machine learning & knowledge extraction

MDPI

Article

Visual Reasoning and Multi-Agent Approach in Multimodal Large Language Models (MLLMs): Solving TSP and mTSP Combinatorial Challenges

Mohammed Elhenawy [1,2], Ahmad Abutahoun [3], Taqwa I. Alhadidi [3], Ahmed Jaber [4], Huthaifa I. Ashqar [5,6], Shadi Jaradat [1,2], Ahmed Abdelhay [7], Sebastien Glaser [1] and Andry Rakotonirainy [1,*]

1 Accident Research and Road Safety Queensland, Queensland University of Technology, Brisbane, 130 Victoria Park Rd, Kelvin Grove, QLD 4059, Australia; mohammed.elhenawy@qut.edu.au (M.E.); shadi.jaradat@hdr.qut.edu.au (S.J.); sebastien.glaser@qut.edu.au (S.G.)
2 Centre for Data Science, Queensland University of Technology, Brisbane, 2 George St, Brisbane, QLD 4000, Australia
3 Civil Engineering Department, Al-Ahliyya Amman University, Amman 19328, Jordan; abutahouna@gmail.com (A.A.); t.alhadidi@ammanu.edu.jo (T.I.A.)
4 Department of Transport Technology and Economics, Faculty of Transportation Engineering and Vehicle Engineering, Budapest University of Technology and Economics, Műegyetem Rkp. 3., H-1111 Budapest, Hungary; ahjaber6@edu.bme.hu
5 Civil Engineering Department, Faculty of Engineering, Arab American University, Jenin P.O. Box 240, Palestine; huthaifa.ashqar@aaup.edu
6 AI@Columbia, Fu Foundation School of Engineering and Applied Science, Columbia University, New York, NY 10027, USA
7 Computer and Systems Engineering Department, Faculty of Engineering, Minia University, Minia 2431436, Egypt; ahmelhuy@gmail.com
* Correspondence: r.andry@qut.edu.au

Citation: Elhenawy, M.; Abutahoun, A.; Alhadidi, T.I.; Jaber, A.; Ashqar, H.I.; Jaradat, S.; Abdelhay, A.; Glaser, S.; Rakotonirainy, A. Visual Reasoning and Multi-Agent Approach in Multimodal Large Language Models (MLLMs): Solving TSP and mTSP Combinatorial Challenges. *Mach. Learn. Knowl. Extr.* **2024**, *6*, 1894–1921. https://doi.org/10.3390/make6030093

Academic Editor: Karin Verspoor

Received: 25 June 2024
Revised: 2 August 2024
Accepted: 9 August 2024
Published: 13 August 2024

Abstract: Multimodal Large Language Models (MLLMs) harness comprehensive knowledge spanning text, images, and audio to adeptly tackle complex problems. This study explores the ability of MLLMs in visually solving the Traveling Salesman Problem (TSP) and Multiple Traveling Salesman Problem (mTSP) using images that portray point distributions on a two-dimensional plane. We introduce a novel approach employing multiple specialized agents within the MLLM framework, each dedicated to optimizing solutions for these combinatorial challenges. We benchmarked our multi-agent model solutions against the Google OR tools, which served as the baseline for comparison. The results demonstrated that both multi-agent models—Multi-Agent 1, which includes the initializer, critic, and scorer agents, and Multi-Agent 2, which comprises only the initializer and critic agents—significantly improved the solution quality for TSP and mTSP problems. Multi-Agent 1 excelled in environments requiring detailed route refinement and evaluation, providing a robust framework for sophisticated optimizations. In contrast, Multi-Agent 2, focusing on iterative refinements by the initializer and critic, proved effective for rapid decision-making scenarios. These experiments yield promising outcomes, showcasing the robust visual reasoning capabilities of MLLMs in addressing diverse combinatorial problems. The findings underscore the potential of MLLMs as powerful tools in computational optimization, offering insights that could inspire further advancements in this promising field.

Keywords: combinatorial optimization; traveling salesman problem; multimodal large language models; visual reasoning

1. Introduction

The Traveling Salesman Problem (TSP) and its multi-salesmen variant (mTSP) are classic challenges in combinatorial optimization, recognized for their NP-hard complexity

and practical relevance in logistics, planning, and network design [1,2]. TSP involves finding the shortest route for a single salesman to visit a set of cities and return to the start, while mTSP extends this to several salesmen, each covering different subsets of cities. These problems are computationally tough, especially as the number of cities grows, despite many advanced algorithms developed over the years.

Various techniques, including Integer Programming, Constraint Programming (CP), and Local Search [3], were employed to approach the TSP, a classic problem. More recent research has explored innovative methods, such as utilizing reinforcement learning and deep neural networks to optimize reward strategies for solving the TSP [4]. Solutions to the TSP have practical implications in transportation, logistics, and automation, necessitating efficient computation to meet real-time demands [5]. In the domain of heuristic algorithms, the significance of transfer functions was underscored. Studies have demonstrated the sequential resolution of the TSP using the Gravitational Search Algorithm (GSA) and Neural Networks (NN) [6]. Additionally, the application of neural network solvers for graph combinatorial optimization problems like the TSP has attracted considerable interest, although challenges persist in scaling these solutions efficiently beyond graphs with a few hundred nodes [7]. This highlights ongoing efforts to enhance the scalability and efficacy of large language models in addressing complex optimization challenges. Furthermore, the incorporation of advanced technologies such as multi-agent reinforcement learning has exhibited promise in tackling intricate optimization problems, like multi-vehicle routing with soft time windows [8]. Research has also focused on reducing the size of combinatorial optimization problems using innovative approaches like the Operator Vaccine by Fuzzy Selector with Adaptive Heuristics, showcasing successful reductions in instances of the TSP [9]. Traditional solutions to TSP and mTSP typically rely on distance matrices and explicit calculations of routes based on node coordinates [10,11]. However, human problem-solving often employs visual and heuristic approaches to generate reasonable solutions without detailed calculations quickly. Inspired by these intuitive human strategies, our research explores a novel visual reasoning approach to solve TSP and mTSP. This approach leverages the power of visual inspection and team-based iterative refinement, bypassing the need for textual data or distance matrices, which are standard in computational methods [10,11].

The advantages of using MLLMs in solving problems include their ability to process and integrate diverse data types such as text [12,13], images [14,15], and videos [16], enabling a more holistic understanding of complex issues. Traditional solutions to TSP and mTSP typically rely on distance matrices and explicit calculations of routes based on node coordinates [10,11]. However, human problem-solving often employs visual and heuristic approaches to generate reasonable solutions without detailed calculations quickly. Inspired by these intuitive human strategies, our research explores a novel visual reasoning approach to solve TSP and mTSP. This approach leverages the power of visual inspection and team-based iterative refinement, bypassing the need for textual data or distance matrices, which are standard in computational methods [10,11].

MLLMs can process and interpret visual data to suggest efficient routes through nodes, bypassing the need for textual information or distance matrices. This aligns with human visual problem-solving capabilities, allowing for a more intuitive and flexible approach to spatial problems. Moreover, MLLMs facilitate a collaborative, iterative process where multiple agents propose, evaluate, and refine solutions, leveraging diverse perspectives and expertise. This teamwork-based strategy not only enhances the quality of solutions by minimizing route intersections and optimizing lengths but also reduces the computational burden associated with traditional methods. Therefore, the use of MLLM in this context is promising as it showcases the potential of advanced AI to replicate human cognitive strategies, leading to significant improvements in solving complex problems like TSP and mTSP.

This study introduces two multi-agent strategies that utilize the visual reasoning of MLLM to solve TSP and mTSP. The first strategy labelled Multi-Agent 1, employs a trio

of MLLM agents—initializer, critic, and scorer—each with distinct roles in proposing, refining, and evaluating routes based on their visual quality. The second strategy, labelled Multi-Agent 2, simplifies the approach by using only the initializer and critic MLLM agents, focusing on rapid iterative refinement without the Scorer. Both methods aim to enhance route optimization by leveraging visual cues, mimicking the human ability to suggest and refine solutions without extensive computational resources. The study assesses the effectiveness of visual reasoning-based strategies for solving the TSP mTSP. It compares their performance to that of the Google OR tools, which are used as baseline solutions. The comparison utilizes metrics such as the mean and standard deviation of the gap percentage and employs statistical validation through the Wilcoxon signed-rank test. Our results demonstrate significant improvements in solution quality and consistency, particularly in smaller and moderately sized problem instances, underscoring the potential of visually driven methodologies in complex problem-solving scenarios. The main contributions of this paper are as follows:

1. We introduce novel strategies for solving the TSP and mTSP using visual reasoning of MLLM alone, bypassing traditional numerical data like node coordinates or distance matrices.
2. We present MLLM as a multi-agent system—initializer, critic, and optionally scorer agents—that iteratively refines routes. The initializer suggests routes visually, the critic improves them iteratively, and the scorer evaluates them based on visual clarity and efficiency.
3. By leveraging iterative refinement, our approach minimizes route intersections, optimizes lengths, and ensures comprehensive node coverage without relying on numerical computations.

The remainder of this paper is organized as follows. In Section 2, we provide relevant work from the literature. In Section 3, we describe our approach and methods for in-context prompting. Section 4 presents the results of experimented methodologies. Finally, Section 5 concludes the study findings.

2. Related Work

Various innovative methods were proposed to tackle TSP, ranging from self-organizing map learning procedures to continuous relaxations and spider monkey optimization [17–19]. These approaches demonstrate the diversity of strategies employed to optimize TSP solutions, showcasing the interdisciplinary nature of research in this field. Furthermore, the utilization of linear function approximation and artificial immune system optimization reflects the continuous exploration of novel techniques to enhance combinatorial optimization outcomes [20,21].

The integration of different optimization algorithms like Ant Colony Optimization (ACO) and Genetic Algorithms (GA) is a common theme in TSP research [22,23]. Studies have focused on improving these algorithms by incorporating local optimization heuristics and adaptive strategies to enhance their performance in solving TSP instances [24,25]. Moreover, the exploration of quantum-inspired methods and hybrid solvers has shown promising results in terms of computational efficiency and solution quality for TSP [26,27]. Researchers have also investigated the application of unconventional approaches such as affinity propagation clustering, producer–scrounger methods, and penguin search algorithms to optimize TSP solutions [19,28,29]. These diverse methodologies highlight the continuous quest for innovative solutions to complex combinatorial optimization problems like TSP. Additionally, the exploration of quantum annealers and hybrid solvers has opened new avenues for addressing TSP challenges, showcasing the evolving landscape of optimization techniques [27].

The recent literature on multi-agent systems (MAS) has increasingly integrated deep reinforcement learning (DRL), as evidenced by Gronauer and Diepold [30]. This review

outlines how DRL methods are structured to train multiple agents, emphasizing their applications in cooperative, competitive, and mixed scenarios. It also identified and addressed specific challenges unique to MAS, proposing strategies to overcome these obstacles and suggesting future research directions. Complementing this perspective, Dorri et al. [31] provided a comprehensive overview of MAS, discussing its definitions, features, applications, and challenges. Similarly, in a paper by Pop et al. [32], researchers evaluated various visual reasoning methodologies against traditional computational techniques. The study comprehensively surveyed mathematical formulations, solution approaches, and the latest advances regarding the generalized traveling salesman problem. Moreover, Yang et al. conducted alike survey on the cooperative control of multi-agent systems. Reinforced algorithms like the Lin–Kernighan–Helsgaun have demonstrated significant advancements in leveraging machine learning for solving TSP [33].

Currently, researchers are exploring the use of Large Language Models (LLMs) to address combinatorial problems like the traveling salesman problem. Liu et al. [34] examined the concept of LLM-driven evolutionary algorithms (LMEA), marking the first attempt to apply LLMs in solving such problems. Meanwhile, Yang et al. [35] proposed Optimization by PROmpting (OPRO), where the LLM generates solutions from prompts containing previously generated solutions and their evaluations. This iterative process enhances solution quality with each step. Additionally, ensemble learning methods combined with LLMs, as discussed in works by Silviu et al. [36], showed promising results in optimizing solutions.

In addressing the mTSP, Zheng et al. [37] focused on optimizing two objectives: minimizing the total length of all tours (minsum objective) and minimizing the length of the longest tour (minmax objective) among all salesmen. Furthermore, combining LLMs with other optimization techniques demonstrates the potential for improving TSP solutions. Each method, whether utilizing zero-shot, few-shot, or Chain-of-Thoughts (CoT) prompting techniques, aimed to enhance the accuracy of LLM responses, as evidenced in recent studies [38,39].

Additionally, a study by Bérczi et al. [40] considered a further generalization of mTSP (many-visits mTSP) where each city has a request of how many times it should be visited by the salesmen. The authors provided polynomial-time algorithms for several variants of the many-visits mTSP that compute constant-factor approximate solutions. Huang et al. [41] explored LLMs' application in vehicle routing problems, demonstrating that direct input of natural language prompts enhances performance. They proposed a self-refinement framework to iteratively improve LLM-generated solutions, stressing the role of detailed task descriptions in boosting performance. Despite excelling in text-based tasks, LLMs face challenges with other data types [15]. However, MLLMs aim to overcome these limitations by integrating diverse data modalities (text, image, video, audio, etc.), broadening LLMs' potential applications beyond traditional text domains.

This study introduces two novel strategies for solving the TSP and mTSP that leverage the visual reasoning capabilities of MLLM. Unlike traditional approaches that rely on numerical data such as node coordinates or distance matrices, this study uses purely visual cues to infer efficient routes. This method mimics human visual problem-solving abilities and provides a more intuitive and flexible solution to spatial problems. While some recent research has explored the use of LLMs in combinatorial problems and optimization, such as LLM-driven evolutionary algorithms and Optimization by PROmpting (OPRO), these studies have primarily focused on text-based tasks. This study extends the application of LLMs to solely visual data, demonstrating the potential of MLLMs to address complex problem-solving scenarios without extensive computational resources.

This study proposed a multi-agent system involving distinct roles for MLLM agents—initializer, critic, and scorer. This collaborative, iterative approach enhances the quality of solutions through diverse perspectives and expertise, which is not extensively covered in prior research. While some studies have discussed the integration of DRL in

multi-agent systems and the cooperative control of such systems, this study uniquely applies visual reasoning in a multi-agent context to optimize routes for TSP and mTSP. By focusing on visual reasoning and iterative refinement in MLLMs, the study minimizes route intersections, optimizes route lengths, and ensures comprehensive node coverage, thus offering a novel methodology that advances the current state of research in visual computational techniques and complex problem-solving.

3. Materials and Methods

Our methodology draws on two concepts inspired by human problem-solving approaches. The first is based on the human ability to suggest efficient routes through nodes without explicit calculations visually. The second concept leverages the advantages of teamwork, where individuals collaboratively refine solutions. In this approach, a team member proposes one or more solutions, which are collectively analyzed. Ineffective solutions are discarded while promising ones are iteratively improved through subsequent proposal, evaluation, and refinement cycles. This iterative team-based strategy enhances the development of practical solutions by leveraging diverse perspectives and expertise.

In this paper, we adopt these ideas to develop two strategies to solve the TSP and mTSP using visual reasoning alone, without relying on textual information about node locations or distance matrices with the use of MLLMs. This approach not only improves the practical application of visual computational techniques but also demonstrates the potential for MLLM to revolutionize complex problem-solving by mimicking human cognitive processes. The following subsection will detail the proposed strategies, explaining how each leverage purely visual cues to infer efficient routes across multiple nodes, thus enhancing our understanding of visual computational capabilities in complex problem-solving.

3.1. Multi-Agent 1

The proposed methodology in Figure 1 adopts a multi-agent strategy to solve the m-salesmen problem where $m \in \{1,2,3\}$. The process involves three agents: the initializer, the critic, and the scorer. Initializer agent visually inspects the nodes' layout without textual information about their coordinates. Using visual reasoning, it suggests an initial order for visiting the nodes, aiming to propose a good solution. The suggested solution by the Initializer is visualized and presented to the critic agent along with a prompt instructing it to inspect the visualization and suggest better routes. The critic agent employs a self-ensemble method by raising its temperature to 0.7, generating seven solutions. These solutions are visualized and passed on to the Scorer Agent. This agent assigns a score to each solution based solely on its visual quality without calculating the actual distances. The solution with the highest score is then selected and returned to the critic. The term "temperature" in GPT-4o refers to a parameter controlling the randomness of the model's output. Higher temperatures produce more diverse responses, while lower temperatures result in more focused and deterministic outputs. These settings can balance the trade-off between diversity and focus on generated responses. In our methodology, we use the temperature setting as an input variable to manage the model's creativity and randomness. By setting the temperature to 0.7, we generate a variety of potential solutions, which are then iteratively refined for improved accuracy and robustness. This value was chosen after multiple trials to balance generating diverse outputs and maintaining necessary focus for accurate solutions.

The steps involving the critic and the scorer continue iteratively until the maximum number of iterations is reached. Finally, the best solution is returned as the proposed solution to the m-salesmen problem. This strategy leverages a visually driven multi-agent strategy where the Initializer proposes, the critic refines, and the scorer evaluates solutions based on visual reasoning using MLLM to iteratively enhance the solution quality for the m-salesmen problem. In this strategy, we used ChatGPT-4o as the initializer, the critic, and the scorer.

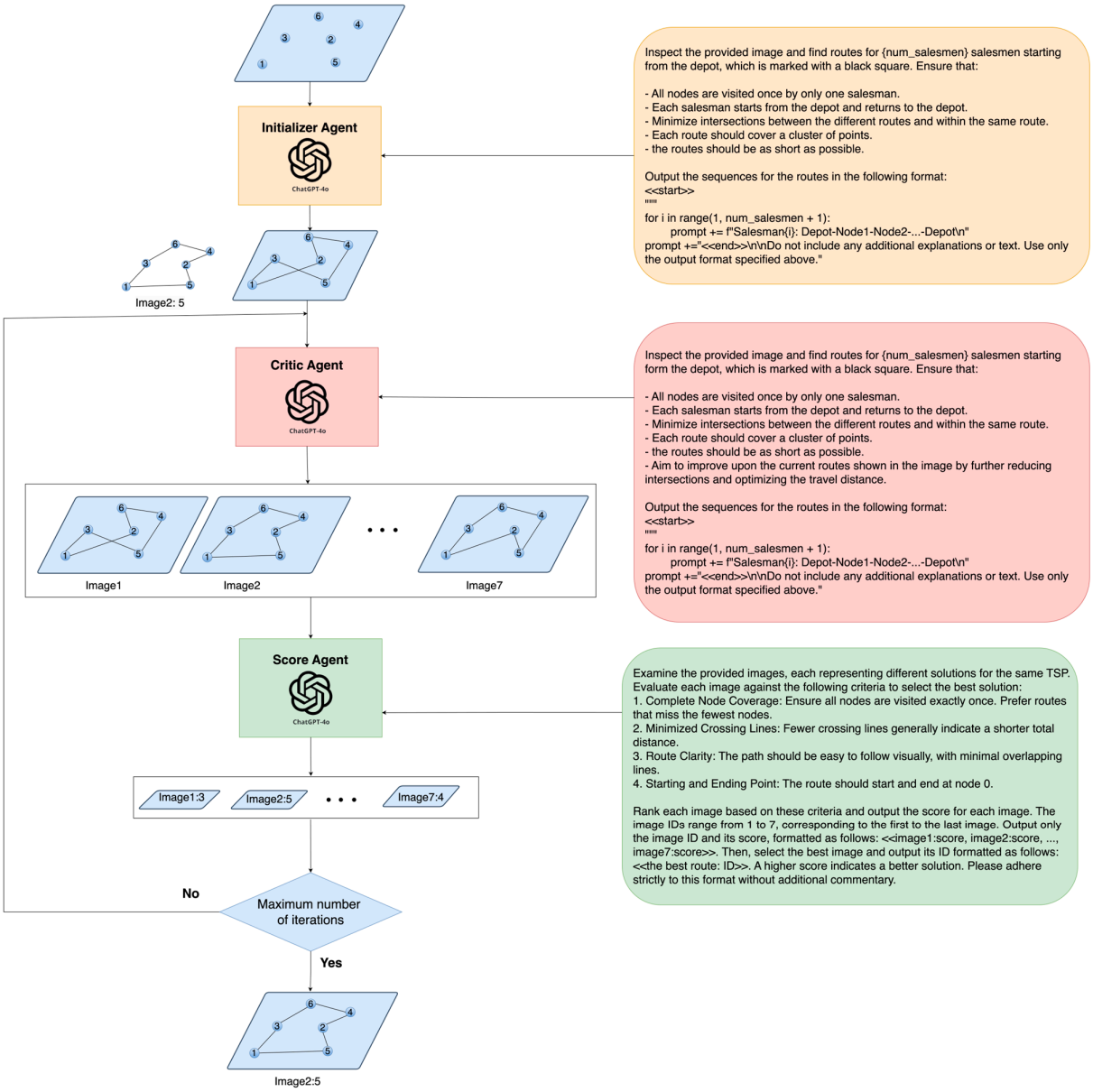


Figure 1. Multi-Agent 1 strategy: the diagram illustrates a three-phase approach using visual reasoning, where an initializer agent proposes routes, a critic agent refines them, and a scorer agent evaluates their visual quality, enhancing the solutions iteratively. Numbers 1–6 represent different nodes visited by a salesman.

3.2. Multi-Agent 2

The simplified Multi-Agent 2 strategy in Figure 2, designed for solving the m-salesmen problem with fewer agents, uses only the initializer and critic agents, eliminating the scorer agent.

In this lighter approach, the initializer agent continues by visually inspecting the nodes and proposing an initial sequence for visiting them, leveraging visual cues instead of relying on coordinate data. This proposed route is then passed to the critic agent, which, unlike its role in Multi-Agent 1, operates under a higher temperature setting of 0.7 to generate a single alternative route per iteration. Each new solution is visually presented back to the critic for further refinement. The iterative process between the initializer and critic continues until the maximum number of iterations is reached, at which point the process

halts. This streamlined version focuses on rapid iteration and refinement without the evaluative step of scoring, aiming to efficiently enhance route optimization through continuous visual feedback and critical adjustments. In this strategy, we used ChatGPT-4o as the initializer and the critic.

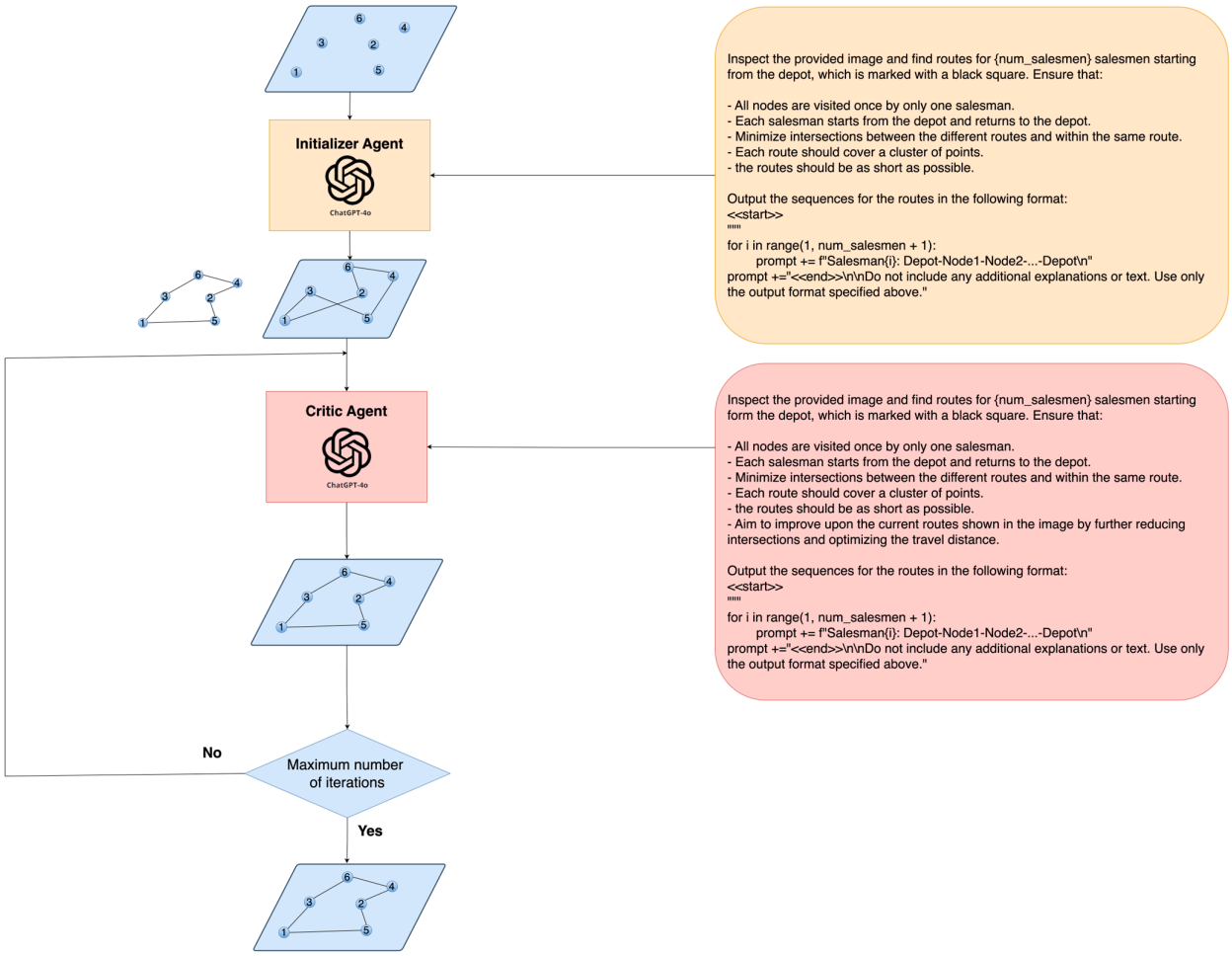


Figure 2. Multi-Agent 2 strategy: the diagram illustrates a streamlined two-agent methodology, using visual reasoning without computational distance measures. The initializer agent proposes routes, which the critic agent refines, iterating until a maximum iteration. Numbers 1–6 represent different nodes visited by a salesman.

3.3. Initializer and the Critic

Both proposed multi-agent strategies employ two core agents: the initializer and the critic. Their roles are adapted to optimize route efficiency in solving the mTSP (Multiple Traveling Salesman Problem) through visual reasoning.

The initializer plays a consistent role across both the Multi-Agent 1 and Multi-Agent 2 strategies. Its primary function is to generate preliminary routes by visually inspecting the distribution of nodes in the input image. This agent proposes an initial sequence of node visits, establishing an initial route that the critic agent will refine. To ensure focus and accuracy, the temperature parameter for the initializer is set to zero, promoting the generation of the most probable and coherent token sequences.

For Multi-Agent 1 strategy, the critic agent evaluates the route proposed by the Initializer to enhance its feasibility and efficiency in the first iteration. It utilizes a self-ensemble method with a temperature setting of 0.7, allowing for the generation of multiple route

alternatives. In subsequent iterations, the critic refines the route selected by the scorer agent using the same self-ensemble method, iteratively improving upon the previous selections.

For Multi-Agent 2 strategy, the critic assesses the initializer's route, operating at a temperature of 0.7 to generate a single and refined route in the first iteration. In following iterations, this agent continuously refines its previously generated route, adhering to this approach until the stopping criteria are met.

3.4. Prompts Engineering

In this subsection, we detail the specific prompts used in ChatGPT-4o to direct the actions of the different agents involved in our multi-agent system for solving the m-salesmen problem using visual reasoning. In Table 1, each agent is tasked with unique functions that contribute to identifying good routes based on visual inputs alone without relying on numerical data like distances or coordinates.

Table 1. Rules and prompts.

Rule	Prompt
Initializer agent	f“““ Inspect the provided image and find routes for {num_salesmen} salesmen starting from the depot, which is marked with a black square. Ensure that: - All nodes are visited once by only one salesman. - Each salesman starts from the depot and returns to the depot. - Minimize intersections between the different routes and within the same route. - Each route should cover a cluster of points. - The routes should be as short as possible. Output the sequences for the routes in the following format: <<start>> “““ for i in range(1, num_salesmen + 1): prompt += f”Salesman{i}: Depot-Node1-Node2-...-Depot\n” prompt += “<<end>>\n\nDo not include any additional explanations or text. Use only the output format specified above.”
Critic agent	f“““ Inspect the provided image and find routes for {num_salesmen} salesmen starting from the depot, which is marked with a black square. Ensure that: - All nodes are visited once by only one salesman. - Each salesman starts from the depot and returns to the depot. - Minimize intersections between the different routes and within the same route. - Each route should cover a cluster of points. - The routes should be as short as possible. - Aim to improve upon the current routes shown in the image by further reducing intersections and optimizing the travel distance. Output the sequences for the routes in the following format: <<start>> “““

	for i in range(1, num_salesmen + 1): prompt += f"Salesman{i}: Depot-Node1-Node2-...-Depot\n" prompt += "<<end>>\n\nDo not include any additional explanations or text. Use only the output format specified above."
Score agent	Examine the provided images, each representing different solutions for the same TSP. Evaluate each image against the following criteria to select the best solution: 1. Complete Node Coverage: Ensure all nodes are visited exactly once. Prefer routes that miss the fewest nodes. 2. Minimized Crossing Lines: fewer crossing lines generally indicate a shorter total distance. 3. Route Clarity: the path should be easy to follow visually, with minimal overlapping lines. 4. Starting and Ending Point: the route should start and end at node 0. Rank each image based on these criteria and output the score for each image. The image IDs range from 1 to 7, corresponding to the first to the last image. Output only the image ID and its score, formatted as follows: <<image1: score, image2: score, …, image7: score>>. Then, select the best image and output its ID formatted as follows: <<the best route: ID>>. A higher score indicates a better solution. Please adhere strictly to this format without additional commentary.

3.5. Initializer Agent Prompt

The prompt provided for the initializer agent is designed to guide the agent in constructing routes for multiple salesmen (the exact number specified by {num_salesmen}) from a visual representation of nodes on an image. The prompt directs the agent as follows:

- Starting Point: The prompt specifies that the routes must start and end at a depot, marked as a black square in the image. This sets a clear starting and returning point for each salesman's route.
- Node Visitation: it is stipulated that all nodes must be visited exactly once by only one salesman, ensuring that the task covers all designated points without overlap between salesmen.
- Route Efficiency: The prompt demands minimizing intersections within and between routes. This aims to reduce potential route conflicts and ensures that the paths taken are as efficient as possible.
- Cluster Coverage: each route should cover a cluster of points, implying that the agent should look for logical groupings of close nodes to form each route, enhancing practicality and efficiency.
- Route Length: the emphasis is also on keeping routes short, prioritizing direct paths and proximity among nodes within a route.
- Output Format: The expected output format is very structured, asking for the route of each salesman to be listed sequentially from the depot, through each node, and back to the depot. The format is specified to begin with <<start>> and end with <<end>>, and only the routes are to be listed without any additional text or explanation.

3.6. Critic Agent Prompt

The prompt for the critic agent builds on the initializer's task by refining the suggested routes. It is structured to guide the critic in analyzing the existing routes provided by the initializer and then improving them based on specific criteria. Here is how the prompt directs the agent:

- Initial Instructions: like the initializer agent, the critic starts with the same basic guidelines regarding the depot, unique node visits by each salesman, minimizing route intersections, and covering clustered points.
- Optimization Focus: The key addition for the critic is the instruction to "improve upon the current routes shown in the image by further reducing intersections and optimizing the travel distance." This pushes the agent to look for enhancements over the already suggested routes, focusing on increased efficiency and reduced travel distances.
- Output Format: The format for output remains structured and specific. The critic must list each salesman's route starting and ending at the depot in a precise sequence without additional commentary. The sequence is strictly formatted from the depot to the last node and back, maintaining clarity and consistency.

3.7. Score Agent Prompt

The prompt for the score agent is designed to evaluate and rank multiple solutions for the Traveling Salesman Problem (TSP) based on visual clarity and efficiency. Here is how the prompt directs the agent:

- Evaluation Criteria:
  - Complete Node Coverage: this ensures all nodes are visited precisely once, emphasizing routes that do not skip any nodes.
  - Minimized Crossing Lines: routes with fewer intersections are preferred as they generally suggest a shorter overall path.
  - Route Clarity: the paths should be straightforward to trace visually, with minimal overlapping, which enhances the readability and practicality of the route.
  - Starting and Ending Point: it is essential that the route begins and ends at the same point, labeled as node 0, ensuring a closed loop that is typical for TSP solutions.
- Scoring and Ranking: Each route is assigned a score based on how well it meets the above criteria. The scores are directly related to the route's efficiency and clarity. The prompt specifies that each image representing a different route solution should be scored and then listed with its corresponding score, maintaining a clear format for easy comparison.
- Output: The scores are to be formatted concisely: <<image1: score, image2: score, …, image7: score>>. Additionally, the highest-scoring route is to be highlighted as the best solution with its specific ID in the format: <<the best route: ID>>.
- Format and Procedure: The agent is instructed to strictly adhere to the output format without adding any supplementary explanations or textual content. This structured approach ensures that the outputs are standardized and focused solely on numerical scoring and ranking.

3.8. Test Data

The test data described in this paper are generated through a process where each node's coordinates (x and y) are independently sampled from a uniform distribution ranging from zero to five. This method ensures that all data points are evenly and randomly distributed within a 5×5 square area. The uniform distribution is used to simulate an equal probability of any point within the chosen specified range, which helps create a diverse set of scenarios for evaluating the model's performance on spatial problems like the TSP. This approach is typical in computational experiments where the goal is to

assess algorithmic efficiency under varied yet controlled conditions to ensure that the performance metrics are not biased by any specific configuration of the nodes [11].

The primary reason for using simulated data is to ensure that our MLLM model is tested in scenarios that they have not encountered during training. Given the vast data exposure of MLLMs, including potential benchmark datasets, our approach aims to evaluate the models' generalization capabilities effectively. Additionally, the random generation of nodes across numerous instances ensures a high variability in node distribution and proximity, which rigorously tests the models' adaptability to different spatial challenges.

Our methodology incorporates a novel approach utilizing multiple specialized agents within the MLLM framework. However, to address hallucination in testing results, we took several measures. This approach assigns specific tasks to each agent and enables cross-verification among them, reducing the chances of hallucination [42]. The design patterns, which include reflection and multi-agent collaboration, were shown in the literature to significantly enhance the reliability of AI outputs through iterative refinement and mutual critique among agents [43,44]. Furthermore, although careful prompt engineering was employed to reduce the risk of hallucination, this might not create a complete solution. Fine-tuning the LLM with task-specific and validated data could significantly improve performance, although this requires substantial resources and the availability of the model for fine-tuning, which is currently a limitation for GPT-4o. Despite these constraints, our approach aims to demonstrate the inherent capabilities of MLLMs in their default state, primarily providing results for future improvements. In addition to prompt engineering and agentic design, we are exploring the integration of Retrieval-Augmented Generation (RAG) with our multi-agent system to further reduce hallucinations [45]. This ongoing experiment, still in its early stages due to funding constraints, aims to enhance the robustness of our methodology.

3.9. Ground Truth Solutions

In this study, we employed the Google OR-Tools framework to address the mTSP [46,47], a variant of the TSP, where more than one salesman is involved. The mTSP is known for its NP-hard complexity, making exact solutions impractical for large datasets. To manage the routing challenges efficiently, we utilized the pywrapcp package. The Routing Model from the OR-Tools suite is a powerful tool designed to streamline the process of defining, solving, and analyzing routing problems. The Routing Model is configured with a Euclidean distance matrix representing the distances between nodes. This matrix is crucial for the algorithm as it serves as the foundation upon which the routing decisions are based. To generate initial feasible solutions rapidly, we employed the SAVINGS heuristic, a commonly used approach for vehicle routing problems that provides a good starting solution for further improvement. This heuristic is integrated into the OR-Tools through the routing_enums_pb2.FirstSolutionStrategy.SAVINGS.

Further refinement of the initial solution is achieved using the GUIDED_LOCAL_SEARCH metaheuristic, which is part of the OR-Tools' local search algorithms. This metaheuristic helps navigate the solution space effectively, improving the quality of solutions by escaping local optima—a frequent challenge in route optimization problems. The local search parameters are tuned to balance between computation time and solution quality, making the approach feasible for larger datasets.

Although the solutions derived from this methodology are not guaranteed to be globally optimal due to the heuristic nature of the algorithms, they are generally close to optimal and computationally efficient [46,47]. This approach allows us to handle more significant problem instances that are typically infeasible with exact methods, providing practical solutions within a reasonable timeframe. Thus, the use of Google OR-Tools, particularly the pywrapcp and RoutingModel, represents a robust method for tackling complex routing problems like the mTSP in applied research settings [46,47].

4. Results

This section will explore the solution quality of the two proposed strategies. In our evaluation metrics, we followed the following steps:

1. For each problem size n, we generated 30 distinct instances. The coordinates for each node within these instances were independently sampled from a uniform distribution across a 5 × 5 grid. This controlled setup allows for a comprehensive assessment of the proposed strategies against varied yet uniformly distributed scenarios.
2. The Google OR-Tools framework, employing an Euclidean distance matrix and the SAVINGS heuristic, was used to obtain a benchmark solution for each instance. These solutions serve as our "ground truth (i.e., baseline)" against which the proposed multi-agent strategies are compared.
3. For each instance, the gap is calculated using the following equation:

$$Gap = 100 \times \frac{(Proposed\ Multiagent\ Solution\ Distance - Google\ OR\ Baseline\ Distance)}{(Google\ OR\ Baseline\ Distance)}$$

This metric reflects the percentage improvement or deterioration of our proposed solution relative to the Google OR-Tools solution.

4. For each problem size, the mean and standard deviation of the gap percentages are computed from the 30 instances. These statistics provide insights into the average performance and consistency of the proposed solutions across different scenarios.

To ensure a thorough evaluation of our proposed multi-agent strategies, we compare performance metrics—specifically the mean and standard deviation of the gap percentage—of these strategies against those of zero-shot solutions. Zero-shot learning (ZSL) is a method where models classify new classes they have not seen during training, using semantic understanding to associate indirect information about data. This allows models to make predictions or recognize entities without prior explicit examples of those entities. In the context of this paper, we present the initializer with images that illustrate point distributions on a two-dimensional plane and prompt it to generate a solution without supplying any pre-solved examples. This approach tests the model's capacity to deduce and resolve the task based solely on the visual data provided. Additionally, the solution produced by the initializer is accepted as the final output without further refinement from either the critic or score agent. Moreover, the gap for the zero-shot solutions is calculated using the formula:

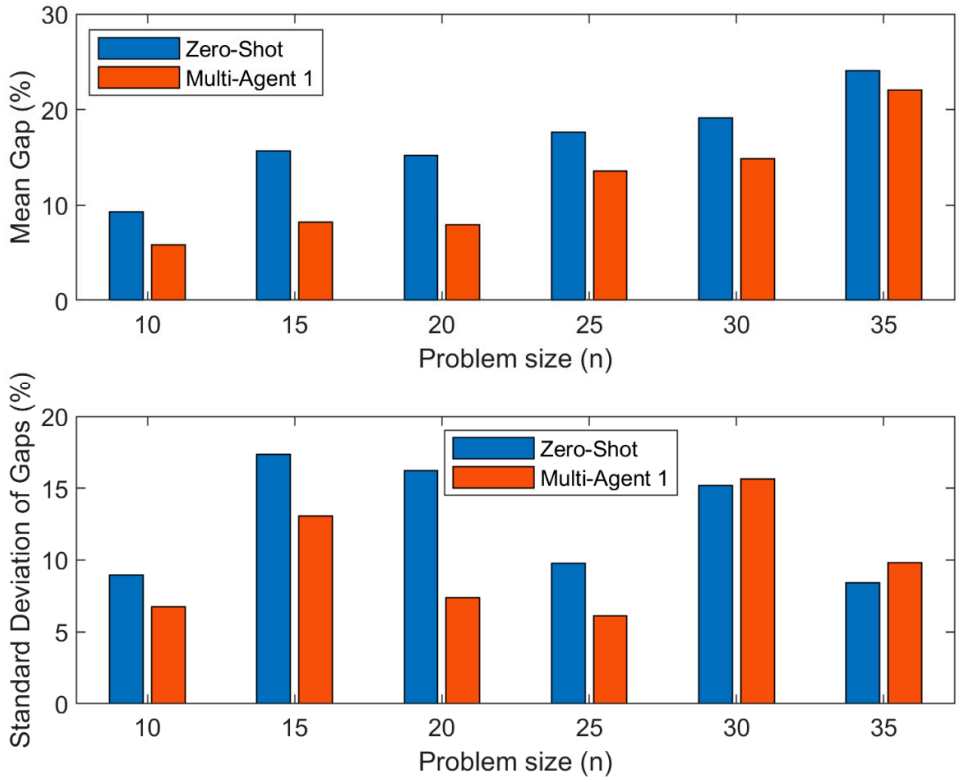
$$Gap = 100 \times \frac{(\text{Zero-shot}\ Solution\ Distance - Google\ OR\ Baseline\ Distance)}{(Google\ OR\ Baseline\ Distance)}$$

4.1. Multi-Agent 1 Solution Quality

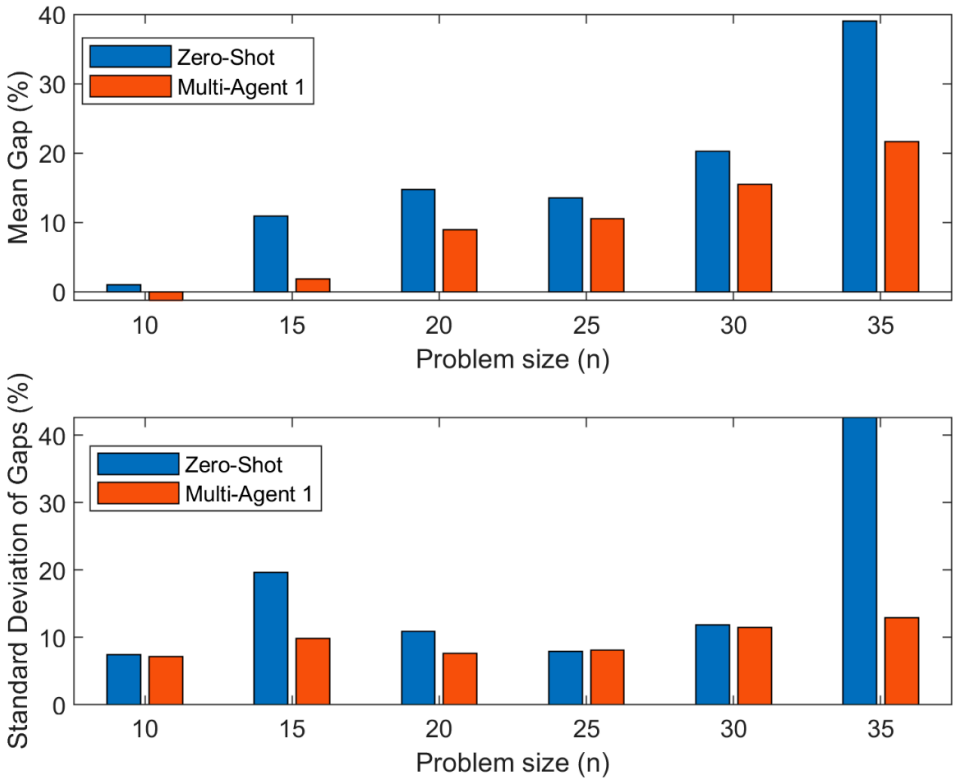
The presented analysis in Figure 3 across three distinct scenarios—solving the traveling salesman problem with one, two, and three salesmen—demonstrates a comparative evaluation of solution quality measured by the mean and standard deviation of gap percentages as the problem size increases. For configurations involving two and three salesmen, the observed negative mean gap values at smaller problem sizes indicate superior performance of the proposed strategies over the solutions generated by the Google OR tool.

Regarding solution times, the Google OR tools were set with a cap of 120 s. For the multi-agent strategies, we opted for a limit of 10 iterations rather than a similar time constraint. This approach was necessary due to the OpenAI API's limitation on the number of model responses allowed per minute, preventing us from setting a fixed time-based stopping criterion like that used for the Google OR tools. This suggests that as the complexity introduced by additional salesmen increases, the proposed strategies are better equipped to handle such complexities, outperforming the baseline available tools set by Google's optimization tools, particularly at smaller problem scales.

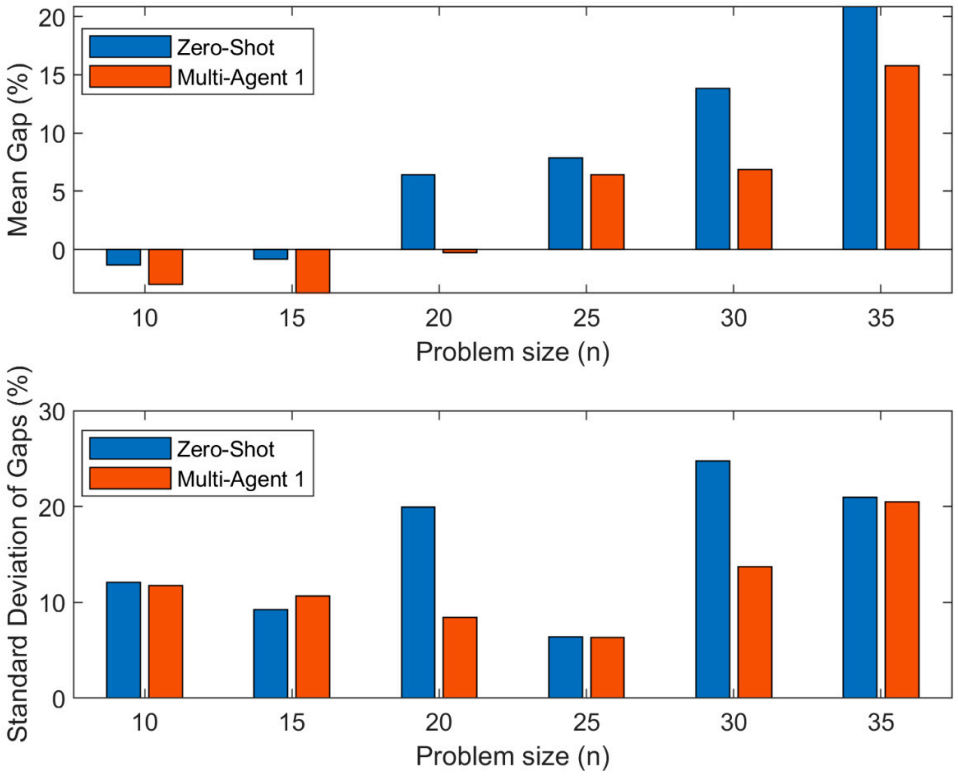
Further, the standard deviation of the gaps consistently illustrates lower variability for the Multi-Agent 1 strategy compared to the baseline method across all problem sizes, indicating a more consistent output from the proposed strategies. This trend is consistent even as the problem size increases, underlining the challenges posed by larger problems and showcasing the proposed solutions' robustness.



(**a**)


Zero-Shot
Multi-Agent 1
Mean Gap (%)
Problem size (n)
Standard Deviation of Gaps (%)
Problem size (n)

(**b**)


Zero-Shot
Multi-Agent 1
Mean Gap (%)
Problem size (n)
Standard Deviation of Gaps (%)
Problem size (n)

(**c**)

Figure 3. Comparative performance of zero-shot and Multi-Agent 1 strategy across different salesmen configurations: analysis of mean gap and standard deviation by problem size: (**a**) one salesman; (**b**) two salesmen; (**c**) three salesmen.

Table 2 presents a comprehensive analysis using the Wilcoxon signed-rank test to statistically validate the improvement in route quality offered by the Multi-Agent 1 strategy over the zero-shot approach. In this paired, two-sided test, the null hypothesis posits that the difference between the routes generated by zero-shot and Multi-Agent 1 strategies (denoted as x–y) has a median of zero. The results indicate a significant improvement with the Multi-Agent 1 approach, as evidenced by the consistently low *p*-values across various problem sizes, particularly for smaller problems.

Table 2. Statistical analysis of route quality improvement using Multi-Agent 1 strategy over zero-shot.

	$m = 1$		$m = 2$		$m = 3$	
Problem Size	***p*-Value**	**Number of Pairs**	***p*-Value**	**Number of Pairs**	***p*-Value**	**Number of Pairs**
10	0.0010	27	0.0002	24	0.0004	29
15	0.0003	27	0.0001	28	0.0020	22
20	0.0004	24	0.0002	20	0.0010	22
25	0.0156	28	0.0005	18	0.0156	19
30	0.0020	16	0.0156	20	0.1250	10
35	0.5000	9	1.0000	7	0.0313	13

As problem sizes increase, the initializer agent encounters more difficulties in effectively managing node visits, leading to an increased rate of hallucinations where the proposed routes fail to visit all designated nodes. This effect of hallucinations decreases the number of valid problem instances available for paired comparisons. Hallucinations are more pronounced in larger problems due to the heightened complexity involved in routing.

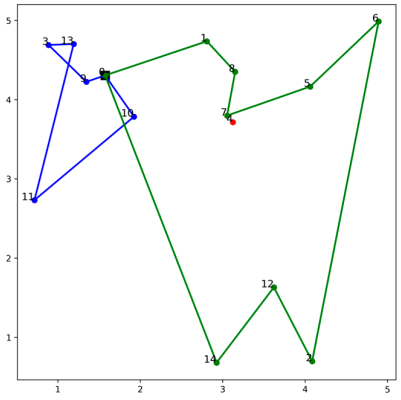
The critic agent plays a pivotal role in this context by identifying and rectifying incomplete or erroneous routes that miss crucial nodes, thereby ensuring the routes are optimized to include all necessary stops. This intervention by the critic agent is essential for enhancing route quality, particularly when initial outputs from the initializer may be flawed. For our statistical analysis, we exclude any problem instances where the zero-shot strategy results in incomplete routes, reducing the number of pairs available for the Wilcoxon signed-rank test. This limitation can affect our ability to detect significant differences between the Multi-Agent 1 strategy and the zero-shot approach, especially when employing non-parametric tests that have less power than the parametric tests.

4.2. Multi-Agent 1 Example 1

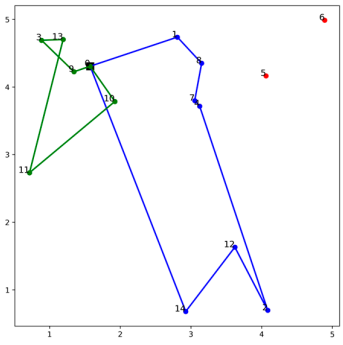
In this example, the multi-agent system comprises an initializer and a critic agent and shows an illustration of an incomplete route hallucination. The process starts with the initializer generating an initial solution which is visualized in the first row of the images. Notably, this initial solution misses one crucial node. The incomplete solution is then presented to the critic agent, which evaluates it and suggests seven possible solutions to improve the initial output. These solutions are visually represented in Figure 4. Subsequently, these images are passed to a score agent, which assigns a score to each solution based on its effectiveness and completeness.

The scoring details reveal that solutions number 1 and 3 are identical, and both received the same score, which indicates consistency in the scoring process by the score agent. Furthermore, solutions number 4 and 7 achieved the highest score of 4, suggesting they are the most promising solutions offered by the critic. It can be observed that the

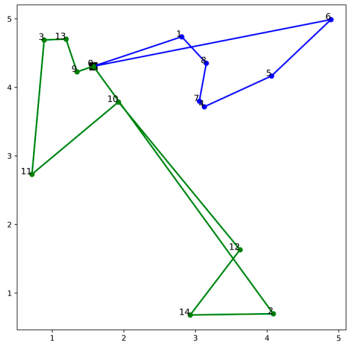
initial solutions displayed in Figures 4 and 5 are generated by the initializer, which processes images illustrating point distributions on a two-dimensional plane. Following this scoring process, the solution with the highest score, in this case, solution number 4, is selected to be passed back to the critic agent for the next iteration. This iterative process aims at refining the solution until it covers all required nodes, which, as noted, happens after the first iteration—indicating that the initial solution's shortcomings are effectively addressed by the multi-agent system. This example illustrates the iterative interaction between the initializer, critic agent, and score agent in improving a solution within a multi-agent system, ultimately leading to a good solution.



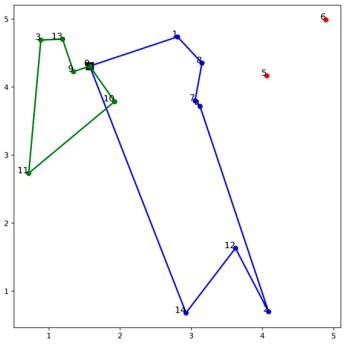
(**a**) Initial solution



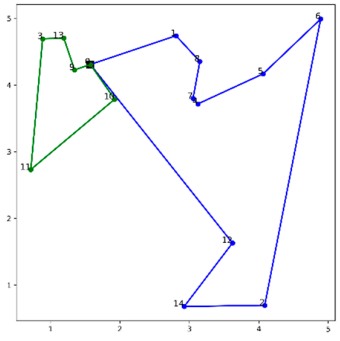
(**b**) Suggested solution #1 score = 3



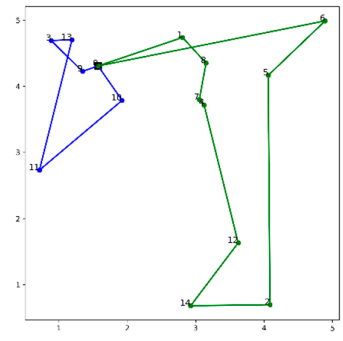
(**c**) Suggested solution #2 score = 2



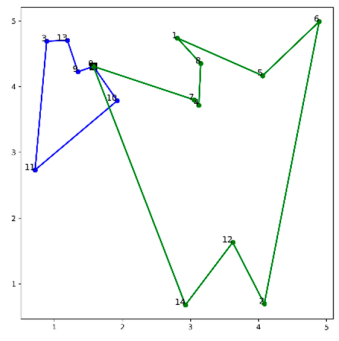
(**d**) Suggested solution #3 score = 3



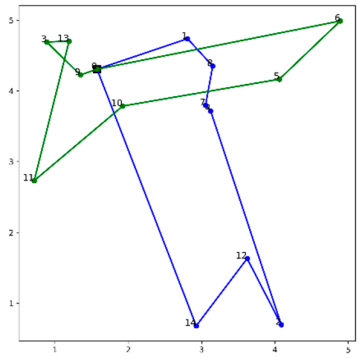
(**e**) Suggested solution #4 score = 4



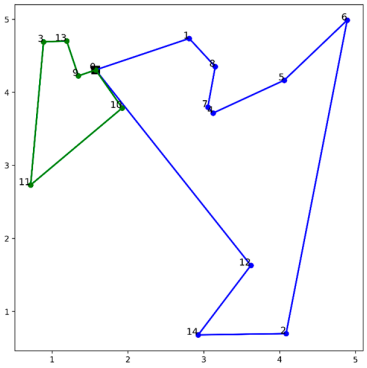
(**f**) Suggested solution #5 score = 1



(**g**) Suggested solution #6 score = 2

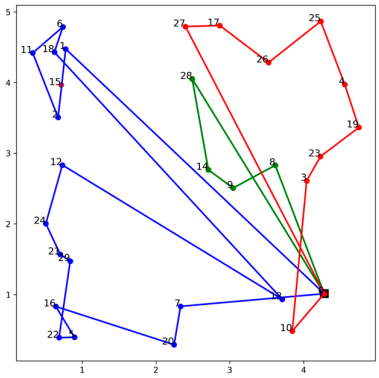


(**h**) suggested solution #7 score = 4

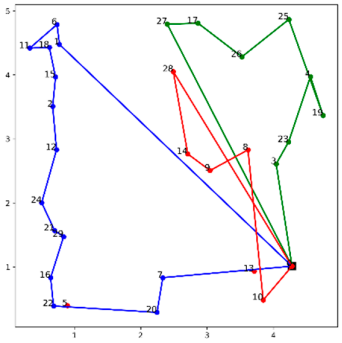


(**i**) Final solution

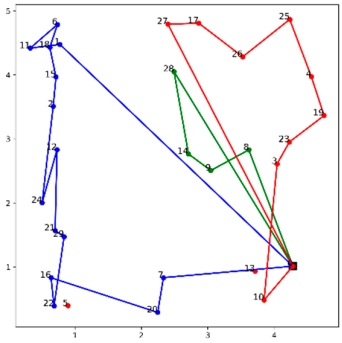
Figure 4. Visual representations of seven proposed solutions by the critic agent, along with scores assigned by the score agent, highlighting the iterative improvement process in the multi-agent system: (**a**) initial solution, (**b**–**e**,**h**) are suggested solutions, and (**i**) is the final solution. Different lines color represent different journeys. Numbers represent the nodes visited by a salesman.



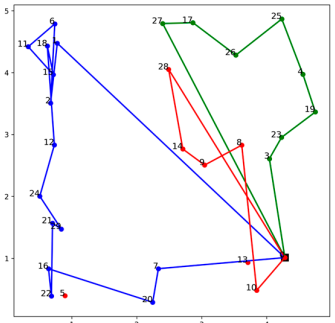
(**a**) Initial solution



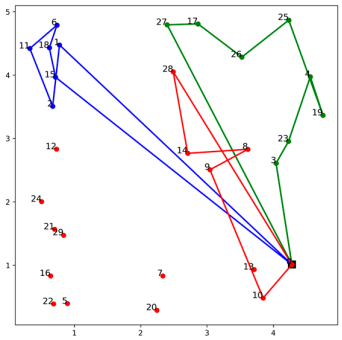
(**b**) Suggested solution #1 score = 3



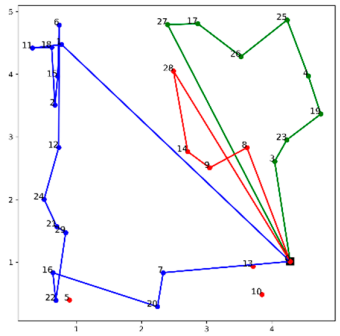
(**c**) Suggested solution #2 score = 4



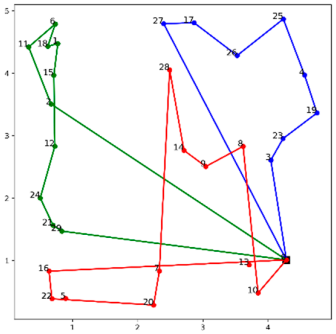
(**d**) Suggested solution #3 score = 3



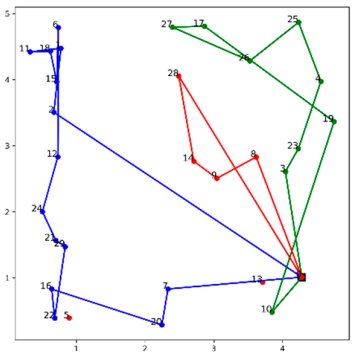
(**e**) Suggested solution #4 score = 1



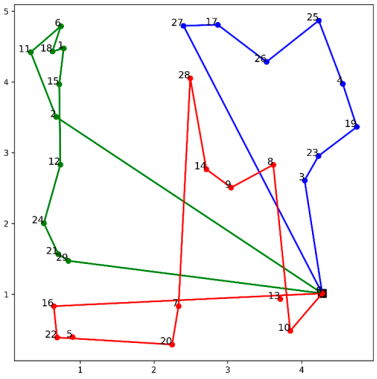
(**f**) Suggested solution #5 score = 2



(**g**) Suggested solution #6 score = 5



(**h**) Suggested solution #7 score = 3



(**i**) Final solution

Figure 5. Evaluation of proposed solutions in the second iteration with scores indicating the completeness of node coverage in a 30-node network. Different lines color represents different journeys. Numbers represent the nodes visited by a salesman.

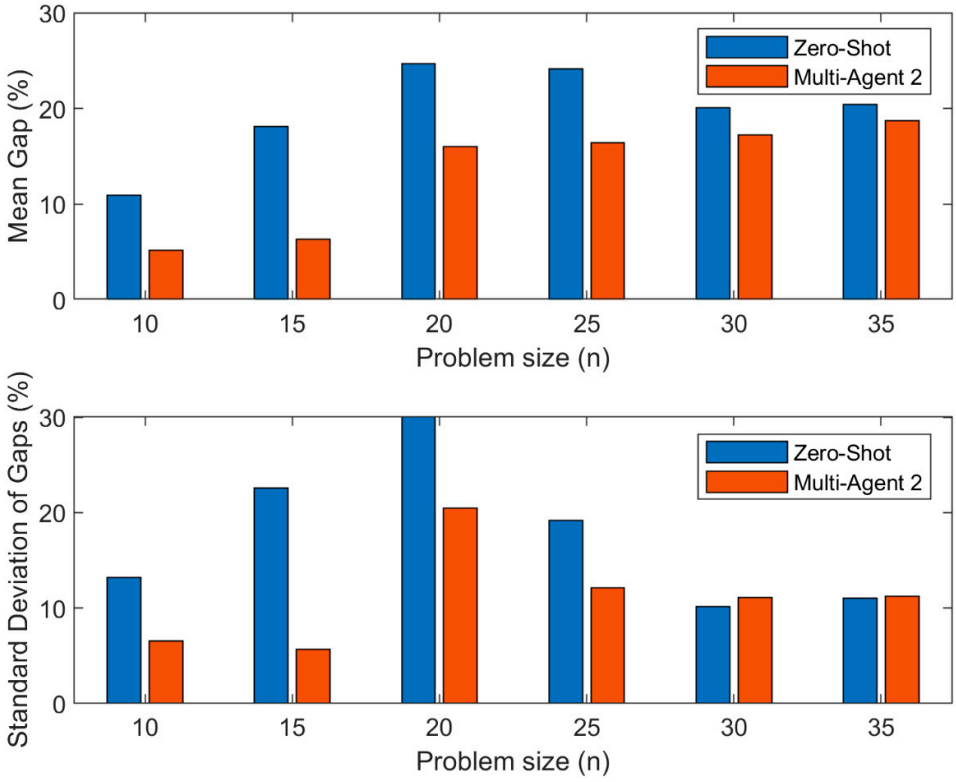
4.3. Multi-Agent 1 Example 2

Figure 5 shows the third iteration of solving a problem with a network comprising 30 nodes. The progression begins with the leading solution from the previous iteration, showcased at the top of the visual table. This serves as the benchmark for the subsequent evaluations conducted by the critic agent. The additional solutions proposed are visually represented and scored based on their coverage and accuracy in addressing the node network. These solutions are listed as follows: Suggested Solution #1 with a score of 3; Suggested Solution #2 with a score of 4; Suggested Solution #3 also scoring 3; Suggested Solution #4 with the lowest score of 1; Suggested Solution #5 scoring 2; Suggested Solution #6 with the highest score of 5; and Suggested Solution #7 scoring 3. The scoring outcomes indicate a clear correlation between the completeness of node coverage and the assigned scores, with missing nodes notably reducing the effectiveness of the solutions. The highest score, a 5, reflects a solution that significantly improves node coverage, suggesting near-optimal completeness. This methodical, iterative approach, facilitated by the interactions between the critic and scoring agents, underscores the efficacy of the multi-agent system in refining solutions to ensure comprehensive node coverage.

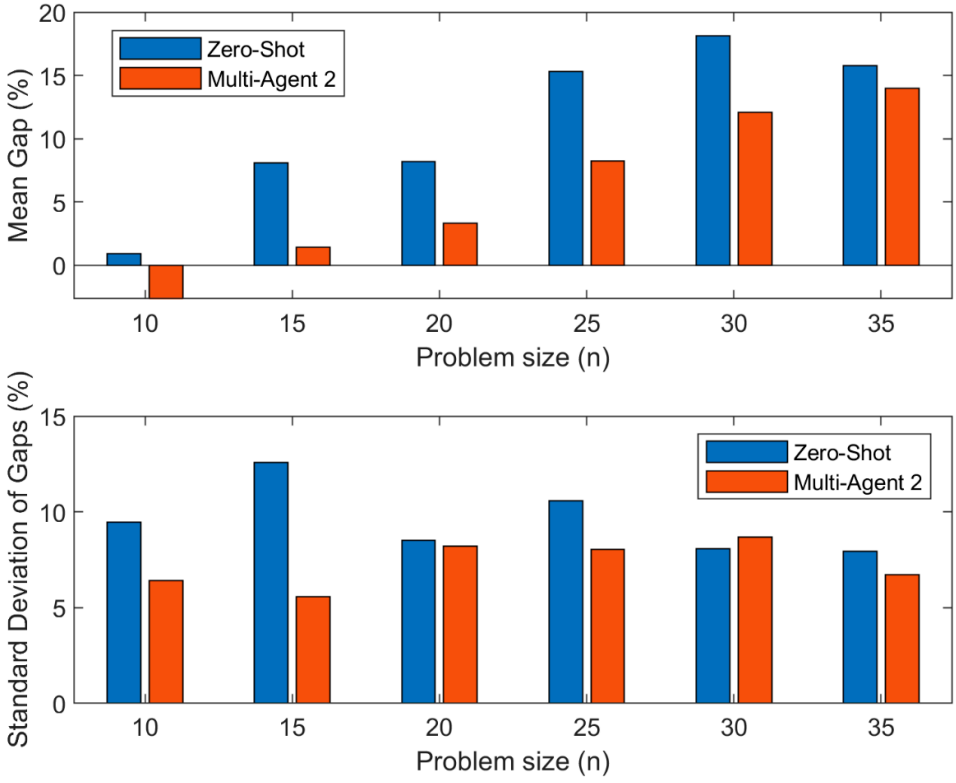
4.4. Multi-Agent 2 Solutions Quality

The analysis of the Multi-Agent 2 strategy performance in addressing the m-Salesmen problem demonstrates a significant enhancement in solution quality, particularly when contrasted with the zero-shot strategy. As outlined in Figure 6, the mean gap of gaps across the different salesman configurations shows that Multi-Agent 2 consistently reduces the solution gaps, even as problem sizes increase. Additionally, the standard deviation of the gap is equal to or improved in Multi-Agent 2 compared to the zero-shot strategy. This is indicative of its robustness in managing complex scenarios more effectively than the zero-shot approach.

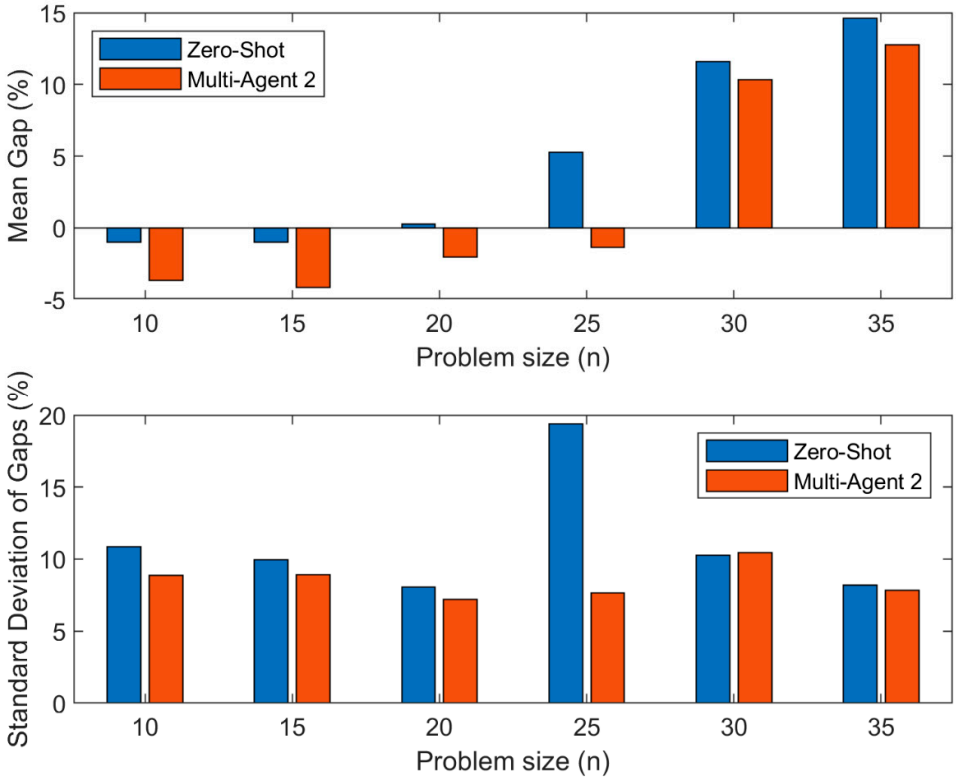
The statistical analysis presented in Table 3 further underscores this improvement. Utilizing the Wilcoxon signed-rank test, the results reveal that the differences in route quality between Multi-Agent 2 and zero-shot strategies are statistically significant, with p-values consistently lower across various configurations and problem sizes. Notably, the number of pairs remains high across all problem sizes for Multi-Agent 2, suggesting fewer instances of hallucinations by the initializer, which are commonly observed in larger problems. This indicates that Multi-Agent 2, even without the scoring phase by a scorer agent, maintains high fidelity in route generation and optimization, emphasizing its efficiency in iterative refinement through the critic agent's interventions.


Mean Gap (%)
Zero-Shot
Multi-Agent 2
Problem size (n)
Standard Deviation of Gaps (%)
Zero-Shot
Multi-Agent 2
Problem size (n)

(**a**)


Mean Gap (%)
Zero-Shot
Multi-Agent 2
Problem size (n)
Standard Deviation of Gaps (%)
Zero-Shot
Multi-Agent 2
Problem size (n)

(**b**)



(**c**)

Figure 6. Comparative performance of zero-shot and Multi-Agent 2 strategy across different salesmen configurations: analysis of mean gap and standard deviation by problem size: (**a**) one salesman; (**b**) two salesmen; (**c**) three salesmen.

This reduction in hallucinations and the subsequent improvement in route quality highlight Multi-Agent 2's effectiveness in leveraging visual reasoning for complex route optimization tasks. The streamlined process, focusing solely on the initializer and critic Agents, proves to be highly effective, particularly in environments where rapid route generation and iterative refinement are crucial. These findings advocate for the potential of simplified visual reasoning models in solving not just theoretical problems but potentially real-world logistical challenges.

Table 3. Statistical analysis of route quality improvement using Multi-Agent 2 strategy over zero-shot.

	$m = 1$		$m = 2$		$m = 3$	
Problem Size	***p*-Value**	**Number of Pairs**	***p*-Value**	**Number of Pairs**	***p*-Value**	**Number of Pairs**
10	0.0004	30	0.0001	30	<0.0001	30
15	0.0001	28	0.0002	26	0.0001	25
20	0.0001	24	0.0005	20	0.0039	19
25	0.0020	19	0.0002	19	0.0005	19
30	0.0020	15	0.0078	16	0.1250	21
35	0.0313	17	0.1250	16	0.0625	14

Figure 7 illustrates the difference in the mean gap percentage between the zero-shot strategy and the two multi-agent strategies across varying problem sizes for $m = 1$, $m = 2$, and $m = 3$ salesmen. The analysis focuses on comparing the improvement each multi-

agent strategy offers over the zero-shot by evaluating the reduction in the mean gap percentage.

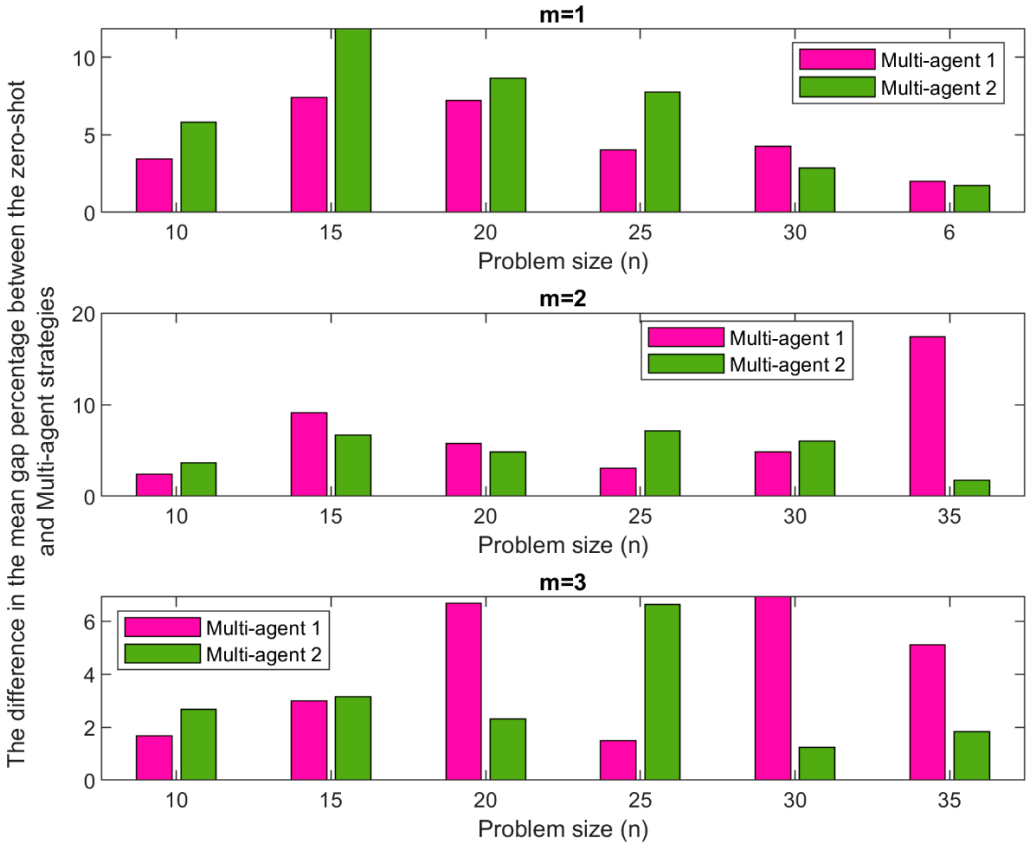
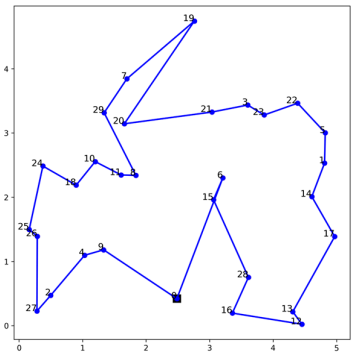


Figure 7. Comparative analysis of mean gap percentage reductions: evaluating Multi-Agent 1 and Multi-Agent 2 against zero-shot across different problem sizes and salesman configurations.

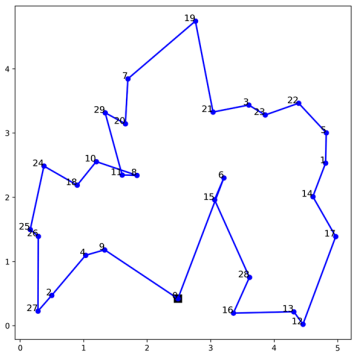
For $m = 1$ (i.e., single salesman), Multi-Agent 2 generally outperforms Multi-Agent 1 at smaller problem sizes, indicating a more effective optimization in simpler scenarios. However, as the problem size increases, the performance advantage of Multi-Agent 2 diminishes slightly, suggesting that the Multi-Agent 2 strategy may be better suited for less complex problems. As the number of salesmen increases to two and three (i.e., $m = 2$ and $m = 3$), the performance dynamics shift. Multi-Agent 1 starts to show more significant improvements over the zero-shot, particularly in medium-sized problems. For instance, at a problem size of 35 for $m = 2$, Multi-Agent 1 substantially outperforms Multi-Agent 2, suggesting that its strategies may be better suited to handling more complex scenarios with multiple salesmen. This visualization underscores that while Multi-Agent 2 excels in simpler setups, Multi-Agent 1 may offer more robust solutions in more challenging environments, effectively addressing zero-shot's shortcomings in larger and more complex problem configurations

4.5. Multi-Agent 2 Example 1

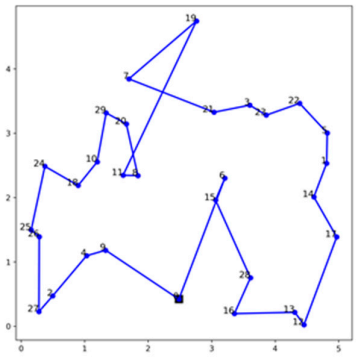
Figure 8 illustrates the Multi-Agent 2 solution using a complex network comprising 30 nodes. This figure showcases the ongoing refinement, beginning with the leading solution from the previous iteration, which serves as a benchmark for subsequent evaluations.



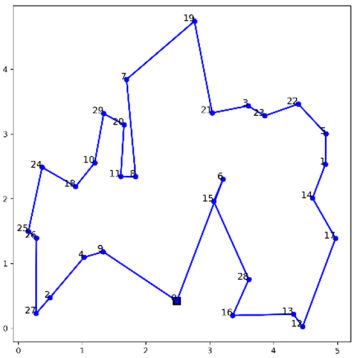
(**a**) Initializer solution
(*total distance* = 24.24)



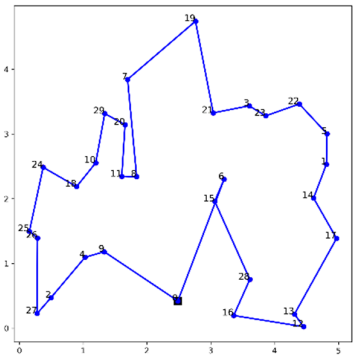
(**b**) Critic solution iteration #1
(*total distance* = 22.86)



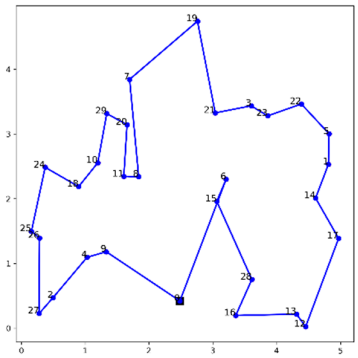
(**c**) Critic solution iteration #2
(*total distance* = 24.73)



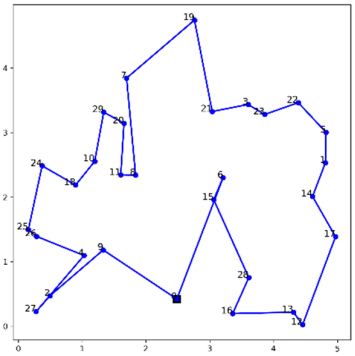
(**d**) Critic solution iteration #3
(*total distance* = 23.56)



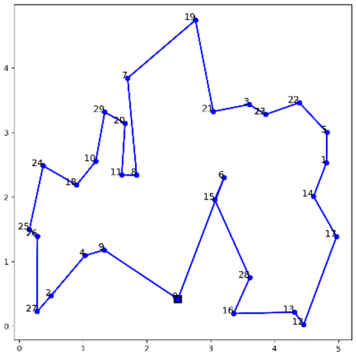
(**e**) Critic solution iteration #4
(*total distance* = 23.60)



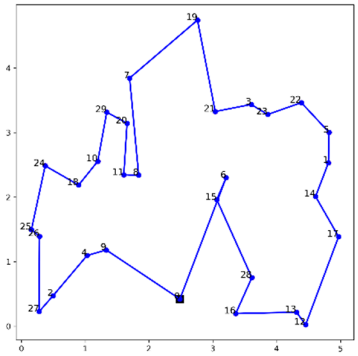
(**f**) Critic solution iteration #5
(*total distance* = 23.56)



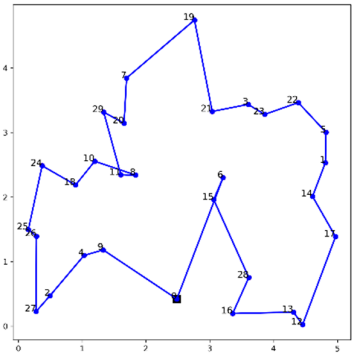
(**g**) Critic solution iteration #6
(*total distance* = 24.31)



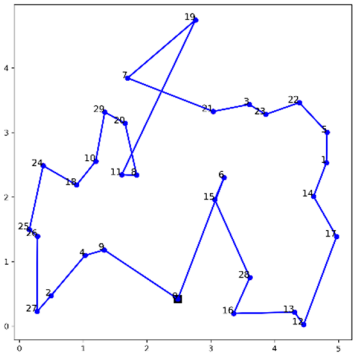
(**h**) Critic solution iteration #7
(*total distance* = 23.56)



(**i**) Critic solution iteration #8
(*total distance* = 23.56)



(**j**) Critic solution iteration #9
(*total distance* = 22.86)



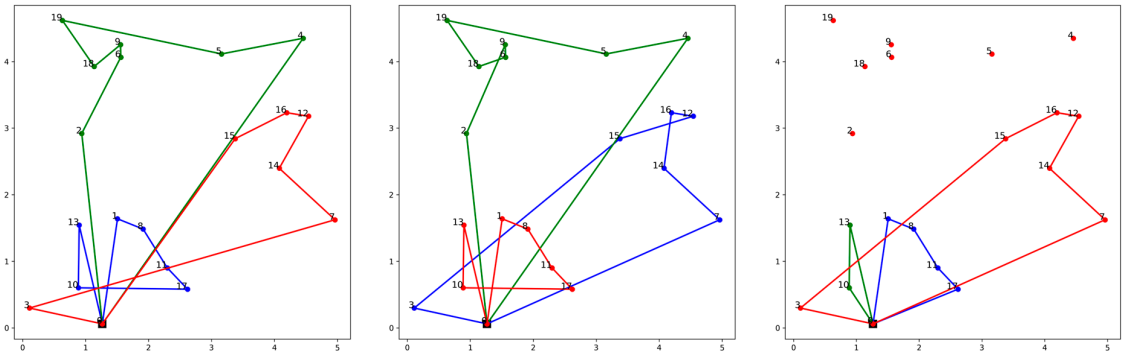
(**k**) Critic solution iteration #10
(*total distance* = 24.73)

Figure 8. Comparative analysis of mean gap percentage reductions: evaluating Multi-Agent 1 and Multi-Agent 2 against zero-shot across different problem sizes and salesman configurations.

The Multi-Agent 2 strategy, which involves only the initializer and critic agents, was shown to be effective in refining routes for a given problem in the previous example. To better understand how this strategy works, let us examine the example in detail. By employing an iterative and systematic approach, the dynamic interaction between the critic and scoring agents effectively highlights the efficacy of the multi-agent system in refining and optimizing solutions. This method ensures comprehensive node coverage and optimizes network connectivity, which is essential for complex system analyses in network optimization studies. In this strategy, the initializer starts with a solution that has a distance of 24.24 units. Over the course of ten iterations, the critic agent works to refine the initial path, and the first iteration achieves the most significant reduction in distance, bringing it down to 22.86 units, which is the optimal outcome in the series of iterations. However, subsequent iterations display inconsistent results, with distances fluctuating and even reverting to 24.73 units at times. This variability underscores the potential of the critic agent to enhance the initial route proposed by the initializer, but it also highlights the inconsistency in the optimization process across different iterations.

4.6. Multi-Agent 2 Example 2

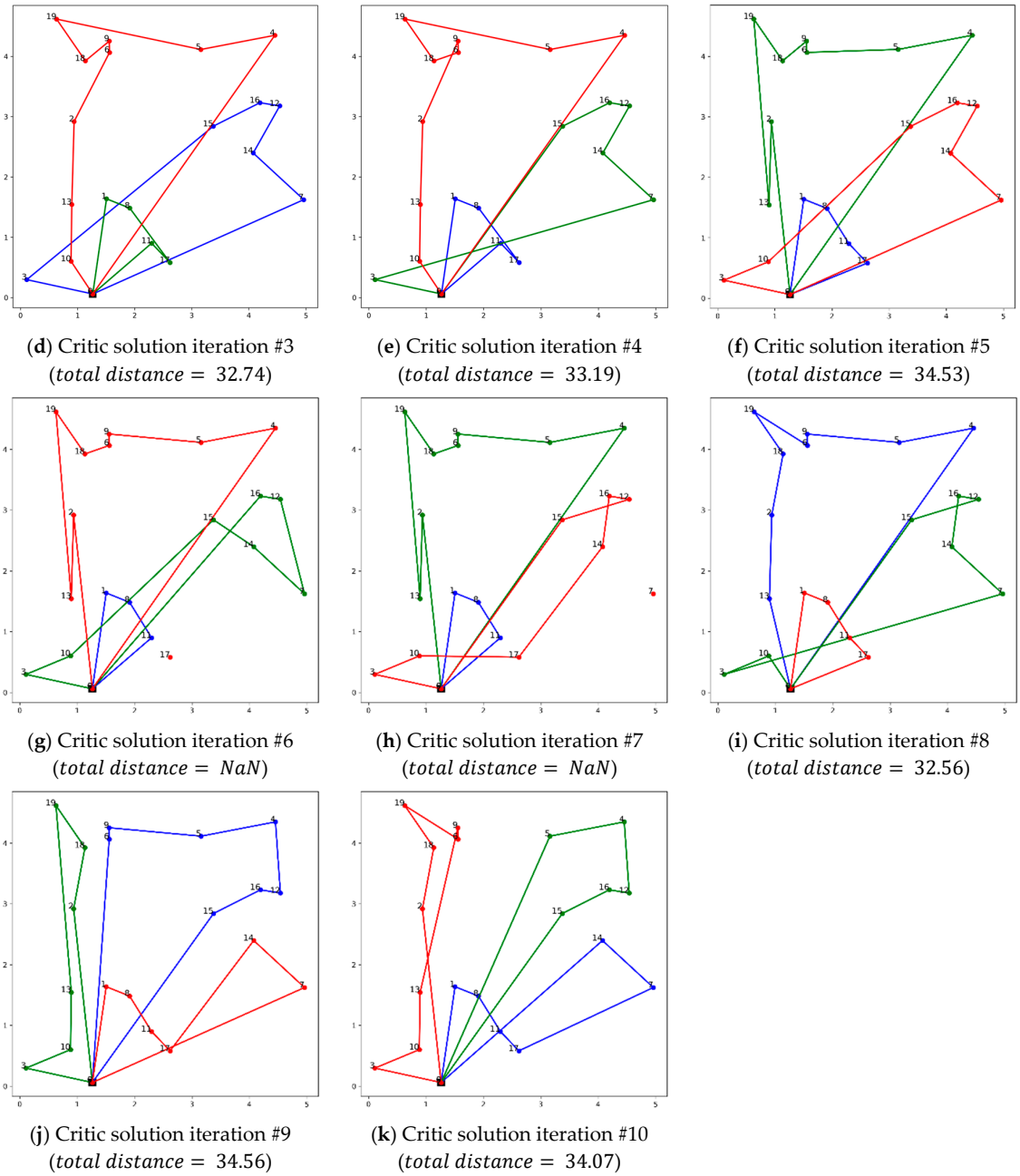
Figure 9 illustrates the intricate and evolving process of route optimization through the Multi-Agent 2 strategy. This example highlights the critic agent's exceptional ability to correct initial errors and make necessary adjustments. The series of iterations exemplifies the critic agent's thorough and deliberate approach to identifying and correcting misconceptions, particularly in instances where vital junctions were omitted during the initial proposals. To address the issue of error rates, we observed that while the Critic and Score Agents significantly reduce errors, they do not eliminate them. Mistakes still occur, particularly in more complex scenarios with larger problem sizes. These errors typically involve incomplete routes where not all nodes are visited. Analysis of the error rates and specific instances of mistakes in the different trials in our experiment were included to highlight the system's robustness and areas for future improvement.



(**a**) Initializer solution ($total\ distance\ =\ 35.43$)

(**b**) Critic solution iteration #1 ($total\ distance\ =\ 35.38$)

(**c**) Critic solution iteration #2 ($total\ distance\ =\ NaN$)



(**d**) Critic solution iteration #3 ($total\ distance = 32.74$)
(**e**) Critic solution iteration #4 ($total\ distance = 33.19$)
(**f**) Critic solution iteration #5 ($total\ distance = 34.53$)
(**g**) Critic solution iteration #6 ($total\ distance = NaN$)
(**h**) Critic solution iteration #7 ($total\ distance = NaN$)
(**i**) Critic solution iteration #8 ($total\ distance = 32.56$)
(**j**) Critic solution iteration #9 ($total\ distance = 34.56$)
(**k**) Critic solution iteration #10 ($total\ distance = 34.07$)

Figure 9. Comparative analysis of mean gap percentage reductions: evaluating Multi-Agent 1 and Multi-Agent 2 against zero-shot across different problem sizes and salesman configurations. the second iteration with scores indicating the completeness of node coverage in a 30-node network. Different lines color represents different journeys. Numbers represent the nodes visited by a salesman.

Initially, the initializer provides a solution with a total distance of 35.43 units. The critic's first two iterations make minor adjustments with negligible reductions in total distance, reflecting the critic's initial struggle to optimize the route effectively. Notably, iterations 2 and 6 demonstrate the occurrence of hallucinations, where the critic agent fails to include all nodes, leading to a NaN (i.e., Not a Number) total distance, indicating an incomplete or erroneous route computation.

Despite these setbacks, the critic agent remarkably recovers in subsequent iterations. Iteration 3 shows a significant improvement with a reduced total distance of 32.74 units, Iterations 4 through 8 oscillate in effectiveness, but each maintains a coherent route. Following other hallucinations in iterations 6 and 7, the agent recovers again in iterations 9 and 10, finalizing with a distance of 34.07 units, demonstrating an overall enhancement from the initial route.

5. Conclusions

This study introduces a pioneering framework for tackling the TSP and mTSP solely through visual reasoning using ChatGPT-4o as an example of MLLMs, marking a departure from conventional reliance on numerical data. By leveraging a multi-agent system in MLLMs, our approach systematically improves route quality by iteratively refining proposed solutions. This iterative process ensures comprehensive node coverage, minimal route intersections, and optimized path lengths without explicit computation of distances. Comparative analyses underscore the competitiveness of our approach against industry-standard tools, validating its potential for practical applications in diverse domains requiring efficient routing solutions

The results showed that both Multi-Agent 1 (which contains an initializer, critic, and scorer) and Multi-Agent 2 (which only contains an initializer and a critic) led to a significant enhancement in the quality of solutions for TSP and mTSP problems. Using MLLM as an agent utilized visual reasoning to bypass traditional computational complexities, offering a novel problem-solving method that mirrors human intuitive processes. The results indicate that Multi-Agent 1, with its trio of initializer, critic, and scorer agents, excels in environments necessitating strict route refinement and evaluation, suggesting a robust framework for managing intricate optimization situations. On the other hand, Multi-Agent 2 streamlines the optimization process by concentrating on iterative refinements conducted by the initializer and critic, proving effective in fast decision-making contexts.

Nevertheless, as problem sizes increase, the emergence of hallucinations—instances where routes fail to visit all designated nodes—highlights a critical flaw in the scaling capabilities of the proposed strategies using MLLMs. This phenomenon raises questions about the dependability of these systems in larger, more intricate situations, a significant limitation in real-world applications like urban planning and logistics. Future research will focus on several key areas to enhance the robustness and applicability of our methodology. First, we will explore other open-source MLLMs to evaluate their performance and ensure generalizability across different models. Second, we plan to experiment with three-dimensional data points (x, y, t) to address more complex time window scenarios and further evaluate our models' adaptability. Third, we will incorporate additional valuation metrics, such as the combination of time and distance costs, to provide a more comprehensive assessment of routing efficiency. Additionally, we will develop and test a heuristic approach for handling nodes in close proximity by representing groups of closely situated nodes as a single virtual node, thereby simplifying complex node arrangements while maintaining route accuracy. Finally, we will segment the entire node layout into patches, feeding them sequentially into the model with explicit positional indications, allowing the model to deduce comprehensive routes that encompass all nodes within these patches. These steps will address the current limitations and extend the applicability of our methodology to various real-world scenarios.

Author Contributions: Conceptualization, M.E., A.A. (Ahmad Abutahoun), T.I.A., A.J., H.I.A., S.J., A.A. (Ahmed Abdelhay), S.G. and A.R.; methodology, M.E.; software, M.E.; validation, M.E., A.A. (Ahmad Abutahoun), T.I.A., A.J., H.I.A. and S. J.; resources, M.E., S.G. and A.R.; writing—original draft preparation, A.A. (Ahmad Abutahoun), T.I.A., A.J., H.I.A., S.J. and A.A. (Ahmed Abdelhay); writing—review and editing, H.I.A., S.G. and A. R.; visualization, A.A. (Ahmed Abdelhay); supervision, S.G. and A.R.; project administration, S.G. and A.R.; funding acquisition, M.E., S.G. and A.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are available upon request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Liu, S.; Chen, C.; Qu, X.; Tang, K.; Ong, Y.-S. Large Language Models as Evolutionary Optimizers. *arXiv* **2023**, arXiv:2310.19046.
2. Yang, C.; Wang, X.; Lu, Y.; Liu, H.; Le, Q.V.; Zhou, D.; Chen, X. Large Language Models as Optimizers. *arXiv* **2023**, arXiv2309.03409.
3. Bellodi, E.; Bertagnon, A.; Gavanelli, M.; Zese, R. *Improving the Efficiency of Euclidean TSP Solving in Constraint Programming by Predicting Effective Nocrossing Constraints*; Springer International Publishing: Cham, Switzerland, 2021; pp. 318–334. https://doi.org/10.1007/978-3-030-77091-4_20.
4. Antuori, V.; Hebrard, E.; Huguet, M.-J.; Essodaigui, S.; Nguyen, A. *Leveraging Reinforcement Learning, Constraint Programming and Local Search: A Case Study in Car Manufacturing*; Springer International Publishing: Cham, Switzerland, 2020; pp. 657–672. https://doi.org/10.1007/978-3-030-58475-7_38.
5. Hudson, B.; Li, Q.; Malencia, M.; Prorok, A. Graph Neural Network Guided Local Search for the Traveling Salesperson Problem. *arXiv* **2021**, arXiv:2110.05291. https://doi.org/10.48550/arxiv.2110.05291.
6. Saremi, S.; Mirjalili, S.; Lewis, A. How Important Is a Transfer Function in Discrete Heuristic Algorithms. *Neural Comput. Appl.* **2014**, *26*, 625–640. https://doi.org/10.1007/s00521-014-1743-5.
7. Joshi, C.K.; Cappart, Q.; Rousseau, L.-M.; Laurent, T. Learning the Travelling Salesperson Problem Requires Rethinking Generalization. *Constraints* **2022**, *27*, 70–98. https://doi.org/10.1007/s10601-022-09327-y.
8. Zhang, K.; He, F.; Zhang, Z.; Lin, X.; Li, M. Multi-Vehicle Routing Problems with Soft Time Windows: A Multi-Agent Reinforcement Learning Approach. *Transp. Res. Part. C Emerg. Technol.* **2020**, *121*, 102861. https://doi.org/10.1016/j.trc.2020.102861.
9. Montiel, O.; Delgadillo, F.J.D. Reducing the Size of Combinatorial Optimization Problems Using the Operator Vaccine by Fuzzy Selector With Adaptive Heuristics. *Math. Probl. Eng.* **2015**, *2015*, 713043. https://doi.org/10.1155/2015/713043.
10. Xu, X.; Yuan, H.; Liptrott, M.; Trovati, M. Two Phase Heuristic Algorithm for the Multiple-Travelling Salesman Problem. *Soft Comput.* **2018**, *22*, 6567–6581.
11. Cheikhrouhou, O.; Khoufi, I. A Comprehensive Survey on the Multiple Traveling Salesman Problem: Applications, Approaches and Taxonomy. *Comput. Sci. Rev.* **2021**, *40*, 100369.
12. Jaradat, S.; Alhadidi, T.I.; Ashqar, H.I.; Hossain, A.; Elhenawy, M. Exploring Traffic Crash Narratives in Jordan Using Text Mining Analytics. *arXiv* **2024**, arXiv:2406.09438.
13. Radwan, A.; Amarneh, M.; Alawneh, H.; Ashqar, H.I.; AlSobeh, A.; Magableh, A.A.A.R. Predictive Analytics in Mental Health Leveraging LLM Embeddings and Machine Learning Models for Social Media Analysis. *Int. J. Web Serv. Res. (IJWSR)* **2024**, *21*, 1–22.
14. Ashqar, H.I.; Alhadidi, T.I.; Elhenawy, M.; Khanfar, N.O. The Use of Multimodal Large Language Models to Detect Objects from Thermal Images: Transportation Applications. *arXiv* **2024**, arXiv:2406.13898.
15. Alhadidi, T.I.; Jaber, A.; Jaradat, S.; Ashqar, H.I.; Elhenawy, M. Object Detection Using Oriented Window Learning Vi-Sion Transformer: Roadway Assets Recognition. *arXiv* **2024**, arXiv:2406.10712.
16. Tami, M.A.; Ashqar, H.I.; Elhenawy, M. Using Multimodal Large Language Models for Automated Detection of Traffic Safety Critical Events. *arXiv* **2024**, arXiv:2406.13894.
17. Faigl, J.; Hollinger, G.A. *Self-Organizing Map for the Prize-Collecting Traveling Salesman Problem*; Springer International Publishing: Cham, Switzerland, 2014; pp. 281–291. https://doi.org/10.1007/978-3-319-07695-9_27.
18. Sahai, T.; Ziessler, A.; Klus, S.; Dellnitz, M. Continuous Relaxations for the Traveling Salesman Problem. *Nonlinear Dyn.* **2019**, *97*, 2003–2022. https://doi.org/10.1007/s11071-019-05092-5.
19. Akhand, M.A.H.; Ayon, S.I.; Shahriyar, S.A.; Siddique, N.; Adeli, H. Discrete Spider Monkey Optimization for Travelling Salesman Problem. *Appl. Soft Comput.* **2020**, *86*, 105887. https://doi.org/10.1016/j.asoc.2019.105887.
20. Guang, R.; Khodadian, S. Linear Function Approximation as a Resource Efficient Method to Solve the Travelling Salesman Problem. *J. Stud. Res.* **2022**, *10*. https://doi.org/10.47611/jsrhs.v10i4.2143.
21. Mandal, R.K.; Mukherjee, P.; Maitra, M. Solving Travelling Salesman Problem Using Artificial Immune System Optimization (AISO). *J. Sci. Res.* **2022**, *66*, 114–120. https://doi.org/10.37398/jsr.2022.660416.
22. Chen, Y.; Yang, J. Research on Traveling Salesman Problem Based on the Ant Colony Optimization Algorithm and Genetic Algorithm. *Open Autom. Control Syst. J.* **2015**, *7*, 1329–1334. https://doi.org/10.2174/1874444301507011329.
23. Barán, B.; Gómez, O.M. Omicron ACO. A New Ant Colony Optimization Algorithm. *Clei Electron. J.* **2018**, *8*. https://doi.org/10.19153/cleiej.8.1.5.
24. Shahadat, A.S.B.; Akhand, M.A.H.; Kamal, A.S. Visibility Adaptation in Ant Colony Optimization for Solving Traveling Salesman Problem. *Mathematics* **2022**, *10*, 2448. https://doi.org/10.3390/math10142448.
25. Xu, H.; Lan, H. An Adaptive Layered Clustering Framework with Improved Genetic Algorithm for Solving Large-Scale Traveling Salesman Problems. *Electronics* **2023**, *12*, 1681. https://doi.org/10.20944/preprints202302.0412.v1.

26. Herrera, B.A.L.d.M.; Coelho, L.d.S.; Steiner, M.T.A. Quantum Inspired Particle Swarm Combined With Lin-Kernighan-Helsgaun Method to the Traveling Salesman Problem. *Pesqui. Oper.* **2015**, *35*, 465–488. https://doi.org/10.1590/0101-7438.2015.035.03.0465.
27. Evangelos, S.; Papalitsas, C.; Andronikos, T. Experimental Analysis of Quantum Annealers and Hybrid Solvers Using Benchmark Optimization Problems. *Mathematics* **2022**, *10*, 1294. https://doi.org/10.48550/arxiv.2202.08939.
28. El-Samak, A.F.; Ashour, W.M. Optimization of Traveling Salesman Problem Using Affinity Propagation Clustering and Genetic Algorithm. *J. Artif. Intell. Soft Comput. Res.* **2015**, *5*, 239–245. https://doi.org/10.1515/jaiscr-2015-0032.
29. Mzili, I.; Mzili, T.; Riffi, M.E. Efficient Routing Optimization with Discrete Penguins Search Algorithm for MTSP. *Decis. Mak. Appl. Manag. Eng.* **2023**, *6*, 730–743. https://doi.org/10.31181/dmame04092023m.
30. Jiang, M.; Ruan, Y.; Huang, S.; Liao, S.; Pitis, S.; Grosse, R.B.; Ba, J. Calibrating Language Models via Augmented Prompt Ensembles. 2023. Available online: https://openreview.net/forum?id=L0dc4wqbNs#all (accessed on 8 August 2024).
31. Pitis, S.; Zhang, M.R.; Wang, A.; Ba, J. Boosted Prompt Ensembles for Large Language Models. *arXiv* **2023**, arXiv:2304.05970.
32. Kojima, T.; Gu, S.S.; Reid, M.; Matsuo, Y.; Iwasawa, Y. Large Language Models Are Zero-Shot Reasoners. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 22199–22213.
33. Zheng, J.; He, K.; Zhou, J.; Yan, J.; Li, C.-M. Reinforced Lin-Kernighan-Helsgaun Algorithms for the Traveling Salesman Problems. *arXiv* **2022**, arXiv:2207.03876. https://doi.org/10.48550/arxiv.2207.03876.
34. Min, S.; Lyu, X.; Holtzman, A.; Artetxe, M.; Lewis, M.; Hajishirzi, H.; Zettlemoyer, L. Rethinking the Role of Demonstrations: What Makes In-Context Learning Work? 2022. Available online: https://openreview.net/forum?id=cnRGMv-Ak7u (accessed on 8 August 2024).
35. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models Are Few-Shot Learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
36. Zhang, Z.; Zhang, A.; Li, M.; Smola, A. Automatic Chain of Thought Prompting in Large Language Models. *arXiv* **2022**, arXiv:2210.03493.
37. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 24824–24837.
38. Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; Zhou, D. Self-Consistency Improves Chain of Thought Reasoning in Language Models. *arXiv* **2022**, arXiv:2203.11171.
39. Wu, J.; Gan, W.; Chen, Z.; Wan, S.; Yu, P.S. Multimodal Large Language Models: A Survey. In Proceedings of the 2023 IEEE International Conference on Big Data (BigData), Sorrento, Italy, 15–18 December 2023.
40. Huang, Y.; Zhang, W.; Feng, L.; Wu, X.; Tan, K.C. How Multimodal Integration Boost the Performance of Llm for Optimization: Case Study on Capacitated Vehicle Routing Problems. *arXiv* **2024**, arXiv:2403.01757.
41. Huang, Z.; Shi, G.; Sukhatme, G.S. From Words to Routes: Applying Large Language Models to Vehicle Routing. *arXiv* **2024**, arXiv:2403.10795.
42. Zhang, J.; Li, J.; Zhang, Y.; Wu, Q.; Wu, X.; Shu, F.; Jin, S.; Chen, W. Collaborative Intelligent Reflecting Surface Networks with Multi-Agent Reinforcement Learning. *IEEE J. Sel. Top. Signal Process* **2022**, *16*, 532–545.
43. Li, Y.; Ren, S.; Wu, P.; Chen, S.; Feng, C.; Zhang, W. Learning Distilled Collaboration Graph for Multi-Agent Perception. *Adv. Neural Inf. Process Syst.* **2021**, *34*, 29541–29552.
44. Mazumder, S.; Liu, B.; Ma, N.; Wang, S.; Amazon, A.I. Continuous and Interactive Factual Knowledge Learning in Verification Dialogues. In Proceedings of the NeurIPS-2020 Workshop on Human and Machine in-the-Loop Evaluation and Learning Strategies, December 2020. Available online: https://www.cs.uic.edu/~liub/publications/Neurips_workshop_HAMLETS_camera_ready.pdf (accessed on 8 August 2024).
45. Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.; Rocktäschel, T. Retrieval-Augmented Generation for Knowledge-Intensive Nlp Tasks. *Adv. Neural Inf. Process Syst.* **2020**, *33*, 9459–9474.
46. Deudon, M.; Cournut, P.; Lacoste, A.; Adulyasak, Y.; Rousseau, L.-M. Learning Heuristics for the Tsp by Policy Gradient. In Proceedings of the Integration of Constraint Programming, Artificial Intelligence, and Operations Research: 15th International Conference, CPAIOR 2018, Delft, The Netherlands, 26–29 June 2018; Proceedings 15; Springer: Berlin/Heidelberg, Germany, 2018; pp. 170–181.
47. Chen, W.; Zhang, Y.; Zhou, Y. Integrated Scheduling of Zone Picking and Vehicle Routing Problem with Time Windows in the Front Warehouse Mode. *Comput. Ind. Eng.* **2022**, *163*, 107823.