LLM-Enhanced Constraint Programming for Task Planning in Heterogeneous Multi-Agent Systems

Jozsef Palmieri[1], Martina Lippi[2], Alessandro Marino[1]

***Abstract*—Multi-agent systems (MAS) of humans and robots are increasingly applied to complex tasks, yet face challenges in translating natural language instructions into executable plans and managing heterogeneous agents. Traditional planning methods provide guarantees but are rigid and knowledge-intensive, while Large Language Models (LLMs) offer flexibility but lack reliability. We propose a hybrid framework that integrates LLMs with Constraint Programming (CP), combining intuitive task decomposition with formal optimization. Simulation experiments in object-sorting and structure-building scenarios show that our approach delivers more feasible, efficient, and optimized plans than LLM-only baselines, while reducing computation time. The results highlight the value of uniting AI-based reasoning with optimization techniques for reliable task planning in heterogeneous MAS.**

I. Introduction

Multi-agent systems (MAS), especially those involving heterogeneous agents such as robots and humans (Figure 1), are playing an increasingly critical role in addressing complex tasks under dynamic and uncertain conditions. The reason for this growing adoption can be attributed to their capacity to combine the unique cognitive strengths of humans and the physical ones of the robots, thereby enhancing productivity and operational efficiency. Their applications span a wide range of domains, including intelligent manufacturing, urban search and rescue, logistics, and precision agriculture [1], [2], [3], [4], [5]. However, within this domain, several critical challenges remain still unresolved, including: i) the translation of natural language instructions, frequently ambiguous and differing in their level of detail, into efficient and executable plans for heterogeneous agents, and ii) the management of system heterogeneity, which entails addressing inherent differences in task execution. Addressing this second challenge involves optimally allocating and scheduling activities across agents to improve performance while meeting the given constraints. In the last few decades, such kind of problems have been mainly addressed using formal planning techniques capable of producing optimal or sub-optimal solutions. Such approaches, ranging from classical planners relying on the Planning Domain Definition Language (PDDL) to Constraint Programming (CP) and Mixed Integer Linear Programming (MILP), provide solid guarantees with respect to plan correctness, feasibility, and resource allocation [6], [7]. Nonetheless, because all these approaches rely on a strict and structured formalism, the attainment of a complete and accurate formalization of the scenario, that is, the representation of the problem domain, agent capabilities, and task dependencies, requires considerable effort, profound domain knowledge, and advanced expertise in logic and planning languages. Therefore, this reliance can be regarded as the main bottleneck limiting the adoption of such approaches in complex real-world scenarios. A further limitation is the need for substantial re-engineering of the underlying models whenever the system must be adapted to new tasks or to accommodate unforeseen changes. Recent advances in the Large Language Models (LLMs) field have given rise to new paradigms in robot task planning, promoting the development of more flexible and intuitive methodologies. Thanks to their advanced natural language understanding, extensive world knowledge, and high-level reasoning abilities, LLMs are able to correctly interpret human instructions and generate adaptive task plans [8], [9]. In particular, they can decompose complex tasks into hierarchical structures [9],

Fig. 1. Example of a multi-agent system involving both humans and robots cooperating to achieve a common goal.

[1]J. Palmieri and A. Marino are with the Department of Electrical and Information Engineering, University of Cassino and Southern Lazio, Via G. Di Biasio 43, Cassino, Italy,{jozsef.palmieri, al.marino}@unicas.it.

[2] M. Lippi is with the Department of Civil, Computer Science and Aeronautical Technologies Engineering, Roma Tre University, Via della Vasca Navale 79, Roma, Italy, {martina.lippi}@uniroma3.it.

This work was supported by Project CONCERTO-A COgNitive arChitecture for sEamless human-Robot inTeractiOn, CUP H53C24001050006, funded by EU in NextGenerationEU plan through the Italian "Bando Prin 2022 - D.D. 104 del 02-02-2022"

generate plans, and manage the allocation and scheduling of activities. Therefore, LLM-based frameworks, such as the one proposed in [10] that translates natural language into action sequences for embodied agents, can be regarded as powerful and intuitive interfaces, enabling even non-expert users to engage effectively with MAS. Although promising, LLM-based frameworks are nonetheless subject to several important limitations. Most importantly, the plans they generate often lack formal correctness, feasibility, and reliability [11], [12]. Indeed, without a formalism that ensures solution correctness, and because of their tendency to produce unreliable or "hallucinated" outputs, LLMs are prone to generating plans that violate critical safety requirements, include actions that cannot be carried out, or assign tasks to agents who lack the skills or resources to perform them [13], [14]. To overcome the inherent limitations of both formal methods and purely LLM-based approaches, we propose a novel hybrid framework that integrates the rigor of Constraint Programming (CP) with the flexibility of Large Language Models (LLMs).

II. Constraint Programming (CP)

Constraint Programming (CP) is a logic-based paradigm originally developed for solving Constraint Satisfaction Problems (CSPs), where the objective is to assign values to variables under predefined constraints and rules. In recent years, however, CP has also been successfully applied to Constraint Optimization Problems (COPs), including timetabling, factory scheduling, and the allocation of time or resources to events [15][16]. A distinctive feature of CP lies in its ability to handle a wide range of decision variables, such as integer, continuous, and interval variables. Interval variables are commonly employed to model temporal windows within which events can take place. Interval variables may also be defined as *optional*, indicating that they can be excluded from the solution and thus be ignored by any constraint or expression involving them. Such variables are particularly useful for modeling optional tasks (i.e., tasks that may not be executed) or for representing alternatives in tasks and resources (i.e., cases where the same task can be performed in multiple ways). Within CP, each decision variable is associated with a domain, namely the set of admissible values it can assume. During the solving process, infeasible values are systematically pruned from these domains through an iterative mechanism known as constraint propagation. Furthermore, CP natively supports a broad range of constraint types, including arithmetic expressions, logical disjunctions, conditional if–then rules, and global constraints. The latter represent a special class of constraints which, in contrast to local or primitive ones (e.g., arithmetic expressions), are defined over a variable number of arguments. The use of these constraints offers two main advantages. First, they facilitate problem modeling, as they encapsulate some of those complex logical or combinatorial concepts commonly found in real-world scenarios (e.g., "*everything must be different*" or "*tasks must not overlap*"). Second, each of them comes with a specialized filtering algorithm capable of pruning the search space far more effectively than would occur if the same constraint were implemented using dozens or even hundreds of local (primitive) constraints. More details on CP can be found in [17].

III. Problem formulation and Proposed Solution

Building on the above considerations, we are now ready to introduce the main problem addressed in this work.

Problem 1. *Let us consider a heterogeneous multi-agent system composed of human and robotic agents, each endowed with distinct skills and located in a specific place. In addition, let us assume that a set of resources (e.g., raw materials for mixture preparation or for the construction of objects and structures) is available in the shared workspace, each characterized by distinctive properties. Finally, let us suppose the availability of a high-level instruction, expressed in natural language, outlining the scenario, the team, and the mission to be achieved. The objective is twofold:*

1) *to derive all required tasks and their physical (e.g., object properties, tools) and temporal (e.g., precedence) constraints;*
2) *to compute the optimal allocation and scheduling of activities to complete the mission while minimizing a cost function that balances robotic energy, human workload, and overall makespan, under all execution constraints.*

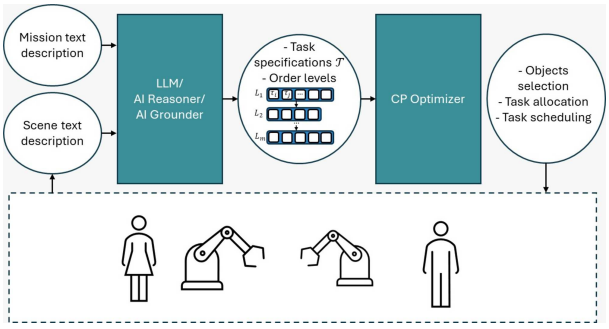

Fig. 2. Proposed architecture.

Figure 2 shows the proposed architecture for solving Problem 1. In particular, it is a two-layer architecture that combines the rigor of constraint programming with the flexibility of LLMs. The first layer, based on an LLM, leverages its world knowledge, reasoning capabilities, and grounding mechanisms to transform the given natural language scenario description into a structured and ordered sequence of subtasks. The second layer, by contrast, builds on a CP solver to handle the complex tasks of combinatorial optimization and correctness verification. Specifically, it translates the structured plan produced by the first layer into a Constraint Programming formulation and ensures that the resulting plan is logically consistent, feasible, and optimized with respect to the defined performance metrics. We validated our framework against a purely LLM-based approach in a multi-agent blocksworld-inspired scenario, showing that the proposed method markedly

enhances plan reliability, feasibility, and optimality, while retaining the natural language flexibility and generalization capabilities of LLMs.

The proposed framework was validated through several simulation experiments. More in detail, The validation process addressed two representative case studies: object sorting and structure construction. The first required allocating objects into boxes subject to predefined constraints, whereas the second involved arranging them on a grid to replicate target structures with minimal resource utilization. The assessment of performance was conducted in terms of solution quality, computational efficiency, and feasibility. The obtained findings corroborated the effectiveness of the proposed framework. In particular, they highlighted that the proposed framework is able to generate feasible solutions with lower objective function values, reduced makespan, and decreased energy consumption, while also achieving the lowest computation times across all experiments. In contrast, the baseline method exhibited higher error rates, longer computation times, and a lower proportion of feasible solutions. Overall, these results demonstrate the effectiveness of the proposed approach and emphasize the value of integrating optimization techniques with AI-based reasoning for the management of complex tasks.

REFERENCES

[1] S. Kumar, C. Savur, and F. Sahin, "Survey of human–robot collaboration in industrial settings: Awareness, intelligence, and compliance," *IEEE Trans. Syst. Man Cybern.: Syst.*, vol. 51, no. 1, pp. 280–297, 2021.

[2] F. A. Cheein, D. Herrera, J. Gimenez, R. Carelli, M. Torres-Torriti, J. R. Rosell-Polo, A. Escolà, and J. Arnó, "Human-robot interaction in precision agriculture: Sharing the workspace with service units," in *IEEE Int. Conf. Ind. Technol.*, 2015, pp. 289–295.

[3] M. Lippi and A. Marino, "Human multi-robot physical interaction: a distributed framework," *J. Intell. Robot. Syst.*, vol. 101, no. 2, p. 35, 2021.

[4] R. Maderna, M. Poggiali, A. M. Zanchettin, and P. Rocco, "An online scheduling algorithm for human-robot collaborative kitting," in *IEEE Int. Conf. Robot. Autom.*, 2020, pp. 11 430–11 435.

[5] S. Ramadurai and H. Jeong, "Effect of human involvement on work performance and fluency in human-robot collaboration for recycling," in *ACM/IEEE Int. Conf. Human-Robot Interaction*, 2022, pp. 1007–1011.

[6] K. Obata, T. Aoki, T. Horii, T. Taniguchi, and T. Nagai, "LiP-LLM: Integrating Linear Programming and dependency graph with Large Language Models for multi-robot task planning," *arXiv preprint arXiv:2410.21040*, 2024.

[7] S. Bezrucav, Y. Liu, and B. Corves, "Optimization-based or ai task planning for scenarios with cooperating mobile manipulators?" in *Proceedings of the 18th International Conference on Informatics in Control, Automation and Robotics - ICINCO*, INSTICC. SciTePress, 2021, pp. 115–122.

[8] S. S. Kannan, V. L. N. Venkatesh, and B.-C. Min, "SMART-LLM: Smart multi-agent robot task planning using large language models," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*. IEEE, 2024.

[9] P. Gupta, D. Isele, B. Dariush, E. Sachdeva, P.-H. Huang, S. Bae, and K. Lee, "Generalized Mission Planning for Heterogeneous Multi-Robot Teams via LLM-constructed Hierarchical Trees," *arXiv preprint arXiv:2501.16539*, 2025, v1.

[10] C. H. Song, J. Wu, C. Washington, B. M. Sadler, W.-L. Chao, and Y. Su, "Llm-planner: Few-shot grounded planning for embodied agents with large language models," 2023.

[11] I. Obi, V. L. Venkatesh, W. Wang, R. Wang, D. Suh, T. I. Amosa, W. Jo, and B.-C. Min, "SafePlan: Leveraging Formal Logic and Chain-of-Thought Reasoning for Enhanced Safety in LLM-based Robotic Task Planning," *arXiv preprint arXiv:2503.06892*, 2025.

[12] Z. Zhou, K. Yao, J. Song, Z. Shu, and L. Ma, "ISR-LLM: Iterative Self-Refined Large Language Model for Long-Horizon Sequential Task Planning," *arXiv preprint arXiv:2308.13724*, 2023, v1.

[13] J. Huang, R. Wang, J. Li, and M. Yang, "A survey on zero-shot planners with large language models," in *International Conference on Machine Learning*, 2022.

[14] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 824–24 837, 2022.

[15] P. Baptiste, C. Le Pape, and W. Nuijten, *Constraint-based scheduling: applying constraint programming to scheduling problems*. Springer Science & Business Media, 2001, vol. 39.

[16] Y. Bukchin and T. Raviv, "Constraint programming for solving various assembly line balancing problems," *Omega*, vol. 78, pp. 57–68, 2018.

[17] F. Rossi, P. van Beek, and T. Walsh, *Handbook of Constraint Programming*. USA: Elsevier Science Inc., 2006.