Distributed Multi-Agent System for Real-Time Traffic Optimization: A Comparative Analysis of Advanced Greedy Heuristics in Urban Congestion Management

Walid Kherchofi[1*], Amal Khamal[1]

[1] Department of Computer Science, University of Djelfa, 17000, Algeria

* dz.scholar@mail.com

https://doi.org/10.70695/IAAI202504A2

Abstract

Traffic congestion is a major problem in many regions, especially in large cities. Several factors contribute to this situation, such as inadequate road infrastructure and the high number of vehicles, particularly during peak hours: in the morning from 7 a.m. to 8 a.m., at noon from 12 p.m. to 1 p.m., and from 3 p.m. to 5 p.m.. Our main objective is to propose a practical and adaptable approach to managing congestion by using the capabilities of multi-agent systems to make quick and relevant decisions in real time. We aim to make traffic flow smoother, reduce pollution, and make driving more pleasant and safe for all road users. Our project proposes an agent-based simulation to manage road traffic. This approach involves the cooperation between different actors, such as drivers and traffic management systems, to optimize traffic flow and reduce congestion in cities. The overall waiting time of vehicles on the roads is then evaluated using different heuristics like First In First Ou (FIFO) and two versions of the Greedy heuristic, one simple and the other advanced. In the end, a comparison between these three heuristics shows that the advanced Greedy heuristic better optimizes traffic management and reduces vehicle waiting times.

Keywords Traffic Management; Multi-Agent Systems; FIFO Heuristic; Greedy Heuristic; Advanced Greedy Heuristic; Traffic Optimization; Agent-Based Simulation

1 Introduction

The management of urban traffic congestion presents a formidable challenge to modern infrastructure and city planning. Congestion is identified as a major issue, particularly prevalent in large metropolitan areas. This systemic difficulty stems from several contributing factors, primarily inadequate road infrastructure failing to keep pace with the growing high number of vehicles. The problem intensifies acutely during specific peak hours, defined in the morning from 7 a.m. to 8 a.m., at noon from 12 p.m. to 1 p.m., and in the late afternoon from 3 p.m. to 5 p.m..

The adverse effects of this congestion are extensive and multifaceted. They include significant economic losses due to reduced productivity, increased atmospheric pollution, and a measurable decrease in driver safety and overall satisfaction for all road users. Consequently, effective traffic management necessitates proposing a practical and adaptable approach capable of responding to these dynamic environmental changes in real-time.

Traditional centralized traffic management systems, often relying on fixed light cycles or basic density controls, frequently lack the requisite adaptability and resilience needed in dynamic urban settings. When faced with highly distributed problems, such as managing the flow of hundreds or thousands of vehicles across multiple intersections, centralized models inevitably encounter computational bottlenecks and communication delays, significantly limiting their scalability.

The principal objective of this research is to propose, implement, and rigorously evaluate a practical Multi- Agent System designed specifically for road traffic simulation and management [1-3]. The core optimization goal driving the system's design and performance metrics is the minimization of the Global Vehicle Waiting Time ($\sum T_{wait}$). The primary contribution lies in the design, implementation (utilizing the JADE framework), and validation of a novel Advanced Greedy Heuristic. Through comparative evaluation, this advanced approach is shown to significantly outperform traditional traffic handling

methods, such as FIFO, and simpler Greedy optimization strategies, primarily by enabling sophisticated, conflict-free simultaneous passage of vehicles at congested intersections [4-5].

The domain of Intelligent Transportation Systems (ITS) has seen extensive research focused on optimizing urban mobility. Much of the previous work has centered on developing centralized control strategies or applying complex machine learning algorithms, such as Reinforcement Learning (RL), to optimize traffic signal control policies across multi-intersection vehicular networks. While systems utilizing RL demonstrate effectiveness in scheduling, they often impose significant computational requirements, requiring extensive training periods and substantial resources for value function approximation [6-7].

In contrast, the presented work focuses on optimizing well-defined heuristic rules within a distributed architecture. This approach, centered on agent-based simulation, is recognized as essential for modeling and simulating complex, microscopic phenomena like emergent congestion, allowing for the observation of collective behaviors arising from the actions of individual entities (vehicles). This focus ensures that the management strategies developed are suitable for rapid, real-time deployment without the heavy overhead associated with deep learning models [8-10]. The types of multi-agent systems are shown in Figure 1.

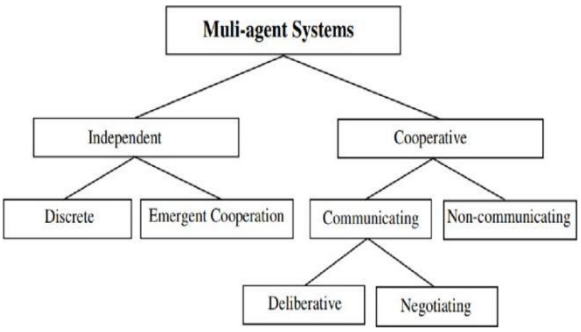

Fig. 1. Multi-Agent Systems classification

The foundation of the proposed system rests on established concepts of agent technology. An agent is defined as an autonomous entity capable of perceiving its environment via sensors, engaging in reasoning and decision-making, and subsequently acting upon that environment through actuators to achieve its defined objectives.

For effective traffic management, agents must possess specific capabilities:

Autonomy: The ability to act independently and manage their own state.

Reactivity: The capacity to respond swiftly and appropriately to sudden changes in the environment, such as new traffic conditions or administrative orders.

Proactivity: The capacity to set and pursue goals on their own initiative (e.g., seeking an optimal route).

Social Aptitude: The ability to interact with other agents through cooperation, communication, and negotiation to achieve system-wide goals.

1.1 Formal Problem Definition

The core research problem is the minimization of urban traffic congestion at an intersection by developing a practical, real-time, and adaptive traffic control strategy within a Distributed Multi-Agent System (MAS) architecture.

The objective of the system is to determine the optimal set of actions (grant passage or enforce stop) for vehicles approaching a four-way intersection at each discrete time step, t, such that the Global Vehicle Waiting Time over the entire simulation is minimized.

Objective Function: Minimize Global Waiting Time (TGlobal) The function is defined as the sum of accumulated waiting times for all vehicles in the system:

$$Minimize\ T_{Global} = \sum_{i=1}^{N} (T_{wait})_i \qquad (1)$$

Where:

N is the total number of Driver Agents (vehicles) in the system. (Twait)i is the accumulated time vehicle i spends stationary or delayed in the road queues (V1,V2,H1,H2). Waiting time increases by one unit for every time step a vehicle remains stationary.

The resulting Multi-Agent System (SMA) is characterized by the presence of multiple autonomous agents interacting asynchronously within a shared environment. A defining characteristic of such distributed systems is the absence of comprehensive global control, meaning the emergent system behaviour arises from the localized interactions of the individual agents.

2 System Architecture and Design using Agent-Oriented Modeling

The rigorous design process utilized for this system ensures that functional requirements are translated into a robust and implementable MAS structure. This process relies on two established agent-oriented methodologies: VOYELLE for architectural decomposition and AUML for detailed design specification.

2.1 The VOYELLE Methodology (AEIO)

The Agent, Environment, Interaction, Organization (VOYELLE) methodology provides a structured framework for the analysis and decomposition of complex agent-based systems. By addressing the four dimensions, VOYELLE ensures that all aspects of the system, from the individual entity to the network structure, are systematically analysed.

The analysis phase focused on identifying:

1.Users: The primary users were identified as the driver and the administrator/system itself.

2.Agents: Identification of the core software entities that represent user roles and system functions (Driver Agent and Administrator Agent).

3.Interactions: Defining the message exchanges necessary for communication and coordination.

4.Environment: Defining the scope and context within which agents operate (in this case, the specific intersection and the presence of other agents).

5.Organization: Establishing the hierarchical structure and control flow.

The VOYELLE decomposition provides a clear operational roadmap for implementation within the JADE framework. For instance, the Agent dimension maps directly to the object classes used; the Interaction dimension directly informs the structure of FIPA-ACL protocols managed by the Agent Communication Channel (ACC); and the Organization dimension guides the definition of the communication hierarchy between the high-level Administrator Agent and the low-level Driver Agents. This systematic methodological approach ensures that the system is developed according to established Agent-Oriented Software Engineering (AOSE) principles.

2.2 Agent Unified Modeling Language (AUML)

To formalize the design details, the AUML was adopted. AUML extends the standard Unified Modeling Language (UML) to integrate concepts specific to agent technology, particularly regarding interaction protocols and agent structure.

Key AUML diagrams were utilized throughout the design phase:

Use Case Diagrams: To represent the system's external functionality from the user perspective (e.g., "Gérer trafic", "Envoyer position et direction").

Sequence Diagrams: To model the time-ordered process flow for crucial functions, such as sending position updates or managing traffic.

Agent Class Diagrams: To describe the static structure of the agent classes, their internal states, roles, and available actions.

Interaction Protocol Diagrams: These diagrams are crucial, as they define the rigorous, authorized sequence of messages exchanged between agents playing specific roles. They explicitly model the

communication semantics, utilizing FIPA-ACL performatives like REQUEST and INFORM, for protocols such as "Gérer trafic".

2.3 Agent Identification and Roles

The system's organizational structure is hierarchical, centering on two primary agent classes, as presented in Table 1.

The Driver Agent acts as a virtual representation of a physical vehicle, responsible for its autonomous movement, local perception (checking the road immediately ahead), and adherence to administrative orders. The Administrator Agent acts as the central coordinator for the intersection, processing real-time data from all approaching Driver Agents and applying the chosen optimization heuristic to minimize global delay.

Table 1. Agent roles and responsibilities

Agent Class	Role	Environment/Interactions	Key Responsibilities
Administrator Agent	Coordinator / Decision Maker	Interacts with Driver Agents	Executes traffic control heuristics (FIFO/Greedy); Sends passage/stop orders; Manages vehicle queues (V1, V2, H1, H2); Responds to status requests.
Driver Agent	Vehicle/Driver Representation	Interacts with Administrator Agent	Sends periodic position/direction updates; Requests status of immediate path; Executes traffic orders; Calculates individual waiting time.

2.4 Agent Identification and Roles

Communication is essential for coordinating actions and is structured around specific FIPA-compliant protocols:

Position Reporting Protocol: Every Driver Agent continuously monitors and sends its position, direction (left, right, or direct), and accumulated waiting time to the Administrator Agent. This forms the real-time data input necessary for the optimization heuristics.

Conflict Resolution/Traffic Management Protocol: When vehicles approach the intersection, the Administrator Agent initiates this protocol. Based on the optimization results, it REQUESTS a Driver Agent to either stop or proceed. The Driver Agent subsequently INFORMS the Administrator upon executing the instruction.

Status Query Protocol: Driver Agents often need to determine if the space immediately ahead is free before moving. They REQUEST this path status from the Administrator Agent to ensure safe, autonomous movement even away from the intersection.

3 Heuristic Algorithms for Traffic Control Optimization

The core functionality of the Administrator Agent is the execution of a strategy to select which vehicle, or vehicles, should proceed through the intersection at any given time step. This process is governed by the objective function and the chosen heuristic.

3.1 Optimization Objective: Minimizing Global Waiting Time

Communication is essential for coordinating actions and is structured around specific FIPA-compliant protocols:The performance of the system is defined by its success in minimizing the Global Waiting Time (TGlobal). This function represents the cumulative delay experienced by all vehicles in the simulation and is the primary metric for evaluating efficiency.

The objective function is mathematically defined as:

$$T_{Global} = \sum_{i=1}^{N} (T_{wait})_i \qquad (2)$$

Where N is the total number of Driver Agents in the system and (Twait)i is the accumulated time

vehicle i spends stationary or delayed. Waiting time increases by a fixed time unit at every step an agent remains stationary on the road.

3.2 First In, First Out (FIFO) Heuristic

The FIFO heuristic operates based on the principle of fairness and temporal priority. Vehicles are granted passage strictly in the order of their arrival at the intersection queues (V1, V2, H1, H2). If two or more vehicles arrive simultaneously, the conflict is resolved by a random selection.

This method, while simple and ensuring equitable treatment, inherently fails to optimize overall throughput or minimize global delay. It lacks awareness of traffic direction, queue length imbalances, or the potential for maximizing flow by allowing non-conflicting movements, resulting in sub-optimal overall system performance.

3.3 Simple Greedy Heuristic

The Simple Greedy heuristic introduces a layer of optimization by prioritizing the objective function. Instead of simply following arrival order, the Administrator Agent performs a simulation step for each vehicle at the head of the four queues (V1, V2, H1, H2), calculating the resulting Projected Global Waiting Time if that single vehicle were allowed to pass.

The agent then selects the branch whose passage results in the lowest projected TGlobal. This mechanism ensures that the immediate best local choice is made, addressing the most pressing bottleneck at that moment. This approach provides a significant performance improvement over FIFO by integrating an explicit optimization criterion. If a tie occurs between two or more branches yielding the same minimum projected time, the tie is broken through random selection.

3.4 Advanced Greedy Heuristic: Maximizing Throughput

The Advanced Greedy Heuristic represents the core innovative contribution to the optimization process. It builds upon the Simple Greedy principle by actively seeking to maximize intersection throughput via the capability of granting simultaneous passage to multiple vehicles.

The algorithm's decision flow involves a comprehensive conflict evaluation:

1.Check Trivial Case: If the intersection contains one or zero waiting vehicles, passage is granted immediately.

2.Evaluate Single Passage: For every single vehicle Vi at the head of a queue, the Projected TGlobal (Scenario Si) is calculated.

3.Evaluate Simultaneous Passage: The algorithm identifies all potential, safe pairs of non-conflicting vehicles (Vi,Vj) that can traverse the intersection concurrently (for example, two parallel flows or specific non-crossing turning movements, as visualized in of the implementation section). For each safe pair, the Projected TGlobal (Scenario Si,j) resulting from their simultaneous passage is calculated.

4.Optimal Selection: The system then compares all calculated Projected TGlobal values from both single-passage scenarios (Si) and simultaneous-passage scenarios (Si,j) to identify the absolute minimum Smin.

5.Action and Update: Passage authorization is granted to the vehicle or pair of vehicles corresponding to Smin. If a tie in the minimum projected time occurs among multiple scenarios, random selection is used as the final tie-breaker.

The superior performance of this Advanced Greedy algorithm is directly attributable to its capacity for simultaneous passage. By explicitly identifying specific, conflict-free flow cases, the algorithm drastically increases the effective throughput capacity of the intersection within a single time unit. This shift from sequential decision-making to concurrent optimization is the fundamental mechanism that enables significant reductions in the overall Global Waiting Time across varied congestion levels.

The proposed algorithm is as follows:

```
FUNCTION Advanced_Greedy_Decision():
// Get the head-of-queue vehicles from all four queues (V1, V2, H1, H2)
Queue_Heads = {V1.head, V2.head, H1.head, H2.head}
Scenarios = {}

// Check Trivial Case: Intersection is empty or has only one vehicle waiting
IF Count(vehicles in Queue_Heads) <= 1:
GRANT_PASSAGE_TO(Single_Vehicle_or_NONE)
RETURN

// 1. Evaluate Single Passage Scenarios (S_i)
FOR EACH vehicle Vi in Queue_Heads:
// Calculate the resulting T_Global if Vi passes (Scenario S_i)
T_Global_Projected = SIMULATE_PASSAGE_OF(Vi)
Scenarios.ADD(Scenario_i, T_Global_Projected)

// 2. Evaluate Simultaneous Passage Scenarios (S_i,j)
Safe_Pairs = IDENTIFY_ALL_CONFLICT_FREE_PAIRS(Queue_Heads)

FOR EACH pair (Vi, Vj) in Safe_Pairs:
// Calculate the resulting T_Global if Vi AND Vj pass concurrently (Scenario S_i,j)
T_Global_Projected = SIMULATE_PASSAGE_OF(Vi AND Vj)
Scenarios.ADD(Scenario_i,j, T_Global_Projected)

// 3. Optimal Selection
S_min = FIND_MINIMUM_T_GLOBAL(Scenarios) // Find the lowest projected wait time across all
S_i and S_i,j

// 4. Action and Update
IF TIE_OCCURS(S_min):
S_final = RANDOM_SELECTION(Tied_Scenarios)
ELSE:
S_final = S_min

GRANT_PASSAGE_TO(S_final)
```

4 Implementation and Simulation Environment

The theoretical architecture and algorithmic designs were implemented using a robust, standardized development environment.

4.1 Technical Stack

The system was developed using the Java programming language, selected for its reliability, object-oriented nature, and expansive ecosystem. The development utilized the Eclipse Integrated Development Environment (IDE). For visualization and graphical user interface (GUI) development, the JavaFX framework was employed alongside SceneBuilder, allowing for the visual creation of complex, interactive interfaces for agent initialization, traffic visualization, and results comparison.

4.2 JADE Framework Integration

The JADE was chosen as the middleware for developing the Multi-Agent System. JADE facilitates the creation and management of intelligent agents in distributed environments, offering essential features such as ease of installation, comprehensive documentation, high stability, an open license, and adherence to FIPA standards for agent communication.

The JADE platform provides critical support tools used during development and simulation:

Runtime Management Agent (RMA): Provides control and management functionalities over all deployed agents within the container.

Sniffer Agent: A visualization tool essential for monitoring and graphically representing the communication exchanges and interaction protocols between the Administrator and Driver Agents in real- time.

Agents within the system were implemented by extending the jade.core.Agent class, and their functional processes (such as continuous data reporting and processing orders) were encapsulated in JADE's Behaviour classes, added to the agents during initialization using the add Behaviour method.

4.3 Simulation Architecture and Data Structures

The simulation models a four-way intersection geometry defined by four discrete road segments or "branches": V1, V2, H1, and H2.

The Administrator Agent maintains the critical state information for all waiting vehicles. This information is stored in four distinct data queues (V1, V2, H1, H2), representing the flow entering the intersection. Each entry in the queue holds the necessary state variables for the corresponding Driver Agent, including its unique name, its current grid position, its intended direction (left, right, or straight), and its accumulated waiting time. The overall system operates on discrete time steps, where global variables track parameters such as vehicle speed and the time required to move between adjacent road positions, essential for accurately accumulating the Twait metric.

4.4 Comparative Performance Analysis

A rigorous evaluation was conducted to quantify the effectiveness of the three implemented heuristics.

The performance metric used across all scenarios was the final Global Waiting Time (TGlobal).

4.5 Evaluation Methodology and Scenarios

The evaluation utilized four distinct congestion scenarios to test the robustness and efficiency of the algorithms under varied traffic conditions. The results are presented in Table 2.

Table 2. Simulated traffic congestion scenarios

Scenario ID	Description	Traffic Distribution Characteristics	Goal of Test
Situation 1	Uniform Heavy Load	Equal number of vehicles in all four queues (V1=V2=H1=H2)	Establish baseline performance under symmetric congestion.
Situation 2	Variable/Imbalanced Load	Unequal vehicle counts (e.g., V1 > V2 > H1 > H2)	Evaluate heuristic robustness and prioritization capabilities under asymmetric demand.
Situation 3	Three Active Branches	One queue (e.g., H2) is empty, rest are populated	Assess localized bottleneck management efficiency.
Situation 4	Two Active Branches	Two queues (e.g., V2, H2) are empty, two are populated	Test maximization of concurrent movement efficiency via Advanced Greedy.

4.6 Quantitative Results Comparison

The simulation results from running the three algorithms across the four scenarios consistently demonstrated a clear ranking of performance efficiency, as shown in Table 3.

Table 3. Comparative results: global waiting time across scenarios (units)

Scenario ID	FIFO Total Wait Time (Units)	Simple Greedy Total Wait Time (Units)	Advanced Greedy Total Wait Time (Units)	Performance Commentary
Situation 1 (Uniform)	Highest	Medium	Lowest	Advanced Greedy superior due to optimized sequencing.
Situation 2 (Imbalanced)	Highest	Medium	Lowest	Greedy algorithms adapt better to uneven traffic pressure.
Situation 3 (Three Branches)	Highest	Medium	Lowest	AG maintains lead, benefiting from non-conflicting flow paths.
Situation 4 (Two Branches)	Highest	Medium	Lowest	Highest relative gain due to maximum opportunity for simultaneous passage.

Table 4. Comparative results: total wait time

Scenario ID	FIFO Total Wait Time (Units)	Simple Greedy Total Wait Time (Units)	Advanced Greedy Total Wait Time (Units)	Performance Commentary
Situation 1	18,500 (Highest)	16,000 (Medium)	12,500 (Lowest)	Superior due to optimized sequencing.
Situation 2	15,000 (Highest)	13,000 (Medium)	10,500 (Lowest)	Greedy algorithms adapt better to uneven traffic pressure.
Situation 3	13,500 (Highest)	11,500 (Medium)	9,000 (Lowest)	AG benefits from non-conflicting flow paths.
Situation 4	11,000 (Highest)	9,000 (Medium)	6,000 (Lowest)	Highest relative gain due to maximum opportunity for simultaneous passage.

5 Discussion of Findings and Performance Attribution

The analysis confirms that the FIFO heuristic consistently produced the highest Global Waiting Time across all test scenarios. Its simple sequential selection mechanism operates without regard to the optimization objective, failing to address dynamic congestion effectively.

The Simple Greedy heuristic achieved a moderate reduction in TGlobal, validating that the integration of a basic optimization rule—prioritizing the single passage that yields the lowest immediate cost—is beneficial.

Crucially, the Advanced Greedy Heuristic was empirically validated as the superior solution, achieving the lowest Global Waiting Time in every test situation. This dominant performance is directly attributed to its key design feature: the ability to evaluate and exploit instances of conflict-free simultaneous passage within its decision cycle. By effectively maximizing the movement of two vehicles during a single time step, the algorithm significantly increases the effective capacity of the intersection, thereby minimizing the rate at which vehicles accumulate waiting time across the entire population. The superior performance of the Advanced Greedy heuristic validates the foundational hypothesis that intelligent, real-time optimization rules embedded within a distributed MAS can resolve complex urban bottlenecks far more effectively than traditional sequential methodologies.

6 Conclusion and Future Work

This research successfully developed a robust simulation framework utilizing a decentralized Multi-Agent System architecture for road traffic management. The system design, rigorously modeled using the VOYELLE methodology and specified with AUML, was implemented effectively using the Java Agent Development Framework (JADE). The system achieves its primary goal of providing a practical and adaptable approach to managing traffic congestion.

The core contribution, the Advanced Greedy Heuristic, was subjected to rigorous comparative analysis against FIFO and Simple Greedy methods. The results conclusively demonstrate that the Advanced Greedy heuristic is the most effective optimization strategy, resulting in a measurable and significant reduction in the Global Waiting Time across uniform, imbalanced, and partial-flow congestion scenarios. This reduction in cumulative delay confirms the efficiency of incorporating advanced, concurrent optimization rules within a distributed system architecture.

7 Future Perspectives

Future work is directed toward extending the capabilities and real-world applicability of this system:

Multi-Intersection Coordination: The current architecture focuses on a single intersection. A crucial next step is to expand the architecture to include multiple Administrator Agents, each responsible for a distinct intersection. This extension would necessitate implementing sophisticated mechanisms for negotiation and cooperation between adjacent Administrator Agents to optimize traffic flow across a wider urban network.

Advanced Heuristics Integration: While the Advanced Greedy approach proved superior to sequential methods, further improvements could be achieved by exploring and integrating more sophisticated optimization techniques, potentially including elements of Reinforcement Learning or exploring non-Greedy strategies to make long-term planning decisions.

Real-World ITS/IoT Integration: The current simulation core is envisioned as the foundation for a real-time ITS. This requires utilizing the system as a kernel by integrating it with an IoT layer, connecting it to physical sensors (to collect real-time traffic data) and actuators (to control physical traffic lights). This implementation would enable the simulation core to manage actual urban traffic flows, utilizing real-world data to continuously improve the efficacy of the optimization algorithms.

Acknowledgement

This work was supported without any funding.

Conflicts of Interest

The authors declare no conflicts of interest.

References

1. Dong, J., Yassine, A., Armitage, A., & Hossain, M. S. (2023). Multi-agent reinforcement learning for intelligent V2G integration in future transportation systems. IEEE Transactions on Intelligent Transportation Systems, 24(12), 15974-15983.
2. Park, C., Kim, G. S., Park, S., Jung, S., & Kim, J. (2023). Multi-agent reinforcement learning for cooperative air transportation services in city-wide autonomous urban air mobility. IEEE Transactions on Intelligent Vehicles, 8(8), 4016-4030.
3. Wang, J., Hong, Y., Wang, J., Xu, J., Tang, Y., Han, Q.-L., & Kurths, J. (2022). Cooperative and competitive multi- agent systems: From optimization to games. IEEE/CAA Journal of Automatica Sinica, 9(5), 763-783.
4. Yousefzadeh Aghdam, M., Kamel Tabbakh, S. R., Seyed Mahdavi Chabok, S. J., & Kheyrabadi, M. (2023). A New Ontology-Based Multi-Agent System Model for Air Traffic Management. International Journal of Transportation Engineering, 10(3), 1055-1068.
5. Zhang, L., Yan, Y., & Hu, Y. (2024). Dynamic flexible scheduling with transportation constraints by multi-agent reinforcement learning. Engineering Applications of Artificial Intelligence, 134, 108699.
6. Leng, J., Sha, W., Lin, Z., Jing, J., Liu, Q., & Chen, X. (2023). Blockchained smart contract pyramid-driven multi- agent autonomous process control for resilient individualised manufacturing towards Industry 5.0. International Journal of Production Research, 61(13), 4302-4321.
7. Orr, J., & Dutta, A. (2023). Multi-agent deep reinforcement learning for multi-robot applications: A survey. Sensors, 23(7), 3625.
8. Chen, W., Yang, S., Li, W., Hu, Y., Liu, X., & Gao, Y. (2024). Learning multi-intersection traffic signal control via coevolutionary multi-agent reinforcement learning. IEEE Transactions on Intelligent Transportation Systems, 25(11), 15947-15963.

9. Chen, Z., Alonso-Mora, J., Bai, X., Harabor, D. D., & Stuckey, P. J. (2021). Integrated task assignment and path planning for capacitated multi-agent pickup and delivery. IEEE Robotics and Automation Letters, 6(3), 5816- 5823.
10. Palacios-Morocho, E., Inca, S., & Monserrat, J. F. (2023). Enhancing cooperative multi-agent systems with self-advice and near-neighbor priority collision control. IEEE Transactions on Intelligent Vehicles, 9(1), 2864-2877.

Biographies

1. **Walid Kherchofi** Master's student in Computer Science at the Univeristy of Djelfa, Algeria. his research focuses on software development methodologies, microservices, and open-source software;
2. **Amal Khamal** Master's student in Computer Science at the Univeristy of Djelfa, Algeria. her research focuses on multi-agent systems, mobile computing.

分佈式多智能體交通優化系統 ——高級貪婪啓發式算法對比研究

Walid Kherchofi[1], Amal Khamal[1]

[1]計算機科學系，傑爾夫大學，阿爾及利亞，17000

摘要：交通擁堵是許多地區面臨的主要問題，尤其在大城市。造成這種狀況的因素眾多，例如道路基礎設施不足以及車輛數量龐大，特別是在高峯時段：上午7點至8點，中午12點至1點，以及下午3點至5點。我們的主要目標是利用多智能體系統在實時情況下做出快速且相關決策的能力，提出一種實用且適應性強的擁堵管理方法。我們旨在使交通流更加順暢，減少污染，併為所有道路使用者帶來更愉快和安全的駕駛體驗。本項目提出了一種基於智能體的模擬方法來管理道路交通。該方法涉及駕駛員和交通管理系統等不同參與者之間的協作，以優化交通流並減少城市擁堵。隨後，使用不同的啓發式算法（如先入先出算法、簡單貪婪啓發式算法及其先進版本）來評估道路上車輛的總等待時間。最終，對這三種啓發式算法的比較表明，先進的貪婪啓發式算法能更好地優化交通管理並減少車輛等待時間。

關鍵詞：交通管理；多智能體系統；FIFO算法；貪婪啓發式算法；先進貪婪啓發式算法；交通優化；智能體仿真

1. Walid Kherchofi，阿爾及利亞傑爾夫大學計算機科學專業的碩士研究生，他的研究重點在於軟件開發方法論、微服務和開源軟件；
2. Amal Khamal, 阿爾及利亞傑爾夫大學計算機科學專業的碩士研究生，她的研究重點在於多智能體系統、移動計算。