Knowledge and Information Systems (2022) 64:415–439
https://doi.org/10.1007/s10115-021-01638-5

REGULAR PAPER

An efficient lightweight coordination model to multi-agent planning

Leonardo Henrique Moreira[1] · Célia Ghedini Ralha[1]

Received: 26 July 2019 / Revised: 30 November 2021 / Accepted: 3 December 2021 /
Published online: 5 January 2022

Abstract
The main issue in multi-agent planning (MAP) is the agents' coordination process that is a computationally hard problem. Thus, many works focus on the planning strategy considering the computational process, agents' distribution roles, information privacy, and the resources coupling level. But domain-independent models that explore the balance between coordination process and privacy leading to efficient execution are missing. In this manuscript, we present a Lightweight Coordination Multi-agent Planning (LCMAP), a domain-independent model that balances the coordination process and privacy through three independent phases: (i) verification—each agent verifies its capabilities of reaching the goals; (ii) transformation—the coordinator selects agents through their capabilities and distributes the goals, transforming the original problem into single-agent problems; and (iii) validation—each plan is validated to check whether it can be parallel. LCMAP was compared to the state-of-the-art models to evaluate time efficiency and plan length during the problem-solving process using loosely and tightly coupled domains with specific evaluation metrics inherited from planning competitions. Furthermore, we conducted experiments to evaluate the execution efficiency regarding different configurations concerning planning time and plan length of the models, when LCMAP execution proves to be efficient.

Keywords Automated planning · Evaluation · Multi-agent planning · Multi-agent systems

1 Introduction

Automated planning and scheduling are related to the deliberation process associated with performing tasks. This process deals with defining actions able to change the environment to satisfy the desired configuration. Therefore, the planning process focuses on the choice and organization of actions through their expected effects [17]. The reasoning algorithms and models performed by a single agent define the automated planning area. This area has contributed to heuristics and strategies that improve the agent's deliberation process. Automated planning has been useful in many applications, from robot path planning to transportation

✉ Leonardo Henrique Moreira
leohmoreira@gmail.com

[1] Department of Computer Science, Institute of Exact Sciences, University of Brasília, Brasília, Brazil

logistics [5]. However, the rule of a single agent makes it difficult to solve real-world problems. For such problems, algorithms and models must tackle the fact that multiple actors interact by cooperating and competing in a common environment.

In this context, multi-agent planning (MAP) has gained increasing importance by integrating multi-agent systems with practical reasoning agents focusing means-end decisions with planning capabilities [53,54]. MAP research encompasses factors such as individual or global objectives, coordination and computational processes, communication and interaction among agents, privacy, scalability, plan synthesis and heuristic search [40,41,44–48,52,55]. Involving so many factors, MAP computational complexity is NP-hard, but in some cases, it can scale polynomially considering the number of agents [5].

Usually, MAP studies gather over one factor. In particular, the coordination process and privacy factors play a highlighted role. The coordination depends on the solving strategy applied to the searching process. The challenge faced by the coordination process derives from the complexity that may grow with the number of agents. The distributed approach uses more than one planner, where each agent tries to find its solution, but the centralized one searches for solutions for all agents in a unique episode performed by a single planner agent. Thus, the distributed process may be harder to coordinate, but it can tackle large search spaces compared to a single planner.

Besides, privacy factor defines how the approach treats and preserves information. Agents can exchange knowledge about their actions, goals, and states. If this exchange is undergone without restrictions, information is public and there is no privacy at all. Otherwise, when agents are selective in spreading their knowledge and information, privacy may vary from weak to strong levels.

Considering MAP aspects, we conducted a systematic mapping study to highlight recent works to identify related open problems. This study was enriched by a survey that focused on the most relevant approaches to MAP that took place in the 2015 Competition of Distributed and Multi-Agent Planning [44] (see Sect. 3). An open problem identified was the trade-off between the coordination process and privacy in MAP models. While a centralized approach reduces interaction levels easing coordination among agents, privacy pays the cost by turning private information to public. In summary, to maintain the privacy of the agents, a more sophisticated communication and coordination process is necessary.

Many MAP approaches explore plan refinement process by adding actions to partial ordered plans (based plans) until a solution is found. However, this strategy does not scale due to the message bottleneck. Our approach uses goal delegation transforming MAP to single-agent planning providing conditions to individual and parallel plan searches. Thus, the agents' interactions occur in well-defined moments, and this technique was more efficient than the strong interaction employed by approaches that explore the plan refinement process.

Therefore, this research has focused on the trade-off between the coordination process and privacy of the agents to provide an efficient MAP model. As a result, we propose the Lightweight Coordination Multi-Agent planning (LCMAP) that explores the domain problem by testing the agents' problem-solving capabilities individually, partially or completely during the verification step. Afterward, the goals are assigned to the most capable agents in the transformation phase. Finally, the individual plans are verified for the parallel execution viability, otherwise they are centralized. LCMAP is considered lightweight because the coordination is done at specific moments—before goal assignment and during plan validation—avoiding coordination through message exchange.

LCMAP was compared to different models considering the hypothesis that it is an efficient model when compared to different state-of-the-art MAP models. The experiments conducted to test this hypothesis focus on state-of-the-art MAP models. First, the experimentation used

International Planning Competition (IPC) domains—satellite, rovers, zeno-travel, elevator, depots and logistics—to collect time and plan-length metrics. Afterward, we checked the impact of those metrics over the models regarding different configurations of a weighted average performance metric.

The conducted experiments provide evidence that LCMAP is more efficient than state-of-the-art approaches. Thus, the main contribution of this paper is the efficient lightweight coordination model that balances computational process and privacy. Moreover, the main difference of this work comparing to other MAP approaches is the verification phase carried out in previous step. Once related tools and algorithms start immediately the planning process, LCMAP saves time and resources by predicting an impossible solution by searching for lack of agent capabilities. Also, in the transformation phase, the goal delegation process evaluates agent capabilities while other approaches do not explore this feature, assigning objectives blindly to agents.

The rest of the paper is structured as follows. In Sect. 2, we present important MAP definitions. In Sect. 3, we highlight related work. In Sect. 4, we detail the proposed model. In Sect. 5, we describe the experimental study comparing LCMAP to the state-of-the-art models. Finally, we draw conclusions and suggestions for future work in Sect. 6.

2 Background

A multi-agent system (MAS) is formed by a set of software entities able of sensing (sensors) and changing the environment state using their actuators [53,54]. In a MAS, there are autonomous agents that interact through a defined communication protocol allowing competitive or cooperative behavior with a coordination model. Competitive agents require a negotiation protocol, while cooperative agents need a planning protocol to define individual and group goals. Planning strategies of solving planning problems are implemented centralized or distributed [53]. Hastily, the planning process in the presence of multiple agents is a MAP task [22]. One interesting point is that agents in a MAP model can play planning and/or executor roles.

In this research, we assumed restrictive premises to deal with MAP similarly to other work [5,41,45,47]. A state of the world is formed by a set of logical propositions. The environment is fully observable and agents can access information immediately. Agents are collaborative, so they are not competitive (self-interested) and there is no private goal. Actions are deterministic (well-defined), they always achieve their effects, which are constant, unit cost, instantaneous, and the only reason for world update. No events take place in the environment apart from the actions expected effects of the agents. Finally, communication process is free of failures.

In the sequence, we present the most important concepts regarding MAP study. The definitions of operator, action single-agent and MAP task were based on [17,36]. Most part of MAP setting relies on action specification. But to define actions, operators must be previously defined.

Definition 2.1 An operator is a schema that defines actions using variables, (parameters). An operator is a tuple $\theta = < name(\theta), pre(\theta), eff(\theta) >$, where

- $name(\theta)$ is an identification to the operator;
- $pre(\theta)$ is formed by a set of precondition that stands for literals required to apply the operator;
- $eff(\theta)$ is formed by a set of effects that stands for literals which are added (eff^+) or deleted (eff^-)from the state of the world after executing the operator.

Definition 2.2 An action is an instantiated operator. In other words, an action is an operator where parameters were changed by objects.

When there is only one agent, the propositions, actions and goals sets belong to a single source. Every environment change results from this agents actions. Hence, it is impossible to delegate responsibilities. For the same reason, there is no need for a coordination protocol to manage agents' cooperation or competition. A single-agent setting defines the planning tasks.

Definition 2.3 A single-agent planning (SAP) task can be formalized by a tuple $\Pi =< F, A, I, G >$, where

- F is formed by a set of propositions;
- A is formed by a set of grounded actions, derived from operators;
- I is an initial state, $I \subseteq F$;
- G is formed by a set of goals, $G \subseteq F$.

In the presence of more agents, a MAP model must coordinate interdependences among agents and their tasks. For each agent ag_i, there is a SAP task $\Pi_{ag_i} =< F_{ag_i}, A_{ag_i}, I_{ag_i}, G_{ag_i} >$.

A mutex is an inconsistence among propositions. We assume that in the MAP setting, there are no mutexes in the initial state and the set of goals among agents. Therefore, $I = \bigcup_{i=1}^{m} I_{ag_i}$ and $G = \bigcup_{i=1}^{m} G_{ag_i}$.

A MAP task describes the combination of all tasks $\{\Pi_{ag_1}, \Pi_{ag_2}, \ldots, \Pi_{ag_m}\}$.

Definition 2.4 A multi-agent planning task can be formalized by a tuple $\tau =< Ag, \bar{F}, \bar{A}, I, G >$, where

- Ag = $\{ag_1, ag_2, \ldots, ag_m\}$ is formed by a finite set of agents;
- $\bar{F} = \bigcup_{i=1}^{m} F_{ag_i}$ is formed by a set of propositions;
- $\bar{A}$ is formed by a set of grounded actions,$\bar{A} = \bigcup_{i=1}^{m} A_{ag_i}$;
- I is an initial state, $I \subseteq \bar{F}$;
- G is formed by a set of goals, $G \subseteq \bar{F}$.

The transition caused by an action α applied in a state s is defined by Eq. 1.

$$\gamma(s, \alpha) = (s \backslash eff^{-}(\alpha)) \cup eff^{+}(\alpha) \tag{1}$$

The result of a planning task, SAP or MAP, is a plan $\pi = [a_1, a_2, \ldots, a_n]$, namely, a finite sequence of actions. When a plan π is applied sequentially from the initial state I, it would generate in a state s_n, where $G \subseteq s_n$. The evolution from the initial state to s_n is present by Eq. 2.

$$\begin{aligned}\gamma(I, \pi) &= \gamma(I, [a_1, a_2, \ldots, a_n]) \\ &= \gamma(\gamma(I, a_1), [a_2, \ldots, a_n]) \\ &= \gamma(\gamma(\gamma(I, a_1), a_2), [a_3, \ldots, a_n]) \\ &\ \vdots \\ &= \gamma(s_{n-1}, a_n) = s_n \end{aligned} \tag{2}$$

When two or more agents execute their plans simultaneously, they can compete for some resources or even undo the effects of each other's actions. However, these plans can provide cooperation with a minimum level of coordination when plans are independent.

Definition 2.5 Two plans π_1 and π_2 are independent iff:

$$({}^*\delta_{\pi_1} \cup \delta^*_{\pi_1}) \cap ({}^*\delta_{\pi_2} \cup \delta^*_{\pi_2}) = \emptyset, \text{ where:}$$

$$ {}^*\delta_\pi = \bigcup_{i=1}^{n} pre(\alpha_i) | \alpha_i \in \pi = [a_1, \ldots, a_n],$$

$$\delta^*_\pi = \bigcup_{i=1}^{n} eff(\alpha_i) | \alpha_i \in \pi = [a_1, \ldots, a_n].$$

The complete listing of operators, actions, initial state and goals regarding the domains is available in the project repository.[1]

3 Literature review

In order to identify state-of-the-art MAP research works, we conducted a systematic mapping study according to guidelines provided by Kitchenham et al. [23], Cooper [11]. The study included a protocol defining four research questions to accept or refuse the efficiency hypotheses of our lightweight coordination model focusing on coordination process and privacy aspects. The key points of the protocol include the definition of

– Problem: What are the effects of balancing the coordination process and privacy in a MAP model?
– Intervention: What are the approaches that explore agent selection, validation, and verification in MAP?
– Comparison: How efficient is planning time compared to other MAP models?
– Outcome: What are the effects of executing a pre-planning[2] phase in a MAP?

The systematic mapping was done from February to June 2017 and resulted in 292 works. The study focused on journal articles and proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)[3] from 2011 to 2017. Moreover, after reviewing some works, we investigated their references in order to include related and helpful work. The studied journal articles and conference proceedings were available in premier computer area digital libraries such as

– The IEEE Xplore digital library provides proceedings of important conferences such as the IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT); the Symposium on Computational Intelligence in Control and Automation (CICA); and the Brazilian Conference on Intelligent Systems (BRACIS) - the leading AI research conference in Brazil;
– In the ACM Digital Library, there are proceedings from important conferences, such as (i) International Conference on Autonomous Agents and Multi-agent Systems (AAMAS) the world's leading scientific conference for research in autonomous agents and multi-agent systems; and (ii) European Conference on Artificial Intelligence (ECAI), the Europe premier AI research venue;

[1] https://gitlab.com/InfoKnow/AutomatedPlanning/LeonardoMoreira-LCMAP.

[2] Pre-planning tries to guarantee that the agents' plans can be combined into a solution that satisfies the goals of the MAP task [44].

[3] http://www.icaps-conference.org/.

Table 1 Questions of quality criterion

Item	Question
Q1	Does the study discuss the soundness and/or completeness of the proposed algorithm/model?
Q2	Does the proposed model discuss scalability aspects?
Q3	Does the study present evaluated results and comparison with other studies?
Q4	Do the tests use the IPC domains?
Q5	Does the model use a standard planning language (PDDL, ADL, STRIPS)?
Q6	Is there any executable implementation available to reproduce the experiments?
Q7	Is there any agent selection prior to planning?
Q8	Is there any problem checking prior to planning?

- The Springer-Link digital library includes important journals in the domain of intelligent planning such as (i) Applied Intelligence; (ii) Journal of Research on Intelligent Systems for Real Life Complex Problems; and (iii) Knowledge and Information Systems (KAIS).

Regarding the ICAPS proceedings, it is considered the premier forum for exchanging news and research results on theory and applications of intelligent planning and scheduling technology, enabling smart decision-making in autonomous systems. As a part of the workshop on Distributed and Multi-agent Planning (DMAP) at the ICAPS 2015 Conference, the Competition of Distributed and Multi-Agent Planners (CoDMAP) was organized with the aim to consolidate the planners in terms of input format; to promote development of multi-agent planners both inside and outside of the multi-agent research community; and to provide a proof-of-concept of a potential future multi-agent planning track of the International Planning Competition (IPC).

3.1 Research criteria

The research work inclusion was based mainly in two criteria: scope of application and publication vehicle quality (impact factor). The work should present MAP models or algorithms independent of domain. As a result, 35 research works were chosen.

In relation to the quality criterion, a set of eight binary questions (yes/no) were defined (Table 1). Works were classified through the sum of affirmative answers. Table 2 presents the 35 included works and highlights six works with the sum of affirmative answers greater than or equal to three. This value was stipulated from the average of the works that presented at least one positive answer. In Table 3, we present the works used in the comparative study to validate our hypotheses.

3.2 Comparative study works

FMAP is presented in [47], a domain-independent solver that integrates planning and coordination in a distributed environment preserving information privacy. Agents interact toward a solution by refining partial plans defined by a round-robin leader. FMAP uses an interaction protocol with lot of message exchange during the plan refinement process. Thus, the number of agents increase affects the model performance and scalability.

MADLA is presented in [41], a domain-independent MAP centralized solver but unlike others, it combines two types of heuristics. The agents' information is processed isolatedly in order to compute the heuristic value which is shared to define a final value. Authors claim that such heuristic combination increases the possibility of solving problems, but no experimental validation was presented.

Chouhan and Niyogi [9] suggest a domain-independent approach that consider the capability of agents to solve MAP problems with cooperative goals involving joint actions. In the proposal, coined as *MAPJA*, the original problem is checked whether it can be solved by any available multi-agent planner or it must be processed in a centralized way.

In [3], the strategy of transforming the original problem into smaller instances (SAP tasks—Definition 2.3) is employed. In the *MAPR* model, agents interact through a ring structure in which the first one becomes responsible for the goals it can satisfy removing them from the set of targets. Then, the set of targets is sent to the next agent until the set is empty. MAPR is a centralized coordination model.

Agents' interaction in [55] is discussed and a formal characterization of situations where cooperation is required to achieve goals in a multi-agent environment is presented. The key contribution is the discussion of the question: Under what conditions are multiple agents need to solve a planning problem? A centralized planner, Multi-agent Planner for Required Cooperation (MARC), is proposed which aims to use the minimum amount of agents to satisfy the goals.

In [28], the resource saving aspect is discussed when goals are distributed to agents in MAP. Although experiments were also limited to scenarios without cooperation requirements, the results highlighted that a delegation strategy is able to improve planning time about 96%.

3.3 MAP solvers taxonomy

Concerning the presented works, it is possible to create a taxonomy that classifies the MAP solvers according to the following features (Table 3):

- Evaluation (F1): the evaluation of agents' capabilities of reaching goals is one important property, but rarely presented in MAP models. Therefore, this evaluation may enhance the planning process since it avoids searching for impossible solutions;
- Cooperation analysis (F2): through cooperation analysis, the agents' interaction is checked in order to define the coupling level among agents;
- Agent selection (F3): by selecting only the agents that may contribute to problem-solving, MAP models may improve their performance and also adopt an efficient resource usage;
- Transformation (F4): the original problem is transformed into many instances of simpler cases by agent delegation and goal assignment;
- Coordination process (F5): planning design that can be centralized (C) or distributed (D);
- Privacy (F6): agents' information privacy, that can be flexible (F), ignored (I) or preserved (P).

In addition to our systematic study, a MAP survey considering the 2015 Competition of Distributed and Multi-agent Planning approaches was presented by Torreño et al. [44]. This survey highlights the state-of-the-art solvers. Concerning coordination process, seven works from 23 adopted distributed approaches: FMAP [47], MH-FMAP [48], MAP-POP [45], MAPlan [16], PSM [50], MAFS [32] and MAD-A* [32]. Most of the work did not address privacy issues and only two addressed in a significant way [6,27].

Although our systematic mapping study and the survey had different purposes and specific periods, they reached similar results. Our study considered explicitly nine of the 23 solvers highlighted in the survey: MARC [55], FMAP [47], MADLA [41], MH-FMAP [48], MAP-POP [45], MAPR [3], PSM [50], MAFS [32] and MAD-A*[32]. However, if the evaluations

Table 2 Reviewed works

Item	Work	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Total
1	[2]	N	N	N	N	N	N	N	N	0
2	[3]	N	N	Y	Y	Y	Y	N	N	4
3	[7]	N	N	N	N	N	N	N	N	0
4	[8]	N	N	N	Y	Y	N	N	N	2
5	[9]	Y	N	Y	N	Y	N	Y	N	4
6	[10]	N	N	N	N	Y	N	N	N	1
7	[13]	N	N	N	Y	N	N	N	N	1
8	[14]	N	N	N	Y	N	N	N	N	1
9	[15]	N	N	N	N	Y	N	N	N	1
10	[18]	N	Y	N	N	N	N	N	N	1
11	[19]	N	N	N	N	N	N	N	N	0
12	[20]	N	N	N	N	N	N	N	N	0
13	[24]	N	N	N	N	N	N	N	N	0
14	[26]	Y	N	N	Y	N	N	N	N	2
15	[28]	N	Y	Y	N	Y	N	N	N	3
16	[30]	N	N	N	N	N	N	N	N	0
17	[31]	N	N	N	N	N	N	N	Y	1
18	[32]	N	N	N	N	N	N	N	N	0
19	[33]	N	N	N	N	N	N	N	N	0
20	[34]	N	N	N	N	N	N	N	N	0
21	[35]	N	N	N	N	N	N	N	N	0
22	[37]	N	N	N	N	N	N	N	N	0
23	[38]	N	N	N	N	N	N	N	N	0
24	[39]	N	N	N	N	N	N	N	N	0
25	[40]	N	N	N	Y	Y	N	N	N	2
26	[41]	N	N	Y	Y	Y	Y	N	N	4
27	[42]	N	N	N	N	N	N	N	N	0
28	[43]	N	N	N	N	N	N	N	N	0
29	[46]	N	Y	N	N	N	N	N	N	1
30	[47]	Y	Y	Y	Y	N	Y	N	N	5
31	[48]	N	Y	Y	N	N	N	N	N	2
32	[49]	N	N	N	Y	Y	N	N	N	2
33	[51]	N	N	N	N	N	N	N	N	0
34	[52]	N	Y	N	N	Y	N	N	N	2
35	[55]	Y	N	Y	N	Y	Y	Y	Y	6

of MAPR [3] were also considered CMAP [4] and PMR [25], the intersection between both reviews would be 11 works. This survey completes our systematic mapping study.

Regarding an initial version published by Moreira and Ralha [29], LCMAP was designed to solve only loosely coupled domains. In this work, LCMAP is updated to tackle privacy on tightly coupled domains as detailed in Sect. 4.

Table 3 Related work features

Work	Features					
	F1	F2	F3	F4	F5	F6
FMAP [47]	–	–	–	–	D	P
MADLA [41]	–	–	–	✓	C	P
MAPJA [9]	–	–	✓	✓	C	I
MAPR [3]	–	–	–	✓	D	P
MARC [55]	–	✓	✓	–	C	I
[28]	–	–	✓	✓	C	I

4 LCMAP overview

LCMAP focuses on the planning strategy mainly considering efficient coordination process. The coordination relies on assigning goals concerning the capabilities of the agents, keeping information privacy considering the resources coupling level. In Fig. 1, the lightweight MAP approach is illustrated through an efficient coordination process. The arrows in Fig. 1 define important steps of the workflow regarding problem definition, as follows:

- C: the agent sends its capabilities to the coordinator;
- G: the coordinator delegates goals to an agent regarding its capabilities;
- π: agent sends its individual plan to the coordinator;
- CP: the coordinator performs a centralized planning whenever individual plans cannot be carried out in parallel or goals cannot be delegated.

The efficiency of LCMAP is the result of using the characteristics of the planning problem to leverage the search for a solution. Whenever the agents can work in isolation regarding their capabilities (C), their commitments (G) are organized at specific moments, before assignment of goals (pre-planning) and during the validation of the plans (π), avoiding the need of message exchange to provide coordination. Otherwise, the coordinator assumes a centralized planning (CP) function and sends agents their plans rather than goals. Therefore, efficiency is supported by pre-planning and validation coordination steps. Independent of the path followed in the workflow, there are no exchange of messages among agents. Therefore, the coordination process and privacy coexist in the LCMAP model.

LCMAP explores the transformation of a MAP problem into simpler instances (SAP). Although other works explore this strategy [3,12,28], the verification of agents' capabilities and the goals each agent can satisfy is a novel approach. In order to identify those capabilities and goals, LCMAP exploits a delete-relaxation algorithm to check the reachablity of agents' goals as detailed in Sect. 4.1. If every single goal, $g \in G$ is satisfied at least by one agent, goals can be indiscriminately distributed among agents. But at the distribution moment, it is not possible to guarantee that the individual calculated plans are independent, so they are checked later during the validation phase. When the distribution of goals is possible, agents carry out their plans independently, being accompanied by the coordinator that assigns goals and validate plans. But when the delegation of goals is impossible or plan validation fails, the coordinator carries out a centralized planning process to find a new valid plan.

Regarding the privacy issues, LCMAP holds a flexible approach. When the transformation of the problem is possible in a SAP, agents, but the coordinator, do not need to know each other and the information privacy is preserved. Otherwise, when centralized planning is done, the coordinator is the only agent that knows other agents' information. Therefore, privacy management is considered flexible because:

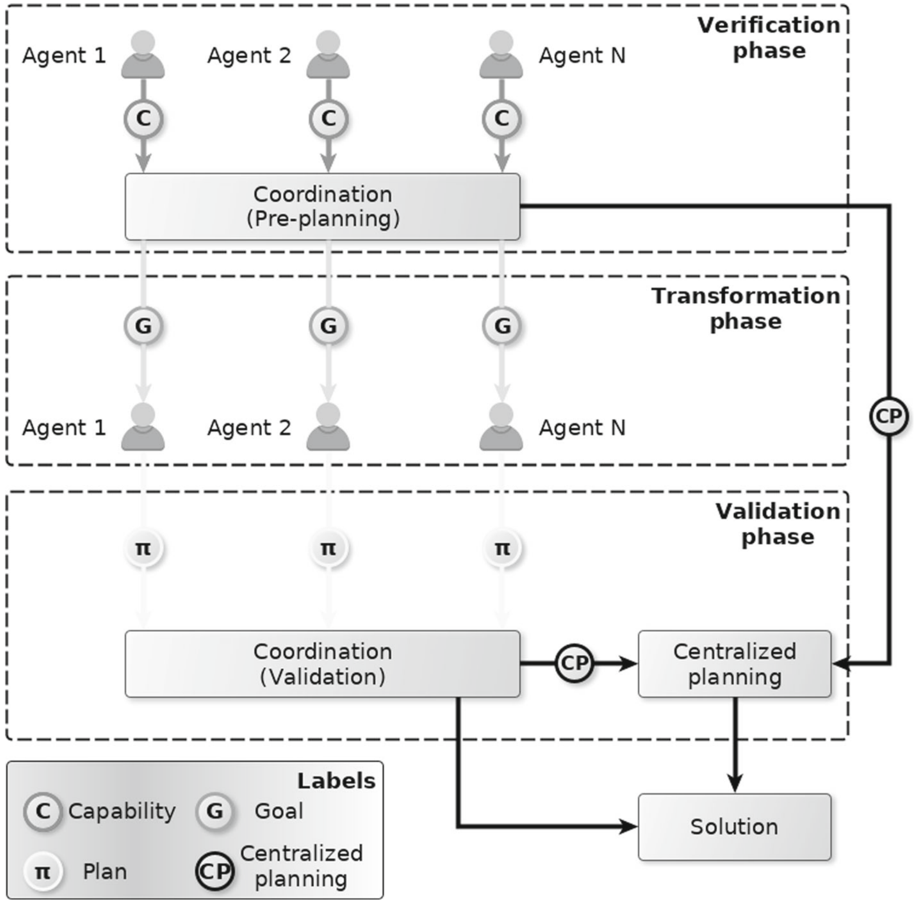


Fig. 1 LCMAP workflow

- there is no information exchange among agents when the goals can be distributed;
- agent's capabilities are known only by the coordinator and its owner when a centralized planning is carried out.

Concerning the computational process that relates the agent's roles and capabilities, the information privacy and the problem coupling level, LCMAP explores these factors in three different phases: verification, transformation and validation. The LCMAP phases and components are presented in Fig. 2 and the details of each component are described in the sequence. Whenever possible, the planning problem is transformed into smaller instances easier to solve. While the components within the agent blocks of Fig. 2 are responsible for verification and planning tasks, those inside the coordinator block guarantee the transformation and validation of the path toward a solution.

Therefore, the design of the LCMAP model explores the possibility of easing the search for a plan by verifying the agents' capabilities preliminary, by transforming the original problem and checking whether the decisions were correct. In summary, LCMAP balances the tasks' coordination and planning process along its three phases.

4.1 Verification

In this phase, each agent verifies its capabilities of reaching the goals by itself. The agents are defined in an input file of the user interface module. The input handler component identifies each agent to instantiate them in parallel processes by the agent instantiator. During the instantiation phase, every agent uses its parser to process the domain and problem files in order to gather information about initial state, goals and available actions. Then, the verification phase, which was started by the user interface module, is carried out by the agents' modules to identify their capabilities. Agents also have a planner component which provides planning autonomy.

Parser and verifier

Using the Planning Domain Definition Language (PDDL) [1], agents are described by domain and problem files that are computed by the parser component. These files contain the propositions used to generate the complete set of possibilities that define any environment state.

The set of all propositions is generated by a Cartesian product over each domain's variables. This information is computed in parser, and it is used by the verifier and planner components.

In the verifier component, each agent uses a delete-relaxation heuristic (DRH) to test its capabilities [17]. Briefly, this technique explores the actions of a single agent, adding new effects and never removing old ones. A relaxed state $\hat{s}$ is a state in which new and old effects coexist. This heuristic family returns a cost value of reaching a state from another. LCMAP model is worried about the possibility of reaching the goals, therefore the heuristic was adapted to return only True when the target is reached, or False, otherwise.

An action a is applicable in a relaxed state $\hat{s}$ if $\hat{s}$ satisfies $pre(a)$. The transition from a state $\hat{s}$ after applying an action a is defined by Eq. 3.

$$\gamma^{+}(\hat{s}, a) = \hat{s} \cup \gamma(s, a) \tag{3}$$

Algorithm 1 presents the delete-relaxation approach used. The set of actions A_{ag_i} is an input parameter. The applicable actions are identified (Line 5) and the $\hat{s_k}$ results from unifying the effects of all operators $a \in A_k$ (Lines 6 and 7). The algorithm is finished when $\hat{s_k} = \hat{s}_{k-1}$ (Line 9), which means that no new effect was added. Otherwise, the set goals are satisfied (Line 4) by $\hat{s_k}$. In summary, the purpose of Algorithm 1 is to simulate all possible executions by adding the effects of the available actions until no further update is caused to the environment state. In the end, the output will be the assessment of the possibility of reaching the goals.

This heuristic is used when each agent needs to identify which goals it may satisfy by itself. Following Algorithm 1 design, the set of goals is first evaluated in a global way, to check if there is a state in which every $g \in G$ holds. Otherwise, the agent checks each goal g individually.

Another important function of the verifier is the definition of rigid literals, propositions that remain constant from its initial state assignment. Thus, the preconditions $pre(\alpha)$ and effects $eff(\alpha)$ of each action are processed to identify elements that are not modified. This information is important in the plan validation phase.

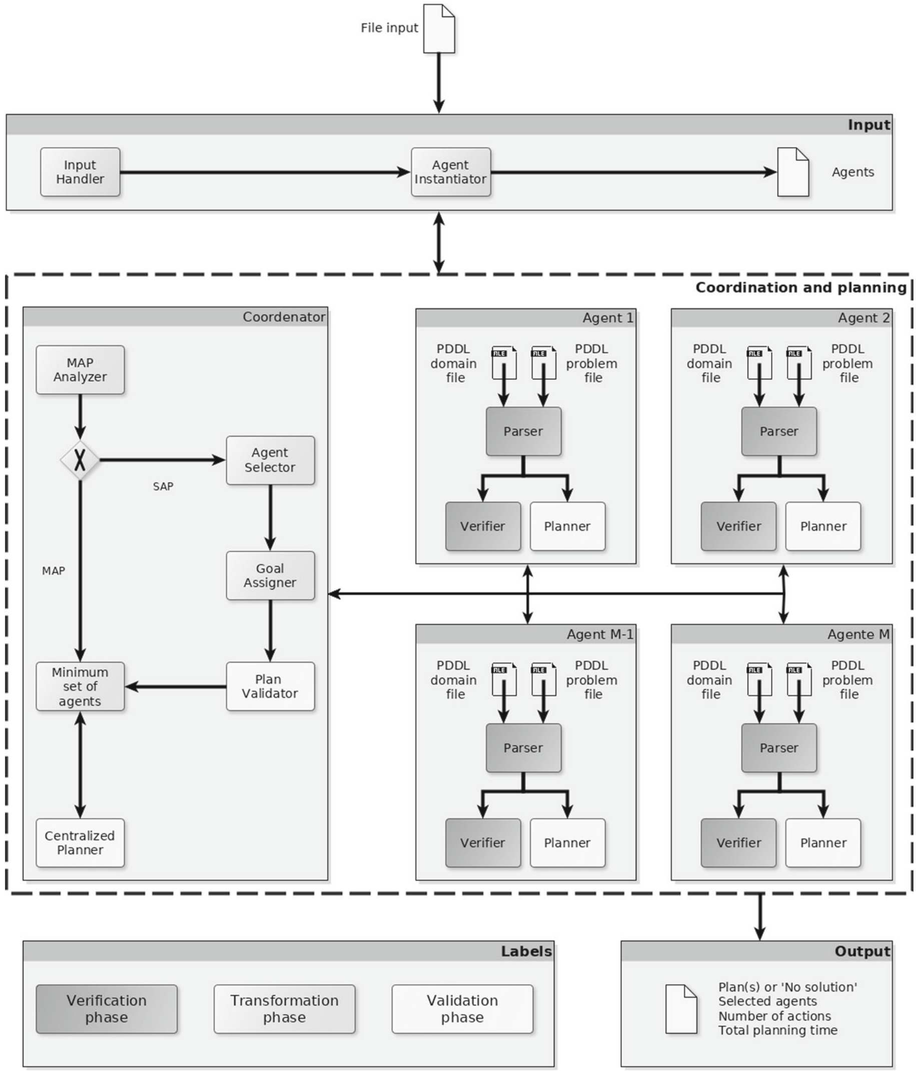


Fig. 2 LCMAP phases and components

4.2 Transformation

In the transformation phase, the coordinator selects agents regarding their capabilities and distributes the goals transforming the original problem into single-agent problems. In this sense, the original problem is transformed by assigning goals after each agent checks the goals it may reach without cooperation or interaction.

The coordinator queries all agents to unify information in the tuple τ (Definition 2.4) and with it, the selection of agents is performed. Following, the coordinator starts the assignment process using a load-balance or saving-resource strategy, based on standard round-robin and greedy techniques, respectively. The strategy is defined as an input parameter. For instance,

Algorithm 1 Delete-relaxation

```
Require: $A_{ag_i}$, $I_{ag_i}$, $G$
Ensure: $True$ or $False$
$\hat{s}_0 \leftarrow I_{ag_i}$
$A_0 \leftarrow \emptyset$
$k \leftarrow 1$
while $\nexists s \subseteq s_k$ that satisfies $G$ do
    $A_k \leftarrow$ {all applicable actions in $\hat{s}_{k-1}$}
    for $a \in A_k$ do
        $\hat{s_k} \leftarrow \hat{s_k} \cup \{\gamma^+(\hat{s}_{k-1}, a)\}$
    end for
    if $\hat{s} = \hat{s}_{k-1}$ then
        return $False$
    end if
    $k \leftarrow k + 1$
end while
return $True$
```

under a load-balance strategy, LCMAP will assign each agent only one goal per round according their capabilities. When LCMAP adopts a saving-resource policy, it employs the lowest number of agents by assigning goals to an already selected agent as much as possible. Whatever the strategy is, there is no plan length estimation during the goal assignment phase. Furthermore, this allocation is based purely on the number of goals and capabilities of the agents. Any assessment of the size of the plans is possible only at the end of the transformation phase, when the agents plan considering the assigned goals.

MAP analyzer

After unifying all information in tuple τ, the coordinator queries agents about their reachable goal (RGoals) and rigid literals. Thus, it analyzes the MAP problem to check whether it can be transformed into simpler instances (SAP), regarding agents' capabilities. Therefore, the union of the RGoals sets is compared to the original set G.

This union and G can only be different if there is a goal that is not reached by any agent. Therefore, the cardinality ($\sharp$) of these sets is enough to define if the transformation is possible. If Eq. 4 is satisfied, the coordinator starts the agent selection phase; otherwise, it assumes a centralized planning.

$$\sharp \bigcup_{i=1}^{m} RGoals_{ag_i} = \sharp G \tag{4}$$

Agent selector and goal assigner

This component aims to identify agents capable of individual planning and execution. In this sense, the coordinator analyzes the $RGoals$ sets of all agents. The collection of agents chosen is defined as $\{ag \in Ag | \sharp RGoals_{ag} > 0\}$.

After the verification phase, agents discover which goals they may satisfy returning no estimate of cost. Although disregarding cost estimate avoids sorting agents regarding their efforts, it does not impair the transformation of the problem since all agents chosen can reach some goal. Once the selector chooses agents, the coordinator assigns goals to avoid agents working toward the same goal simultaneously.

LCMAP can carry out this assignment following two strategies defined as an input parameter. In the first option, the coordinator allocates the smallest number of agents to be used in the planning phase. Therefore, even if all goals are reachable by all agents, only one of them is allocated. In this sense, the saving-resource strategy is based on a greedy algorithm. The agent with the largest number of RGoals items will be the first to be assigned goals. In the load-balance strategy, LCMAP organizes agents in a ring topology, thus selecting agents in a round-robin way and allocating one goal per iteration.

After the assignment, each chosen agent is committed to fulfill a part of the planning problem. This commitment is formed by a set of goals, presented in Eq. 5.

$$G_{ag_i} = \{g | g \subseteq RGoals_{ag_i} \subseteq G\}. \tag{5}$$

Minimum set of agents

Once individual plans were considered invalid or the transformation MAP to SAP was not possible (Eq. 4 was not held), the coordinator starts defining the minimum set of agents to the planning phase. Agents are organized in teams, initially in pairs, and the goals' reachability is evaluated using the modified delete-relaxation heuristic (see Algorithm 2). Unlike the verification phase, here the coordinator uses all operators from all team members. Whenever the current evaluation returns *False*, the team is updated by changing its members or increasing its size.

Algorithm 2 highlights the team definition. First, two agents are randomly chosen (Line 1), and this team is evaluated by the delete-relaxation heuristic function (Line 3) which runs slightly different from Algorithm 1 since it considers the set of actions from all team members (Line 15–16). In each iteration, the main loop (Lines 2–8) checks each team and whenever a failure (*False*) is returned by DRH function, this team is updated (Line 6). The algorithm is finished when a team shows enough condition to satisfy the goals (Line 3) or after all agent combinations are evaluated as failure. In summary, the purpose of Algorithm 2 is to define the smallest team of agents to meet the goals. In this sense, the team of agents is increased step by step and continuously evaluated to determine the possibility of finding a solution. Similarly to Algorithm 1, Algorithm 2 highlights a possibility for a solution rather than a plan.

4.3 Validation

Once every goal has been assigned to a specific agent, each agent starts the planning process individually. After finding their plans (π_{ag_i}), agents send them to the coordinator that analyzes their interaction. Therefore, the validation phase checks whether the plans, individually defined, can be carried out centralized or distributed in parallel (see Definition 2.5). Individual plans are validated whenever Eq. 6 holds. In other words, if every two different assigned agents have independent plans, their plans will be valid.

$$\begin{aligned} \forall ag_i, ag_j \in Ag(((^*\delta_{\pi_{ag_i}} \cup \delta^*_{\pi_{ag_i}}) \cap (^*\delta_{\pi_{ag_j}} \cup \delta^*_{\pi_{ag_j}}) = \emptyset) \\ \wedge (ag_i \neq ag_j) \wedge (\#G_{ag_i} > 0) \wedge (\#G_{ag_j} > 0)) \end{aligned} \tag{6}$$

Algorithm 2 Minimum set of agents

```
Require: $I, G$
Ensure: $True$ or $False$
$team \leftarrow$ two randon agents
while $size(team) \neq \sharp Ag$ do
    if $DRH(I, G, team) = True$ then
        return $team$
    else
        $update(team)$
    end if
end while
return $False$
function DRH($I, G, team$)
    $\hat{s}_0 \leftarrow I_{agId}$
    $A_0 \leftarrow \emptyset$
    $k \leftarrow 1$
    while $\nexists s \subseteq s_k$ that satisfies $G$ do
        for $ag \in team$ do
            $A_k \leftarrow$ {all actions $\in A_{ag}$ applicable in $\hat{s}_{k-1}$}
            for $a \in A_k$ do
                $\hat{s_k} \leftarrow \hat{s_k} \cup \{\gamma^+(\hat{s}_{k-1}, a)\}$
            end for
        end for
        if $\hat{s} = \hat{s}_{k-1}$ then
            return $False$
        end if
        $k \leftarrow k + 1$
    end while
    return $True$
end function
```

Planner and plan validator

The LCMAP model is designed to be modular, and its components can be replaced without impairing its usage. Therefore, each agent can use a different planning tool. After the goal assignment phase, each agent starts its planning using the defined planner tool as an external procedure call. So, it must be able to retrieve and compute the output to identify the plan or a failure.

Regarding the coordinator agent, this planner component has an extra task. It must gather all information from the defined team in a pair of PDDL files, a domain file in which actions, predicates, and types are unified. In the problem file, the initial state and objects are listed.

Finally, the plan found by each planner must be mapped into a sequence of operators, with the sets of preconditions and effects. This translation is important because the plan validator requires a well-defined data structure that is expected and used during all phases of the model.

Once every agent allocated in the goal assignment phase has received its goals and found its plan, the coordinator tests whether all agents' solutions can be performed parallel. Thus, plans are analyzed two by two to check whether they are independent (Definition 2.5). Each plan is visited and from every action, the preconditions and effects are visited in order to compute $^*\delta_\pi$ and δ_π^*. Whenever the plans are not considered independent, the coordinator executes a centralized planning. The whole process finishes when the problem has no solution or when the final plan is present, as the result of the union of individual plans or as a centralized planning response.

5 Experiments and results

In this section, we first present the experimental settings including setup (Sect. 5.1), domains (Sect. 5.2) and computational resources (Sect. 5.3). In the sequence, we describe the sets of experiments and the comparison with other state-of-the-art MAP models (Sect. 5.4). All the results are available in the project repository.[4]

5.1 Setup description

The hypothesis is whether LCMAP is an efficient model when compared to state-of-the-art MAP models. We use two metrics to test the performance of each model for a given MAP task τ attributing zero score when the planner did not solve the planning task.

The *time2*[5] metric computes the score of a planner for a given task as the quotient between the minimum time, T^*, required by any planner to solve the same task and the time, T, a particular planner took to solve the same task, as presented in Eq. 7.

$$time2 = \frac{log(1+T^*)}{log(1+T)}, \text{ such that } time2 \in [0,1] \tag{7}$$

The *quality* metric computes the quality of the best plan found, Q^*, for the task with the quality of the plan, Q, produced by a particular planner (Eq. 8). We use the *quality* metric to report the plan length score. For instance, let π_A, π_B, π_C be plans computed by three different planners (A, B and C) for the same planning problem. Now, consider the number of actions in each plan as $length(\pi_A) = 10, length(\pi_B) = 20, length(\pi_C) = 100$. Clearly, the best planner is A because of its smallest length, hence $Q^* = 10$. The quality of π_A is $\frac{10}{10} = 1$. Regarding π_B and π_C, the qualities are $\frac{10}{20} = 0.5$ and $\frac{10}{100} = 0.1$, respectively. Therefore, the greater the quality is, the better the plan is. The same reasoning can be applied to *time2*.

$$quality = \frac{Q^*}{Q}, \text{ such that } quality \in [0,1] \tag{8}$$

Experiments were defined as configuration tests regarding domains, number of agents, number of goals and the MAP model used. Each configuration was repeated three times, and the average value was used.

5.2 Domains

The IPC[6] domains are used to compare to the state-of-the-art MAP models. Regarding the most used domains described in the related work, the domains chosen from IPC are

- Satellite (loosely coupled)—each agent symbolizes a satellite that is defined by its attributes regarding position, orientation (direction) and available instruments. Although they have different qualities, this condition favors the transformation of the original problem because agents do not need cooperation to carry out their actions. The problem is to scale satellite observations that include collecting and storing data using different instruments to observe a collection of targets;

[4] https://gitlab.com/InfoKnow/AutomatedPlanning/LeonardoMoreira-LCMAP.

[5] http://www.plg.inf.uc3m.es/sw-ipc2011/IPCReport.

[6] IPC is a biennial event organized in the International Conference on Planning and Scheduling with objectives such as to provide a forum for an empirical comparison of planning systems. (http://www.plg.inf.uc3m.es/ipc2011-deterministic/).

Table 4 Setup description

Experiment	Domains											
	Rovers		Satellite		Zeno-travel		Logistics		Elevators		Depots	
	A	G	A	G	A	G	A	G	A	G	A	G
1	1	3	1	3	1	3	3	4	3	3	5	4
2	2	3	2	5	2	4	4	8	4	3	8	4
3	3	6	3	8	3	7	5	10	5	3	–	–
4	4	8	4	9	5	9	–	–	–	–	–	–
5	5	12	5	13	–	–	–	–	–	–	–	–

- Rovers (loosely coupled)—motivated by the exploration missions of Mars. The goal is to use a set of robots (agents) to traverse planet points by performing a variety of sample collection and data transmission operations to the base. The problem includes the visibility constraints of this base from various positions and the ability of each agent to traverse paths between pairs of points. Robots are differentiated by the set of equipment they own and use to perform the tasks. There is no need for cooperation during the execution, and the only interaction is when an agent collects a sample and therefore turns it unavailable to the other agents;
- Zeno-travel (loosely coupled)—related to a matter of transport in which people should board airplanes, travel between localities and then disembark. Airplanes use fuels at different rates according to the speed of the displacement;
- Depots (tightly coupled)—comprises two types of agents, warehouses and trucks, which must cooperate to meet the objectives. This is the most complex case of the tests and configures a tightly coupled domain with many dependencies between the agents;
- Elevators (tightly coupled)—the agents in this field are elevators that vary in speed, being able to be fast or slow, and in the floors that can access. This last feature defines the need for cooperation enabling objectives execution. For example, an elevator may not stop at a certain floor, then it will go to an intermediate floor so that the passenger can enter another elevator that reaches the final destination;
- Logistics (tightly coupled)—the agents in this domain are airplanes and trucks. The delivery of some packages involves the cooperation of elements of both types, since the trip between cities is realized by airplanes and the displacement within the same city is the responsibility of the trucks.

The experimental setup description regarding the number of agents (column A) and goals (column G) in each domain is summarized in Table 4.

5.3 Computational resources

We carried all experiments out in a single computer with a Dual-Core Intel Core i5-2410M with four executable threads and 6 GB RAM. The operational system was Linux Mint 18.1 Serena 64-bit.

The reported times describe all complete processes, from the input to the plan generation or failure. In case the approach is executed by the command line (java or shell script), we use the time[7] command to the measure time between the beginning and the end of the process.

[7] https://linux.die.net/man/1/time.

Table 5 Time2 metric comparison

Model	Rovers	Satellite	Zeno-travel	Logistics	Elevators	Depots
FMAP	0.491	0.459	**0.951**	0.121	**0.930**	0.101
LCMAP-LB	**0.984**	0.977	0.636	**1.000**	0.857	**1.000**
LCMAP-SR	0.953	**0.985**	0.630	0.948	0.845	0.522
MADLA	0.220	0.109	0.069	0.021	0.000	0.063
MAPJA	–	–	–	–	–	–
MAPR	–	–	–	–	–	–
MARC	–	–	–	–	–	–

In case the approach does not terminate at the end of the planning, for example, when there is a graphical user interface, we cannot apply this technique. However, typically time spent by each agent is displayed as soon as the plan is found, and therefore can be assumed as the highest values presented.

The planning tool defined by all agents was the Fast-Forward [21]. Since LCMAP provides two different assignment strategies as presented in Sect. 4.2. We labeled the results with LCMAP-LB and LCMAP-SR, which stands for load-balance and saving-resource, respectively.

5.4 Comparison with MAP models

In this section, we compare LCMAP results against other MAP approaches. Specifically, we compared the results against FMAP [47] and MADLA [41] for particular reasons. According to the detailed results of CoDMAP-15,[8] MARC [55] solved only one instance of the logistic and none of the other domains. Therefore, we did not run experiments with MARC. We asked the authors of MAPJA [9] for sharing the code but we had no answer. Experiments with MAPR [3] were also impossible because some libraries required to compile the code were no longer available.

In the loosely coupled domains (satellite, rovers, zeno-travel), experiments ranged the number of agents from one to five. In tightly coupled domains (depots, elevators, logistics), the limits were defined from three until eight, regarding the problem conditions. In both types of domain, the upper limit was defined as the configuration (number of agents) that the other approaches begun to halt without solving the problem. The lower limit in tightly coupled domain was defined as the minimum configuration that guaranteed the solution.

The first evaluation explores the *time2* metric. To improve the visualization of the values, the comparison of the performance in Table 5 is facilitated by the results shown in Fig. 3 and in the project repository. Each cell in Table 5 shows the average of *time2* metric values from the different experiment configurations. The highest values are highlighted.

The variation in values was significant, being able to reach a 100× increase between the simulations of one and five agents (in the satellite and zeno-travel domains). Therefore, the graphs in Fig. 3 follow a logarithmic scale in the Y axis to ensure that all values are displayed. The reason for a suddenly drop in MADLA values with four and five agents is because it was unable to solve the problem and finished with error.

[8] http://agents.fel.cvut.cz/codmap/results/presentation-RESULTS.pdf.

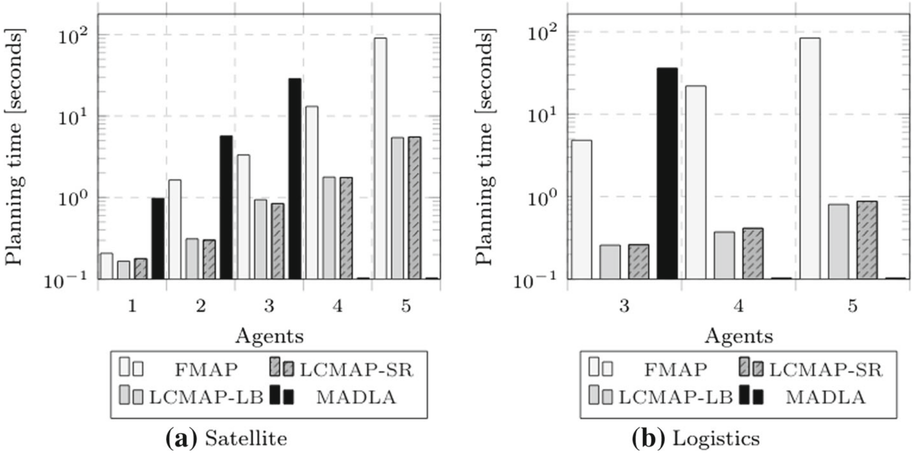


Fig. 3 Planning time

From rovers, satellite and zeno-travel domains, LCMAP results were better than other models in 71.43% (10 in 14) of the experiments. This behavior highlights the lack of cooperation requirements among agents, both in planning and execution phases, as an alternative to mitigate the effects of increasing the number of agents. While FMAP uses a strong agents' interaction during plan refinement, LCMAP explores the agents' decoupling, reaching better results. MADLA had the worst results in planning time. Regarding coverage, MADLA could not solve five problems, while FMAP just one. LCMAP solved all tests.

From depots, elevators and logistics domains, LCMAP results were better than other approaches in 75% (6 in 8). In these cases, despite verification, transformation and validation phases, the gain was because of the centralized role of the coordinator that planned without interacting with other agents. In the experiments in which LCMAP did not have the best results (elevators with four and five agents), it was 2× slower. However, in the other experiments, it was about 100× faster (logistics with five agents). Regarding the two types of domain, LCMAP was superior in 72.72% (16 of 22) of the experiments, regardless the objective strategy attribution.

The second evaluation explores the *quality* metric. Concerning the size of the plans, LCMAP showed solutions with fewer actions than FMAP and MADLA in nine experiments. Although LCMAP plans were about one or two actions shorter than FMAP solutions, this property must be considered as an advantage of the proposed model over other planners.

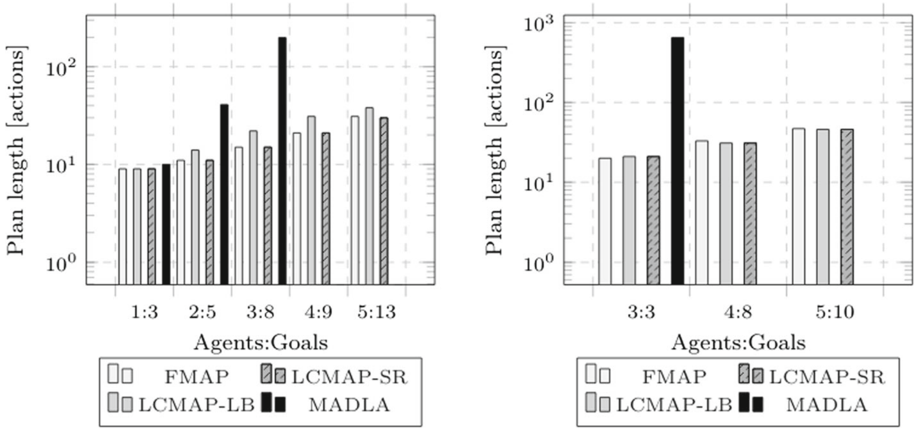
To improve the visualization of the values, the comparison of the performance in Table 6 is facilitated by the results shown in Fig. 4. The values are presented in Table 6 as average, similarly to Table 5. The highest values are highlighted.

Figure 4 uses a logarithmic scale in the Y axis to ensure that all values are displayed. MADLA results are higher than other values; therefore, they can impair the representation of smaller values.

Regarding both metrics, LCMAP presents the best two results. The difference between LCMAP and FMAP results answers the hypothesis about efficiency. LCMAP metric scores are better than FMAP in tightly coupled domains. The transformation phase leverages the highest part of this result by improving *time2* metric. The plan length metric by itself causes an impact lower than 1.0 and can not significantly influence the results. The balance between the computational process and privacy is the key reason for his behavior, since LCMAP

Table 6 Plan length quality metric comparison

Model	Rovers	Satellite	Zenotravel	Logistics	Elevators	Depots
FMAP	**0.991**	0.936	**0.917**	0.973	0.933	0.429
LCMAP-LB	0.926	0.748	0.906	**0.984**	**1.000**	**1.000**
LCMAP-SR	0.892	**1.000**	0.911	**0.984**	0.939	**1.000**
MADLA	0.522	0.249	0.255	0.010	0.000	0.200
MAPJA	–	–	–	–	–	–
MAPR	–	–	–	–	–	–
MARC	–	–	–	–	–	–



(a) Satellite

(b) Logistics

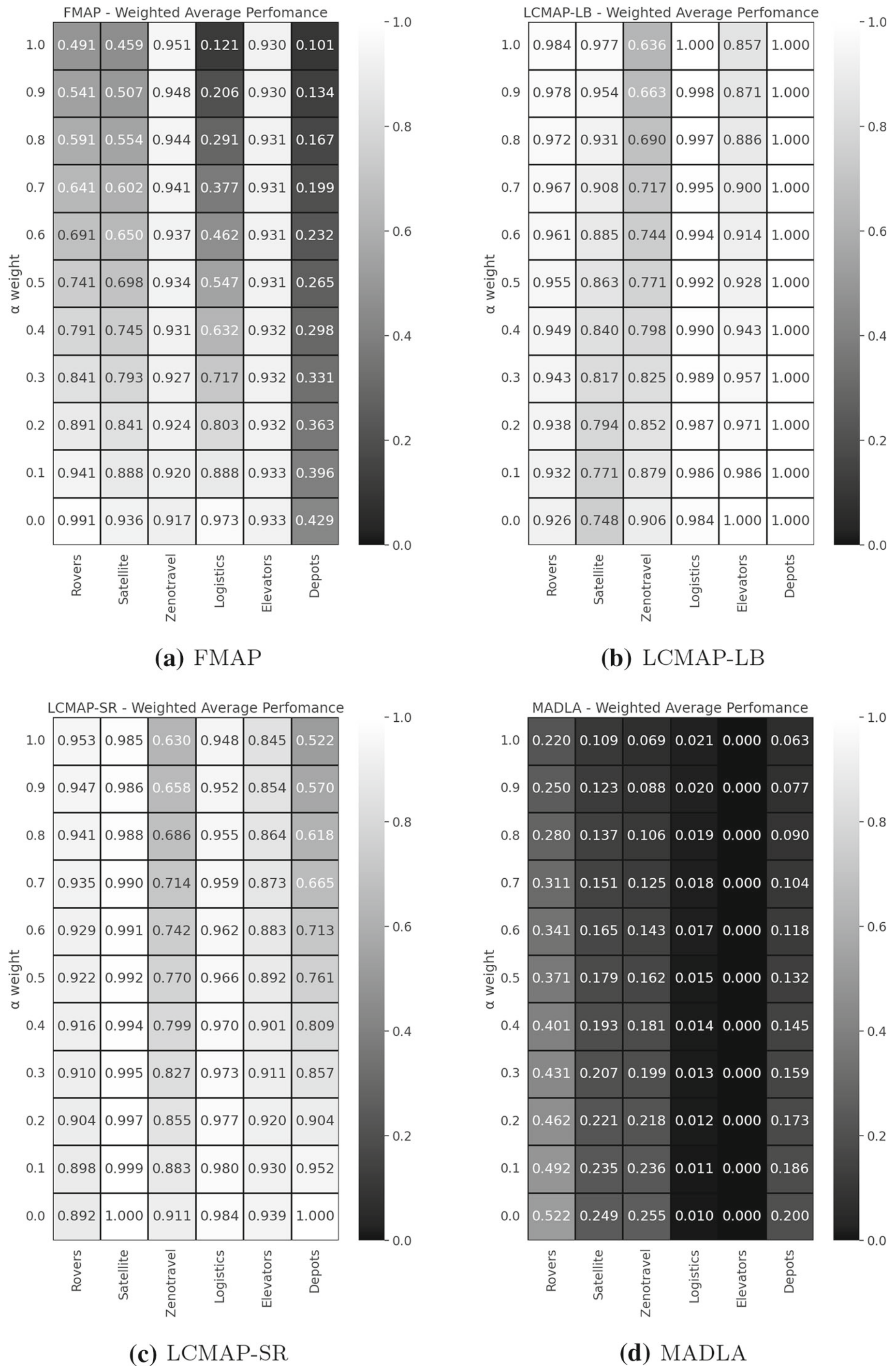
Fig. 4 Plan length

explores the decoupling level of planning tasks. MADLA is excluded from this comparison since it could not perform.

A performance metric is computed concerning the results of each model, using a weighted average of *time2* and *quality*, shown in Tables 5 and 6, following Eq. 9. Here, the purpose is to check each model regarding the impact of planning time and number of actions over its performance. This evaluation is important because, in some cases, the time spent during planning process is critical, for instance, in rescue operations, where time is key to mission success. In other conditions, agents must care about their resources, such as battery and fuel, more than the time. Thus, in scenarios such as robotics path finding, smaller plans avoid resource misuses resulting in a better final plan.

$$performance = \frac{\alpha * time2 + \beta * quality}{\alpha + \beta} \tag{9}$$

The evaluation of the performance metric is presented by heat maps in Fig. 5 where each model is checked regarding the experiments and varying the weight of *alpha* from 0.0 to 1.0. High levels of α indicate that time is more important than plan size. The performance of LCMAP-LB (Fig. 5b) improves when the weight related to *time2* increases in four domains, but zeno-travel and elevators. On the other hand, the performance of LCMAP-SR (Fig. 5c) and FMAP (Fig. 5a) is impaired by the increase in α in all domains but rovers and zeno-travel,



(a) FMAP

(b) LCMAP-LB

(c) LCMAP-SR

(d) MADLA

Fig. 5 Performance metric

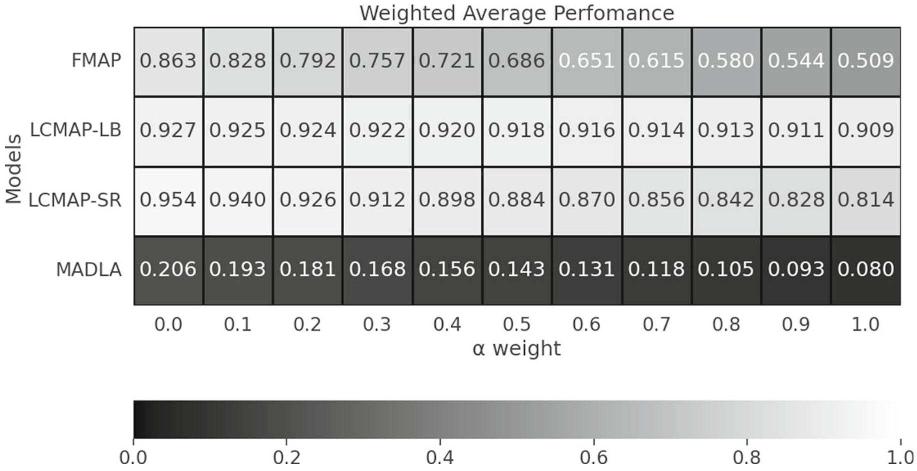


Fig. 6 Final comparison

respectively. At last, MADLA (Fig. 5d) had the worst behavior, no matter the level of α. The performance metric shows that a quick planning process costs non-optimal plans and the search for saving resources depends on a greater consumption of time.

Finally, a final comparison is studied considering all the experiments. The mean values[9] of *time2* and *quality* of each model are used, according to Eq. 10, to rank the models under different conditions. The variation in the final performance is presented in Fig. 6. It is important to highlight that LCMAP had the best scores regardless the value of $alpha$. Moreover, from $alpha = 0.3$, LCMAP-LB shows the best performance, indicating that it can balance planning time and plan size in most of the problems.

$$performance = \frac{\alpha * \overline{time2} + \beta * \overline{quality}}{\alpha + \beta}. \tag{10}$$

6 Conclusions

The main contribution of this work is the presentation of the Lightweight Coordination Multi-Agent Planning model that balances coordination process and privacy through three independent phases: verification, transformation and validation. LCMAP is proved to be an efficient solution to MAP when compared to the state-of-the-art models regarding time efficiency and plan length during the problem solving process. LCMAP has been tested on IPC loosely and tightly coupled domains proving to be a domain-independent model.

The comparison among LCMAP and other related models shows that balancing computational process and privacy provides efficiency to MAP models. Hence, the hypothesis whether LCMAP is an efficient model when comparing to state-of-the-art can be accepted.

Although LCMAP outperforms the state-of-the-art models, such as FMAP and MADLA, some future works can be suggested. Other MAP implementations and planning domains may be evaluated to reinforce the flexibility and modularity aspects of the proposed model.

[9] $\overline{metric}$ describe the arithmetic mean of the values of the model considering all the experiments.

The support to probabilistic models and more dynamic planning aspects would be integrated into the LCMAP model to decrease distance to real-world problems.

Acknowledgements Prof. Célia G. Ralha thanks the support received from the Brazilian National Council for Scientific and Technological Development (CNPq) for the research productivity Grant Number 311301/2018-5.

References

1. PDDL Resources - IPC - 2008 (2016) Available at http://icaps-conference.org/ipc2008/deterministic/PddlResources.html
2. Baldoni M, Baroglio C, Micalizio R (2015) Social continual planning in open multiagent systems: a first study. In: PRIMA, pp 575–584
3. Borrajo D (2013) Multi-agent planning by plan reuse. In: International conference on autonomous agents and multiagent systems (AAMAS), IFAAMAS, pp 1141–1142
4. Borrajo D, Fernández S (2015) MAPR and CMAP. In: Competition of distributed and multi-agent planners (CoDMAP 2015), pp 1–3
5. Borrajo D, Fernández S (2019) Efficient approaches for multi-agent planning. Knowl Inf Syst 58(2):425–479
6. Brafman R (2015) A privacy preserving algorithm for multi-agent planning and search. In: Proceedings of the 24th international conference on artificial intelligence. AAAI Press, pp 1530–1536
7. Cardoso R, Bordini R (2016) Allocating social goals using the contract net protocol in online multi-agent planning. In: Proceedings of the 5th Brazilian conference on intelligent systems (BRACIS), IEEE, pp 199–204
8. Chouhan SS, Niyogi R (2016) Multi-agent planning with quantitative capability. In: International conference on advances in computing, communications and informatics (ICACCI), IEEE, pp 602–606
9. Chouhan SS, Niyogi R (2017) MAPJA: multi-agent planning with joint actions. Appl Intell 47(4):1044–1058
10. Chouhan SS, Singh A, Niyogi R (2015) Multi-agent planning with joint actions. In: 2015 International conference on advances in computing, communications and informatics (ICACCI), IEEE, pp 1284–1290
11. Cooper ID (2016) What is a mapping study? J Med Libr Assoc 104(1):76
12. Crosby M, Jonsson A, Rovatsos M (2014) A single-agent approach to multiagent planning. In: Proceedings of the 22nd European conference on artificial intelligence (ECAI), pp 237–242
13. Dukeman A (2017) Hybrid mission planning with coalition formation. Ph.D. Thesis, Vanderbilt University
14. Elkawkagy M, Biundo S (2011) Hybrid multi-agent planning. In: Multiagent system technologies, pp 16–28
15. Ferrando SP, Onaindia E (2012) Defeasible argumentation for multi-agent planning in ambient intelligence applications. In: 11th international conference on autonomous agents and multiagent systems (AAMAS), IFAAMAS, pp 509–516
16. FiŠer D, Štolba M (2015). MAPlan. In: Competition of distributed and multi-agent planners (CoDMAP 2015), pp 8–10
17. Ghallab M, Nau D, Traverso P (2016) Automated planning: theory and practice. Cambridge University Press
18. Gordon G, Varakantham P, Yeoh W, Lau HC, Aravamudhan A, Cheng S-F (2012) Lagrangian relaxation for large-scale multi-agent planning. In: International joint conferences on web intelligence and intelligent agent technology, IEEE Computer Society, vol 2, pp 494–501
19. Guzman C, Castejon P, Onaindia E, Frank J (2014) Robust plan execution in multi-agent environments. In: 26th international conference on tools with artificial intelligence (ICTAI), IEEE, pp 384–391
20. Hadad M, Kraus S, Hartman IB-A, Rosenfeld A (2013) Group planning with time constraints. Math Artif Intell 69(3):243–291
21. Hoffmann J, Nebel B (2001) The FF planning system: fast plan generation through heuristic search. J Artif Intell Res 14:253–302
22. Jonsson A, Rovatsos M (2011) Scaling up multiagent planning: a best-response approach. In: International conference on automated planning and scheduling (ICAPS)
23. Kitchenham B, Budgen D, Brereton P (2010) The value of mapping studies-a participant-observer case study. In: EASE, vol 10, pp 25–33

24. Klöpper B, Honiden S, Dangelmaier W (2011) Divide & conquer in planning for self-optimizing mechatronic systems-a first application example. In: Symposium on computational intelligence in control and automation (CICA), IEEE, pp 108–115
25. Luis N, Borrajo D (2014) Plan merging by reuse for multi-agent planning. In: 2nd ICAPS workshop on distributed and multi-agent planning (DMAP'14), pp 38–44
26. Maliah S, Brafman R, Shani G (2017) Increased privacy with reduced communication in multi-agent planning. In: International conference on automated planning and scheduling (ICAPS)
27. Maliah S, Shani G, Stern R (2016) Stronger privacy preserving projections for multi-agent planning. In: Proceedings of the 26th international conference on international conference on automated planning and scheduling. AAAI Press, pp 221–229
28. Moreira LH, Ralha CG (2016) Transforming multi-agent planning into single-agent planning using best-cost strategy. In: Brazilian conference on intelligent systems (BRACIS), IEEE, pp 462–467
29. Moreira LH, Ralha CG (2017) Improving multi-agent planning with unsolvability and independent plan detection. In: Brazilian conference on intelligent systems (BRACIS), IEEE, pp 103–108
30. Nath A, Niyogi R (2015a) An extension of FMAP for joint actions. In 8th international conference on contemporary computing (IC3), IEEE, pp 487–492
31. Nath A, Niyogi R (2015b) Validation and verification of joint-actions in multi-agent planning. In: International conference on information technology (ICIT), IEEE, pp 187–192
32. Nissim R, Brafman R (2012) Multi-agent A* for parallel and distributed systems. In: Proceedings of the 11th international conference on autonomous agents and multiagent systems (AAMAS), AAMAS'12. Richland, SC. IFAAMAS, pp 1265–1266
33. Palomares I, Bauters K, Liu W, Hong J (2016a) A two-stage online approach for collaborative multi-agent planning under uncertainty. In: International conference on scalable uncertainty management. Springer, pp 214–229
34. Palomares I, Killough R, Bauters K, Liu W, Hong J (2016b) A collaborative multiagent framework based on online risk-aware planning and decision-making. In: 28th international conference on tools with artificial intelligence (ICTAI), IEEE, pp 25–32
35. Pardo P, Pajares S, Onaindia E, Godo L, Dellunde P (2011) Multiagent argumentation for cooperative planning in delp-pop. In: 10th international conference on autonomous agents and multiagent systems (AAMAS). International Foundation for Autonomous Agents and Multiagent Systems, vol 3, pp 971–978
36. Pellier D, Fiorino H (2018) PDDL4J: a planning domain description library for java. J Exp Theor Artif Intell 30(1):143–176
37. Qing C, Wen Z, Yulong H (2012) Find a multi-agent planning solution in nondeterministic domain. In: International conference on automatic control and artificial intelligence (ACAI), pp 1053–1056
38. Redding J, Ure K, How J, Vavrina M, Vian J (2012) Scalable, MDP-based planning for multiple, cooperating agents. In: American control conference (ACC), IEEE, pp 6011–6016
39. Sapena Ó, Torreño A, Onaindia E (2011) On the construction of joint plans through argumentation schemes. In: 10th international conference on autonomous agents and multiagent systems (AAMAS), IFAAMAS, vol 3, pp 1195–1196
40. Štolba M, Fišer D, Komenda A (2016a) Potential heuristics for multi-agent planning. In: Proceedings of the 26th international conference on international conference on automated planning and scheduling. AAAI Press, pp 308–316
41. Štolba M, Komenda A (2015) MADLA: planning with distributed and local search. In: Competition of distributed and multi-agent planners (CoDMAP 2015), p 21
42. Štolba M, Tožička J, Komenda A (2016b) Secure multi-agent planning. In: 1st international workshop on AI for privacy and security, ACM, p 11
43. Toniolo A, Norman T, Sycara K (2011) Argumentation schemes for collaborative planning. In: PRIMA. Springer, vol 7047, pp 323–335
44. Torreño A, Onaindia E, Komenda A, Štolba M (2017) Cooperative multi-agent planning: a survey. ACM Comput Surv 50(6):84:1-84:32
45. Torreño A, Onaindia E, Sapena Ó (2012) An approach to multi-agent planning with incomplete information. In: Proceedings of the 20th European conference on artificial intelligence (ECAI). IOS Press, pp. 762–767
46. Torreño A, Onaindia E, Sapena Ó (2014) A flexible coupling approach to multi-agent planning under incomplete information. Knowl Inf Syst 38:141–178
47. Torreño A, Onaindia E, Sapena Ó (2014) FMAP: distributed cooperative multi-agent planning. Appl Intell 41(2):606–626
48. Torreño A, Sapena Ó, Onaindia E (2015) Global heuristics for distributed cooperative multi-agent planning. In: International conference on automated planning and scheduling (ICAPS), pp 225–233

49. Tožička J, Jakubův J, Komenda A (2015) PSM-based planners description for CoDMAP 2015 competition. In: Competition of distributed and multi-agent planners (CoDMAP 2015), pp 29–32
50. Tožička J, Jakubův J, Komenda A, Pěchouček M (2016) Privacy-concerned multiagent planning. Knowl Inf Syst 48(3):581–618
51. Undurti A, How J (2011) A decentralized approach to multi-agent planning in the presence of constraints and uncertainty. In: International conference on robotics and automation (ICRA), IEEE, pp 2534–2539
52. Wang C, Liu P (2013) Multi-agent planning for fleet cooperative air defense decision-making. In: 9th international conference on natural computation (ICNC), IEEE, pp 533–538
53. Weiss G (2013) Multiagent Systems, 2nd edn. The MIT Press
54. Wooldridge M (2009) An Introduction to Multiagent Systems, 2nd edn. Wiley
55. Zhang Y, Sreedharan S, Kambhampati S (2016) A formal analysis of required cooperation in multi-agent planning. In: International conference on automated planning and scheduling (ICAPS). AAAI

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Leonardo Henrique Moreira received a Master degree from the University of Brasília, Brazil, in 2017. He has worked as a software engineer in the Brazilian Army since 2012, focusing on command and control and business intelligence solutions. He is currently a Ph.D. student at the Department of Computer Science, Institute of Exact Sciences, University of Brasília, Brazil. His research interests include multi-agent planning in dynamic environments and recovery strategies, such as replanning and repairing.

Célia Ghedini Ralha holds a Ph.D. in Computer Science from Leeds University, England (1996) and an M.Sc. in Electronic and Computer Engineering from Aeronautics Institute of Technology, Brazil (1990). She is an Associate Professor at the Department of Computer Science, University of Brasília, Brazil. She has participated in many research projects and published several papers in international journals and conferences. She is a senior member of the Brazilian Computer Society and receives a research productivity grant from the Brazilian National Council for Scientific and Technological Development (CNPq). Her current research interests include agent-based simulation, intelligent systems, knowledge-based systems, machine learning, multi-agent planning, and multi-agent system.