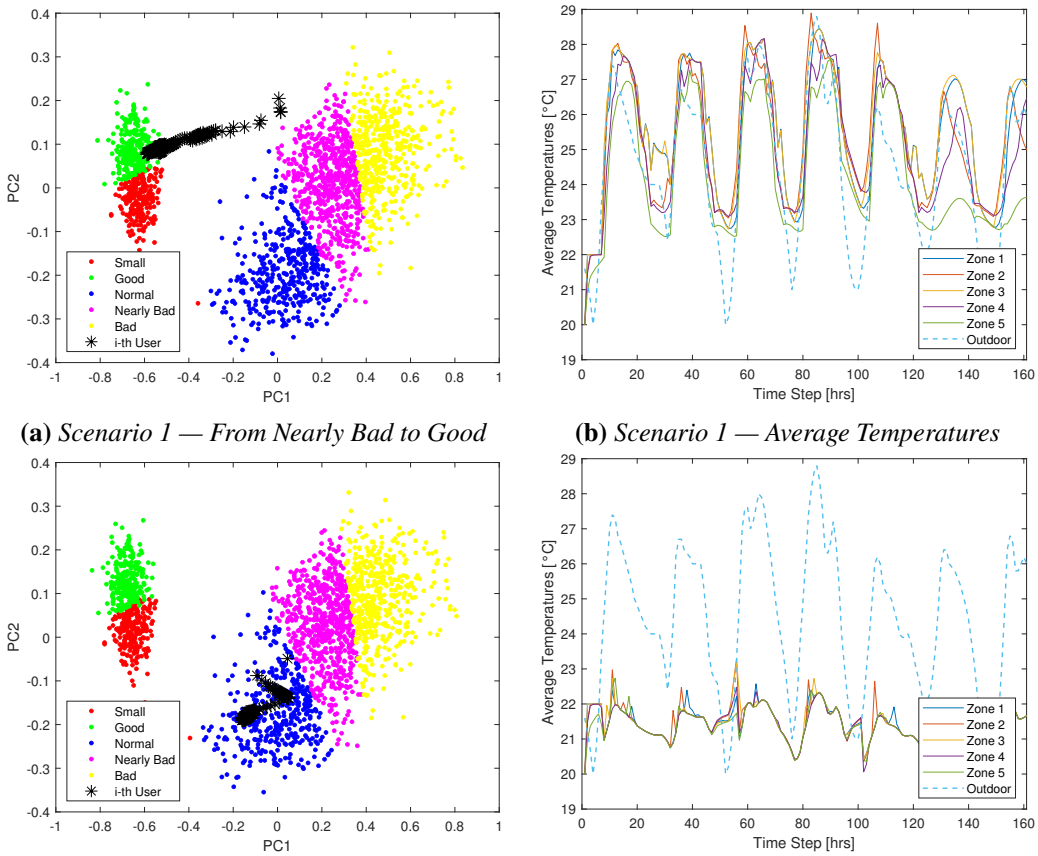
The classes are ranked based on the average indoor temperature compared to the average outdoor temperature, the seasonal average temperature and the temperature prescribed by the law. In addition, the total operational time of the HVAC is considered. The closer the indoor temperature is to the reference values and the less operational time, the better the class is ranked. Based on the above evaluations, we labeled the resulting classes from *Small* (c_1) to *Bad* (c_5).

Now, two *OCFP* scenarios for time frame $h + 1$ are proposed. In that respect, we assume that user u_i has been ranked in class *Nearly Bad* (c_4) for the current time frame h and that he wishes to scale up to class *Good* (c_2) and *Normal* (c_3) for the first and second scenario respectively.

We implement the MPC scheme by using the pre-trained LSTM to model the thermodynamics of the i-th building and we use *EnergyPlus* to simulate the plant. We set the MPC time horizon to $S = 6$ hours and we reiterate the control scheme up to the end of the next billing period. To implement and solve the lexicographic problem (9.6), we use *Python* and the SLQP optimization algorithm provided by the *SciPy* library.

The results of the class following procedure for the time frame $h + 1$ in the two considered scenarios are shown in Fig. 9.9.

In Fig. 9.9a and 9.9c, black star points represent the average centroids calculated over the whole billing period for the i-th user at each time step s for Scenario 1 and 2 respectively. We observe that, in both cases, the points move from the *Nearly Bad* (c_4) cluster to the *Good* (c_2) and *Normal* (c_3) cluster, respectively, by gradually approaching the relative cluster center. Since the *Good* (c_2) class is far more distant from *Nearly Bad* (c_4) than the *Normal* (c_3) class, the initial error for



(a) *Scenario 1 — From Nearly Bad to Good*

(b) *Scenario 1 — Average Temperatures*

(c) *Scenario 2 — From Nearly Bad to Normal*

(d) *Scenario 2 — Average Temperatures*

Figure 9.9: *Optimized Class Follower MPC scheme progress over the billing period.*

Scenario 1 is much greater than for Scenario 2. As a result, the path of Scenario 1 looks much more extended than the one of Scenario 2.

The above difference is also observed in Fig. 9.10, which shows the class tracking relative error of Scenario 1 compared to Scenario 2 and confirms that the convergence to a stable residual error of around 0.14, in the first case, takes more time steps than of Scenario 2. However, the residual error of Scenario 2 stabilizes around 0.20, which is higher than that of Scenario 1. As a result, the numerical simulations show that the proposed scheme works better when the initial error is large and the followed class is relatively distant.

In addition, in Fig. 9.9b and 9.9d the average individual indoor temperatures and the average outdoor temperature are shown. In Scenario 1, the indoor temperatures are much closer to the outdoor temperature. On the contrary, in Scenario 2, the

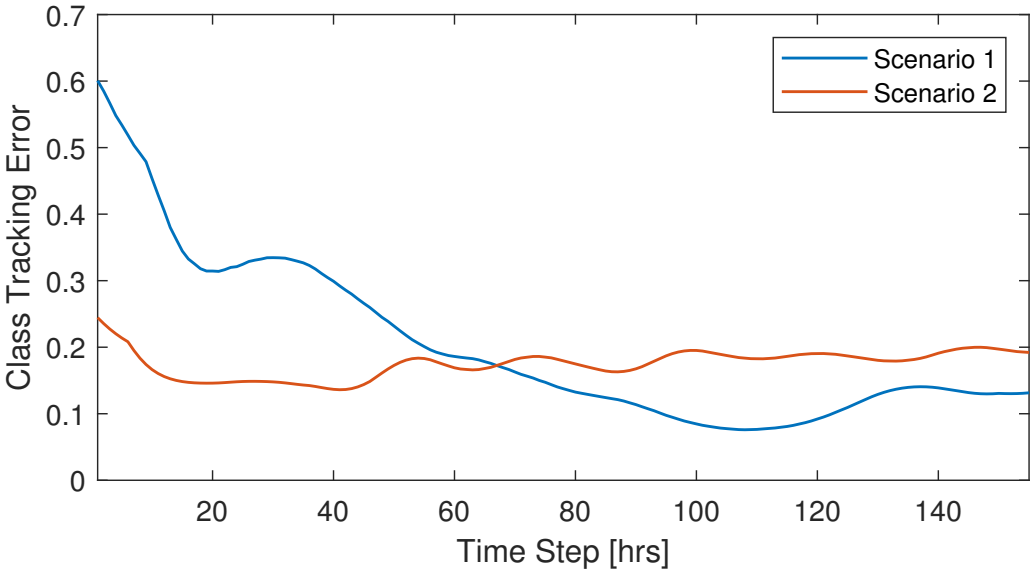


Figure 9.10: *Class Follower Tracking Error*

indoor temperatures are lower than the outdoor temperature. As observed in Table 9.3, the values obtained by the MPC in both scenarios are in line with the respective followed classes *Good* and *Normal*.

Finally, in Fig. 9.11, the cumulative energy consumption in kW over the considered billing period is shown. It is evident that, in Scenario 2, at the end of the week, the HVAC consumed more electricity than in Scenario 1. In Scenario 2, the gap between the indoor and outdoor temperature and the higher operational time leads to higher energy consumption by the HVAC to fit the following class requirements.

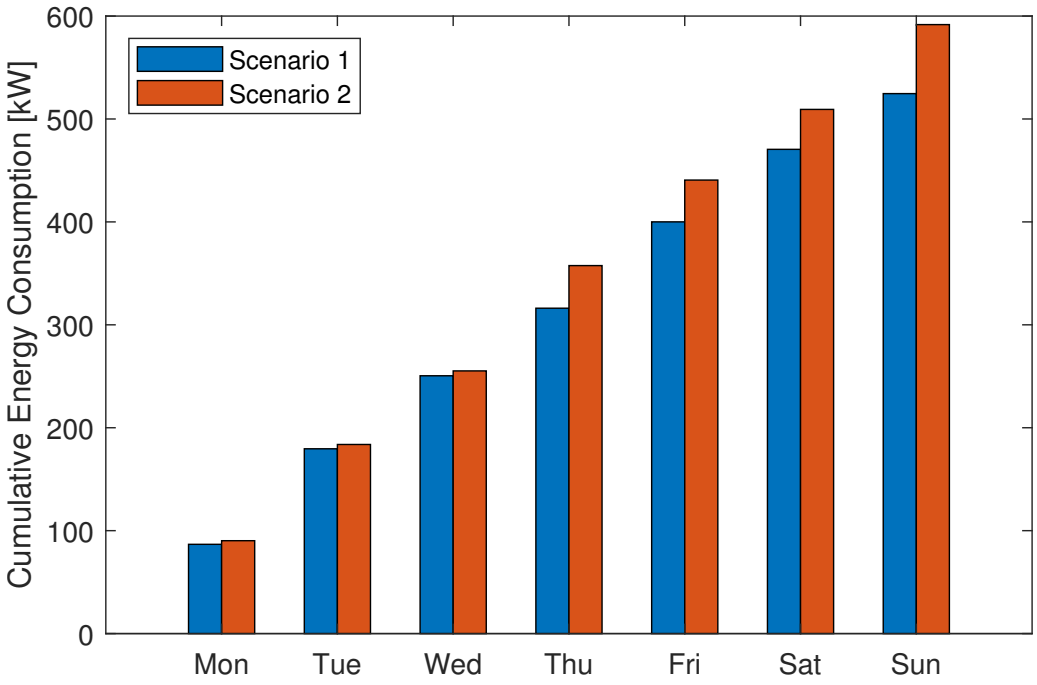


Figure 9.11: *Cumulative Energy Consumption*

9.8 Conclusion

This chapter introduces a novel approach to managing the HVAC control system at the DEMS level to optimize the user HVAC consumption. The HVAC control system includes the synergistic use of blockchains, MPC strategies and LSTM networks. In particular, the district users are divided into consumption classes, and each class is associated with a suitable energy price based on consumption behaviors. A blockchain platform ensures data integrity and facilitates transparent transactions, enables dynamic categorization of users based on their energy consumption behaviors, and processes payments on-chain. Moreover, the users can modify their behaviors and pass to a less expensive class. To this aim, the MPC strategy is proposed to provide the thermostat set points and the time intervals in which the HVAC must be switched off or switched on.

The application of the proposed model to a case study of a smart district shows promising results. The system effectively guides users from less virtuous energy consumption classes towards more energy-efficient ones while maintaining thermal comfort. Moreover, it is observed that the control scheme performs better with larger initial tracking errors, suggesting its robustness in scenarios requiring significant user behavioral adjustments.

In conclusion, the presented research highlighted the potential of combining blockchain with advanced control techniques to improve energy systems management within smart urban districts.

Future researches will study the system's scalability to larger districts, the integration with other smart grid components and the extension of extensive tests under different seasonal conditions with a broader set of user profiles. Additionally, there is an opportunity to refine the predictive algorithms, considering the impact of integrating renewable energy sources into the proposed model.

Part V

Transferability to Organizational Operations

10 Transferability Analysis across Domains

The systematic exploration of optimization paradigms across the heterogeneous landscape of cyber-physical systems undertaken in the preceding chapters reveals a fundamental architectural invariance that transcends the superficial distinctions between vehicular coordination networks, thermodynamic control systems, and organizational resource allocation frameworks. This chapter undertakes a rigorous comparative analysis that validates the methodological coherence underlying the seemingly disparate applications of Deep Reinforcement Learning for autonomous vehicle routing, Model Predictive Control augmented with Long Short-Term Memory networks for building automation, and Integer Linear Programming for personnel allocation. More critically, it establishes the theoretical foundations for a unified optimization framework whose applicability extends beyond the specific instantiations investigated herein.

The empirical evidence accumulated through these diverse deployments demonstrates that the transition from learning-based approaches in stochastic, partially observable environments to exact optimization methods in deterministic, fully specified domains represents neither methodological inconsistency nor technological regression. Rather, it constitutes a sophisticated calibration of computational strategies to the intrinsic characteristics of each problem space, where the selection of optimization paradigms emerges from principled analysis of state space complexity, temporal dynamics, information completeness, and computational constraints rather than *a priori* technological preferences. The analytical framework developed in this chapter systematically deconstructs the architectural patterns, constraint structures, and algorithmic adaptations that manifest consistently across domains, revealing how the three-layer architecture introduced in Section 4.1.1 serves not merely as a descriptive taxonomy but as a prescriptive design pattern.

Through detailed examination of the mathematical isomorphisms between collision avoidance in autonomous intersections and conflict resolution in office allocation, the dynamic threshold mechanisms linking blockchain consensus protocols to constraint relaxation strategies, and the temporal scalability enabling microsecond decisions in vehicular networks while accommodating quarterly planning cycles in organizational contexts, this analysis establishes that the observed transferability emerges from fundamental optimization principles rather than fortuitous structural similarities.

10.1 Methodological Evolution and Computational Strategies

The progression from Deep Reinforcement Learning architectures in autonomous vehicle coordination through hybrid Model Predictive Control frameworks in building automation to Integer Linear Programming formulations for organizational resource allocation embodies a deliberate calibration of computational paradigms to the fundamental characteristics distinguishing each problem domain. This methodological trajectory emerges from rigorous analysis of state space properties, temporal dynamics, information availability, and operational constraints that collectively determine the computational complexity spectrum $C : \mathcal{P} \rightarrow \mathcal{M}$, mapping problem characteristics $\mathcal{P}$ to optimal methodologies $\mathcal{M}$, where stochastic environments with partial observability naturally align with reinforcement learning, systems exhibiting strong temporal correlations benefit from predictive control, and discrete allocation problems with complete information enable exact methods guaranteeing global optimality.

The vehicular routing environment detailed in Chapter 8 necessitates Deep Reinforcement Learning precisely because the combinatorial explosion of continuous position variables, discrete traffic states, and stochastic arrival patterns renders both exhaustive enumeration and closed-form optimization computationally intractable. The modular training architecture that partitions the urban graph into N subsets where $E = E_1 \cup E_2 \cup \ldots \cup E_N$ demonstrates how hierarchical decomposition strategies manage complexity while preserving solution quality through carefully orchestrated coordination mechanisms, requiring millions of simulated trajectories for policy convergence yet enabling microsecond inference once deployed.

Conversely, the thermodynamic processes governing HVAC systems exhibit temporal dependencies and physical constraints that favor Model Predictive Control augmented with LSTM-based system identification, as established in Chapter 9. The lexicographic formulation

$$\text{lex min} \left[\sum_{k=1}^{4} (k_j(h)_k - \alpha_k(h+1))^2, \sum_{s=1}^{S} E(y_{h+1}(s), H_{h+1}(s)) \right] \tag{10.1}$$

prioritizes occupant comfort as an inviolable constraint while subsequently optimizing energy consumption within the comfort-optimal solution space, reflecting the hierarchical nature of building management objectives where human satisfaction defines system purpose rather than constituting a tradeable commodity. The LSTM component captures nonlinear relationships between weather conditions,

thermal mass, and occupancy patterns through historical sequences $\mathbf{x}_{t-w:t}$, enabling anticipatory control that exploits predictable periodicities while maintaining computational tractability through minute-scale optimization horizons.

The Integer Linear Programming approach for personnel allocation examined in Chapter 11 capitalizes on the discrete nature of office assignments $u_{nsd} \in \{0, 1\}$, complete information availability, and periodic decision horizons that characterize organizational planning processes. The objective function

$$\min \left[\lambda z - (1 - \lambda) \frac{1}{N^2 SK} \sum y_{nmsd} \right], \tag{10.2}$$

further discussed in Section 11.3.5, elegantly balances minimax fairness through the auxiliary variable z with total allocation efficiency, where the computational investment ranging from seconds for pure efficiency optimization ($\lambda = 0$) through hours for balanced multi-objective scenarios ($\lambda \approx 0.5$) remains acceptable given quarterly planning cycles. This tolerance for extended computation enables global optimality guarantees through systematic branch-and-bound exploration, validating that operational tempo τ_{op} fundamentally constrains methodological selection where $\tau_{comp} \ll \tau_{op}$ permits exact methods while $\tau_{comp} \approx \tau_{op}$ necessitates heuristic approaches.

The empirical validation presented throughout Chapters 8 through 11 confirms that matching computational strategies to problem properties yields demonstrably superior outcomes compared to universal application of any single paradigm. The implementation reality further reinforces this alignment principle through data requirements —millions of trajectories for DRL versus decades of building science encoded in differential equations for MPC versus direct policy formalization for ILP— establishing that successful optimization transcends algorithmic sophistication to encompass the complete ecosystem of data availability $\mathcal{D}$, computational resources $\mathcal{R}$, and domain expertise $\mathcal{E}$ that collectively determine feasible solution strategies. The transition from approximate to exact methods across this spectrum thus represents not a hierarchy of sophistication but rather a taxonomy of fit where each paradigm exhibits superior performance within its natural domain while proving inadequate when misapplied to foreign contexts.

10.2 Structural Parallels and Architectural Adaptations

The mathematical structures underlying the optimization frameworks deployed across autonomous vehicle coordination, building automation, and personnel allocation domains exhibit profound isomorphisms that transcend superficial implementation differences, revealing fundamental patterns in constraint formulation, adaptive mechanism design, and architectural decomposition that constitute the theoretical foundation for methodological transferability. The recurrence of these structural invariants across ostensibly disparate application contexts validates not merely the existence of universal optimization principles but more fundamentally demonstrates how abstract mathematical formulations naturally specialize to address domain-specific requirements while preserving their essential algebraic properties and computational characteristics.

The constraint structures governing resource exclusion and conflict prevention manifest with remarkable consistency across domains through binary decision variables coupled via linear inequalities, where the personnel separation constraint $u_{nsd} + u_{msd} \leq 1$ preventing conflicting individuals from occupying identical office spaces represents the same fundamental mathematical pattern as collision avoidance protocols in autonomous intersection management detailed in Chapter 5, albeit instantiated through different physical interpretations and temporal scales. This algebraic isomorphism extends beyond notational similarity to encompass the entire solution space topology, where the feasible region defined by mutual exclusion constraints exhibits identical geometric properties whether preventing vehicular collisions through spatial-temporal reservation or ensuring regulatory compliance through personnel separation, with both formulations generating polytopes whose vertices correspond to valid assignments and whose dimensionality scales linearly with the number of entities requiring coordination. The computational implications of this structural parallel prove equally significant, as solution algorithms developed for one domain transfer directly to another through appropriate variable remapping, e.g., branch-and-bound strategies for office allocation apply equally to intersection scheduling once temporal dimensions are discretized into reservation slots.

The emergence of adaptive mechanisms across all investigated domains reveals a universal requirement for dynamic response to exceptional conditions, where the mathematical formulation must gracefully transition between operational modes without compromising system integrity or requiring manual reconfiguration. The dynamic threshold formulation, as presented in Chapter 6, is developed for a blockchain consensus in vehicular networks, which modulates validation requirements

based on event severity and network conditions, finds its structural analog in the constraint relaxation mechanisms employed for personnel allocation where binary switching variables $\delta \in \{0, 1\}$ transition between aspirational targets and physical constraints when mathematical feasibility cannot be maintained. Both mechanisms encode the recognition that exceptional circumstances necessitate departure from standard protocols, implementing this adaptation through continuous functions in the vehicular domain where severity exists on a spectrum, and through discrete switches in the allocation domain where constraints either bind or relax completely. The mathematical elegance of these formulations lies in their ability to encode complex conditional logic within the optimization framework itself rather than requiring external intervention, enabling autonomous response to conditions ranging from emergency events requiring immediate information dissemination to pandemic-induced occupancy restrictions necessitating constraint relaxation.

The architectural evolution from distributed consensus in adversarial vehicular networks through hybrid trust models in building automation to centralized optimization in organizational contexts reflects a sophisticated calibration of trust mechanisms to the governance structures and threat models characterizing each domain. The vehicular environment's deployment of FilterContract validation, VotingContract consensus, and TokenContract incentivization responds to the fundamental absence of central authority and the presence of potentially malicious actors who might benefit from false information propagation, necessitating elaborate cryptographic proofs and economic mechanisms to ensure system integrity. The progressive simplification through HVAC systems, where single administrative control eliminates adversarial considerations while external market interfaces maintain blockchain integration for auditability, to personnel allocation operating entirely within organizational boundaries demonstrates that architectural complexity should align with trust requirements rather than technological capabilities. This principle extends to the computational infrastructure itself, where the transition from distributed GPU clusters training Deep Reinforcement Learning agents to single-node CPLEX optimization reflects not computational limitations but rather the recognition that solution guarantees and interpretability requirements vary fundamentally across domains.

The three-layer architectural pattern consistently manifests across all domains while adapting its specific instantiation to accommodate unique requirements, demonstrating how abstract design principles guide concrete implementations without imposing rigid structural constraints. Layer 1's data integrity mechanisms specialize from SHA-256 hashing of intersection geometries and blockchain notarization in vehicular systems through IoT sensor validation and cryptographic signatures in building automation to constraint consistency verification and canoni-

calized storage in personnel allocation, yet all instantiations address the fundamental challenge of establishing authoritative ground truth in environments subject to noise, corruption, or deception. Layer 2's decision algorithms similarly adapt from neural network policies encoding learned behaviors through model-based predictive controllers leveraging system dynamics to exact optimization engines guaranteeing global optimality, while maintaining the consistent role of transforming validated inputs into optimal actions subject to domain constraints. Layer 3's integration mechanisms evolve from TraCI-mediated simulation control through EnergyPlus co-simulation interfaces to web-based visualization platforms, yet consistently manage the complexity of translating abstract decisions into concrete system modifications while providing stakeholders with interpretable feedback regarding system performance.

The empirical validation detailed in Table 10.1 confirms that these architectural adaptations preserve functional equivalence while optimizing for domain-specific requirements, with each instantiation achieving performance metrics appropriate to its operational context, e.g., millisecond response times for vehicular routing, minute-scale optimization for HVAC control, and hour-scale computation for quarterly personnel planning. The consistency of this pattern across radically different deployment contexts, from real-time safety-critical systems to strategic organizational planning, validates its fundamental soundness as an architectural paradigm for cyber-physical optimization systems. Furthermore, the successful integration of these systems with existing infrastructure, whether through REST APIs for vehicle fleet management, BACnet protocols for building automation, or Excel interfaces for human resources departments, demonstrates that the three-layer architecture facilitates not only internal coherence but also external interoperability.

The synthesis of these structural parallels reveals that successful transfer of optimization methodologies requires not mechanical replication of techniques but rather thoughtful adaptation of core principles to domain characteristics, where the preservation of mathematical structures ensures theoretical soundness while implementation flexibility enables practical deployment. The identification of these transferable patterns, from mutual exclusion constraints through adaptive thresholds to layered architectures, provides a systematic framework for extending these methodologies to novel domains, whether in supply chain optimization, healthcare resource allocation, or smart grid management, where similar patterns of resource competition, dynamic adaptation, and multi-stakeholder coordination arise. This structural analysis thus establishes that the contributions detailed in Chapters 5, 6, 8, and 9 constitute not isolated solutions but instances of a broader optimization paradigm whose applicability extends throughout the spectrum of cyber-physical systems requiring intelligent coordination of scarce resources under

complex constraints.

Table 10.1: *Domain-specific instantiation of the three-layer architecture defined in Section 4.1.1.*

Architecture Layer	**Autonomous Vehicles**	**HVAC Control**	**Personnel Allocation**
Layer 1:	SHA-256(intersection geometry) → Merkle root → blockchain notarization via FilterContract validation	IoT anomaly detection → cryptographic signatures → on-chain storage with tamper-evident trails	Preference matrix validation → constraint consistency → canonicalized storage with version control
Layer 2:	DRL: DDPG/PPO agents with 10^6-transition replay buffer, multi-objective reward	MPC: 24h horizon, LSTM predictions, cost	ILP: CPLEX
Layer 3:	SUMO/TraCI protocol, REST endpoints, real-time dashboard, metric aggregation	EnergyPlus co-sim, demand response contracts, BACnet legacy integration, web scheduler	Web UI, Excel I/O, floorplan SVG rendering, constraint violation reports with Farkas certificates

The empirical validation across three independent deployments confirms that the abstract architectural principles articulated in Section 4.1.1 translate effectively into production systems, with the layered decomposition facilitating both rapid prototyping through component reuse and long-term maintainability through separation of concerns.

10.3 Cross-Domain Insights and Validation

The methodological convergence observed across the investigated domains of vehicular coordination, building automation, and organizational resource management transcends mere algorithmic similarity to reveal fundamental invariants in the structure of cyber-physical optimization problems, where the recurrence of specific mathematical patterns, architectural decompositions, and parametric trade-off mechanisms suggests the existence of a universal grammar for optimization that manifests with remarkable consistency regardless of whether the system coordinates silicon-based sensors or carbon-based stakeholders. The empirical validation of this methodological unity, documented through the implementations spanning Chapters 8 through 11, establishes not only the practical efficacy of individual techniques but more profoundly demonstrates how the systematic application of cryptographic identity, distributed consensus, and multi-objective optimization constitutes a coherent design philosophy whose principles remain invariant even as their technological instantiations adapt to domain-specific constraints.

The emergence of the parametric weight λ as a universal mechanism for navigating multi-objective trade-offs reveals a deeper truth about the nature of optimization in complex systems where the mathematical impossibility of simultaneously maximizing all desirable properties necessitates explicit mechanisms for encoding contextual priorities. The sensitivity analysis of this parameter across domains uncovers a fascinating phenomenon wherein the computational complexity landscape exhibits domain-specific topology: while the vehicular routing domain demonstrates smooth performance degradation as λ varies across the unit interval, reflecting the continuous nature of the underlying state space where incremental priority adjustments produce proportional behavioral modifications, the personnel allocation domain exhibits phase transitions at critical λ values where the problem structure fundamentally transforms from easily solvable to computationally intractable configurations. This differential sensitivity to parametric variation illuminates how the same mathematical formalism of weighted objective combination manifests through radically different computational signatures depending on whether the underlying decision space exhibits continuous, discrete, or hybrid characteristics, providing crucial insights for practitioners seeking to calibrate these parameters in novel application domains.

The methodological trajectory from distributed consensus mechanisms in adversarial vehicular networks through hybrid trust models in building automation to centralized optimization in organizational contexts reveals not a linear progression from complex to simple architectures but rather a sophisticated matching between trust requirements and architectural choices that reflects the fundamental

game-theoretic structure of each domain. The vehicular environment's inherent adversarial dynamics, where independent operators possess economic incentives to misreport traffic conditions or claim false priority, necessitate the elaborate cryptographic protocols and economic mechanisms detailed in Chapter 6, whose computational overhead becomes justified by the prevention of systemic manipulation that could otherwise degrade network performance or compromise safety. The building automation context exhibits an intermediate trust structure where internal sensors operate under unified administrative control, eliminating intra-system adversarial concerns, yet the interface with external energy markets and regulatory compliance requirements motivates selective deployment of blockchain technology for specific transactions requiring auditability and non-repudiation. The organizational allocation problem's complete absence of adversarial dynamics, operating entirely within a single trust boundary where the organization possesses both the authority to enforce decisions and the responsibility for their consequences, renders distributed consensus not merely redundant but actively detrimental, validating the principle that technological sophistication must align with actual trust requirements rather than being pursued for its own sake.

The temporal scalability demonstrated across applications, from microsecond-latency requirements in intersection management through minute-scale optimization horizons in climate control to quarterly planning cycles in space allocation, validates a crucial design principle wherein the relationship between decision frequency and computational intensity exhibits inverse proportionality that enables consistent solution quality across vastly different temporal scales. The autonomous vehicle domain's requirement for sub-second routing decisions precludes online optimization, motivating the extensive offline investment in Deep Reinforcement Learning where millions of simulated trajectories train policies subsequently deployed for near-instantaneous inference, effectively amortizing computational cost across countless runtime decisions. The building control system's hourly optimization cycles permit real-time Model Predictive Control where each iteration leverages warm-starting from previous solutions and linear approximations of system dynamics to achieve tractable computation times while maintaining prediction accuracy. The personnel allocation system's annual or quarterly execution frequency enables exhaustive optimization through Integer Linear Programming where solution times measured in hours remain acceptable because the strategic nature of space planning tolerates computational latency in exchange for provable optimality guarantees. This temporal adaptation mechanism reveals how the same fundamental optimization objectives (safety, efficiency, fairness) fmanifest through entirely different computational strategies depending on the characteristic timescales of system dynamics and decision requirements.

The consistent emergence of three-layer architectural patterns across all investigated domains, despite radical variations in implementation technologies and application contexts, suggests that this decomposition into data integrity, intelligent decision-making, and system integration components reflects not arbitrary design choices but rather fundamental functional requirements that any cyber-physical optimization system must address. The architectural analysis reveals how each layer serves a distinct epistemological function: Layer 1 establishes ground truth by answering "*what is the current state?*", Layer 2 determines optimal actions by resolving "*what should we do?*", and Layer 3 implements decisions by executing "*how do we actuate change?*". This functional decomposition proves remarkably stable across domains, even as the specific technologies employed within each layer vary dramatically, from SHA-256 hashing in vehicular systems to constraint validation in personnel allocation (Layer 1), from neural networks in traffic routing to linear programming in office assignment (Layer 2), from SUMO simulation interfaces to web-based visualization dashboards (Layer 3). The persistence of this pattern suggests that future applications in unexplored domains will likely exhibit similar architectural structures, providing a template for rapid prototyping and systematic development of novel cyber-physical systems.

The qualitative validation obtained through stakeholder engagement across all three domains reveals an unexpected convergence in user requirements that transcends technical specifications to encompass fundamental expectations regarding transparency, controllability, and interpretability that optimization systems must satisfy to achieve practical adoption. The unanimous preference for explicit parameter control, where users can adjust λ and observe resulting trade-offs rather than accepting black-box recommendations, reflects a deep-seated need for human agency in automated decision-making that persists regardless of whether the stakeholders are traffic engineers, facility managers, or human resource professionals. The requirement for interpretable outputs that explain not merely what decisions were made but why specific alternatives were selected or rejected, operationalized through constraint violation reports, Pareto frontier visualizations, and sensitivity analyses, demonstrates that successful cyber-physical systems must bridge the semantic gap between mathematical optimization and human intuition. The iterative refinement workflows observed across all domains, where initial solutions undergo successive modification through parameter adjustment and constraint relaxation until achieving stakeholder acceptance, reveals that optimization technology serves not to replace human judgment but to augment it through systematic exploration of solution spaces too vast for manual consideration.

The synthesis of these cross-domain insights establishes definitively that the contributions presented throughout this dissertation constitute not isolated tech-

nical achievements but rather elements of a coherent methodological framework whose applicability extends far beyond the specific systems investigated. The systematic validation across domains as diverse as autonomous transportation, building automation, and organizational management demonstrates that the proposed optimization strategies exhibit both the theoretical depth necessary for academic contribution and the practical robustness required for real-world deployment. The successful transfer of core principles (cryptographic identity establishment, multi-objective trade-off navigation, temporal adaptation, and architectural decomposition) across radically different application contexts validates the fundamental soundness of the approach while revealing specific adaptation patterns that guide future extensions. This dissertation thus confirms that the methodological framework developed herein provides a principled foundation for addressing the full spectrum of cyber-physical optimization challenges, from purely technological systems operating at microsecond timescales to human-centric processes unfolding over months, unified by common mathematical structures and architectural patterns that transcend surface-level differences to reveal deeper optimization universals.

11 Optimizing Personnel Allocation: an Integer Linear Programming Problem for Enhanced Workplace Efficiency

11.1 Introduction

In large organizations and corporations with extensive office space and numerous personnel, optimizing staff distribution within available workspaces poses significant challenges. The global shift towards hybrid work models, accelerated by the COVID-19 pandemic, has further complicated this task. Flexible schedules and fluctuating occupancy rates require dynamic solutions to ensure efficient space utilization while maintaining team cohesion and productivity. Contemporary approaches, such as smart working, co-working, and agile frameworks, reduce pressure on facilities by carefully planning human resources and scheduling on-site presence in advance. For example, structures can accommodate more workstations within the same spaces while still complying with regulations on distancing and the minimum required space per worker However, these new practices increase the complexity of assigning personnel to rooms, as management must consider absence days, minimize maximum room occupancy to prevent overcrowding, and preserve the unity of organizational units by avoiding unnecessary staff dispersion across different rooms.

A seemingly straightforward solution might involve dynamically rotating personnel among rooms, assigning them daily. However, studies have shown that this approach can negatively impact employee well-being and productivity. In fact, employees benefit from stable assignments, even to the same workstation within the same room, as they enhance their sense of belonging and reduce disruptions. In addition, dynamic assignments can increase the time it takes employees to settle in at the beginning of their workday, thereby reducing overall efficiency. The most conventional approach would be to assign all personnel from the same organizational unit to the same room. Although this may seem logical, it often leads to suboptimal space utilization and does not adequately address the complexities of modern workspace requirements [178–181].

In this article, our objective is to optimize the distribution of personnel in office environments to improve operational efficiency. We achieve this by developing an Integer Linear Programming (ILP) model that addresses the limitations of existing methods. The proposed model integrates modern working practices, individual attendance schedules, room capacity constraints, and organizational unit cohesion into a unified optimization framework. Taking into account the variability in employee attendance and the need for consistent and stable desk assignments, the approach provides a balanced solution that improves workspace efficiency while maintaining employee well-being and team collaboration.

Nevertheless, while the proposed framework accounts for flexible schedules and varying attendance patterns over a defined planning horizon (typically a working week), it does not continuously react to or adapt to minor real-time perturbations, such as sudden absenteeism or last-minute scheduling changes. The presented model emphasizes the balance between flexibility and stability, thereby providing robust solutions suitable for practical implementation without the need for constant revision of allocation decisions.

A critical aspect of developing an effective allocation strategy involves taking into account the inherent fluctuations in the presence of the employee. Personnel may take leave, maintain flexible hours, work remotely, or encounter unforeseen circumstances that prevent scheduled office attendance. Planning for full occupancy is thus impractical and can lead to inefficiencies or overcrowded workspaces. To address this, the model presented incorporates the maximum desired occupancy levels for each room, which serve as target occupancy rates below the physical capacity. This consideration accounts for attendance variations and helps prevent overcrowding, ensuring that the workspace remains comfortable and functional despite unexpected attendance fluctuations. By integrating these reduced occupancy levels into the optimization, this model aims to create a practical and resilient allocation strategy that aligns with actual usage patterns and enhances overall workplace efficiency. In addition, accommodating attendance variability enables the model to provide stable, consistent desk assignments, thus contributing to employee well-being and fostering effective team collaboration.

The remaining parts of the chapter are organized as follows: Section 11.2 provides a review of the literature and outlines the main contribution, Section 11.3 describes the problem from a mathematical point of view and identifies the decision variables, the problem's constraints, and the objective function. Section 11.4 presents a case study from the real world, outlining the outputs we expect from the system. Section 11.5 introduces the publicly accessible tool that has been developed. Eventually, Section 11.6 draws the conclusions.

11.2 Literature review and study contribution

In the literature, very few studies have explored the application of optimization methods to human resource management, and even fewer have specifically focused on the context of office space allocation.

Bosch-Sijtsema et al. [178] provided a critical examination of dynamic staff allocations and investigated the impact of drop-in desks in a large technology firm. Even if such layouts encouraged interaction and a vibrant work atmosphere, they also hindered productivity through navigation challenges, distractions, and a diminished sense of team identity. Complementing this, Nithin and Suma [182] presented a workspace management system for a software organization that incorporates activities such as layout configuration and hot desk. Although effective in managing workspaces, it primarily focused on technical tools rather than optimizing resource allocation.

Recent advances in multi-objective optimization and robust methodologies have significantly improved human resource management applications. These improvements particularly benefit personnel scheduling and allocation within healthcare systems. Shiri et al. [183] proposed a robust multi-objective framework for home healthcare scheduling and routing under uncertainty, focusing on cost minimization, skill-level optimization, and qualitative criteria in facility selection. However, their methodology presented certain limitations. Specifically, it lacked explicit constraints that addressed employee cohesion, stable workspace assignments, and the prevention of overcrowding. These factors constitute critical elements in modern office-space allocation scenarios, significantly influencing both operational efficiency and employee satisfaction. Additionally, their approach does not consider generalized workspace dynamics common in hybrid working contexts and does not account for personnel attendance variability.

Expanding on these aspects, Hamid et al. [184] recently proposed a multi-objective mathematical model for nurse scheduling that explicitly incorporates human factors, including personnel satisfaction, skill distribution, and compatibility in decision-making styles. This advancement provides valuable elements of interpersonal compatibility and satisfaction previously overlooked by Shiri et al. [183], demonstrating a deeper integration of human factors into personnel scheduling problems.

Nevertheless, several methodological limitations remain evident in their approach. Firstly, their modeling framework is highly tailored to nurse rostering within healthcare, limiting its direct applicability to broader office environments or agile workplace management contexts. Secondly, their reliance primarily on metaheuristic techniques, while effective for large-scale NP-hard problems,

lacks rigorous optimality guaranties, potentially yielding suboptimal solutions. In contrast, the approach presented here adopts an ILP-based methodology, solved via established commercial optimization solvers, and proposes a generalized formulation directly applicable to modern office working contexts.

Moreover, the work by Cakir et al. [185] presented an ILP model for multi-mode resource-constrained discrete-time cost trade-off problems in the context of software engineering projects. Their framework effectively balances conflicting project requirements under limited resources, demonstrating the potential of computational optimization methods to manage complex resource allocation challenges. Nevertheless, a key methodological limitation of their approach lies in its complexity and computational effort, especially when handling large-scale instances typical of real-world environments. Furthermore, despite introducing constraint programming and metaheuristic solutions, their model was not designed to incorporate the fluctuations typical of modern workplace attendance patterns and the variability in personnel's daily schedules. These methodological gaps are addressed in this chapter by developing an ILP-based optimization model tailored for dynamic workspace allocation.

Muñoz et al. [186] recently proposed an ILP model aimed at optimizing staff scheduling and shelf-stocking activities for retailers, considering different alternatives to shelf-stocking schedules, including day, night and semi-night shifts. Their approach offers valuable insights into workforce management in grocery retail settings, explicitly addressing cost minimization while accounting for shelf capacity constraints, labor costs, and stock-out penalty costs. Despite the effectiveness of their model in devising optimal shift schedules for shelf-stockers, a significant methodological limitation arises: their framework focuses exclusively on minimizing economic costs associated with labor allocation and stock replenishment, without considering broader socio-organizational factors such as personnel cohesion, employee satisfaction, or stable spatial allocation. Additionally, as their research is specifically tailored to the retail industry, the proposed methodology lacks the necessary generalizability for broader office workspace management contexts, thereby necessitating the development of more comprehensive approaches that transcend sector-specific constraints while maintaining the rigor of multi-objective optimization frameworks.

More specifically, regarding workspace allocation, Chiera's paper [187] on High-Performance Work Organization provides foundational principles that shape modern workspace management methodologies such as smart working and agile working. Despite its pioneering role in the literature, Chiera emphasizes the importance of a collaborative, integrated approach between management and workforce to ensure workplace efficiency and employee well-being. Nevertheless,

due to its historical context, this contribution lacks more recent advances, such as quantitative optimization techniques, formal mathematical modeling frameworks, and computational optimization methodologies, necessary to address contemporary personnel allocation challenges systematically.

A further method in resource-optimization contexts is offered by event-driven control strategies applied to hybrid discrete-event systems, particularly those modeled as generalized batch Petri nets. Liu et al. [188] introduced an ON/OFF event-driven control problem solved via linear programming (LP) formulations that drive hybrid Petri nets from blocking initial configurations to attractive steady-state regions. In a subsequent work [189], they refined this paradigm, presenting advanced event-driven controllers that maximize throughput and optimize temporal performance while maintaining batch coherence constraints and ensuring physical system stability. Both studies employed rigorous event-based LP optimization methods, conceptually similar to the ILP-based approach introduced in this work. Nonetheless, while their methodological foundations align with the optimization-oriented perspective proposed in this chapter, significant limitations arise regarding human-centric office space management. Specifically, their frameworks inherently focus on physical-process modeling aspects prevalent in high-throughput manufacturing, neglecting central human factors and socio-organizational constraints ubiquitous in modern agile workplaces. Such considerations precisely motivate the ILP formulation proposed in the current chapter.

The present work aligns with these studies by focusing on optimizing office space allocation through a collaborative, adaptive framework, specifically targeting human resources within contemporary working methodologies.

Xu et al. [190] present the development of a multiple ontology workspace management system and its performance assessment. The system aims to facilitate the storage and manipulation of ontology models for knowledge-driven manufacturing execution systems. User authentication is introduced to maintain data privacy and integrity. Performance tests demonstrate that the system offers adequate performance for a large number of users. Although their work involves managing multiple ontology workspaces, it does not directly address optimizing office space utilization in the context of modern working methodologies.

In the realm of human resource allocation, Ma et al. [191] propose a novel approach based on decision tree algorithms to allocate human resources across project groups dynamically. The authors establish a multi-objective optimization model to minimize project duration and total cost loss by dynamically allocating human resources, adhering to principles that maximize the capabilities of project management personnel, and prioritizing the needs of high-priority projects. While their methodology seeks to reduce project costs and duration, thereby improving

overall benefits for construction enterprises, their perspective differs from that adopted in this chapter, as they do not consider office space optimization.

Xiao [192] introduces an enterprise human resources optimal allocation model using particle swarm optimization and big data fusion techniques. The method constructs a database model and uses OLAP tools to build a table model for optimal human resource allocation. While the proposed method achieves better information fusion and optimized allocation of enterprise human resources, it does not specifically address the challenges of optimizing office space utilization in consideration of modern working practices, such as smart working, co-working, and agile working.

In addition to traditional methods, recent advances have seen the rise of artificial intelligence in human resources. Gupta and Kumar [193] provide a comprehensive overview of how Artificial Intelligence (AI) addresses various aspects, including recruitment, learning and development, employee experience, and data-driven decision-making. Moreover, the authors emphasize the role of AI in enhancing hiring processes and enabling data-driven decision-making. Nonetheless, they do not optimize office space allocation. Deng et al. [194] propose a simulation and optimization framework to improve thermal comfort in buildings by jointly controlling room environmental conditions and assigning occupants to rooms based on their personalized comfort preferences. They utilize machine learning models to predict individual thermal comfort levels and formulate the problem as a joint optimization of occupant-room assignment and room condition control. While their work represents a cornerstone in applying optimization techniques to personnel placement, it focuses on thermal comfort rather than the broader challenges of space optimization in office environments. However, AI models often require large datasets for training, may lack transparency in their decision-making processes, and can be challenging to interpret and validate in practice.

A notable application of optimization in resource allocation is presented by Nguyen et al. [195], who developed a solver for military training scheduling using Knuth's Dancing Links scheme. While distinct in context, their approaches share the same goal of efficient resource assignment under complex constraints, highlighting methodologies that could inspire innovations across various domains, including the optimization strategies examined in this work.

Contribution: Given the gaps in existing literature, the new contributions of this chapter are as follows: (1) an ILP model is formulated for optimizing office space allocation, specifically addressing the unique challenges in human resources management; (2) the model incorporates real-world constraints, including room capacities, personnel schedules, and organizational policies, alongside individual preferences for realistic and applicable solutions; (3) the proposed system is

validated through a detailed case study, highlighting its ability to satisfy imposed constraints while enhancing space utilization; (4) a practical tool integrated into a web interface enables human resources departments to optimize workspace allocation without requiring advanced technical expertise.

11.3 Problem formulation

In this section, we describe the problem under consideration and then define the decision variables and the constraints of the ILP.

11.3.1 Problem statement

In large organizations, staff distribution within office facilities presents significant challenges, particularly when implementing contemporary work arrangements including smart working, co-working, and agile methodologies. We consider a set of N individuals who must be allocated to S rooms for each working day of the week. Each room s is equipped with P_s workstations, and only one workstation can be assigned to each individual.

The allocation process considers several preferences and constraints:

i) **Team Cohesion:** Individuals are divided into working groups or teams. It is preferable to assign members of the same team to the same room to foster collaboration, improve communication, and enhance overall productivity.

ii) **Separation of Specific Individuals:** Certain pairs of employees cannot be assigned to the same room due to specific organizational requirements, which may include conflicts of interest, confidentiality concerns, or interpersonal issues.

iii) **Flexible Work Schedules:** Employees have the option to select which days of the week they prefer to work from home. This results in variable office attendance that must be considered in the allocation to ensure rooms are neither overutilized nor underutilized on any given day.

iv) **Consistent Workstation Assignments:** To enhance employee comfort and reduce disruptions, it is desirable for each individual to retain the same workstation on the days they are present in the office.

v) **Maximum Desired Capacity:** To prevent overcrowding and comply with health and safety regulations, each room has a maximum desired occupancy level, which is a fraction of its total capacity.

The goal is to develop an allocation strategy that respects these preferences and constraints, resulting in the optimal assignment of personnel to rooms and workstations over the planning period (e.g., one week). To address this problem, we formulate a multi-objective function aiming to:

1. **Minimize Maximum Room Occupancy:** Reduce the highest occupancy percentage among all rooms on various days to prevent overcrowding and ensure balanced space utilization.

2. **Maximize Team Cohesion:** Assign members of the same team to the same rooms whenever possible to foster collaboration and enhance group dynamics.

11.3.2 Parameters

For definitions of the parameters and notation used throughout this chapter, refer to Table 11.1.

Table 11.1: *Summary of Parameters and Notation*

Symbol	Description
N	Total number of people.
$S_R = \{s \mid s = 1, \ldots, S\}$	Set of rooms.
S	Total number of rooms.
P_s	Set of workstations in room s, with $s = 1, \ldots, S$.
I	Set of pairs of people who should *not* be assigned to the same room on the same day. Each element in I is a pair (n, m) representing individuals who should be kept separate.
G_P	Set of preference groups (e.g., organizational units or teams). Each group $G \in G_P$ is a subset of individuals who are encouraged to be assigned to the same room to promote collaboration.
$d \in \{1, \ldots, K\}$	$d = i$ for $i = 1, \ldots, K$ denotes the i-th day of the week.
K	Total number of days for which the assignment is being planned.
$a_{nd} \in \{0, 1\}$	$a_{nd} = 1$ if person n is scheduled to be in the office on day d.
D_n	Ordered set of the week days in which person n is expected to be in the office, $D_n = \{d \mid \text{person } n \text{ is in the office on day } d\}$.
$D_n(j)$	j-th element of D_n.
$max\_capacity$	Maximum desired room capacity (in the following $max\_capacity = 0.8$).
$BigM$	A big number used to linearize alternative constraints. $BigM$ is chosen to be sufficiently large to ensure that the constraints are always satisfied when necessary. In this model, $BigM$ is defined as: $BigM = 20 \times \max(N, \max_s P_s)$.

11.3.3 Decision variables

For specifications of the decision variables, please refer to Table 11.2.

Table 11.2: *Summary of Decision Variables and Notation*

Symbol	Description
$u_{nsd} \in \{0, 1\}$	$u_{nsd} = 1$ if person n is in room s during day d; 0 otherwise. This variable indicates the room occupancy of person n without specifying the exact workstation.
$v_{nspd} \in \{0, 1\}$	$v_{nspd} = 1$ if person n is assigned to workstation $p \in P_s$ in room s during day d (where d represents a specific day); 0 otherwise. It determines the specific workstation assignment.
$switch \in \{0, 1\}$	Used to select between constraints. $switch = 1$ activates the maximum desired capacity constraint, while $switch = 0$ activates the absolute maximum capacity constraint.
$z \geq 0$	An auxiliary variable representing the maximum percentage of occupancy in any room during the week, used in the minmax function.
$y_{nmsd} \in \{0, 1\}$	$y_{nmsd} = 1$ if both person n and m (where $m < n$) are assigned to room s on day d; 0 otherwise. This variable tracks the co-location of personnel.

11.3.4 Constraints

Alternative Constraints: maximum desired capacity

The first two constraints use the $switch$ binary variable that enables the model to select between the maximum desired capacity and the absolute maximum capacity constraints. The use of $BigM$ allows for the deactivation of one constraint when the other is in effect. The binary nature of $switch$ ensures that only one constraint is active at any given time, allowing the model to adapt based on specific solution needs.

Consider, for instance, the scenario in which the human resources department sets a very restrictive target for room occupancy, for example, $max\_capacity = 0.2$. In practice, this would mean that in a room with 10 available workstations $P_s = 10$, at most 2 people can be assigned to the room daily. For most real-world organizations, this constraint would be overly restrictive, potentially making the problem infeasible. Consequently, the optimization procedure would not be able to return any valid assignment, which would significantly impact the usability and practical relevance of the entire system. The introduction of the $switch$ binary variable and the $BigM$ parameter allows the proposed model to effectively handle such infeasible scenarios by dynamically activating a fallback constraint, namely the *absolute maximum capacity of a room*. In doing so, the model dynamically adapts to overly restrictive user choices, ensuring that a feasible and practical solution is always provided.

Alternative Constraint 1 - Maximum desired capacity of a room

To prevent overcrowding, the total number of personnel assigned to room s on day d must not exceed the maximum desired capacity, defined as a fraction max_capacity of the total number of available workstations P_s. This condition is expressed by:

$$\sum_{n=1}^{N} u_{nsd} \leq \text{max\_capacity} \times P_s + \text{BigM}(1 - switch) \quad \forall s \in S_R, \text{ and } d = 1, \ldots, K \tag{11.1}$$

Alternative Constraint 2 - Absolute maximum capacity of a room

As an alternative to the maximum-capacity constraint, we impose an absolute constraint that ensures the number of people in a room never exceeds the total number of workstations available there.

$$\sum_{n=1}^{N} u_{nsd} \leq P_s + BigM(switch) \quad \forall s \in S_R, \text{ and } d = 1, \ldots, K. \tag{11.2}$$

This constraint serves as a fallback if problems arise with the previous constraint or if no feasible solutions can be found under the maximum desired capacity constraint.

Constraint on the uniqueness of the room for each person for each day

To prevent overlaps and conflicts in room assignments, each person n must be assigned to at most one room on any day d when they are scheduled to be in the office.

$$\sum_{s=1}^{S} u_{nsd} \leq a_{nd} \quad \text{for } n = 1, \ldots, N;\ d = 1, \ldots, K. \tag{11.3}$$

If $a_{nd} = 1$, meaning person n is scheduled to be in the office on day d, then the left-hand side ensures that they are assigned to at most one room on that day. If $a_{nd} = 0$, indicating they are not scheduled to be in the office.

Constraint on the uniqueness of the workstation for each person per day

Each workstation p in room s on day d is assigned to at most one individual n:

$$\sum_{n=1}^{N} v_{nspd} \leq 1 \quad \forall s \in S_R, p \in P_s, d = 1, \ldots, K \tag{11.4}$$

Constraint on the specific assignment of a desk within the room

For each person n, on each day d, and in each room s, the sum of the workstation assignment variables v_{nspd} over all desks p in room s must equal the room assignment variable u_{nsd}:

$$\sum_{p=1}^{P_s} v_{nspd} = u_{nsd} \quad \forall s \in S_R, p \in P_s, d = 1, \ldots, K. \tag{11.5}$$

If $u_{nsd} = 1$, then person n is assigned to room s on day d and the summ $\sum_{p=1}^{P_s} v_{nspd}$ must equal 1. This means that person n is assigned to exactly one specific desk p within that room s on day d.

Constraint on the consistency of the specific desk

To maintain consistency in desk assignments and enhance personnel's work experience, each person n should retain the same desk p in room s on all days they are scheduled to be in the office. This consistency reduces logistical complexities and contributes to a more efficient and comfortable working environment [178–181]. The imposed constraint is the following:

$$v_{nspD_n(j)} = v_{nspD_n(j+1)} \quad \forall s \in S_R, p \in P_s, d = 1, \ldots, K \text{ and } j = 1, \ldots, \mathrm{Card}(D_n) - 1 \tag{11.6}$$

Constraint on the assignment of the person present to a desk

To ensure that each person n is assigned to exactly one desk on each day $d \in D$ when they are scheduled to be present in the office, the following condition must be satisfied:

$$\sum_{s=1}^{S} \sum_{p=1}^{P_s} v_{nspd} = a_{nd} \quad \text{for } n = 1, \ldots, N;\ d = 1, \ldots, K. \tag{11.7}$$

Constraints on the preference of units in rooms

To promote organizational cohesion and enhance collaboration among personnel, the model aims to assign individuals from the same preference group to the same room whenever feasible.

For each group $G \in G_P$, and for all pairs of personnel $n, m \in G$ with $n < m$, the following constraints link the auxiliary variable y_{nmsd} to the room assignments:

$$\begin{cases} y_{nmsd} \leq u_{nsd} & \forall n, m \in G, s \in S_R, d = 1, \ldots, K \\ y_{nmsd} \leq u_{msd} & \forall n, m \in G, s \in S_R, d = 1, \ldots, K \\ y_{nmsd} \geq u_{nsd} + u_{msd} - 1 & \forall n, m \in G, s \in S_R, d = 1, \ldots, K \end{cases} \tag{11.8}$$

This set of constraints ensures that y_{nmsd} equals 1 if both n and m are assigned to room s on day d. The condition $n < m$ ensures that each pair is considered only once, avoiding redundant calculations.

Constraint on the separation of specific people

A separation constraint is introduced to ensure that certain people, identified by the set I, are not assigned to the same room on a given day. The followin constraints prevent any pair of people (n, m) belonging to the set I from being assigned to the same room s on the same day d:

$$u_{nsd} + u_{msd} \leq 1 \quad \forall (n, m) \in I, \forall s \in S_R, d = 1, \ldots, K. \tag{11.9}$$

11.3.5 Objective Function

To achieve the outlined objectives outlined in Subsection 11.3.1, we formulate a multi-objective function combining occupancy minimization with team cohesion. Since the optimization problem is formulated as a *minimization*, we incorporate the team cohesion objective by minimizing the negative of the team cohesion term.

Our initial multi-objective function can be expressed as:

$$\min \Bigg[\lambda \cdot \max_{s,d} \left(\frac{\sum_{n=1}^{N} u_{nsd}}{P_s} \right) - (1 - \lambda) \cdot \left(\frac{1}{N^2 S K} \sum_{n,m,s,d} y_{nmsd} \right) \Bigg]. \tag{11.10}$$

Note that:

- $\lambda \in [0, 1]$ is a weighting factor balancing the emphasis on minimizing occupancy and maximizing team cohesion.
- $\max_{s,d} \left(\frac{\sum_{n=1}^{N} u_{nsd}}{P_s} \right)$ represents the maximum room occupancy ratio.
- $\sum_{n,m,s,d} y_{nmsd}$ measures the total team cohesion.

To linearize the objective function and remove the max operator, we introduce an auxiliary variable z and the following constraint:

$$z \geq \frac{\sum_{n=1}^{N} u_{nsd}}{P_s}, \quad \forall s \in S_R, d = 1, \ldots, K. \tag{11.11}$$

Thanks to (11.11), z is greater than or equal to the occupancy ratio of any room s on any day d.

With the introduction of z, we can reformulate the multi-objective function in a linearized form:

$$\min \left[\lambda z - (1 - \lambda) \cdot \left(\frac{1}{N^2 SK} \sum_{n,m,s,d} y_{nmsd} \right) \right]. \tag{11.12}$$

The first term targets the minimization of the maximum room occupancy ratio by varying the auxiliary variable z. This clearly aligns with the objective of reducing crowding across the available office spaces.

The second term is used to maximize the team cohesion. By dividing the team cohesion term by N^2SK, we ensure that this component's value ranges between 0 and 1, thereby aligning it with the range of z.

11.3.6 ILP formulation

In this subsection, we report the mathematical formulation of the proposed optimization problem:

$$\text{Minimize} \quad \lambda z - (1 - \lambda) \frac{1}{N^2 SK} \sum_{n,m,s,d} y_{nmsd}$$

subject to:

$$\sum_{n=1}^{N} u_{nsd} \leq \text{max\_capacity} \times P_s + \text{BigM}(1 - switch)$$

$$\forall s \in S_R, \text{ and } d = 1, \ldots, K$$

$$\sum_{n=1}^{N} u_{nsd} \leq P_s + BigM(switch) \forall s \in S_R, \text{ and } d = 1, \ldots, K$$

$$\sum_{s=1}^{S} u_{nsd} \leq a_{nd} \quad \text{for } n = 1, \ldots, N;\ d = 1, \ldots, K$$

$$\sum_{n=1}^{N} v_{nspd} \leq 1 \quad \forall s \in S_R, p \in P_s, d = 1, \ldots, K$$

$$\sum_{p=1}^{P_s} v_{nspd} = u_{nsd} \quad \forall s \in S_R, p \in P_s, d = 1, \ldots, K$$

$$v_{nspD_n(j)} = v_{nspD_n(j+1)} \quad \forall s \in S_R, p \in P_s, d = 1, \ldots, K \text{ and } j = 1, \ldots, \mathrm{Card}(D_n) - 1$$

$$\sum_{s=1}^{S} \sum_{p=1}^{P_s} v_{nspd} = a_{nd} \quad \text{for } n = 1, \ldots, N;\ d = 1, \ldots, K$$

$$\begin{cases} y_{nmsd} \leq u_{nsd} & \forall n, m \in G, s \in S_R, d = 1, \ldots, K \\ y_{nmsd} \leq u_{msd} & \forall n, m \in G, s \in S_R, d = 1, \ldots, K \\ y_{nmsd} \geq u_{nsd} + u_{msd} - 1 & \forall n, m \in G, s \in S_R, d = 1, \ldots, K \end{cases}$$

$$u_{nsd} + u_{msd} \leq 1 \quad \forall (n, m) \in I, \forall s \in S_R, d = 1, \ldots, K$$

$$z \geq \frac{\sum_{n=1}^{N} u_{nsd}}{P_s}, \quad \forall s \in S_R, d = 1, \ldots, K \tag{11.13}$$

11.4 Case study

This section presents the application of the described ILP problem to a real-world scenario. The case study examined staff distribution within a General Directorate of the *Ministry of the Republic of Italy*.

It is worth noting that the maximum desired room occupancy threshold of 0.8 was determined based on an internal preliminary analysis conducted by the personnel management department of the administration under consideration in this study. This analysis identified significant underutilization of existing office space, indicating an optimal occupancy threshold of 80%. Indeed, a higher theoretical occupancy would likely yield limited practical benefits due to typical underutilization caused by unpredictable absences (e.g., sick leave, training, and organizational mobility). Thus, setting the occupancy threshold at 0.8 results in realistic yet notably improved outcomes, effectively enhancing workspace quality without compromising practical feasibility.

To facilitate the validation of the optimization tool's functionality, we selected a structure with rooms specifically sized to accommodate entire organizational units. This design choice allows us to effectively monitor and assess how well the optimization model adheres to the constraints and objectives defined in the earlier sections.

Employees were instructed to choose up to 2 days of smart working or a similar arrangement, while also considering the possibility of full office presence. The choice was based on the current institutional policy and the specific needs of each employee, with the sole request to keep this day "stable" during the experimentation period. As discussed in Section 11.1, the capacity indicated here will only be the maximum possible, with other factors intervening within human resources, which inevitably lead to this value decreasing. In order to verify the functioning of the optimization system, and without prejudice to the principle whereby each employee is permanently assigned to a workstation as indicated by the constraint (11.6), the employees were distributed in the rooms starting from the assumption that the workstations were available to everyone, given the organization of the institution. The intent, therefore, is to reduce pressure on the structure by reducing overcrowding in the rooms, rather than accommodating a greater number of employees on a limited number of workstations. By doing so, the case study is sized to ensure sufficient workstations are available for all people in the organization.

The proposed case study includes twenty-five people ($N = 25$) divided into four distinct organizational units, which correspond to the preference groups G_P. There are five rooms ($S = 5$) offering a total of forty workstations ($P_1 = 8$, $P_2 = 7$,

$P_3 = 10$, $P_4 = 8$, $P_5 = 7$).

The personnel are grouped into the following preference groups:

- **Group A** ($G_A = \{1, 2, 3, 4, 23\}$).
- **Group B** ($G_B = \{5, 6, 7, 8, 9, 10, 11, 12, 24\}$).
- **Group C** ($G_C = \{13, 14, 15, 16, 17, 25\}$).
- **Group D** ($G_D = \{18, 19, 20, 21, 22\}$).

These groups represent the organizational units within the General Directorate and are used in the model to promote team cohesion by assigning members of the same group to the same rooms whenever possible. Furthermore, the model reflects the strategic decision to distribute the last three newly hired personnel units across the first three offices.

As indicated in Section 11.3 and in Constraint (11.9), certain pairs of individuals must *not* be assigned to the same room in accordance with specific organizational requirements. In the case study, we have restricted the set of such pairs to $I = \{(1, 18)\}$.

Table 11.3: *Schedule of alternative frameworks to in-person work*

Personnel	Mon	Tue	Wed	Thu	Fri
Alessandro				X	
Alessia			X		
Alice				X	
Andrea				X	
Anna					X
Aurora		X			
Chiara				X	
Davide					X
Emma					X
Francesco	X				
Gabriele	X				
Giorgia					X
Giulia			X		
Greta			X		
Lorenzo		X			
Luca				X	
Marco		X			
Martina				X	
Matteo			X		
Riccardo	X				
Simone				X	
Sofia			X		
Grazia			X		
Graziella		X	X		
Gabriella	X			X	

Each individual specified their preferred days for working remotely, resulting

in variable weekly attendance, represented in Table 11.3.

11.4.1 The optimization results

The model was implemented using the Python API of IBM ILOG CPLEX Optimization Studio®, specifically the latest version available at the time of writing (22.1.1). We executed the code on a server equipped with an AMD Ryzen 9 7950X3D processor (16 cores and 32 threads), 1.9TB of disk space, and 128GB of RAM. The choice of a high-performance machine was motivated by the need to run the script multiple times with different λ values to obtain the results presented in Table 11.4. We utilized Python libraries such as *openpyxl*, *numpy*, *pandas*, *seaborn*, and *matplotlib* for data handling and visualization.

Table 11.4: *Lambda Iterations*

Lambda	Av. Occ.	Same Team Allocation	Execution Time [H]	Ticks	Decision Variables
1.0	0.48	13	0.41	416	197
0.9	0.49	21	0.27	459	2245
0.8	0.49	21	0.26	457	2251
0.7	0.49	21	0.28	495	2336
0.6	0.49	21	1.31	2631	2297
0.5	0.49	21	4.21	6778	2262
0.4	0.49	21	6.55	10338	2251
0.3	0.49	21	7.21	11127	2692
0.2	0.49	21	15.56	20435	2445
0.1	0.49	21	22.10	29018	2255
0.0	0.50	25	0.59	1374	2367

The solutions were obtained considering different values of λ, and to analyze the results with the greatest possible level of detail, logs were also inserted into this script to automatically extrapolate significant data and merge them into a CSV file whose results are reported in Table 11.4. This table presents the optimization results and technical data, including the decision variables evaluated during the optimization process.

Upon reviewing the experimental data, it becomes evident that the system identifies three main scenarios, each reflecting a unique balance between average room occupancy and team cohesion. These solutions demonstrate the model's effectiveness in balancing the inherent trade-offs involved in personnel allocation.

In the first scenario, the system prioritizes maximizing occupancy, potentially at the expense of team cohesion ($\lambda = 1$). This solution focuses on minimizing space utilization and ensuring rooms are filled to their optimal capacity, with less emphasis on grouping individuals by team.

The second scenario, which is obtained using lambda values ranging from 0.1 to 0.9, offers a more balanced compromise. Here, the model optimizes both occupancy and team cohesion to an intermediate level, achieving a practical balance between spatial efficiency and organizational unity. This point embodies a harmonious trade-off where neither objective is strongly favored over the other.

Finally, the third scenario suggests a configuration that prioritizes team cohesion ($\lambda = 0$). In this scenario, the model prioritizes maximizing team member groupings, even if it results in lower occupancy rates. This highlights a focus on social optimization, enhancing team interaction and collaboration within shared spaces.

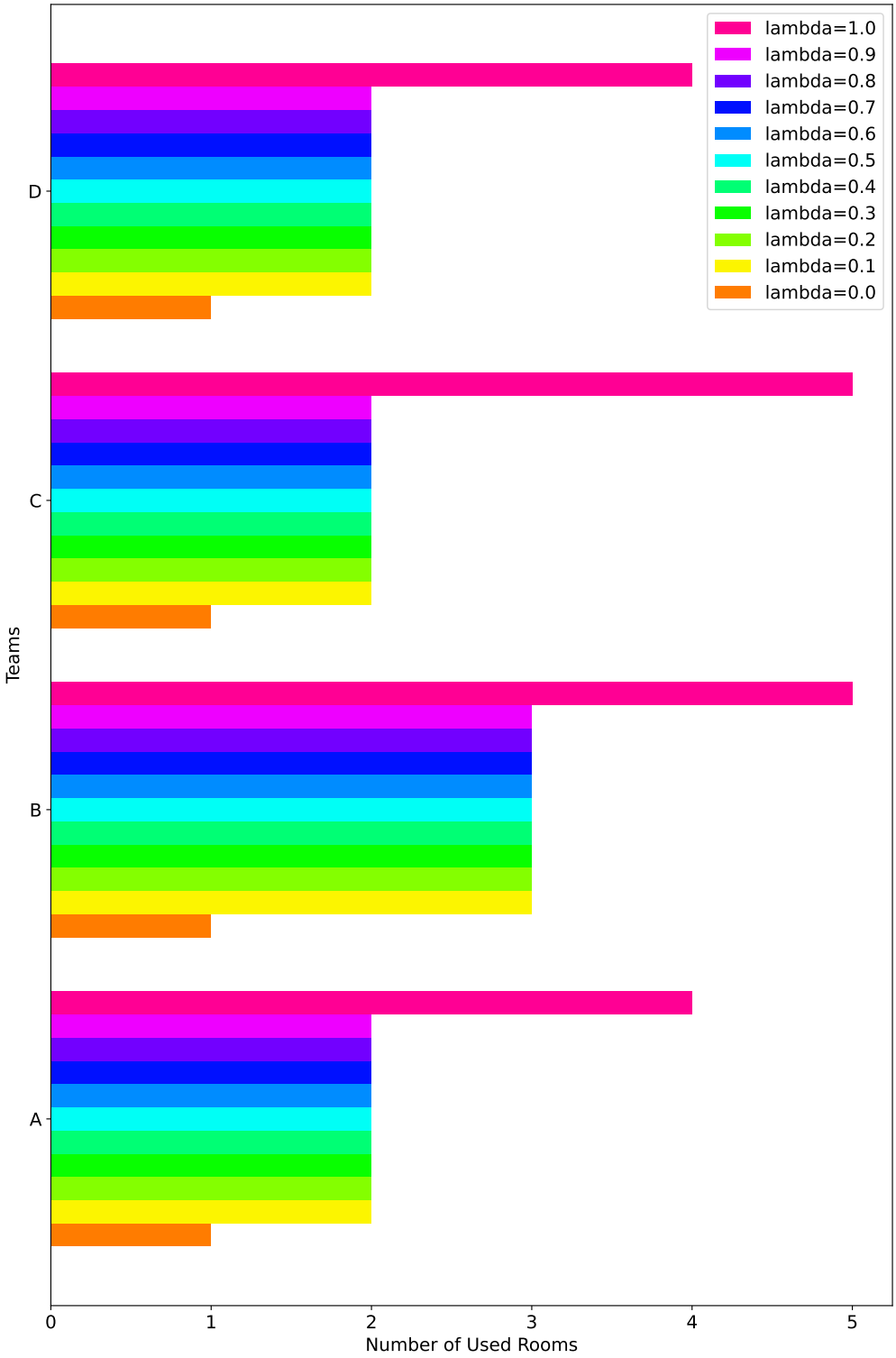


Figure 11.1: *Relationship between team distribution and the number of used rooms for* $\lambda \in [0.0, 1.0]$.

This interpretation is further supported by the trends observed in Fig. 11.1, where the relationship between organizational units and room allocation varies as a function of λ. The graph shows that as λ decreases from 1.0 to 0.0, the allocation strategy shifts in response to the multi-objective function employed. Specifically, when $\lambda = 1.0$, the model prioritizes minimizing the maximum room occupancy percentage. This leads to a configuration in which a single team uses more rooms, with more teams co-located in the same spaces, thereby maximizing space efficiency while maintaining acceptable occupancy levels.

Conversely, as λ decreases, the second term of the objective function (which favors the co-location of personnel of the same team) gains more influence. This results in a shift towards a configuration that prioritizes social optimization and team cohesion over strict occupancy efficiency. The model thus begins to agglomerate teams in the same rooms, even if this means utilizing space less efficiently and saturating occupied rooms.

This phenomenon is most evident when $\lambda = 0$, at which point the first objective of the multi-objective function is effectively nullified, leaving only the summation of y_{nmsd} over all indices n, m, s, and d to guide the optimization process. Under this extreme configuration, the model prioritizes the co-location of organizational units to the greatest extent, resulting in a scenario where all teams are allocated to a single room. This completely disregards room occupancy efficiency, as the model focuses exclusively on maximizing the team allocation preferences.

In this particular case, where the organizational units outnumber the available rooms (please note that the problem was structured with four units as opposed to five rooms), one room is left entirely vacant. A more comprehensive view of the results is depicted in Appendix A, Fig. A.1. The results shown highlight how the teams are distributed across the room for different values of λ. If $\lambda = 0$ people pertaining to the same team are assigned to only one room. On the contrary, if $\lambda = 1$ people of the same team are distributed in different rooms by maximizing the occupancy of the rooms. Intermediate solutions are obtained for $0 < \lambda < 1$, emphasizing the trade-off between optimizing for team cohesion and space efficiency.

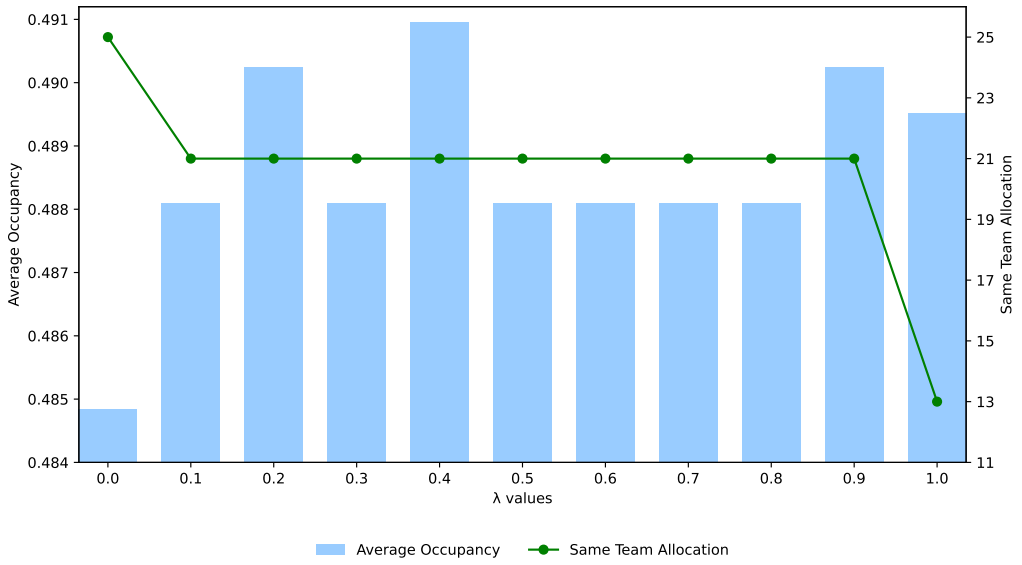


Figure 11.2: *Average occupancy and same team allocation as functions of lambda.*

For a deeper analysis of the results, Figure 11.2 simultaneously illustrates the average room occupancy and the total number of personnel allocated to the same rooms as their team colleagues across different values of λ. A superficial inspection of this graph might lead to identifying the configuration corresponding to $\lambda = 0$ as seemingly ideal. Indeed, this setting exhibits simultaneously the highest team cohesion (i.e., the maximum number of people allocated with colleagues from their teams) and the lowest corresponding average room occupancy.

Nonetheless, this conclusion is misleading when investigated more thoroughly. Figure 11.3 clearly highlights a significant drawback associated explicitly with the $\lambda = 0$ scenario, which remains unnoticed when observing only the aggregated metrics. Specifically, by closely examining the heatmap, it becomes clear that when $\lambda = 0$, one room (Room_02) remains completely unused for the entire scheduling period. Such a scenario is inherently inefficient from the perspective of facility resource allocation.

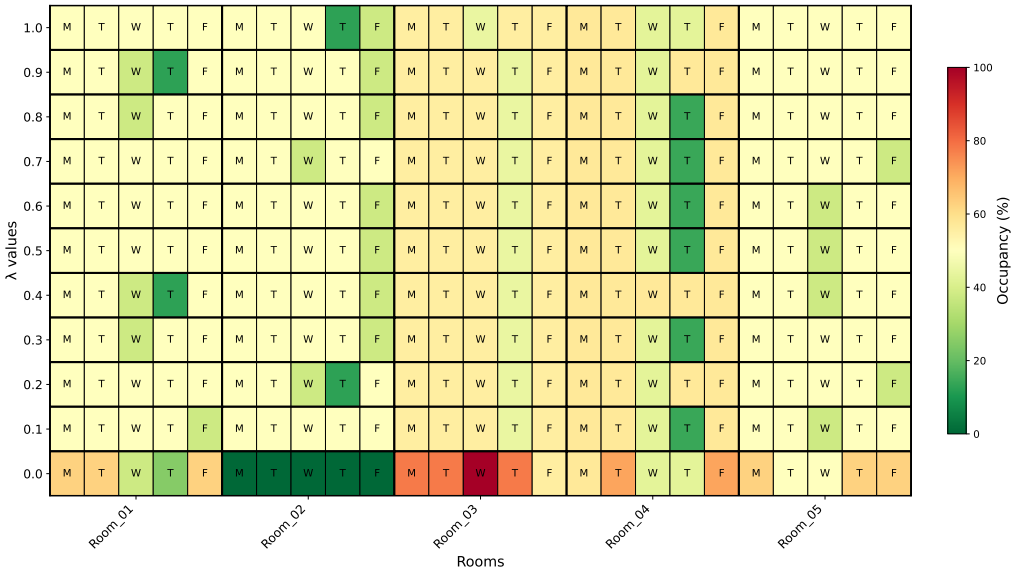


Figure 11.3: *Heatmap representing daily occupancy of rooms for different lambda values.*

Conversely, intermediate scenarios clearly emerge as the most effective solutions overall. Although Figure 11.2 does not show significant differences in numerical performance between these solutions and other intermediate ones ($\lambda \in [0.1, 0.9]$), especially with regard to the analysis of same-team allocation, the detailed occupancy analysis presented in Figure 11.3 reveals additional advantages. In particular, carefully inspecting the heatmap for $\lambda = 0.5$ and $\lambda = 0.6$, these solutions appear superior as they explicitly relieve occupancy on Wednesday and even more so on Thursday in Room_04, which is overall among the most crowded rooms. This detailed insight underscores the value of the multi-objective optimization model and the critical role of graphical visualizations in guiding informed decisions toward the optimal allocation scenario.

The obtained results confirm the model's correctness, demonstrating its ability to navigate between competing objectives depending on the chosen λ value.

To further validate the proposed ILP model and underline its advantages, we briefly analyze two alternative simplified heuristic approaches. A first candidate is the *First-Fit Decreasing* heuristic [196], commonly applied in bin packing problems, where individuals are ordered by decreasing weekly attendance frequency and sequentially allocated into the minimum number of rooms, saturating each room before opening a new one. Although straightforward, this approach tends to saturate rooms, which conflicts with our goal of minimizing maximum occupancy to ensure

workplace flexibility. A second heuristic might evenly distribute employees across available rooms, regardless of their individual attendance schedules, thereby ignoring each worker's weekly attendance variability. Such heuristic inevitably results in a suboptimal weekly occupancy distribution. Conversely, our proposed ILP explicitly considers individuals' attendance variability and optimizes weekly space usage accordingly. Moreover, thanks to its multi-objective formulation based on the parameter λ, the ILP framework offers multiple solutions that balance occupancy and team cohesion, thereby providing decision-makers with enhanced flexibility tailored to their managerial requirements.

However, based on the results discussed in this section, an organization with limited computational resources may focus exclusively on exploring solutions for $\lambda \in \{0, 0.6, 1\}$. Notably, the solution for $\lambda = 0.6$ is highly similar to that for $\lambda = 0.5$, but with a reduced computational time, according to Table 11.4.

11.5 Front-End

The final objective of the work is to provide an optimization tool for the offices responsible for personnel management. To achieve this, the structures must be equipped with a simple tool for loading data and reading outputs. Considering the project's institutional target, we ensured that the web interface can read data from a spreadsheet, with a supplied template ready for compilation from the same interface, as shown in Fig. 11.4. The tool discussed in this study is currently available at the following web address: `https://rgsgo.it/roomsoptimization`.

The back-end receives the file, extracts the data, and processes it directly. Subsequently, the Python code extracts the data from the solver and organizes it into a JSON file. This makes it easy to present the data on the front end and reorganize it into a new spreadsheet. Therefore, the human resources office can upload the staff list and their schedules to a spreadsheet, as well as the list of rooms and their corresponding positions to a second spreadsheet, which is then uploaded to the web platform. The back-end solver performs the optimization and returns a new spreadsheet ready for download containing the assignments. Furthermore, the web interface provides graphs showing the optimization results.

The developed web-based tool is characterized by strong computational efficiency, as the presented ILP formulation provides quick optimization, with typical runtimes in the order of seconds or minutes for realistic-size problems. Directly uploading Excel spreadsheets enhances ease of use, allowing the personnel management office to quickly adopt the tool without extensive technical setup or specific informatics competencies.

From a usability standpoint, the only relevant challenge encountered was related to input data preparation. Indeed, users manually compile Excel sheets containing employee schedules and room information, where the complexity of preparation scales with organization size. Hence, for larger scenarios, the greatest barrier to usability resides in data compilation and formatting. Nevertheless, initial tests with end-users reported a positive user experience due to the intuitive and straightforward interface.

Potential improvements can focus on enhancing scalability and practicality by integrating this prototype into existing management software through dedicated Application Programming Interfaces (APIs). In such scenarios, employee schedules and room assignments can be automatically retrieved, significantly reducing the data preparation burden and improving overall user experience in larger-scale organizational contexts.

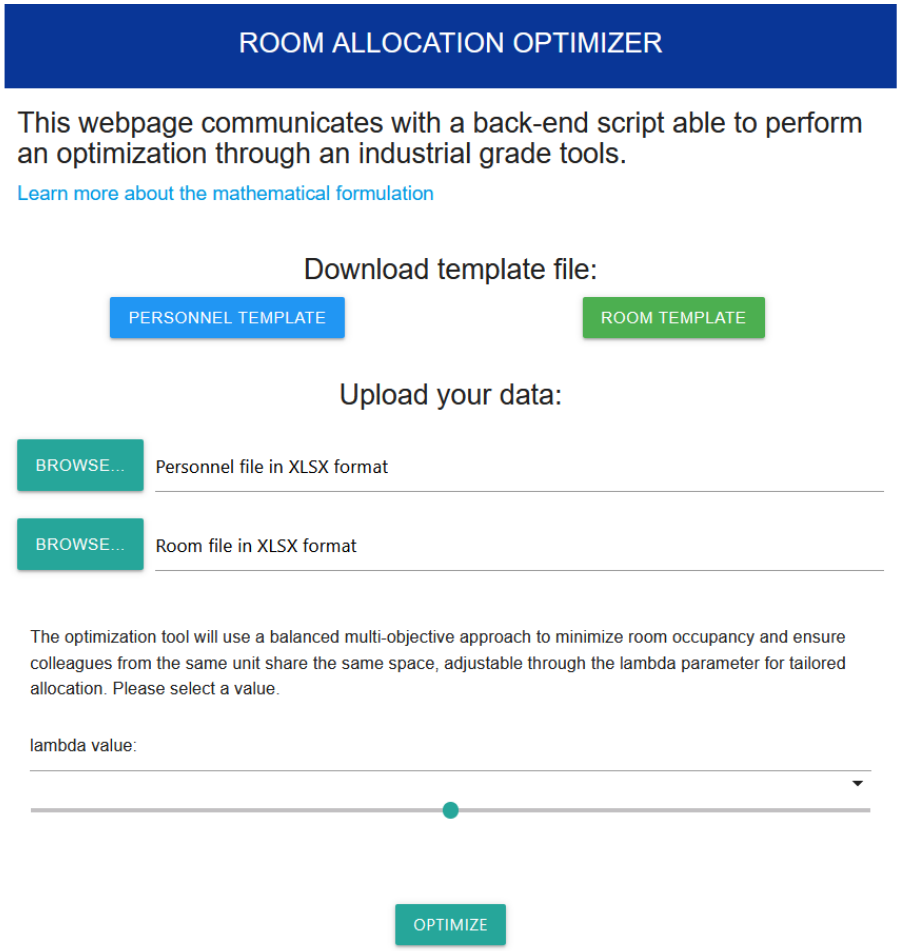


Figure 11.4: *Front-End webpage*

Given the ultimate goal of this work, the outputs that are useful to a human resources coordination office are listed in the following.

- **Distribution of people**: This output shows where each person sits in each room and on each day. In other words, the layout generated here shows for each person n in which room s they are located during day d, according to the variable u_{nsd}.

- **Room layout**: Based on the previous output, the system can generate a basic room layout indicating the workstation and room to which employees are assigned.

- **List of employee assignments**: A list containing all employees as well as the room and workstation to which they are assigned.
- **Percentage of occupancy per room and day**: For each room s and each day d, it is known how full every room actually is.

At the end of the optimization, it is possible to obtain both the graphs and files representing the optimal solution found by the implemented tool.

11.6 Conclusions

This chapter aimed to design an effective tool for optimizing staff distribution in the available rooms of companies or public buildings. To this aim, a multi-objective ILP problem is formulated to account for competing priorities, ensuring that the final solution achieves spatial efficiency while aligning with the organization's social and operational requirements.

When applying the solution in a real-world scenario, the importance of organizational units is underlined. Although we have provided the option to prioritize grouping personnel within the same team, organizational units may still emerge fragmented from a reorganization process that strongly emphasizes reducing average occupancy rather than aggregating personnel from the same units. Our approach provided management offices with a tool designed to optimize space allocation, ensure proper personnel management, and identify potential critical issues that warrant evaluation during the implementation phase.

Although the proposed approach was designed primarily for stable, mid-term planning rather than real-time adjustments, future research might explore ways to enhance model adaptability to unexpected disruptions, such as sudden absenteeism or evolving regulations. Additionally, extending our ILP framework to non-human contexts (e.g., dynamically allocating machinery, equipment, or sensors) is another promising research direction, especially when integrating *machine learning* algorithms to improve model efficiency and scalability further.

Part VI

Conclusions

12 Overall Conclusions

The present *Doctor of Philosophy in Electrical and Information Engineering* thesis, "Governance and Optimization models for Autonomous and Smart Urban Systems: A Blockchain-Enabled and Learning-Based Approach", has established a comprehensive methodological framework for designing trustworthy cyber-physical systems through the systematic integration of distributed ledger technologies with machine learning paradigms and optimization strategies. The research demonstrates that the apparent heterogeneity among autonomous vehicle coordination systems and intelligent building management infrastructures and organizational resource allocation mechanisms masks fundamental structural invariants that enable the development of transferable optimization methodologies applicable across the entire spectrum of urban cyber-physical systems.

The primary contribution lies in the formalization and empirical validation of a three-layer architectural pattern that transcends organizational convenience to represent a fundamental decomposition of cyber-physical system requirements. The foundational layer establishes cryptographic identity and data integrity through deterministic hashing mechanisms integrated with blockchain notarization protocols to provide the mathematical certainty upon which distributed trust mechanisms operate. The intermediate layer implements intelligent decision-making capabilities through the coordinated deployment of Deep Reinforcement Learning algorithms alongside Model Predictive Control strategies and Integer Linear Programming formulations while demonstrating that computational strategies must achieve alignment with inherent problem characteristics rather than pursuing technological sophistication without methodological justification. The uppermost layer bridges the semantic gap between mathematical optimization frameworks and practical deployment constraints through systematic integration protocols that ensure algorithmic sophistication translates into tangible operational improvements measurable through domain-specific performance metrics.

Empirical validation conducted across three distinct yet interconnected domains revealed critical insights into the nature of distributed cyber-physical optimization. The integration of hash-based intersection identification mechanisms with blockchain-mediated consensus protocols in autonomous vehicle coordination established a novel paradigm for trustworthy information dissemination that operates without central authorities while achieving convergence in three hours compared to seven hours required by monolithic approaches and simultaneously improving mean rewards by thirty-two percent relative to baseline implementations. The inves-

tigation of intelligent building management systems demonstrated how blockchain technology transforms traditional HVAC control architectures into participatory ecosystems where user behavior patterns converge with energy pricing dynamics and system optimization objectives through carefully orchestrated incentive mechanisms that achieve twenty percent energy reduction while maintaining thermal comfort through the synergistic integration of Long Short-Term Memory networks for thermodynamic prediction with Model Predictive Control for anticipatory optimization. The extension to organizational resource allocation validated that methodologies developed for highly dynamic stochastic environments transfer effectively to deterministic discrete optimization problems when appropriate adaptations preserve the underlying structural patterns of multi-objective trade-offs and constraint relaxation mechanisms that characterize cyber-physical optimization across domains.

The theoretical implications extend beyond specific technical contributions to illuminate fundamental principles governing trustworthy computation in distributed environments. The consistent emergence of parametric trade-off mechanisms exemplified by the weighting parameter λ that manifests across all investigated domains reveals that multi-objective optimization in cyber-physical systems invariably requires explicit mechanisms for navigating competing priorities whose relative importance varies according to stakeholder preferences and operational constraints. The differential sensitivity exhibited by this parameter across domains manifests as smooth performance degradation in continuous vehicular routing problems while producing discrete phase transitions in personnel allocation scenarios that provide crucial calibration insights for practitioners deploying these systems in novel contexts.

The methodological framework developed throughout this dissertation, while demonstrating substantial efficacy across the investigated domains, exhibits several limitations that warrant explicit acknowledgment and circumscribe the boundaries of immediate applicability. These constraints emerge from fundamental trade-offs inherent in the integration of distributed ledger technologies with learning-based optimization paradigms, and their systematic examination provides essential guidance for practitioners considering deployment in contexts beyond those explicitly validated herein.

The computational overhead imposed by blockchain operations constitutes a primary constraint that manifests with particular severity in resource-constrained edge devices demanding sub-millisecond response latencies for safety-critical applications. The gas costs associated with on-chain storage and computation, despite the optimization strategies detailed in 2.2, remain non-negligible when transaction volumes scale to metropolitan-level deployments involving thousands

of concurrent participants. The mitigation strategies explored through Layer-2 solutions and application-specific subnets reduce but do not eliminate this overhead, and the fundamental requirement for consensus among distributed validators introduces irreducible latency that conflicts with the temporal constraints of real-time vehicular collision avoidance or immediate building system fault response. Furthermore, the economic sustainability of token-based incentive mechanisms depends critically on maintaining equilibrium between reward emission and redemption demand, a balance that proves challenging to calibrate during initial deployment phases when network effects have not yet achieved critical mass.

The sample complexity inherent in Deep Reinforcement Learning algorithms presents substantial barriers to rapid adaptation when urban configurations evolve or emergency scenarios demand immediate policy recalibration. The modular training architecture presented in Chapter 8 requires approximately three hours of computation to achieve convergence even with the proposed decomposition strategy, rendering real-time adaptation to major infrastructure changes operationally infeasible. The policies learned through extensive simulation may exhibit degraded performance when deployed in environments whose statistical characteristics diverge from training distributions, a phenomenon particularly concerning for safety-critical applications where rare but consequential events by definition appear infrequently in training data. The reward function design, while carefully crafted to balance multiple objectives, encodes implicit assumptions about the relative importance of competing goals that may not align with stakeholder preferences in all deployment contexts, and the opacity of neural network decision-making processes complicates the attribution of responsibility when autonomous systems produce undesirable outcomes.

The LSTM-based system identification approach employed for building thermodynamic modeling in Chapter 9 exhibits sensitivity to distribution shift that manifests when building usage patterns, occupancy schedules, or equipment characteristics evolve beyond the conditions represented in training data. The six-hour prediction horizon, while sufficient for the investigated scenarios, may prove inadequate for buildings with substantially different thermal mass characteristics or for climatic conditions exhibiting rapid meteorological transitions. The integration with blockchain-based user classification introduces additional complexity that may exceed the technical capabilities of facility management personnel in organizations lacking dedicated data science expertise, potentially limiting adoption despite demonstrated energy savings.

The Integer Linear Programming formulation for personnel allocation, despite achieving global optimality guarantees, exhibits computational scaling characteristics that become problematic as organizational size increases beyond the

validated case study. The solution times ranging from seconds to hours observed in Table 11.4 suggest that organizations with hundreds of employees and dozens of rooms may encounter prohibitive computation requirements, particularly when exploring the multi-objective trade-off space through systematic variation of the weighting parameter λ. The assumption of stable weekly attendance patterns encoded in the constraint structure may inadequately capture the volatility characteristic of contemporary hybrid work arrangements where employee preferences evolve continuously in response to personal circumstances and organizational dynamics.

Beyond these domain-specific limitations, several cross-cutting constraints affect the framework's generalizability. The reliance on deterministic canonicalization for cryptographic commitment generation assumes that all participating systems implement identical serialization protocols, an assumption that may prove fragile when integrating legacy infrastructure or accommodating future protocol revisions. The reputation mechanisms designed to incentivize truthful reporting operate effectively only when the population of participants remains sufficiently large to provide statistical robustness against coordinated manipulation, a condition that may not hold during early deployment phases or in geographically isolated regions with limited vehicular density. The privacy implications of blockchain-based activity logging, despite the pseudonymity afforded by cryptographic addresses, warrant careful consideration in jurisdictions with stringent data protection requirements, as the immutability that provides integrity guarantees simultaneously prevents the deletion of historical records that privacy regulations may mandate.

Future research should address these limitations through several complementary directions. The integration of zero-knowledge proof systems would enable verification of computational claims without revealing underlying data, potentially reconciling the tension between transparency and privacy that currently constrains certain applications. The development of continual learning mechanisms that enable incremental model updates without catastrophic forgetting of previously acquired knowledge would enhance adaptability to evolving environments while preserving the benefits of extensive initial training. The exploration of federated learning architectures would distribute computational burden across edge devices while maintaining privacy guarantees that centralized training approaches cannot provide. The investigation of formal verification techniques for neural network policies would strengthen safety guarantees in domains where empirical validation alone proves insufficient to establish the confidence required for deployment in safety-critical contexts.

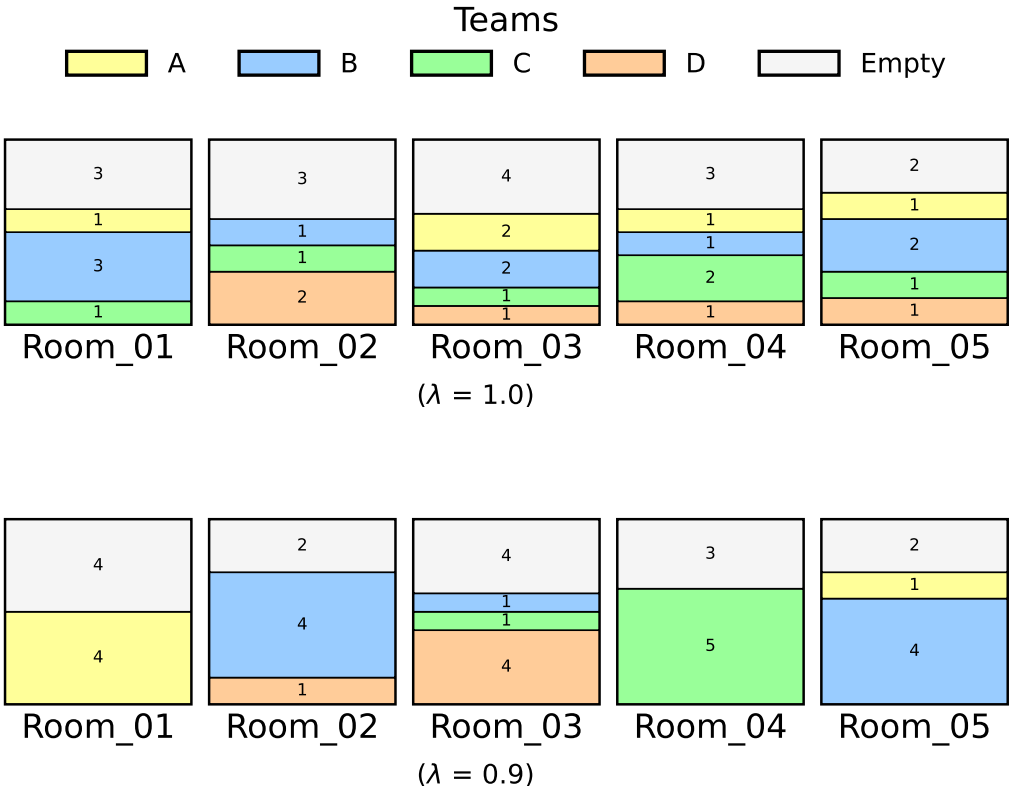
This dissertation advances the state of knowledge in cyber-physical system optimization through theoretical contributions that establish fundamental principles

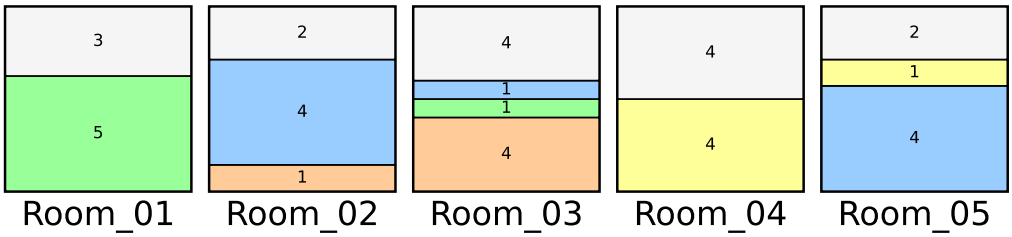
alongside practical demonstrations that validate real-world applicability. The unified methodological framework validated through diverse deployments across heterogeneous domains establishes that trustworthy distributed intelligence emerges from systematic engineering practices rather than accidental convergence of independent development efforts. As urban environments increasingly depend on autonomous systems for critical infrastructure management and service delivery the methodologies developed herein provide essential foundations for ensuring these systems operate reliably and efficiently and equitably while ultimately fostering the development of smarter and more responsive cities that enhance quality of life for their inhabitants.

Appendix

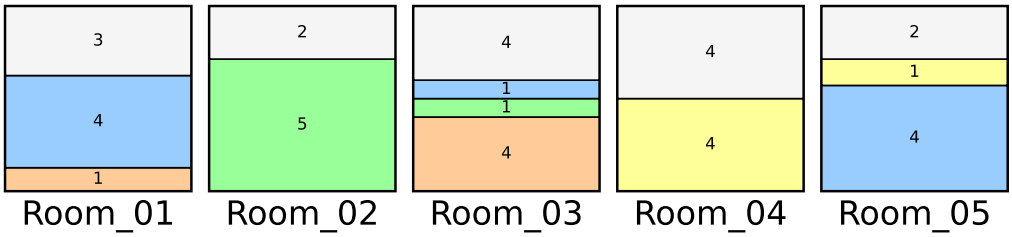
A Room Assignments Graph

In this appendix, we report additional schematic visualizations (Fig. A.1) of personnel allocation across rooms as a function of the values of λ, discussed in Chapter 11, that modify the impact of the occupancy ratio and the team presence in the rooms. In each room, different colors are assigned to each team.

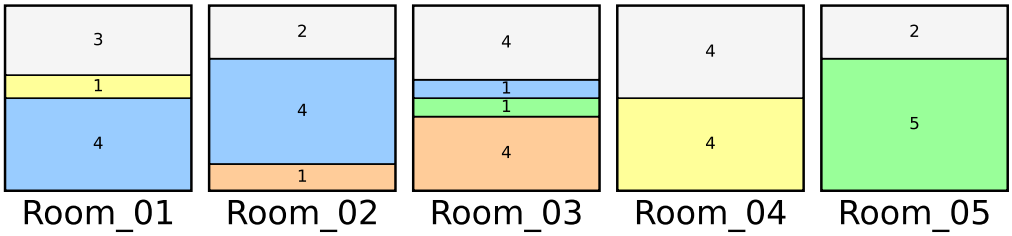




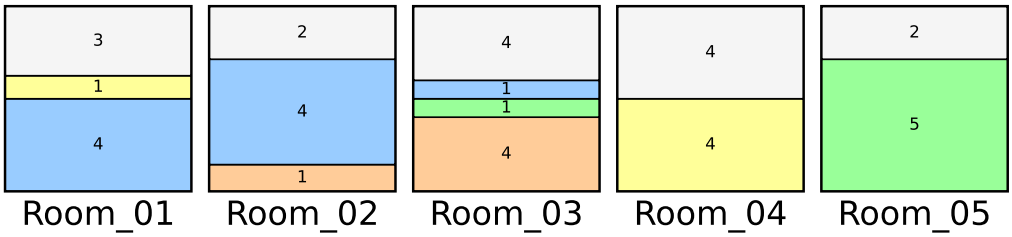
($\lambda = 0.8$)



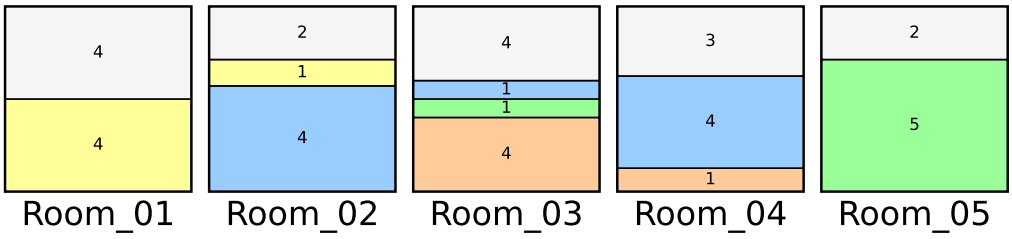
($\lambda = 0.7$)



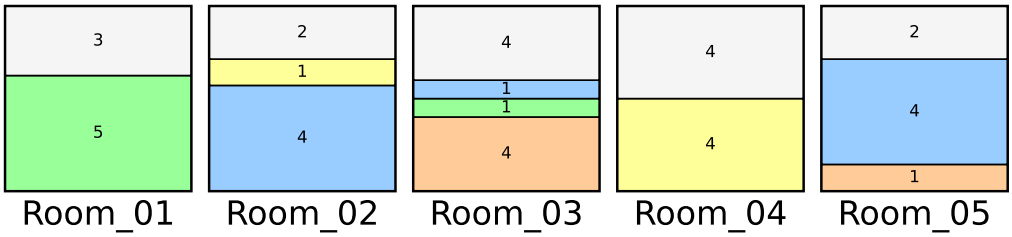
($\lambda = 0.6$)



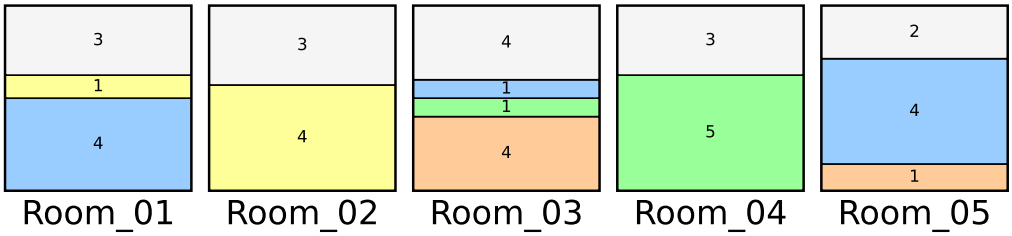
($\lambda = 0.5$)



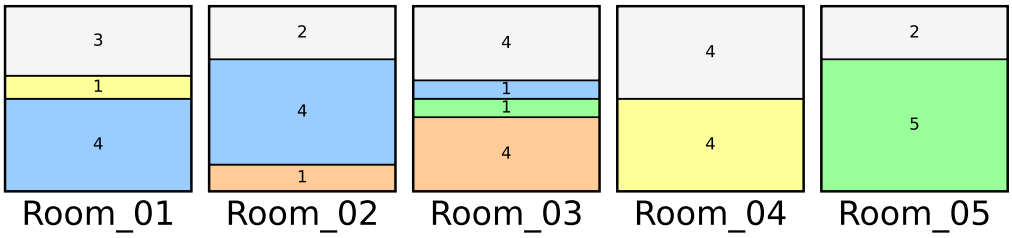
(λ = 0.4)



(λ = 0.3)



(λ = 0.2)



(λ = 0.1)

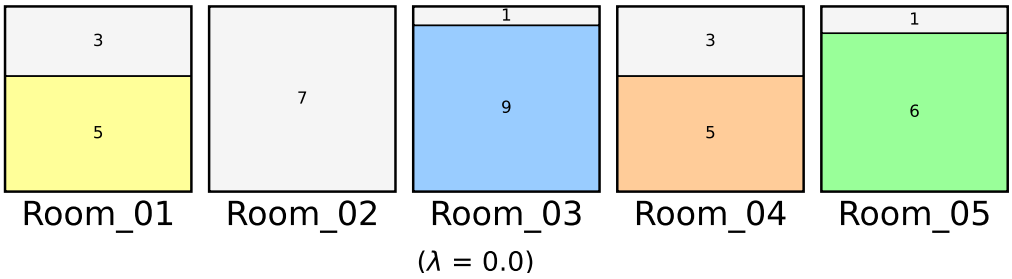


Figure A.1: *Room Assignments Graph for* $\lambda \in [0.0, 1.0]$.

B Software Disclosure

The mathematical optimization model presented in Chapter 11 was solved using IBM ILOG CPLEX Optimization Studio. Access to the fully functional version of this software, with no limitations on model size or search tree depth, was granted through the IBM Academic Initiative program. The license was obtained using institutional credentials provided by the Polytechnic University of Bari, in compliance with the Academic Initiative terms for students and researchers.

Grammarly Premium was employed for final language polishing of this manuscript. The tool relies on automated methods, including artificial intelligence-based features, to provide grammar checking and proofreading suggestions. Its use was strictly limited to surface-level linguistic refinement, such as spelling corrections, punctuation adjustments, and stylistic consistency.

The entire intellectual content, structure, and organization of this thesis remain the original work of the author, who retains full responsibility for all scientific claims and textual formulations presented herein.

Bibliography

[1] "Magic roundabout in swindon, england," 2023, [Accessed: Nov. 2, 2025]. [Online]. Available: https://goo.gl/maps/56kepQSZ8TbVr3LN6

[2] H. kng, "Map of the magic roundabout in swindon," 01 2010, [Accessed: Nov. 2, 2025]. [Online]. Available: https://commons.wikimedia.org/w/index.php?curid=9034460

[3] S.L., Freepik Company, Ning Nong, Nuaba, HAJICON, Uniconlabs, Good Ware, Talha Dogar, "Icons from flaticon used in this work," 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://www.flaticon.com

[4] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*, 2nd ed. Chapman and Hall/CRC, 2014.

[5] Q. H. Dang, "Recommendation for applications using approved hash algorithms," National Institute of Standards and Technology (NIST), Tech. Rep. SP 800-107 Rev. 1, 2012. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-107r1.pdf

[6] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL: CRC Press, 1996.

[7] National Institute of Standards and Technology, "Digital signature standard (dss)," National Institute of Standards and Technology, FIPS Publication 186-5, 2023, [Accessed: Nov. 2, 2025]. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf

[8] B. Laurie, A. Langley, and E. Kasper, "Certificate transparency," RFC 6962, Internet Engineering Task Force (IETF), jun 2013, [Accessed: Nov. 2, 2025]. [Online]. Available: https://www.rfc-editor.org/rfc/rfc6962

[9] G. Bertoni, J. Daemen, M. Peeters, and G. V. Assche, "Sponge functions," in *ECRYPT Workshop on Cryptographic Hash Functions*, Barcelona, Spain, may 2007.

[10] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, "On the indifferentiability of the sponge construction," in *Advances in Cryptology – EUROCRYPT 2008*, N. Smart, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 181–197.

[11] National Institute of Standards and Technology (NIST), "Secure hash standard (shs)," National Institute of Standards and Technology (NIST), Tech. Rep. FIPS PUB 180-4, 2015. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf

[12] A. Rundgren, B. Jordan, and S. Erdtman, "JSON Canonicalization Scheme (JCS)," RFC 8785, jun 2020. [Online]. Available: https://www.rfc-editor.org/info/rfc8785

[13] *IEEE Standard for Floating-Point Arithmetic*, IEEE Computer Society Std. IEEE Std 754-2019 (Revision of IEEE Std 754-2008), 7 2019, [Accessed: Nov. 2, 2025]. [Online]. Available: https://standards.ieee.org/ieee/754/6210/

[14] *Information technology — Microprocessor Systems — Floating-Point arithmetic*, ISO/IEC JTC 1/SC 25 Std. ISO/IEC 60 559:2020, 2020. [Online]. Available: https://www.iso.org/standard/80985.html

[15] C. Bormann and P. E. Hoffman, "Concise Binary Object Representation (CBOR)," RFC 8949, dec 2020. [Online]. Available: https://www.rfc-editor.org/info/rfc8949

[16] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Advances in Cryptology — CRYPTO '87*, C. Pomerance, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1988, pp. 369–378.

[17] M. T. Goodrich and R. Tamassia, "Authenticated data structures," in *Algorithms and Theory of Computation Handbook*, ser. LNCS 2832 (survey reprint/variant available online). Springer, 2003.

[18] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton, NJ: Princeton University Press, 2016.

[19] L. Lamport, R. E. Shostak, and M. C. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, jul 1982.

[20] Y. Li, B. Cao, M. Peng, L. Zhang, L. Zhang, D. Feng, and J. Yu, "Direct acyclic graph-based ledger for internet of things: Performance and security analysis," *IEEE/ACM Transactions on Networking*, vol. 28, no. 4, pp. 1643–1656, 2020.

[21] *Blockchain and distributed ledger technologies — Vocabulary*, International Organization for Standardization (ISO) Std. ISO 22 739:2024, jan 2024, second edition. [Online]. Available: https://www.iso.org/standard/82208.html

[22] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger (yellow paper) — berlin version," Ethereum Foundation, Tech. Rep., 2021, [Accessed: Nov. 2, 2025]. [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf

[23] Ethereum Foundation, "Ethereum virtual machine (evm)," aug 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://ethereum.org/developers/docs/evm/

[24] ——, "Ethereum accounts," jul 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://ethereum.org/developers/docs/accounts

[25] Solidity Team, "Units and globally available variables," 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://docs.soliditylang.org/en/latest/units-and-global-variables.html

[26] Ethereum Foundation, "Oracles," Oct 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://ethereum.org/developers/docs/oracles/

[27] K. Sekniqi, D. Laine, S. Buttolph, and E. Gün Sirer, "Avalanche platform," avalabs.org, Tech. Rep., 2020.

[28] V. Buterin, "Ethereum: A next-generation smart contract and decentralized application platform," White paper, 2014. [Online]. Available: https://ethereum.org/whitepaper/

[29] V. Buterin, E. Conner, R. Dudley, M. Slipper, I. Norden, and A. Bakhta, "Eip-1559: Fee market change for eth 1.0 chain," 2019, [Accessed: Nov. 2, 2025]. [Online]. Available: https://eips.ethereum.org/EIPS/eip-1559

[30] L. Gudgeon, P. Moreno-Sanchez, S. Roos, P. McCorry, and A. Gervais, "Sok: Layer-two blockchain protocols," in *Financial Cryptography and Data Security*, J. Bonneau and N. Heninger, Eds. Cham: Springer International Publishing, 2020, pp. 201–226.

[31] Ethereum Foundation, "Scaling on ethereum — layer 2 overview," Sep 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://ethereum.org/developers/docs/scaling/

[32] ——, "Scaling on ethereum — rollups," Ethereum Foundation, sep 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://ethereum.org/developers/docs/scaling/

[33] ——, "Zero-knowledge rollups," 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://ethereum.org/developers/docs/scaling/zk-rollups/

[34] Y. Hassanzadeh-Nazarabadi and S. Taheri-Boshrooyeh, "Constraint-level design of zkevms: Architectures, trade-offs, and evolution," 2025. [Online]. Available: https://arxiv.org/abs/2510.05376

[35] Ethereum Foundation, "Validium," Ethereum Foundation, 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://ethereum.org/developers/docs/scaling/validium

[36] V. Buterin, D. Feist, D. Loerakker, G. Kadianakis, M. Garnett, M. Taiwo, and A. Dietrichs, "Eip—4844: Shard blob transactions," feb 2022, [Accessed: Nov. 2, 2025]. [Online]. Available: https://eips.ethereum.org/EIPS/eip-4844

[37] O. Labs, "Transaction finality," 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://docs.optimism.io/concepts/transactions/transaction-finality

[38] M. Conforti, G. Cornuéjols, and G. Zambelli, *Integer Programming*, 1st ed., ser. Graduate Texts in Mathematics. Cham: Springer, 2014, vol. 271.

[39] L. A. Wolsey, *Integer Programming*, 2nd ed. Hoboken, NJ, USA: John Wiley & Sons, Inc, 2020.

[40] G. B. Dantzig and P. Wolfe, "Decomposition principle for linear programs," *Operations Research*, vol. 8, no. 1, pp. 101–111, 1960.

[41] J. F. Benders, "Partitioning procedures for solving mixed-variables programming problems," *Numerische Mathematik*, vol. 4, no. 1, pp. 238–252, Dec. 1962.

[42] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997.

[43] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Madison, WI: Nob Hill Publishing, LLC, 2017.

[44] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, 06 1989.

[45] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015.

[46] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York, NY, USA: John Wiley & Sons, Inc, 1994.

[47] Massachusetts Institute of Technology, "10. markov decision processes (6.390 intro to machine learning)," 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://introml.mit.edu/notes/mdp.html

[48] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, may 1992.

[49] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017. [Online]. Available: https://arxiv.org/abs/1707.06347

[50] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. Chichester, U.K.: John Wiley & Sons, Inc, 2001.

[51] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013.

[52] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[53] P. Vamplew, R. Dazeley, A. Berry, R. Issabekov, and E. Dekker, "Empirical evaluation methods for multiobjective reinforcement learning algorithms," *Machine Learning*, vol. 84, no. 1, pp. 51–80, jul 2011.

[54] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 41–48.

[55] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

[56] C. M. Bishop, *Pattern Recognition and Machine Learning*, 1st ed., ser. Information Science and Statistics. New York, NY: Springer, 2006.

[57] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '07)*. New Orleans, LA, USA: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[58] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics*, vol. 3, no. 1, pp. 1–27, 1974.

[59] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, no. 2, pp. 224–227, 1979.

[60] J. C. Gower, "A general coefficient of similarity and some of its properties," *Biometrics*, vol. 27, no. 4, pp. 857–871, 1971.

[61] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wiessner, "Microscopic traffic simulation using sumo," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2575–2582.

[62] M. Contreras and E. Gamess, "Real-time counting of vehicles stopped at a traffic light using vehicular network technology," *IEEE Access*, vol. 8, pp. 135 244–135 263, 2020.

[63] EnergyPlus Development Team, *Engineering Reference — Basis for the Zone and Air System Integration*, U.S. Department of Energy, 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://bigladdersoftware.com/epx/docs/25-1/engineering-reference/basis-for-the-zone-and-air-system-integration.html

[64] ——, *Engineering Reference — Calculation of Zone Air Temperature*, U.S. Department of Energy, 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://bigladdersoftware.com/epx/docs/25-1/engineering-reference/calculation-of-zone-air-temperature.html

[65] *Energy Standard for Sites and Buildings Except Low-Rise Residential Buildings*, ANSI/ASHRAE/IES Std. 90.1-2022, 2022, i-P Edition.

[Online]. Available: https://www.ashrae.org/technical-resources/bookstore/standard-90-1

[66] E. Liang, R. Liaw, P. Moritz, R. Nishihara, R. Fox, K. Goldberg, J. E. Gonzalez, M. I. Jordan, and I. Stoica, "Rllib: Abstractions for distributed reinforcement learning," *arXiv preprint arXiv:1712.09381*, 2018, [Accessed: Nov. 2, 2025]. [Online]. Available: https://arxiv.org/abs/1712.09381

[67] M. W. Hoffman, B. Shahriari, J. Aslanides, G. Barth-Maron, N. Momchev, D. Sinopalnikov, P. Stańczyk, S. Ramos, A. Raichuk, D. Vincent, L. Hussenot, R. Dadashi, G. Dulac-Arnold, M. Orsini, A. Jacq, J. Ferret, N. Vieillard, S. K. S. Ghasemipour, S. Girgin, O. Pietquin, F. Behbahani, T. Norman, A. Abdolmaleki, A. Cassirer, F. Yang, K. Baumli, S. Henderson, A. Friesen, R. Haroun, A. Novikov, S. G. Colmenarejo, S. Cabi, C. Gulcehre, T. L. Paine, S. Srinivasan, A. Cowie, Z. Wang, B. Piot, and N. de Freitas, "Acme: A research framework for distributed reinforcement learning," 2022. [Online]. Available: https://arxiv.org/abs/2006.00979

[68] (2025) Key insights about open energy benchmark. Open Energy Transition. [Accessed: Nov. 2, 2025]. [Online]. Available: https://openenergybenchmark.org/key-insights

[69] Linux Kernel Project, "Control group v2 — the linux kernel documentation," 2025, [Accessed: Nov. 2, 2025]. [Online]. Available: https://www.kernel.org/doc/html/latest/admin-guide/cgroup-v2.html

[70] R. L. Keeney and H. Raiffa, *Decisions with Multiple Objectives: Preferences and Value Trade-Offs*. Cambridge University Press, 1993.

[71] M. Ehrgott, *Multicriteria Optimization*, 2nd ed. Berlin, Heidelberg: Springer, may 2005, originally published as volume 491 in the series Lecture Notes in Economics and Mathematical Systems.

[72] R. V. Lenth, "Some practical guidelines for effective sample size determination," *The American Statistician*, vol. 55, no. 3, pp. 187–193, 2001.

[73] M. D. McKay, R. J. Beckman, and W. J. Conover, "Comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[74] A. M. Law, *Simulation Modeling and Analysis*, 5th ed. New York, NY, USA: McGraw-Hill Education, 2015.

[75] Y. Hochberg and A. C. Tamhane, *Multiple Comparison Procedures*. New York: John Wiley & Sons, Inc, 1987.

[76] B. McEwan, "Bonferroni correction," in *The SAGE Encyclopedia of Communication Research Methods*. Thousand Oaks, CA: SAGE Publications, Inc., 2017, vol. 4, pp. 105–106.

[77] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[78] H. B. Mann and D. R. Whitney, "On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other," *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50 – 60, 1947.

[79] M. Pease, R. Shostak, and L. Lamport, "Reaching agreement in the presence of faults," *J. ACM*, vol. 27, no. 2, p. 228–234, apr 1980.

[80] G. J. Mailath and L. Samuelson, *Repeated Games and Reputations: Long-Run Relationships*. Princeton, NJ, USA: Princeton Univ. Press, 2006.

[81] M. Nowak and K. Sigmund, "A strategy of win-stay, lose-shift that outperforms tit-for-tat in the Prisoner's Dilemma game," *Nature*, vol. 364, no. 6432, pp. 56–58, Jul. 1993.

[82] M. C. Boerlijst, M. A. Nowak, and K. Sigmund, "The logic of contrition," *Journal of Theoretical Biology*, vol. 185, no. 3, pp. 281–293, 1997.

[83] N. Case, "The evolution of trust," 2017, [Accessed: Nov. 2, 2025]. [Online]. Available: https://ncase.me/trust/

[84] X. Xie, W. Liao, H. K. Aghajan, P. Veelaert, and W. Philips, "Detecting road intersections from gps traces using longest common subsequence algorithm," *ISPRS Int. J. Geo Inf.*, vol. 6, p. 1, 2016.

[85] T. Matsushita, T. Tanaka, and M. Yonekawa, "Method for accuracy improvement of gps measurement in the city," in *2009 ICCAS-SICE*, Aug 2009, pp. 3966–3969.

[86] Y.-J. Chang and T. L. Shih, "Intersection location service and performance comparison of three location service algorithms for vehicular ad hoc networks in city environments," in *2008 3rd International Symposium on Wireless Pervasive Computing*, May 2008, pp. 562–565.

[87] J. Gao, D. Wang, C.-P. Lin, C. Luo, Y. Ruan, and M. Yuan, "Detecting and learning city intersection traffic contexts for autonomous vehicles," *Journal of Smart Cities and Society*, vol. 1, no. 3, pp. 213–239, 2022.

[88] M. Khayatian, M. Mehrabian, E. Andert, R. Dedinsky, S. Choudhary, Y. Lou, and A. Shrivastava, "A survey on intersection management of connected autonomous vehicles," *ACM Transactions on Cyber-Physical Systems*, vol. 4, no. 4, pp. 1–27, 2020.

[89] R. Jabbar, E. Dhib, A. B. Said, M. Krichen, N. Fetais, E. Zaidan, and K. Barkaoui, "Blockchain technology for intelligent transportation systems: A systematic literature review," *IEEE Access*, vol. 10, pp. 20 995–21 031, 2022.

[90] M. Eitz and G. Lixu, "Hierarchical Spatial Hashing for Real-time Collision Detection," in *IEEE International Conference on Shape Modeling and Applications 2007 (SMI '07)*. IEEE, 6 2007.

[91] M. Hausknecht, T.-C. Au, and P. Stone, "Autonomous intersection management: Multi-intersection optimization," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2011, pp. 4581–4586.

[92] Y. Li and Q. Liu, "Intersection management for autonomous vehicles with vehicle-to-infrastructure communication," *PLOS ONE*, vol. 15, no. 7, pp. 1–12, 07 2020.

[93] Z. Farkas, A. Mihály, and P. Gáspár, "Analysis of model predictive intersection control for autonomous vehicles," *Periodica Polytechnica Transportation Engineering*, vol. 51, no. 3, p. 209–215, 2023. [Online]. Available: https://pp.bme.hu/tr/article/view/22082

[94] B. Li, Y. Zhang, T. Acarman, Y. Ouyang, C. Yaman, and Y. Wang, "Lane-free autonomous intersection management: A batch-processing framework integrating reservation-based and planning-based methods," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 7915–7921.

[95] D. Isele, R. Rahimi, A. Cosgun, K. Subramanian, and K. Fujimura, "Navigating occluded intersections with autonomous vehicles using deep reinforcement learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 2034–2039.

[96] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Bitcoin.org, Tech. Rep., 2008.

[97] S. P and K. Venkatesh, "An analysis of various techniques in blockchain applications," in *2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS)*, May 2022, pp. 857–860.

[98] N. Khairina, M. K. Harahap, and J. H. Lubis, "The Authenticity of Image using Hash MD5 and Steganography Least Significant Bit," *IJISTECH (International Journal Of Information System Technology)*, vol. 2, no. 1, p. 1, nov 30 2018.

[99] Y. Dupuis, P. Merriaux, P. Subirats, R. Boutteau, X. Savatier, and P. Vasseur, "Gps-based preliminary map estimation for autonomous vehicle mission preparation," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2014, pp. 4241–4246.

[100] H. Min, X. Wu, C. Cheng, and X. Zhao, "Kinematic and dynamic vehicle model-assisted global positioning method for autonomous vehicles with low-cost gps/camera/in-vehicle sensors," *Sensors*, vol. 19, no. 24, p. 5430, 2019.

[101] M. Pál, F. Vörös, I. Elek, and B. Kovács, "Possibilities of high precision gps data in autonomous driving," *Abstracts of the ICA*, vol. 1, p. 286, 2019.

[102] J. V. Sickle, *GPS for Land Surveyors*. CRC Press, 2001.

[103] M. Padhi and R. Chaudhari, "An optimized pipelined architecture of sha-256 hash function," in *2017 7th International Symposium on Embedded Computing and System Design (ISED)*, Dec 2017, pp. 1–4.

[104] R. C. Merkle, "A certified digital signature," in *Advances in Cryptology — CRYPTO' 89 Proceedings*, G. Brassard, Ed. New York, NY: Springer New York, 1990, pp. 218–238.

[105] F. Paparella, G. Volpe, A. M. Mangini, and M. P. Fanti, "Collision avoidance strategy for autonomous intersection management by a central optimizer algorithm," in *2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2023, pp. 2826–2831.

[106] G. Difilippo, M. P. Fanti, and A. Marcello Mangini, "A consensus protocol for connecting automated vehicles at signal-free intersection," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, Dec 2022, pp. 6568–6573.

[107] D. Das, S. Banerjee, P. Chatterjee, U. Ghosh, and U. Biswas, "Blockchain for intelligent transportation systems: Applications, challenges, and opportunities," *IEEE Internet of Things Journal*, vol. 10, no. 21, pp. 18 961–18 970, Nov 2023.

[108] V. Rajkumar, E. Kavitha, E. Ranjith, and R. Aruna Kirithika, "Apco-blockchain integration for data trust and congestion control in vehicular networks," *Telecommunication Systems*, vol. 88, no. 1, p. 15, jan 2025.

[109] J. Cui, F. Ouyang, Z. Ying, L. Wei, and H. Zhong, "Secure and efficient data sharing among vehicles based on consortium blockchain," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 8857–8867, July 2022.

[110] R. Liu, W. Duan, A. M. Mangini, and M. P. Fanti, "K-protection of global secret in discrete event systems using supervisor control," in *2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2023, pp. 2832–2837.

[111] R. Liu, Y. Hu, A. M. Mangini, and M. P. Fanti, "K-corruption intermittent attacks for violating the codiagnosability," *IEEE/CAA Journal of Automatica Sinica*, vol. 12, no. 1, pp. 159–172, January 2025.

[112] D. S. Sarwatt, Y. Lin, J. Ding, Y. Sun, and H. Ning, "Metaverse for intelligent transportation systems (its): A comprehensive review of technologies, applications, implications, challenges and future directions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 7, pp. 6290–6308, July 2024.

[113] N. Alherimi, A. Saihi, and M. Ben-Daya, "A systematic review of optimization approaches employed in digital warehousing transformation," *IEEE Access*, vol. 12, pp. 145 809–145 831, 2024.

[114] S.-K. Kim and H. C. Vong, "Secured network architectures based on blockchain technologies: A systematic review," *ACM Comput. Surv.*, vol. 57, no. 7, Feb. 2025.

[115] T. Vairam and M. Srijeimathy, "Investigation of blockchain for security and transparency in intelligent transportation systems," *Procedia Computer Science*, vol. 252, pp. 851–861, 2025, 4th International Conference on Evolutionary Computing and Mobile Sustainable Networks.

[116] J. Yang, Q. Ni, G. Luo, Q. Cheng, L. Oukhellou, and S. Han, "A trustworthy internet of vehicles: The dao to safe, secure, and collaborative autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 12, pp. 4678–4681, Dec 2023.

[117] J. Rasool and S. Gupta, "Decentralised autonomous organisation based ecosystem structure for commercial companies and organisations," in *2024 First International Conference on Technological Innovations and Advance Computing (TIACOMP)*, June 2024, pp. 212–220.

[118] H. Qin, Y. Tan, Y. Chen, W. Ren, and K.-K. R. Choo, "Tribodes: A tri-blockchain-based detection and sharing scheme for dangerous road condition information in internet of vehicles," *IEEE Internet of Things Journal*, vol. 11, no. 2, pp. 3563–3577, Jan 2024.

[119] G. Olivieri, G. Volpe, A. M. Mangini, and M. Pia Fanti, "Enhancing intersection identification for autonomous vehicles: A hash-based approach," in *2024 10th International Conference on Control, Decision and Information Technologies (CoDIT)*, July 2024, pp. 700–705.

[120] P. Robinson and R. Ramesh, "Layer 2 atomic cross-blockchain function calls," *CoRR*, vol. abs/2005.09790, 2020.

[121] G. Olivieri, "Hardhat simulation repository," [Accessed: Nov. 2, 2025]. [Online]. Available: https://github.com/GSEPE/BC-AV-dataSharing-hardhat.git

[122] L. Liang, H. Ye, and G. Y. Li, "Toward intelligent vehicular networks: A machine learning framework," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 124–135, Feb 2019.

[123] A. Pompigna and R. Mauro, "Smart roads: A state of the art of highways innovations in the smart age," *Engineering Science and Technology, an International Journal*, vol. 25, p. 100986, 2022.

[124] M. P. Fanti, A. Rinaldi, M. Roccotelli, B. Silvestri, S. Porru, and F. E. Pani, "Software requirements and use cases for electric light vehicles management," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2018, pp. 311–316.

[125] M. P. Fanti, A. M. Mangini, M. Roccotelli, B. Silvestri, and S. Digiesi, "Electric vehicle fleet relocation management for sharing systems based

on incentive mechanism," in *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*, 2019, pp. 1048–1053.

[126] A. M. Mangini and M. Roccotelli, "Innovative services for electric mobility based on virtual sensors and petri nets," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 9, pp. 1845–1859, 2023.

[127] X. Ma, Y. Xie, and C. Chigan, "Graph convolutional network based multi-objective meta-deep q-learning for eco-routing," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–16, 2024.

[128] A. Sharma, A. Sharma, P. Nikashina, V. Gavrilenko, A. Tselykh, A. Bozhenyuk, M. Masud, and H. Meshref, "A graph neural network (gnn)-based approach for real-time estimation of traffic speed in sustainable smart cities," *Sustainability*, vol. 15, no. 15, 2023.

[129] G. Qin, Q. Luo, Y. Yin, J. Sun, and J. Ye, "Optimizing matching time intervals for ride-hailing services using reinforcement learning," *Transportation Research Part C: Emerging Technologies*, vol. 129, p. 103239, 2021.

[130] J. Teusch, J. N. Gremmel, C. Koetsier, F. T. Johora, M. Sester, D. M. Woisetschläger, and J. P. Müller, "A systematic literature review on machine learning in shared mobility," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 4, pp. 870–899, 2023.

[131] J. Amarnath J, R. S, and K. K. S, "Route-based user segmentation and clustering: A machine learning approach for enhanced transportation service," in *2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS)*, Dec 2023, pp. 1–4.

[132] C. Chen, L. Li, M. Li, R. Li, Z. Wang, F. Wu, and C. Xiang, "curl: A generic framework for bi-criteria optimum path-finding based on deep reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 2, pp. 1949–1961, 2023.

[133] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. A. Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909–4926, June 2022.

[134] F. Shafique, T. Naeem, and A. H. Farooqi, "Path tracking and obstacle avoidance control using deep reinforcement learning for autonomous vehicles," in *2023 17th International Conference on Open Source Systems and Technologies (ICOSST)*, 2023, pp. 1–6.

[135] B. Durgabhavani, V. B. Reddy Muvva, L. Leo Joseph, D. M. Babu, and M. H. T., "Adaptive path planning for autonomous vehicles in complex traffic scenarios," in *2023 International Conference on Evolutionary Algorithms and Soft Computing Techniques (EASCT)*, 2023, pp. 1–5.

[136] A. Kusari, P. Li, H. Yang, N. Punshi, M. Rasulis, S. Bogard, and D. J. LeBlanc, "Enhancing sumo simulator for simulation based testing and validation of autonomous vehicles," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, 2022, pp. 829–835.

[137] E. Liang, R. Liaw, R. Nishihara, P. Moritz, R. Fox, K. Goldberg, J. Gonzalez, M. Jordan, and I. Stoica, "RLlib: Abstractions for distributed reinforcement learning," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 3053–3062.

[138] X. Deng, Y. Zhang, Y. Zhang, and H. Qi, "Toward smart multizone hvac control by combining context-aware system and deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 9, no. 21, pp. 21 010–21 024, Nov 2022.

[139] O. Van Cutsem, D. Ho Dac, P. Boudou, and M. Kayal, "Cooperative energy management of a community of smart-buildings: A blockchain approach," *International Journal of Electrical Power & Energy Systems*, vol. 117, p. 105643, 2020.

[140] O. Erdinç, A. Taşcikaraoğlu, N. G. Paterakis, and J. P. S. Catalão, "Novel incentive mechanism for end-users enrolled in dlc-based demand response programs within stochastic planning context," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 2, pp. 1476–1487, Feb 2019.

[141] Q. Shi, C.-F. Chen, A. Mammoli, and F. Li, "Estimating the profile of incentive-based demand response (ibdr) by integrating technical models and social-behavioral factors," *IEEE Transactions on Smart Grid*, vol. 11, no. 1, pp. 171–183, Jan 2020.

[142] A. Mnatsakanyan, A. H. AlMazrooqi, E. B. Muruaga, and P. A. Banda, "Electricity market structure with individual pricing mechanism," in *2018*

9th IEEE International Symposium on Power Electronics for Distributed Generation Systems (PEDG), June 2018, pp. 1–6.

[143] I. Brahmia, J. Wang, H. Xu, H. Wang, and L. D. O. Turci, "Robust data predictive control framework for smart multi-microgrid energy dispatch considering electricity market uncertainty," *IEEE Access*, vol. 9, pp. 32 390–32 404, 2021.

[144] M. Biemann, P. A. Gunkel, F. Scheller, L. Huang, and X. Liu, "Data center hvac control harnessing flexibility potential via real-time pricing cost optimization using reinforcement learning," *IEEE Internet of Things Journal*, vol. 10, no. 15, pp. 13 876–13 894, Aug 2023.

[145] J. P. Tan, A. L. A. Ramos, M. V. Abante, R. L. Tadeo, and R. R. Lansigan, "A performance review of recurrent neural networks long short-term memory (lstm)," in *2022 3rd International Conference for Emerging Technology (INCET)*, May 2022, pp. 1–5.

[146] R. Kwadzogah, M. Zhou, and S. Li, "Model predictive control for hvac systems — a review," in *2013 IEEE International Conference on Automation Science and Engineering (CASE)*, Aug 2013, pp. 442–447.

[147] G. Olivieri, G. Volpe, A. M. Mangini, and M. Pia Fanti, "A district energy management approach based on internet of things and blockchain," in *2022 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, Sep. 2022, pp. 1–6.

[148] F. Casino, T. K. Dasaklis, and C. Patsakis, "A systematic literature review of blockchain-based applications: Current status, classification and open issues," *Telematics and Informatics*, vol. 36, pp. 55 – 81, 2019.

[149] M. J. M. Chowdhury, A. Colman, M. A. Kabir, J. Han, and P. Sarda, "Blockchain versus database: A critical analysis," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, Aug 2018, pp. 1348–1353.

[150] P. Haji Mirzaee, M. Shojafar, H. Cruickshank, and R. Tafazolli, "Smart grid security and privacy: From conventional to machine learning issues (threats and countermeasures)," *IEEE Access*, vol. 10, pp. 52 922–52 954, 2022.

[151] M. B. Mollah, J. Zhao, D. Niyato, K.-Y. Lam, X. Zhang, A. M. Y. M. Ghias, L. H. Koh, and L. Yang, "Blockchain for future smart grid: A comprehensive survey," *IEEE Internet of Things Journal*, vol. 8, no. 1, pp. 18–43, 2021.

[152] C. Lazaroiu and M. Roscia, "Smart district through iot and blockchain," in *2017 IEEE 6th International Conference on Renewable Energy Research and Applications (ICRERA)*, 2017, pp. 454–461.

[153] K. Christidis and M. Devetsikiotis, "Blockchains and Smart Contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[154] S. Benedict, P. Rumaise, and J. Kaur, "Iot Blockchain Solution for Air Quality Monitoring in SmartCities," in *2019 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. IEEE, 12 2019.

[155] P. Zhuang, T. Zamir, and H. Liang, "Blockchain for cybersecurity in smart grid: A comprehensive survey," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 1, pp. 3–19, Jan 2021.

[156] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts," in *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 5 2016.

[157] X. Li, H. Jiao, L. Cheng, Y. Yin, H. Li, W. Mu, and R. Zhang, "A Quantitative and Qualitative Review of Blockchain Research from 2015 to 2021," *Sustainability*, vol. 15, no. 6, p. 5067, mar 13 2023.

[158] T. B. Malla, A. Bhattarai, A. Parajuli, A. Shrestha, B. B. Chhetri, and K. Chapagain, "Status, Challenges and Future Directions of Blockchain Technology in Power System: A State of Art Review," *Energies*, vol. 15, no. 22, p. 8571, nov 16 2022.

[159] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, and J. Yellick, "Hyperledger fabric," in *Proceedings of the Thirteenth EuroSys Conference*. ACM, apr 23 2018.

[160] Z. Wu and W. Chu, "Sampling strategy analysis of machine learning models for energy consumption prediction," in *2021 IEEE 9th International Conference on Smart Energy Grid Engineering (SEGE)*, Aug 2021, pp. 77–81.

[161] M. Roccotelli, A. M. Mangini, and M. P. Fanti, "Smart District Energy Management With Cooperative Microgrids," *IEEE Access*, vol. 10, pp. 36 311–36 326, 2022.

[162] D. Muralidar, Meikandasivam, and Vijayakumar, "A review about energy management techniques in industrial buildings," in *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)*. IEEE, 4 2017.

[163] A. Rajith, S. Soki, and M. Hiroshi, "Real-time optimized HVAC control system on top of an IoT framework," in *2018 Third International Conference on Fog and Mobile Edge Computing (FMEC)*. IEEE, 4 2018.

[164] W.-H. Chen, S. Yang, and F. You, "Thermal comfort control on sustainable building via data-driven robust model predictive control," in *2023 American Control Conference (ACC)*, May 2023, pp. 591–596.

[165] R. Eini and S. Abdelwahed, "Learning-based model predictive control for smart building thermal management," in *2019 IEEE 16th International Conference on Smart Cities: Improving Quality of Life Using ICT and IoT and AI (HONET-ICT)*, Oct 2019, pp. 038–042.

[166] R. Z. Homod, K. S. Mohamed Sahari, H. A. Almurib, and F. H. Nagi, "Rlf and ts fuzzy model identification of indoor thermal comfort based on pmv/ppd," *Building and Environment*, vol. 49, pp. 141–153, 2012.

[167] S. Taheri, P. Hosseini, and A. Razban, "Model predictive control of heating, ventilation, and air conditioning (hvac) systems: A state-of-the-art review," *Journal of Building Engineering*, vol. 60, 11 2022.

[168] K. Friansa, J. Pradipta, I. N. Haq, P. H. K. Utama, M. Wasesa, and E. Leksono, "Enhancing hvac electricity load prediction accuracy using bi-lstm method based on daily dataset," in *2023 8th International Conference on Instrumentation, Control, and Automation (ICA)*, Aug 2023, pp. 92–96.

[169] X. Wang, X. Wang, L. Wang, L. Jiang, and Y. Zhan, "A distributed fusion lstm model to forecast temperature and relative humidity in smart buildings," in *2021 IEEE 16th Conference on Industrial Electronics and Applications (ICIEA)*, Aug 2021, pp. 1–6.

[170] R. E. Alden, H. Gong, E. S. Jones, C. Ababei, and D. M. Ionel, "Artificial intelligence method for the forecast and separation of total and hvac loads with application to energy management of smart and nze homes," *IEEE Access*, vol. 9, pp. 160 497–160 509, 2021.

[171] M. Westerkamp, "Verifiable smart contract portability," in *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, May 2019, pp. 1–9.

[172] C. Wu, J. Xiong, H. Xiong, Y. Zhao, and W. Yi, "A review on recent progress of smart contract in blockchain," *IEEE Access*, vol. 10, pp. 50 839–50 863, 2022.

[173] I. Jolliffe and Springer-Verlag, *Principal Component Analysis*, ser. Springer Series in Statistics. Springer, 2002.

[174] A. Pozzi and D. Toti, "Lexicographic model predictive control strategy in ageing-aware optimal charging procedure for lithium-ion batteries," *Computers & Chemical Engineering*, vol. 163, p. 107847, 2022.

[175] J. Bi, X. Zhang, H. Yuan, J. Zhang, and M. Zhou, "A hybrid prediction method for realistic network traffic with temporal convolutional network and lstm," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 1869–1879, July 2022.

[176] H. Wang, B. Lu, J. Li, T. Liu, Y. Xing, C. Lv, D. Cao, J. Li, J. Zhang, and E. Hashemi, "Risk assessment and mitigation in local path planning for autonomous vehicles with lstm based predictive model," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 2738–2749, Oct 2022.

[177] S. J. W. Jorge Nocedal, *Sequential Quadratic Programming*. New York, NY: Springer New York, 2006, pp. 529–562.

[178] P. M. Bosch-Sijtsema, V. Ruohomäki, and M. Vartiainen, "Multi-locational knowledge workers in the office: navigation, disturbances and effectiveness," *New Technology, Work and Employment*, vol. 25, no. 3, pp. 183–195, nov 2010.

[179] G. Ditchburn, "The rise and fall of the hot desk: Say hello to activity-based working," 2014, [Accessed: Nov. 11, 2025]. [Online]. Available: https://theconversation.com/the-rise-and-fall-of-the-hot-desk-say-hello-to-activity-based-working-26622

[180] C. V. Mercer, "Shift work: The social, psychological, and physical consequences," *Social Forces*, vol. 44, no. 4, pp. 593–594, 1966.

[181] C. Shepheard, "Working on the workplace," *Manager Magazine*, vol. Q2, pp. 24–26, 2015, [Accessed: Nov. 11, 2025]. [Online]. Available: https://api.semanticscholar.org/CorpusID:168742267

[182] M. Nithin and V. Suma, "Workspace management and hot-seating," in *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, June 2017, pp. 900–903.

[183] M. Shiri, F. Ahmadizar, and H. Mahmoudzadeh, "A three-phase methodology for home healthcare routing and scheduling under uncertainty," *Computers & Industrial Engineering*, vol. 158, p. 107416, 2021.

[184] M. Hamid, R. Tavakkoli-Moghaddam, F. Golpaygani, and B. Vahedi-Nouri, "A multi-objective model for a nurse scheduling problem by emphasizing human factors," *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, vol. 234, no. 2, pp. 179–199, 2020, pMID: 31755354.

[185] G. Cakir, K. Subulan, S. T. Yildiz, A. Hamzadayi, and C. Asilkefeli, "A comparative study of modeling and solution approaches for the multi-mode resource-constrained discrete time-cost trade-off problem: Case study of an erp implementation project," *Computers & Industrial Engineering*, vol. 169, p. 108201, 2022.

[186] R. Muñoz, J.-C. Muñoz, J.-C. Ferrer, V. I. González, and C. A. Henao, "When should shelf stocking be done at night? a workforce management optimization approach for retailers," *Computers & Industrial Engineering*, vol. 190, p. 110025, 2024.

[187] E. Chiera, "High performance work organization - a promising future for american industry and organized labor," *Control Engineering Practice*, vol. 2, no. 4, pp. 677–687, 1994.

[188] R. Liu, R. Ammour, L. Brenner, and I. Demongodin, "On/off control for reaching a steady state attractive region in batches petri nets," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 9618–9623, 2023, 22nd IFAC World Congress.

[189] ——, "Event-driven control of hybrid systems using batches petri nets: Application to high throughput manufacturing systems," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 11 906–11 919, 2025.

[190] X. Xu, G. M. Luis, A. Lobov, and J. L. Martinez Lastra, "Multiple ontology workspace management and performance assessment," in *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, July 2015, pp. 1063–1068.

[191] C. Ma, S. Zhang, J. Zhuo, Y. Liu, and Y. Zhou, "Research on project group human resource allocation of construction enterprises based on decision tree algorithm," in *2022 2nd International Conference on Networking, Communications and Information Technology (NetCIT)*, Dec 2022, pp. 193–196.

[192] L. Xiao, "Optimal allocation model of enterprise human resources based on particle swarm optimization," in *2020 International Conference on Computer Information and Big Data Applications (CIBDA)*, April 2020, pp. 249–253.

[193] C. P. Gupta and V. V. R. Kumar, "Artificial intelligence application in human resource management: The way forward," in *2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS)*, Jan 2024, pp. 1726–1730.

[194] M. Deng, B. Fu, and C. C. Menassa, "Room match: Achieving thermal comfort through smart space allocation and environmental control in buildings," in *2021 Winter Simulation Conference (WSC)*, Dec 2021, pp. 1–11.

[195] V. Nguyen, V. Mak-Hau, B. Moran, and A. Novak, "An efficient and exact algorithm for military timetabling and trainee assignment problems," *Computers & Industrial Engineering*, vol. 169, p. 108192, 2022.

[196] M. Yue and L. Zhang, "A simple proof of the inequality mffd(l) $\leq$ 71/60 opt(l) + 1, l for the mffd bin-packing algorithm," *Acta Mathematicae Applicatae Sinica*, vol. 11, no. 3, pp. 318–330, 1995.