Multiagent and Grid Systems – An International Journal -1 (2020) 1–22 1
DOI 10.3233/MGS-200327
IOS Press

Plan distance heuristics for task fusion in distributed temporal continuous planning

Gilberto Marcon dos Santos* and Julie A. Adams
Collaborative Robotics and Intelligent Systems Institute, Oregon State University, Corvallis, OR, USA

Received 5 September 2019
Accepted 4 March 2020

Abstract. Automating planning for large teams of heterogeneous robots is a growing challenge, as robot capabilities diversify and domain complexities are incorporated. Temporal and continuous features accurately model real-world constraints, but add computational complexity. Distributed planning methods, such as the Coalition Formation then Planning framework, allocate tasks to robot teams and plan each task separately to accelerate planning. However, the task decomposition limits cooperation between coalitions allocated to different tasks and results in lower quality plans that require more actions and time to complete. Task Fusion estimates couplings between tasks and fuses coupled coalition-task pairs to improve cooperation and produce higher quality plans. Task Fusion relies on existing heuristics, which were ineffective and often resulted in worse results than the baseline framework. This manuscript introduces new heuristics that outperform the existing methods in two complex heterogeneous multi-robot domains that incorporate temporal and continuous constraints.

Keywords: Multiagent planning, coalition formation, temporal continuous planning, plan distance

1. Introduction

Robots are rapidly moving into the commercial, medical, and military domains. The fast-paced development of sensing, processing, and actuation devices at increasingly lower costs is resulting in robots with a growing variety of capabilities, approaching a world where robots are ubiquitous and diverse. Robots have proven potential to assist in response to major disasters, such as search and rescue, bomb defusal, and natural disasters, but currently highly trained operators make most decisions, while the robot decision making is limited to low level actions [2]. Exploiting the potential of autonomous robots will require scalable automated planning capable of modeling complex problems that incorporate a diverse set of robots [2].

First response for natural and man-made disasters requires rapid evaluation and deployment of available personnel and equipment in order to mitigate the situation, which when combined with the robotic aspects, greatly increases the complexity of the deployment allocation and assignment problems. Existing planning methods (e.g., [7,11,20,41]) fail to account for all of the domain's complexities, such as requiring continuous fluents (e.g., fuel capacity), concurrent actions (e.g. simultaneously triaging multiple victims),

*Corresponding author: Gilberto Marcon dos Santos, Collaborative Robotics and Intelligent Systems (CoRIS) Institute, Oregon State University, 2000 SW Monroe Ave, 204 Rogers Hall, Corvallis, OR 97331, USA. E-mail: marcondg@oregonstate.edu.

and real-time results (e.g. planning and executing the plan within the task deadlines). Existing methods can only meet some of these requirements and cannot scale to a large number of heterogeneous robots [23].

Dukeman and Adams [18] developed the hybrid Coalition Formation then Planning (CFP) framework to merge automated planning and coalition formation with improved scalability to dozens of robots when developing continuous temporal plans. The CFP assigns robots to coalitions according to their capabilities and allocates tasks to each coalition. Planning for tasks separately accelerates planning, but limits cooperation between coalitions, lowering plan quality, and requiring more actions and time to execute.

Task Fusion merges coalition-task pairs in order to improve the plan quality by evaluating the coupling of each pair. Task Fusion uses heuristics to estimate coupling and fuse the highest scoring pairs; thus, allowing explicit cooperation between robots in the fused coalitions and improved plan quality. However, prior analysis of Task Fusion effectiveness was inconclusive [18]. While some problems solved using Task Fusion resulted in better quality plans, most produced worse quality plans, due to inaccurate heuristics.

This manuscript's main contribution addresses the limitations of CFP and devises new heuristics that estimate coupling between tasks and coalitions. Detecting couplings allows fusing tightly coupled coalition-task pairs, which improves cooperation and reduces plan length; thus, producing higher quality plans that contain fewer actions and require less time to execute. The new heuristics leverage plan distance as a proxy for the coalition-task coupling estimation. Plan distance heuristics, previously used as a measure of plan diversity [40], were adapted for estimating problem coupling in Task Fusion. Relaxed plans are determined rapidly from the coalition-task problems, and the distance between relaxed plans indicates the level of coupling that informs the Task Fusion. The plan distance heuristics provide better plans that require fewer computational resources for planning. This manuscript introduces new plan distance heuristics, presents and evaluates four new hypothesis, incorporates four new metrics, and evaluates the CFP framework using an additional planner.

2. Literature review

Automated planning for teams of heterogeneous robots is a specific application of multiagent planning. Multiagent planning is the more general field of planning, which goes beyond embodied physical robots and can involve other types of agents, such as software agents. Software agents are abstract decision making entities, such as web crawlers and stock traders, whereas robot systems have physical embodiment in the form of sensors and actuators, often associated with a mobile body [45]. Distributed temporal continuous planning incorporates temporal constraints and continuous numerical fluents to more accurately model real-world problems for multiple robot systems.

Planning for complex domains, such as first response, requires the classical planning model to be extended to incorporate expressive features, such as concurrent action execution (e.g., [9,26,34]) and continuous fluents (e.g., [7,10,11,20]). No single planner incorporates all the necessary features, and the most expressive algorithms are unable to scale to complex problems with multiple robots or task complexities [22]. However, significant performance improvements can be achieved by factoring the problem based on the system and the environment [31]. Multiagent factoring distributes the plan synthesis across multiple planning agents in order to reduce the computational complexity [42]. The planning problem is partitioned into tasks and task plans are devised independently. Planning agents coordinate before planning, to allocate tasks, and after planning, to merge the individual plans and minimize conflicts [16]. Planning coordination remains a challenging problem, especially for problems with tightly coupled tasks [5] that have mutual dependencies, such as shared locations and resources (e.g., tools and assets), and cannot be independently solved. The independent execution of one task can alter the environment and jeopardize the ability to accomplish tasks.

Plan merging algorithms allow agents to coordinate after planning in order to solve action redundancies and consistency flaws [13]. The Multiagent Plan Coordination by Plan Modification Algorithm minimizes the resulting number of actions while merging independently generated plans [13]. A set of actions is replaced by a single redundant action, resulting in merged plans with fewer actions, but the algorithm's scalability to a large number of robots is limited. The Temporal Optimal Conflict Resolution Algorithm employs a search relaxation constant in order to scale to a large number of robots, but cannot scale to a large number of tightly coupled tasks [28]. Another temporal plan merge algorithm generates relaxed plans for each task prior to merging [29], but is not applicable to multiple robot tasks.

Decentralized planing algorithms can use serial plan synthesis for tightly coupled tasks [17]. Robots generate plans iteratively, where each planning agent assumes that its initial state is the prior agent's final state. The planner's goals are concatenated with the goals of the next planning agent in order to guarantee that the next agent will not undo the previous agent's achieved goals. Serial plan synthesis does not require serial plan execution. The serially synthesized plans can be merged for parallel execution; however, most existing decentralized planners assume serial plan execution. Rather, the agents take actions in turns, which hinders applicability to real-world multiple robot systems [42]. Parallel action execution requires sophisticated coordination methods that optimize parallel plan execution, while also minimizing makespan, the plan execution time.

The Multi-Agent Planning by Plan Reuse algorithm performs task allocation then planning using relaxed reachability analysis after generating relaxed plans for all agent-task combinations. However, the method requires homogeneous agents [4]. The algorithm can be applied to a heterogeneous mobile multiple robot system by using actuation maps instead of relaxed plans, but it does not generalize to complex tasks [32].

Task allocation can address coupling and optimize parallel plan execution with problem decomposition [6]. The agent interaction graph minimizes the problem coupling when allocating tasks in order to reduce computational complexity [6]. The problem decomposition is formulated as a constraint satisfaction problem, but solving the constraint satisfaction problem dominates the plan synthesis time, rendering the algorithm inefficient [15]. The Agent Decomposition-Based Planner uses causal graphs to decompose the planning problem [15], and has been extended to support concurrent actions in a real-world industrial mobile manipulator robot domain [14], but does not scale to a large number of robots. The Multiagent Planner for Required Cooperation allocates tasks to m planning agents, which devise plans for n executing agents [39]. The above methods cannot support concurrent action execution, hindering their applicability to real-world multiple robot systems.

Models of capabilities were used recently as a heuristic for state-space forward search [46]. Capabilities are modeled as the agent's likelihoods of achieving a particular state from any other state. A Bayesian network learns the likelihood capabilities from plan traces, but requires a large number of plan execution simulations covering initial and goal states. Buehler et. al [8] define a robot capability as an extended action schema that integrates with the underlying Robot Operating System (ROS) [33]. ROS controls action execution and relays abstracted sensor data to a multiple robot planning and execution architecture. However, neither study uses capability models to improve plan quality or reduce computational cost. The following Section presents the promises and limitations of the Coalition Formation then Planning framework for scalable multiagent planning, and introduces a new family of heuristics that address the shortcomings of the approach.

3. Coalition formation and planning

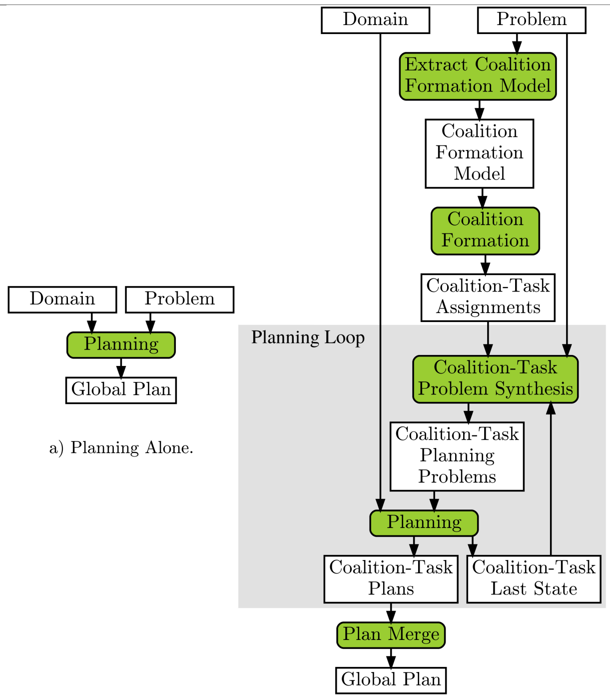
Coalition formation is an alternative task allocation method for multiple robot distributed planning [18]. Coalition formation generalizes task allocation by assigning entities (e.g., robots or humans) to coalitions to perform tasks (e.g., [1,21,38,44]). The entities are grouped into coalitions according to the capabilities offered by the individual entities and the capabilities required to complete the tasks. Capabilities represent resources (i.e., sensor range and battery power) or services, (i.e., distance measurement or image acquisition) [37]. This Section formally defines coalition formation as applied to multiple robot planning, and introduces a new family of heuristics to address the shortcomings of the approach.

The coalition formation problem takes a set of n robots, $\Phi = \{\phi_1, \phi_2, \ldots, \phi_n\}$, and a set of m tasks, $V = \{v_1, v_2, \ldots, v_m\}$. Coalition formation maps tasks to coalitions, $CF : V \to 2^{\Phi}$, which yields a set of m coalition-task pairs $P_m = \{p_1, p_2, \ldots, p_m\} \mid p_i = \langle \Phi_i, v_i \rangle \mid \Phi_i \subseteq \Phi$. A capability c_j is a non-negative real number and each individual robot has a vector of capabilities $C_\phi = \langle c_1^\phi, c_2^\phi, \ldots, c_k^\phi \rangle$, where each vector entry c_j^ϕ represents a capability j offered by robot ϕ, and k is the number of modeled capabilities. The set of all n robots is the capability vector set $C^\Phi = \{C_1, C_2, \ldots, C_n\}$ that associates a capability vector with each robot. Tasks are defined as a vector of required capabilities $C_v = \langle c_1^v, c_2^v, \ldots, c_k^v \rangle$, where each vector entry c_j^v represents a capability j required by task v. The set of m tasks is the task requirement capability vector set $C^V = \{C_1, C_2, \ldots, C_m\}$ that associates a capability vector with each task. A coalition $\Phi_v \subseteq \Phi$ is a subset of robots capable of executing a task v, if $(\sum_{\phi \in \Phi_v} c_j^\phi) \geqslant c_j^v, \forall j \in \{1, 2, \ldots, k\}$. Coalition formation algorithms maximize the individual robot's contributions to the tasks and allow robots to belong to multiple coalitions.

The Hybrid Mission Planning with Coalition Formation (HMPCF) [18] model is represented as a tuple $\langle S, I, A, \Phi, V, M, C \rangle$, where $S = \{s_1, s_2, \ldots\}$ is the state space, $I \subseteq S$ is the initial state, $A = \{a_1, a_2, \ldots\}$ is the action space, $\Phi = \{\phi_1, \ldots, \phi_n\}$ is the set of n robots, the grand coalition, $V = \{v_1, \ldots, v_m\}$ is the set of m tasks, $M : \Phi \to A_\Phi \mid A_\Phi \subseteq A$ is the robot-action mapping function, and C is the tuple $\langle C^\Phi, C^V \rangle$, where $C^\Phi = \{C_1, C_2, \ldots, C_n\}$ is the robot capability vector set and $C^V = \{C_1, C_2, \ldots, C_m\}$ is the task requirement capability vector set. The conjunction of all task conditions defines the goal states $G \subseteq S \mid G = \{s_1, s_2, \ldots, \} \mid s \vdash \bigwedge_{v \in V} v \mid \forall s \in G$. A solution to a HMPCF problem is a plan, π, consisting of a set of scheduled actions assigned to each robot $\phi \in \Phi$. HMPCF uses existing Coalition Formation and Planning algorithms to solve large multiple robot planning problems that incorporate temporal constraints and continuous numerical fluents in a more tractable, albeit centralized manner.

Planning alone, a fully centralized baseline planning method, groups all robots and tasks into a single-agent multi-effector planning problem. Planning Alone synthesizes a goal set G as the conjunction of all task requirements and invokes a domain-independent planner, as indicated in Fig. 1a. The external planner receives the grand coalition's combined action space and attempts to satisfy all task constraints embedded in the goal state set. Planning Alone generates high-quality plans, but scales poorly as the number of robots or the domain complexity increase [18]. The combinatorial complexity of centralized planning limits the problems that can be solved.

Coalition formation then planning (CFP) is a hybrid method that leverages coalition formation to minimize combinatorial complexity and overall planning time. The robot and task capability vector sets and coalition formation algorithms are used to generate the coalitions and assign tasks, invoking planning algorithms for each coalition-task pair. The resulting coalition-task pair plans are merged into a global plan, as presented in Fig. 1b. CFP applies serial plan synthesis and assumes robot coalitions take turns when planning, with coordination occuring before and after planning. Coordination before



b) Coalition Formation then Planning.

Fig. 1. Planning alone (a) and the coalition formation then planning framework (b). Rounded filled shapes represent processes and rectangles represent data. Coalition formation then planning extracts robots' and task's capabilities from the problem description, partitions the robots into coalitions and generates separate plans for each coalition. The coalition plans are merged into a global plan [18].

planning occurs by forming coalitions and allocating tasks, whereas plan merging performs coordination after planning by minimizing action redundancy, while also preventing consistency flaws [13]. Each task planning problem is solved separately, and the task planning problem's goals are concatenated with the goals of the next task planning problem in order to guarantee that the following task plan will not undo the goals achieved by the prior task plan. CFP uses capabilities to inform problem partitioning.

The *Extract Coalition Formation Model* process derives robot and task capability vectors from the problem description [18]. Coalition formation generates coalition-task pairs and the *Coalition-Task Problem Synthesis* process uses the coalition-task pairs, the problem description, and the final state achieved by the latest coalition-task plan to generate separate planning problems for each coalition-task pair. The *Coalition-Task Planning Problems* are solved separately by external planners, such as COLIN [11] or TFD [20]. The planner produces a plan for each coalition-task pair, and the resulting plans are merged into a global plan using a greedy approach [18].

Multiple robot planning is largely an intractable problem, but assigning tasks to the most appropriate robot coalitions scales significantly better. The coalition formation problem is NP-hard [38], but domain-independent planning is EXPSPACE-complete [19]. The plan synthesis time can be orders of magnitude longer than the corresponding coalition formation problems; thus, the overhead created by coalition formation is minimal. The problem complexity is reduced by generating multiple small-action-set plans. The reduced search branching factor permits derivation of plans for significantly larger problems [18]. The CFP framework is agnostic to the coalition formation algorithm adopted and uses external algorithms [36].

Coalition formation can scale planning to larger numbers of robots and more complex tasks, but results in poor quality plans, that have longer makespan than centralized planning (i.e., Planning Alone) [18]. The model of capabilities used by coalition formation does not reveal whether tasks are tightly coupled, limiting cooperation between coalitions allocated to different tasks and results in lower quality plans that require more actions and time to complete. Plan quality can be improved by partitioning the planning problem along coalition and task coupling lines [6]. The most tightly coupled coalition-tasks pairs are fused, whereas the most loosely coupled remain separate. When two coupled coalition-task planning problems are solved separately, the planner considers each tasks' goals individually, and produces potentially redundant action sequences [42]. However, when two coupled coalition-tasks are fused, the actions for one task can contribute to achieving states necessary to achieve another task and can generate higher quality plans. Planning uncoupled coalition-task planning problems together does not improve plan quality, and often increases planning complexity.

3.1. Task fusion

Coalition formation was enhanced with *Task Fusion* in order to account for tightly coupled tasks and generate higher quality plans at a lower computational cost [18]. After coalition formation, tightly coupled coalition-task pairs are fused into larger coalition-task pairs. The fused coalition-tasks plans can be synthesized faster and result in shorter makespan with fewer actions. Fusing allows the planner to address the tasks' mutual dependencies and facilitates cooperation between the fused coalitions. The result of Task Fusion over coalition-task pairs $p_i = \langle \Phi_i, v_i \rangle$ and $p_j = \langle \Phi_j, v_j \rangle$ is a fused coalition-task pair p_f, $p_f = \langle \Phi_f, v_f \rangle = F(p_i, p_j)$, where F is a mapping of pairs of coalition-task pairs $F : p_i \times p_j \to p_f \mid p_i, p_j \in P_m$, Φ_f is the union of robots $\phi \subseteq \Phi_i$ and $\phi \subseteq \Phi_j$, $\Phi_f = \Phi_i \cup \Phi_j$, and $v_f = v_i \wedge v_j$ is the conjunction of task requirements from v_i and v_j.

Task fusion is the fusion of tasks and the assigned coalitions. Both coalitions and tasks are fused. Tasks are fused by concatenating the goals of the original tasks. Coalitions are fused by combining or taking the union of the members of the original coalitions. A coalition-task pair consists of a task and a coalition. Fusing a coalition results in a new coalition where the members of the original coalitions are combined. Fusing tasks results in a new task, where the goals of the original tasks are concatenated. Members from both coalitions will be considered during plan generation, as the goals of both tasks must be satisfied by the resulting plan.

Coalition-task coupling is estimated by a heuristic that maps two coalition-task pairs p_i and p_j, to a coupling estimate, $H(p_i, p_j)$: $p_i \times p_j \to$ [0,1], where $H(p_i, p_j) = 0$ indicates that p_i and p_j are uncoupled and $H(p_i, p_j) = 1$ indicates that p_i and p_j are tightly coupled. The Task Fusion algorithm stops when the ratio of fused coalitions, relative to the original number of coalitions, m, becomes greater than a user-defined threshold f_{max}, the fusion ratio, as presented in Algorithm 1. No coalition is fused when $f_{max} = 0$, and all coalitions are fused when $f_{max} = 1$. A zero fusion ratio, $f_{max} = 0$, is equivalent to the baseline CFP (i.e., no Task Fusion). Fusing all coalitions using $f_{max} = 1$ does not produce the grand coalition, because the algorithm is restricted to pair wise coalition fusion only, in order to avoid the combinatorial complexity of evaluating all possible coalition-task subsets. A grand coalition can only be produced when only two coalitions exist to be fused.

The previously developed heuristics estimate coalition-task coupling based on the coalition formation model of capabilities [18]. The *Coalition Similarity* (CS) heuristic, $\frac{|\Phi_i \cap \Phi_j|}{|\Phi_i \cup \Phi_j|}$, operates on coalition-task pairs that share common robots. Coalition-task pairs that have no common robots score 0 and coalition-task pairs that share all robots score 1. The *Coalition Assistance* (CA) heuristic, $\sum_{r=1}^{k} \frac{c_r^{\Phi_i \cup \Phi_j}}{max(c_r^{v_i}, c_r^{v_j})}$, estimates the

Algorithm 1: The task fusion algorithm.

Data: $P_m = \{p_1, p_2, \ldots, p_m\}$, a set of m coalition-task pairs;
$H(p_i, p_j) : p_i \times p_j \to [0, 1]$;
Result: A set of o coalition-task pairs $P_o = \{p_1, p_2, \ldots, p_o\}$.
Initialize empty set $P_o = \{\emptyset\}$;
Populate list l with all $\binom{m}{2}$ pairs of coalition-task pairs $\langle p_i, p_j \rangle$, $p_i, p_j \in P_m$;
foreach *pair* $\langle p_i, p_j \rangle$ *in list* l **do**
  Compute the heuristic value $h_{ij} = H(p_i, p_j)$;
end
Sort list l relative to the heuristic value h_{ij};
foreach *pair* $\langle p_i, p_j \rangle$ *in list* l **do**
  Remove pair $\langle p_i, p_j \rangle$ from list l;
  Remove all pairs containing p_i or p_j from list l and from set P_m;
  Fuse pair $\langle p_i, p_j \rangle$ into coalition-task pair $p_f = F(p_i, p_j)$;
  Insert coalition-task pair p_f into set P_o;
  if $2 \cdot \|P_o\| > m \cdot f_{max}$ **then**
    break;
  end
end
Return $P_o = P_m \cup P_o$;

ratio of coalition capabilities over task requirement capabilities after fusion, and prioritizes coalition-task pairs that share the same task requirement capabilities. These heuristics do not consider planning-related information, such as robots handling the same logical objects, or sharing the same physical location. Ignoring planning-related information limits the heuristic's accuracy, which can produce plans that take longer to execute and require a larger number of actions. The following section introduces a new family of heuristics that leverage plan distance metrics to improve plan quality and cost.

3.2. New task fusion heuristics with plan distance

Heuristics for Task Fusion can become more accurate and achieve better planning results by incorporating an estimation of plan distance. Plan distance metrics were developed to quantify solution diversity in plan synthesis and can estimate the level of similarity between two plans [40]. Nguyen et al. [30] formulate a distance function between two plans π_i and π_j, that maps to a real-valued distance metric $\delta(\pi_i, \pi_j) : \pi_i \times \pi_j \to [0, 1]$. The action plan distance metric is defined by $1 - \frac{|A(\pi_i) \cap A(\pi_j)|}{|A(\pi_i) \cup A(\pi_j)|}$, where $A(\pi)$ is the set of actions in plan π [30]. The opposite of plan distance, plan similarity, can be approximated by $1 - \delta(\pi_i, \pi_j)$, changing the action plan distance metric to $\frac{|A(\pi_i) \cap A(\pi_j)|}{|A(\pi_i) \cup A(\pi_j)|}$. The similarity between plans can be a proxy for estimating the level of coupling between two coalition-task planning problems. Problems that produce similar plans can be considered more tightly coupled.

Plan distance heuristics can use the plan's logical objects, in addition to the plan's actions. Logical object instances are extracted from the plan actions' argument lists in order to reveal problem details that otherwise are ignored when only actions are considered. The overlap of actions and logical objects between two plans indicates the plans' level of similarity and coupling. Higher overlap of actions and logical objects indicates that the robots interact with common objects and navigate through common locations, which are represented as logical objects. The use of action *sets* makes Nguyen et al.'s heuristic unaware of repeated action instances. *Lists* allow and account for repeated actions, revealing nuances that are otherwise omitted. This manuscript introduces a family of plan distance heuristics that use lists of plan's actions and logical objects to estimate coupling and generate better planning results with Task Fusion.

The introduced plan distance heuristics consider the overlap of actions and logical object occurrences between two plans in order to estimate coupling [27]. The *Object heuristic* (O), the *Action heuristic* (A), and the *Action-Object heuristic* (AO), are based on overlaps in the action, object, and both action and object occurrences, respectively. The time at which each action is scheduled to occur is used to extend each heuristic into three temporal variants: the *Object-Temporal heuristic* (OT), the *Action-Temporal heuristic* (AT), and the *Action-Object-Temporal heuristic* (AOT).

A plan π consists of a list of actions, where each action entry contains a start time τ, robots $\Phi = \{\phi_1, \ldots\}$, and planning-model first-order logic objects $O = \{o_1, \ldots\}$. Plan distance heuristics compile a list of logical object and action occurrences, extracted from each plan action entry. Each action-object occurrence, tagged with the associated action start time, τ, populates the action-object list, $L = \{\langle l_1, \tau_1\rangle, \ldots\}$. The similarities between plans π_i and π_j result in an estimate for the utility of fusing coalition-task pairs p_i and p_j. Let π_i and π_j represent the plans for coalition-task pairs p_i and p_j, respectively. A plan distance heuristic is a function $H(\pi_i, \pi_j) : \pi_i \times \pi_j \rightarrow [0, 1]$ that maps to a utility value. Plan distance heuristics require synthesizing plans π for all m coalition-task pairs p, but can leverage the details from plans that are unavailable via the capabilities or coalition structures from the coalition formation model. The heuristics are agnostic to the origin of the plans adopted and leverage existing planners.

3.2.1. Object, action, and action-object heuristics

The *Object* (O), *Action* (A), and *Action-Object* (AO) heuristics represent the level of overlap between the logical object and action occurrences in plans π_i and π_j for coalition-task pairs p_i and p_j, respectively: $H(p_i, p_j) = \frac{1}{|L_i|\cdot|L_j|} \cdot \sum_{l_i \in L_i} \sum_{l_j \in L_j} (l_i = l_j)$, where $|L_i|$ and $|L_j|$ are list sizes for action-object lists L_i and L_j, respectively. The list elements l represent objects for the Object heuristic, actions for the Action heuristic, and both objects and actions for the Action-Object heuristic. All pairs of entries from both action-object lists are compared. Each heuristic variant populates the plan lists, L_i and L_j. The *Object heuristic* populates lists with logical object occurrences; the *Action heuristic* populates lists with action occurrences; and the *Action-Object heuristic* populates lists with both action and logical object occurrences. The normalizing fraction ensures that the heuristic values are between [0, 1], where 1 indicates maximal task coupling.

A simple first response example is provided. Assume two coalition-task pairs, p_A and p_B, have plans π_A and π_B, respectively. The $\mathit{move}(w_x, w_y)$ action moves a robot from a location w_x to a location w_y, whereas the $\mathit{triage}(v, w)$ action triages a victim v in location w. The respective plan actions are $\{\mathit{move}(w_0, w_1),\ \mathit{triage}(v_1, w_1)\}$ and $\{\mathit{move}(w_0, w_1),\ \mathit{move}(w_1, w_2),\ \mathit{triage}(v_2, w_2)\}$. The Action lists are $L_A = \{\mathit{move}, \mathit{triage}\}$ and $L_B = \{\mathit{move}, \mathit{move}, \mathit{triage}\}$, resulting in three matches and producing an Action heuristic value of $H(p_A, p_B) = 0.500$, due to the normalization factor ($|L_A| = 2$, $|L_B| = 3$, and $|L_A| \cdot |L_B| = 6$). The Object lists are $L_A = \{w_0, w_1, v_1, w_1\}$ and $L_B = \{w_0, w_1, w_1, w_2, v_2, w_2\}$, resulting in five matches and producing an Object heuristic value of $H(p_A, p_B) = 0.208$, due to the normalization factor ($|L_A| = 4$, $|L_B| = 6$, and $|L_A| \cdot |L_B| = 24$). The Action-Object lists are $L_A = \{\mathit{move}, \mathit{triage}, w_0, w_1, v_1, w_1\}$ and $L_B = \{\mathit{move}, \mathit{move}, \mathit{triage}, w_0, w_1, w_1, w_2, v_2, w_2\}$, resulting in six matches and producing an Action-Object heuristic value of $H(p_A, p_B) = 0.111$, due to the normalization factor ($|L_A| = 6$, $|L_B| = 9$, and $|L_A| \cdot |L_B| = 54$). Note that repeated entries are supported, and the lists account for higher coupling, as demonstrated by the repeated use of the action *move* by plan π_B.

3.2.2. Object-temporal, action-temporal, and action-object-temporal heuristics

The *Object-Temporal* (OT), *Action-Temporal* (AT), and *Action-Object-Temporal* (AOT) heuristics integrate temporal dependencies in order to account for action and object interactions at different times throughout the plan. Each heuristic variant populates the plan lists, L_i and L_j, with object occurrences, action occurrences, or both, as was the case in Subsubsection 3.2.1. The temporal heuristics weight each matching list entry with a decaying exponential weighting factor. The weighting ranks pairs that interact with the same objects at similar times higher than pairs that interact with the same objects at different times. The weighting factor is a function of the time difference between each matching list entry: $H(p_i, p_j) = \frac{1}{|L_i| \cdot |L_j|} \cdot \sum_{l_i \in L_i} \sum_{l_j \in L_j} (l_i = l_j) \cdot e^{-|\tau_i - \tau_j|}$, where τ_i and τ_j are temporal timestamps for list entries l_i and l_j, respectively. If $\Delta\tau = |\tau_i - \tau_j| = 0$, (i.e., the object matching occurs at the same time), the weighting factor is 1. If $\Delta\tau \to \infty$, (i.e., the object matching occurs at different times), the weighting factor is 0.

Drawing from the example in Subsubsection 3.2.1, assume the action *triage*(v_1, w_1) was scheduled to execute in plan π_A at time $\tau_i = 10$ minutes, whereas the action *triage*(v_2, w_2) was scheduled to execute in plan π_B at time $\tau_j = 12$ minutes. The time difference between the two actions is $|\tau_i - \tau_j| = 2$ minutes and the temporal weighting factor is $e^{-|\tau_i - \tau_j|} = e^{-2} = 0.607$, causing the action match contribution to be diminished by 39.3%.

Generating full plans to estimate coalition-task coupling can be prohibitively costly, and defeat the purpose of Task Fusion. However, relaxed plans can replace full plans for coalition-task coupling estimation. A relaxation of the problem model, such as to ignore actions' negative effects, can significantly reduce the computation complexity [25]. Relaxed plans offer a rough approximation of the actual plans and are used to inform forward search [25], reachability analysis [4], and distance metrics [12]. Relaxed plans can provide an estimate of the actions and the involved logical objects required by the full plan, yet require significantly less computation.

The heuristics use plan distance to estimate coupling and inform Coalition Formation. Relaxed plans allow evaluating efficiently the planning elements of coalition-task pairs before planning. Coupling across coalition-task pairs is estimated by the actions and logical objects extracted from the relaxed plans, and the most coupled coalition-task pairs are fused.

4. Empirical evaluation

The heuristics were evaluated for two different domains chosen to model the complexity of planning for multiple heterogeneous robot systems. Continuous fluents and temporal constraints allow modeling the numerical and temporal constraints necessary for each domain. Multiple robot planning problems with continuous fluents and temporal constraints are yet unavailable in existing standard planning problem benchmarks. The heterogeneous robot systems contain robots with subsets of the capabilities necessary to accomplish each task, and require robots to cooperate. The heterogeneity of robot capabilities, together with complex problems, generate tightly coupled tasks. Ten coalitions of robots and ten missions were randomly generated and combined to form 100 problems per domain. Each coalition generated ten problems, one for each mission, and each mission generated ten problems, one for each coalition. The resulting plans were evaluated based on the plan outcome, makespan, number of actions, processing time, and memory usage.

4.1. Blocks world domain

The Blocks World Domain [24] was extended [18] to require temporal constraints and continuous fluents, model a variety of end-effectors, block sizes, multiple robot arms, and incorporate two block sizes. A finite sized table holds stacks of blocks that require specific end-effectors. A specific stacking of the blocks determines the initial and goal states, which a team of robot arms seeks to achieve. Each arm has a subset of available end effectors and each block requires a specific type of end effector. Blocks can be either single- or double-weight. Single-weight blocks can be manipulated by a single arm, while double-weight blocks require two arms. Four types of end effectors were used: friction, suction, magnetic, and encompass. Ten coalitions with a minimum of four robot arms and a maximum of eight robot arms were generated. Ten missions with a minimum of eleven tasks and a maximum of 24 tasks were generated. Tighter coupled tasks have blocks that share the same blocks' pile. Each arm and grasper require different amounts of time to grasp, manipulate, and release blocks; thus, introducing durative actions. The time to stack and unstack blocks is also dependent on the arm and the block's initial and final position positions, modeled with continuous fluents.

4.2. First response domain

This first response domain [18] models disaster response problems that require coordinating heterogeneous human-robot teams. Human-robot teams cooperate to rescue victims, collect hazardous objects, clear gas leaks, and clear blocked roads after a natural disaster. Prescription drugs inside pharmacies and weapons at pawn shops must be secured to prevent looting and ensure civilian safety. Victim rescue tasks require a human to triage the victim. The resulting triage level determines how the victim is taken to a hospital, either guided by a quadrotor or transported by a rover. The pawn shop cleanup tasks require a police officer to locate, clear, secure, and load the weapons into the police robot for transport to the police base. The pharmacy cleanup tasks require personnel to locate, clear, and secure all prescription drugs, including loading the drugs into a robot for transport to a hospital. The first response domain generates a plan for both robots and humans.

The first response domain was expanded to permit more complex, but realistic problems. One extension is a model of the robot batteries that drain as a function of robot activity over time. As well, the robot load is a numerical fluent; thus, allowing robots to carry a varying number of objects, dependent on the individual robot load capacity. The number of robots, victims, pawn shops, pharmacies, road blocks, gas leaks, and waypoints was drawn from a uniform distribution. Ten coalitions with a minimum of 15 robots and a maximum of 21 robots were generated, with each coalition having a minimum of 1 robot and a maximum of 6 robots per robot type. Ten missions with a minimum of 13 tasks and a maximum of 24 tasks were generated, with each mission having a minimum of 1 task and a maximum of 15 tasks per task type. The coupling between two tasks is stronger when there is overlap between the locations the robots must traverse. Traveling across the environment and performing each task requires significantly different amounts of time, making continuous and temporal constraints critical to a plan's successful execution.

4.3. Experimental design

The experiment's independent variables are the specific planning methods: Planning Alone (PA), Coalition Formation then Planning (CFP), Task Fusion with the plan distance heuristics: Object (O), Action (A), Action-Object (AO), Object-Temporal (OT), Action-Temporal (AT), and Action-Object-Temporal (AOT), and Task Fusion with the baseline heuristics: Coalition Assistance (CA) and Coalition Similarity (CS). The planning outcomes are: Success, a valid plan is produced; Nonexecutable, no plan can be derived for the task given the coalition's composition and allocated tasks; Time Fail, the time limit is exceeded; and Memory Fail, the memory limit is exceeded. The fusion ratio, f_{max}, limits the number of fused coalition-task pairs and can impact the effectiveness of Task Fusion. Each heuristic was evaluated for fusion ratios $f_{max} = \{0.25, 0.50, 0.75, 1.00\}$, values chosen to uniformly cover the valid [0, 1] range. Each experiment's planning time was capped at one hour and memory usage was limited to 120 GB.

The dependent variables are the planning outcome, makespan, number of actions, processing time, and memory usage. Makespan represents plan length, measured in seconds [35]. The number of actions is the total number of actions required by the plan to accomplish a task [35]. The processing time, measured in minutes, is the time required to solve a problem, which includes the coalition formation, processing heuristics, planning for all tasks, and merging each task plan into a final plan. The memory usage is the maximum amount of memory allocated, in GB. The makespan and the number of actions metrics indicate plan quality. Higher quality plans have lower makespan and fewer of actions; thus, higher quality plans achieve their goals faster and require fewer actions. Plans with lower processing time and memory usage require fewer computing resources.

The TFD [20] and COLIN [11] planners support temporal constraints and continuous fluents, and were adopted for the Blocks World Domain experiment. A dynamic programming coalition formation algorithm was used [37]. COLIN [11] is the only continuous planner that accommodates the time-varying continuous fluents required for the first response domain. RACHNA [43], a market-based coalition formation algorithm, was used for the first response domain experiment. The relaxed plans were generated by a relaxed COLIN planner, which removes actions' delete effects [11]. The experiments were performed on an Intel Xeon CPU E5-1630 v4 @ 3.70 GHz $\times$ 8 workstation with 128 GB memory, running Ubuntu 14.04.5 LTS with the 4.4.0-89-generic Linux kernel. Third party Coalition Formation and Planning systems were compiled using the gcc/g++ compiler version 5.4.0.[1]

Plan distance heuristics aim to estimate coupling and provide higher quality plans requiring less processing time and memory usage. The first hypothesis (H_1) is that the effectiveness of Task Fusion is affected by the heuristics utilized. The second hypothesis (H_2) is that the object oriented plan distance heuristics: Object, Action-Object, Object-Temporal, and Action-Object-Temporal, will outperform the baselines: the Coalition Assistance and Coalition Similarity heuristics, CFP, and Planning Alone. The third and fourth hypothesis are that the object oriented plan distance heuristics will result in better quality plans (H_3) and will require lower computational cost than the baseline approaches (H_4).

4.4. Results

The results are presented by problem domain and planner. Method quality and cost are represented for multiple metrics. High quality methods minimize the plans' makespan and number of actions, while low cost methods minimize processing time and memory usage. The concepts of Pareto Dominance and Pareto Strength [47] were adopted for comparing methods across these metrics. Method t_1 *dominates* method t_2 if all of t_1's metrics' means are better than t_2's. Specifically, t_1's quality dominates t_2's quality if both t_1's mean makespan and mean number of actions are better than the mean makespan and mean number of actions for t_2. The *Pareto Strength* of a method t_i is determined by the number n of methods $t_1, t_2, \cdots, t_n$ that t_i dominates. Methods with higher *Pareto Strength* dominate many other methods.

4.4.1. The blocks world domain with TFD

The Blocks World Domain with TFD planning was characterized by a positive relationship between the evaluated metrics and the Task Fusion ratio f_{max}. Most heuristics offered increasingly better success

[1]The full source code will be made available at the time of publication. The problem set is available at https://gitlab.com/human-machine-teaming-lab-open-repositories/multi-agent-planning-and-coalition-formation/test-set.

Table 1
Blocks world with TFD planning results by method, f_{max}, and percentage for successfully generating a plan, nonexecutable coalition, and no plan generated due to time failure or memory failure

Method	f_{max}	Success	Nonexec	Time fail
Object	*0.25*	38	26	36
	0.50	33	29	38
	0.75	39	16	45
	1.00	39	16	45
Action	*0.25*	29	29	42
	0.50	32	25	43
	0.75	37	13	50
	1.00	37	13	50
Action-object	*0.25*	32	29	39
	0.50	28	28	44
	0.75	34	15	51
	1.00	34	15	51
Object-temporal	*0.25*	35	28	37
	0.50	31	28	41
	0.75	36	17	47
	1.00	36	17	47
Action-temporal	*0.25*	23	32	45
	0.50	31	23	46
	0.75	36	15	49
	1.00	35	16	49
Action-object-temporal	*0.25*	28	32	40
	0.50	30	25	45
	0.75	36	14	50
	1.00	35	15	50
Coalition similarity	*0.25*	23	33	44
	0.50	20	31	49
	0.75	27	20	53
	1.00	27	20	53
Coalition assistance	*0.25*	36	22	42
	0.50	28	23	49
	0.75	42	6	52
	1.00	41	7	52
CFP	N/A	24	34	42
Planning alone	N/A	40	0	60

rates, plan quality, and computational cost for larger f_{max} values. The improved performance saturates at high f_{max} values, with virtually equivalent results being obtained for $f_{max} = 0.75$ and 1.00 across the heuristics.

The Coalition Assistance heuristic ($f_{max} = 0.75$ and 1.00) had the best planning success rates (42% and 41%, respectively) followed by Planning Alone (40%, and the Object heuristic ($f_{max} = 0.75$ and 1.00, both 39%), as shown in Table 1. Planning Alone had the highest rate of time failures (60%), followed by the Coalition Similarity ($f_{max} = 0.75$ and 1.00, both 53%) and Coalition Assistance ($f_{max} = 0.75$ and 1.00, both 52%) heuristics. CFP produced the highest rate of nonexecutable coalitions (34%). No method exceeded the 120 GB memory limit.

Planning Alone (PA), the Object (O) and Coalition Assistance (CA) heuristics produced the highest success rates, as shown in Fig. 2a. The Object heuristic produced the second highest success rates for $f_{max} = 0.25$ and 0.50, whereas the Coalition Assistance heuristic produced the highest success rates for $f_{max} = 0.75$ and 1.00. The Coalition Assistance heuristic, however, resulted in mediocre makespan, number of actions, and memory usage results for all f_{max} values, as presented in Fig. 2b, c and e,

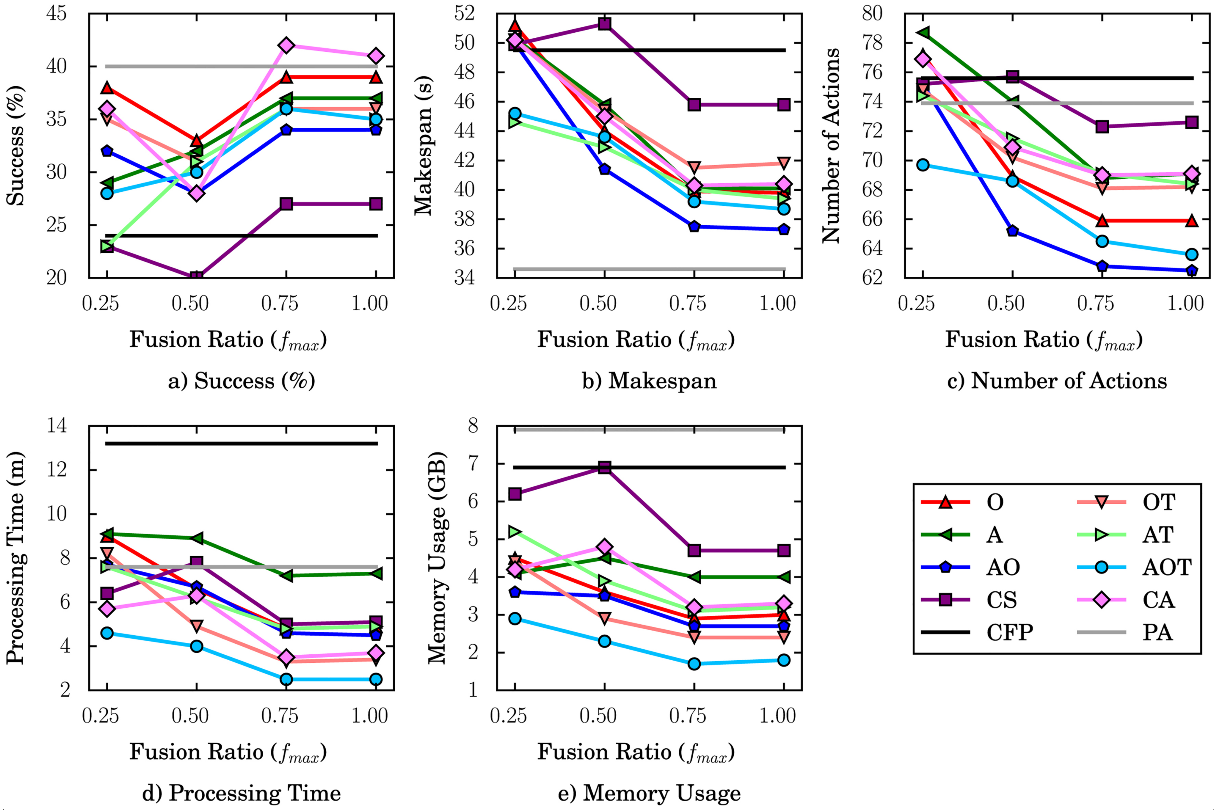


Fig. 2. Blocks world with TFD (a) success, (b) makespan, (c) number of actions, (d) processing time, and (e) memory usage by fusion ratio (f_{max}). Samples were connected to facilitate visualization.

respectively. Planning Alone resulted in the best makespan (Fig. 2b), but among the worst number of actions (Fig. 2c), processing time (Fig. 2d), and the worst memory usage (Fig. 2e).

The Action-Object (AO) heuristic produced the second best makespan and the best number of actions for $f_{max} = 0.50$ through 1.00, as presented in Fig. 2b and c, respectively. The Action-Object-Temporal (AOT) heuristic produced the second best number of actions for $f_{max} = 0.50$ through 1.00 (Fig. 2c) and resulted in the best processing time and memory usage across all f_{max} values, as shown in Fig. 2d and e, respectively. The Coalition Similarity (CS) heuristic resulted in the second worst success rates and makespan for $f_{max} = 0.75$ and 1.00 (Fig. 2a and b), and third worst memory usage for all f_{max} values (Fig. 2e). CFP produced among the worst success rates, makespan, number of actions, memory usage, and among the worst processing time.

The Pareto Strength *quality*, which minimizes plans' makespan and number of actions, was evaluated across all methods. The Action-Object heuristic ($f_{max} = 1.00$ and 0.75) produced the best and second best plan quality (Pareto Strengths 32 and 31, respectively), followed by the Action-Object-Temporal heuristic ($f_{max} = 1.00$ and 0.75), which had the third and fourth best quality (Pareto Strengths 30 and 29, respectively). The Action ($f_{max} = 0.25$), Object ($f_{max} = 0.25$), and Coalition Similarity ($f_{max} = 0.50$) heuristics produced the lowest quality (Pareto Strength 0). The Pareto Strength *cost* minimizes processing time and memory usage. The Action-Object-Temporal heuristic ($f_{max} = 0.75$ and 1.00) resulted in the two lowest costs (Pareto Strengths 33 and 32). Planning Alone (PA), CFP, and the Coalition Similarity heuristic ($f_{max} = 0.50$) had the highest cost (Pareto Strength 0).

Table 2
Blocks world with COLIN planning results

Method	f_{max}	Success	Nonexec	Time fail	Mem fail
Object	*0.25*	53	21	10	16
	0.50	48	16	13	23
	0.75	55	10	17	18
	1.00	55	10	17	18
Action	*0.25*	38	29	17	16
	0.50	41	22	13	24
	0.75	47	15	22	16
	1.00	47	15	22	16
Action-object	*0.25*	49	23	12	16
	0.50	46	19	13	22
	0.75	50	13	21	16
	1.00	50	13	20	17
Object-temporal	*0.25*	48	24	14	14
	0.50	47	16	14	23
	0.75	52	11	20	17
	1.00	52	11	19	18
Action-temporal	*0.25*	43	22	18	17
	0.50	43	16	17	24
	0.75	45	10	29	16
	1.00	45	10	29	16
Action-object-temporal	*0.25*	46	25	13	16
	0.50	45	19	11	25
	0.75	47	12	23	18
	1.00	47	12	22	19
Coalition similarity	*0.25*	37	27	17	19
	0.50	38	21	17	24
	0.75	43	16	22	19
	1.00	43	16	22	19
Coalition assistance	*0.25*	44	27	17	12
	0.50	39	25	20	16
	0.75	50	15	22	13
	1.00	49	15	23	13
CFP	N/A	38	34	12	16
Planning alone	N/A	28	0	42	30

The Action-Object-Temporal heuristic is the best solution to the Blocks World Domain with TFD, as it resulted in among the best makespan and number of actions; and the best processing time, and memory usage. The Action-Object heuristic is the second best, as it resulted in the best quality, but mediocre processing time and memory usage. CFP is the worst solution, resulting in the worst metrics.

4.4.2. The blocks world domain with COLIN

The object oriented heuristics also offered the best solution to the Blocks World Domain with the COLIN planner. The top five success rates were achieved by the plan similarity heuristics, whereas only two of the top five best success rates were plan similarity heuristics when using TFD. The Object heuristic presented better results with increasing f_{max} values.

The Object heuristic ($f_{max} = 0.75$ and 1.00) had the best success rate (both 55%), as presented in Table 2. The Object ($f_{max} = 0.25$, 53%) and Object-Temporal ($f_{max} = 0.75$ and 1.00, both 52%) heuristics were second and the third best, respectively. Planning Alone had zero nonexecutable coalitions, but had the worst success (28%), time failure (42%), and memory failure (30%) rates. CFP produced the most nonexecutable coalitions (34%).

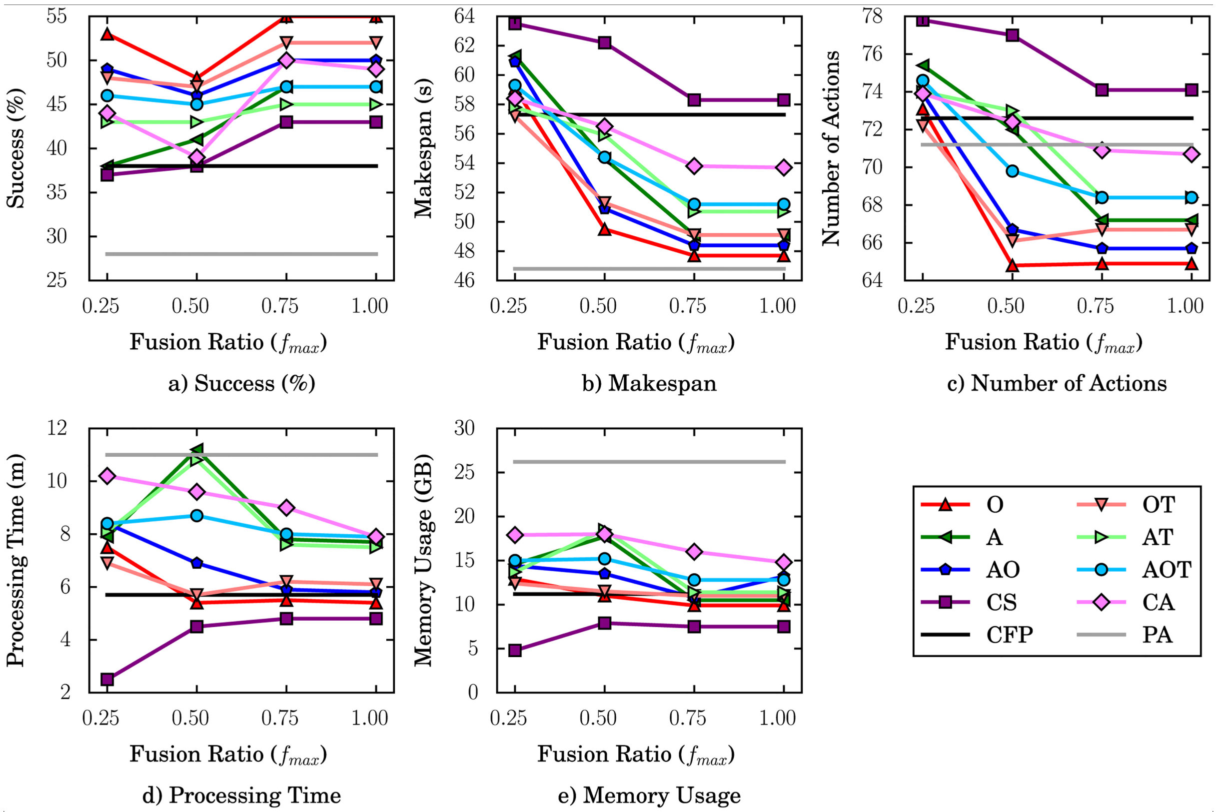


Fig. 3. Blocks world with COLIN (a) success, (b) makespan, (c) number of actions, (d) processing time, and (e) memory usage by fusion ratio (f_{max}). Samples were connected to facilitate visualization.

The Object (O) heuristic resulted in the highest success rates across all f_{max} values, as presented in Fig. 3a whereas Planning Alone (PA) produced the worst. PA produced the best makespan (Fig. 3b), but among the worst number of actions (Fig. 3c). The Object heuristic produced the second best makespan, the best number of actions, and the second best processing time and memory usage for $f_{max} = 0.50$ through 1.00, as shown in Fig. 3b–e. The Coalition Similarity (CS) heuristic generated the lowest cost (Fig. 3d and e), but also produced among the worst success rates and the worst makespan, across all f_{max} values (Fig. 3a–c).

All heuristics produced their maximum success rates for the highest f_{max} values, 0.75 and 1.00, as presented in Fig. 3a, and resulted in monotonically better makespan for larger f_{max} values, as presented in Fig. 3b, meaning that greater f_{max} values resulted in higher success rates for all f_{max} values evaluated. The Object heuristic produced monotonically better makespan, number of actions, processing time, and memory usage for greater f_{max} values, as presented in Fig. 3b–e. The Object heuristic is the best solution to the Blocks World Domain with COLIN, as it resulted in the best quality and second lowest cost across $f_{max} = 0.50$, 0.75, and 1.00.

The Object heuristic ($f_{max} = 0.75$ and 1.00) produced the best and second best plan quality (both Pareto Strength 30). The Action-Object heuristic ($f_{max} = 0.75$ and 1.00) produced the third and fourth best plan quality results (both Pareto Strength 28), followed by the Object ($f_{max} = 0.50$) and Object-Temporal ($f_{max} = 0.75$ and 1.00) heuristics (all Pareto Strength 24). The Coalition Similarity heuristic ($f_{max} = 0.25$) produced the lowest plan quality (Pareto Strength 0), while the Coalition Similarity heuristic ($f_{max} =$

0.50) was slightly better (Pareto Strength 1). The Coalition Similarity heuristic produced the three lowest cost results (Pareto Strengths 33, 31, and 30), with the best result being for the lowest f_{max} value. The Object heuristic ($f_{max} = 0.75$ and 1.00) produced the fourth lowest cost results (both Pareto Strength 27). Planning Alone and the Action heuristic ($f_{max} = 0.50$) produced the worst cost results (both Pareto Strength 0).

4.4.3. The first response domain

The more complex first response domain presented noisier results, compared to the Blocks World Domain. Planning Alone exceeded the processing time limit for all problems, resulting in no plans. The plan distance heuristics produced generally better results for intermediary f_{max} values, whereas the baseline heuristics, Coalition Similarity and Coalition Assistance, performed better for lower f_{max} values. Monotonically worsening success, makespan, number of actions, and memory usage were observed for larger f_{max} values for most baseline methods, while most plan distance heuristics presented convex curves.

The Coalition Similarity heuristic ($f_{max} = 0.25$) produced the best planning success rate (73%), as shown in Table 3. The Coalition Similarity ($f_{max} = 0.50$) and Action-Object-Temporal ($f_{max} = 0.25$) heuristics were the second best (both 66%), followed closely by the Object-Temporal heuristic ($f_{max} =$ 0.50, 65%). The Object heuristic ($f_{max} = 0.25$) produced the highest rate of nonexecutable coalitions (35%), the Coalition Assistance heuristic ($f_{max} = 1.00$) produced the highest rate of time failures (56%), and the Object heuristic ($f_{max} = 0.75$) produced the most memory failures (10%).

Many methods performed best for the intermediary f_{max} values, 0.50 and 0.75, in the first response domain, whereas most methods generated best results for the boundary f_{max} values, 0.25 and 1.00, in the Blocks World Domain. The Object (O) and Object-Temporal (OT) heuristics produced their best success rates for $f_{max} = 0.50$, as shown in Fig. 4a, and produced their best makespan, number of actions, processing time, and memory usage for $f_{max} = 0.75$, as indicated in Fig. 4b–e, respectively. The Action-Object (AO) heuristic produced its best success rate, processing time, and memory usage for $f_{max} = 0.50$, as shown in Fig. 4a, d, and e, respectively, and produced their best makespan and number of actions for $f_{max} = 0.75$, as shown in Fig. 4b and c, respectively. The success rates produced by the Action-Temporal (AT), Action-Object-Temporal (AOT), and Coalition Assistance (CA) heuristics monotonically decreased for greater f_{max} values, as presented in Fig. 4a. The Action-Object-Temporal (AOT) heuristics produced monotonically lower (better) makespan and fewer actions for larger f_{max} values, as shown in Fig. 4 (b and c, respectively). The Action-Temporal (AT) and Coalition Similarity (CS) heuristics produced monotonically worse processing times and memory usage for greater f_{max} values, as indicated in Fig. 4 (d and e, respectively). CFP resulted in among the best success rates, the best processing time and memory usage, but among the worst makespan and number of actions.

The Object heuristic ($f_{max} = 0.75$) produced the overall best plan quality (Pareto Strength 32), dominating all methods. The Object-Temporal heuristic ($f_{max} = 0.75$) had the second best plan quality (Pareto Strength 30), followed by the Action-Object ($f_{max} = 0.75$), Action-Object-Temporal ($f_{max} =$ 1.00) and Object-Temporal ($f_{max} = 1.00$) heuristics (all Pareto Strength 28). The Coalition Similarity ($f_{max} = 0.50$ and 1.00), Coalition Assistance ($f_{max} = 1.00$), and Action-Temporal ($f_{max} = 0.75$) heuristics had the lowest plan quality (all Pareto Strength 0). CFP produced the lowest cost (Pareto Strength 32), dominating all other methods. The Coalition Similarity heuristic ($f_{max} = 0.25$) had the second lowest cost (Pareto Strength 31), followed by the Object heuristic ($f_{max} = 0.75$, Pareto Strength 30).

The best performing method was the Object (O) heuristic with $f_{max} = 0.75$, which provided among the lowest success rates Fig. 4a, but the best makespan, number of actions, and processing time Fig. 4b–e. The action oriented plan distance heuristics, Action and Action-Temporal, offered mediocre results and

Table 3
First response planning results

Method	f_{max}	Success	Nonexec	Time fail	Mem fail
Object	*0.25*	56	35	8	1
	0.50	61	25	10	4
	0.75	33	24	33	10
	1.00	44	10	45	1
Action	*0.25*	51	20	29	0
	0.50	33	20	42	5
	0.75	29	20	51	0
	1.00	33	12	52	3
Action-object	*0.25*	57	30	10	3
	0.50	59	20	21	0
	0.75	42	20	34	4
	1.00	59	10	29	2
Object-temporal	*0.25*	59	24	13	4
	0.50	65	22	12	1
	0.75	43	28	23	6
	1.00	43	10	47	0
Action-temporal	*0.25*	58	30	9	3
	0.50	57	22	19	2
	0.75	44	20	36	0
	1.00	39	10	51	0
Action-object-temporal	*0.25*	66	20	11	3
	0.50	62	23	13	2
	0.75	48	17	35	0
	1.00	34	11	55	0
Coalition similarity	*0.25*	73	20	4	3
	0.50	66	16	16	2
	0.75	44	26	28	2
	1.00	47	18	32	3
Coalition assistance	*0.25*	46	23	31	0
	0.50	41	18	40	1
	0.75	31	15	54	0
	1.00	31	13	56	0
CFP	N/A	62	32	5	1

did not provide the best solution to any of the domains and planners evaluated. The Coalition Assistance heuristic with $f_{max} = 0.75$ was the worst solution, with the second lowest success rate Fig. 4a, and among the quality and cost results Fig. 4b–e.

5. Discussion

The proposed heuristics significantly outperform the baseline methods in the resulting plansąŕ quality and offers a better trade-off between quality and processing cost. While other existing methods make constraining assumptions, such as requiring serial plan execution, or requiring a specific planning algorithm, the framework is demonstrated to outperform baselines on both planning algorithms used.

The first response domain has an underlying routing problem, in that robots must travel across locations in order to perform their location-dependent tasks, such as navigating to a victim before triaging said victim. The randomly distributed victim locations result in a wide variety of complex routing problems, which is a possible cause for the larger variance across the various metrics when compared to the Blocks World Domain. Faster routes involving multiple short hops generate more actions than slower routes with

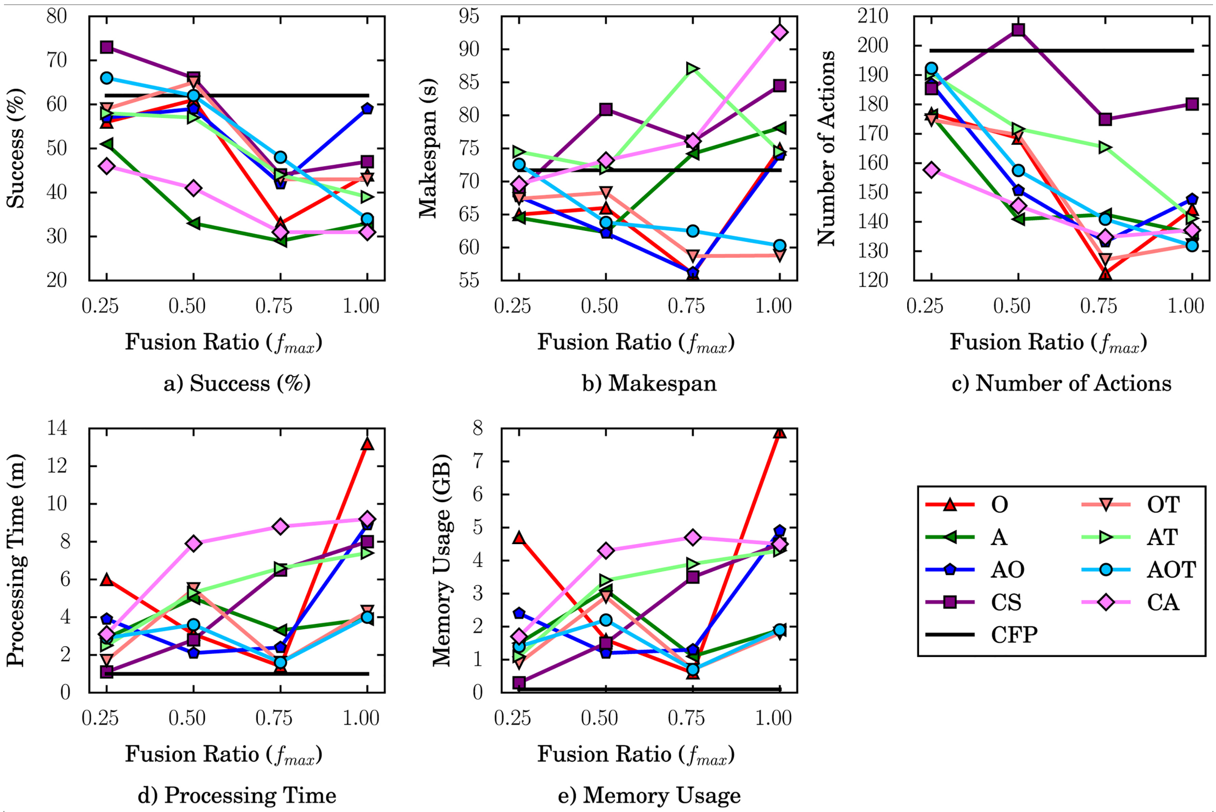


Fig. 4. First response (a) success, (b) makespan, (c) number of actions, (d) processing time, and (e) memory usage by fusion ratio (f_{max}). Samples were connected to facilitate visualization.

fewer long hops. The number of possible alternative routes across the locations' graph connecting the multiple points of interest is larger. Allocating different tasks to different robots can result in plans with a wider variety of makespans and number of actions, due to the fact that the allocated robots perform different paths to achieve their tasks, depending on where the robots were initially located. The distances traveled by robot arms are more uniform in the Blocks World Domain, because there are only two block sizes.

The hypothesis H_1, which states that the effectiveness of Task Fusion is affected by the heuristics utilized, was supported across all experiments. The heuristics had a profound impact on the Task Fusion effectiveness, with the choice of heuristic resulting in success rates ranging from the lowest to the highest. The fusion ratio, f_{max}, also impacted Task Fusion by limiting the number of fused coalition-task pairs. The lower f_{max} values attenuated the negative impacts of the bad heuristics, whereas the higher values enhanced the effectiveness of the good heuristics. The best-performing heuristics had a positive relationship with increasing f_{max}. The Action-Object-Temporal heuristic performed generally better for larger f_{max} values, whereas the Coalition Similarity heuristic performed the worst.

Hypothesis H_2, which states that the object oriented plan distance heuristics, Object, Action-Object, Object-Temporal, and Action-Object-Temporal, outperform the baselines: Coalition Assistance and Coalition Similarity heuristics, CFP, and Planning Alone, was supported. The best solution for each domain and planner was produced by object oriented plan distance heuristics, which account for logical objects common across the task plans to identify and fuse tightly coupled tasks. Task Fusion increases coalition size, which increases the search space; thus, increasing the computational costs. However, when two tightly coupled tasks are fused, the actions accomplishing one task often contribute to achieving states necessary to achieve the other task.

The third hypothesis, H_3, was supported, as the Object and Action-Object heuristics offered the best quality plans across the evaluated domains and planners. The relaxed plan logical objects provide an accurate estimate of the utility of fusing coalition-task pairs. Detecting and fusing tightly coupled coalition-task pairs facilitates planning for tasks that require explicit cooperation between robots. Robots explicitly cooperate and accomplish tasks faster when tightly coupled tasks are fused. The Coalition Assistance and Coalition Similarity heuristics fuse tasks based on coalition formation capabilities and fail to account for task planning elements, such as plan actions and logical objects. Tightly coupled tasks are planned separately and the outcomes of one task increase the planning complexity for other tasks, resulting in worse plan quality.

The final hypothesis, H_4, was not supported, as no method dominated costs across all experiments. The object oriented plan distance heuristics resulted in lower costs compared to the baseline methods for the Blocks World Domain with TFD, but were superseded by the Coalition Similarity heuristic for the same domain with COLIN. Further, the Coalition Similarity heuristics' low cost results are associated with the worst plan quality. The object oriented plan distance heuristics resulted better costs compared to the Coalition Assistance heuristic for all domains and planners, which provides some support for hypothesis H_4.

The manuscript extends and applies distance heuristics $\delta(\pi_i, \pi_j)$ as Task Fusion heuristics $H(\pi_i, \pi_j)$, related by the formula $H = 1 - \delta$. The existing action oriented plan distance heuristic [30] is extended to include plans' logical objects and use lists instead of sets to account for repeated instances. Plans often include repeated instances of actions and logical objects (i.e., the same block is handled multiple times to achieve a task in the Blocks World Domain, or the same location is visited to rescue victims in the first response domain). Accounting for the repeated instances of actions and logical objects allows more accurate coupling estimation. Tightly coupled tasks require robots to handle the same blocks and transition through the same locations more often, increasing the likelihood for dependencies and conflicts. The object oriented plan distance heuristics outperformed the action oriented plan distance heuristics across most evaluated metrics, domains, and planners. The nuances revealed by using lists of logical objects is a potential contribution to diverse planning, which needs to be evaluated as future work.

The heuristics contribute to a more informed task allocation. Coalition formation models operate on robot and task capabilities, but lack planning domain information. The plan distance heuristics use relaxed plans in order to introduce planning domain information into the task allocation process. The added planning domain information supports more accurately estimating the value of fusing coalition-task pairs and results in improved task allocation. The heuristics also contribute to estimating coupling between planning problems. Determining the exact problem coupling by computing the treewidth of the agent interaction graph is an NP-hard problem [3]. The heuristics offer an approximate alternative, which is polynomial on the number of actions and objects in the problem's plan: $O(|L_i| \cdot |L_j|)$, where $|L_i|$ and $|L_j|$ are list sizes for action-object lists L_i and L_j, respectively.

The Task Fusion algorithm considers only pair wise (binary) coalition fusion in order to avoid the complexity of evaluating all possible coalition combinations. Extending the algorithm to support n-ary fusions constitutes future research. Another future research direction is to merge and generate relaxed plans for all $\binom{m}{2}$ pairs of coalition-task pairs from the original set of m coalition-task pairs. New heuristics can compare the resulting plan quality and computational cost to the original coalition-task relaxed plans; however, several issues limit the approach. The first drawback is the combinatorial number of relaxed plans to be generated, which does not scale linearly with the number of agents, and can jeopardize overall scalability. The second limitation is that greedily minimizing each coalition-task pair's makespan and number of actions does not guarantee minimizing the makespan and the number of actions of the resulting global plan. Lastly, the processing time and the memory usage necessary to generate relaxed plans does not necessarily correlate to the computational cost necessary to generate full plans.

6. Conclusion

Plan distance heuristics were introduced to provide a better balance between plan quality and the required processing resources, when planning for multiple heterogeneous robots in complex real-world time-sensitive domains. The heuristics estimate plan distance as a proxy for estimating coalition-tasks coupling. The level of coupling determines which coalition-tasks pairs to fuse, after robots are grouped into coalitions and allocated tasks. Fusing coupled tasks improves plan quality by increasing cooperation between robots, while separating loosely coupled tasks reduces plan synthesis cost. The heuristics use lists of logical object instances, extracted from the plans' action description arguments, to reveal nuances ignored by existing Task Fusion heuristics.

The plan distance heuristics combine aspects of problem coupling and plan distance estimation to improve task allocation. The heuristics generally outperform baselines in both plan quality and computational costs. First response is an example domain that is time-critical. The small reductions in plan execution time can make the difference between mission success or mission failure. The cases in which the heuristics do not perform strictly better still offer a better balance between plan quality and computational cost. As a result, larger planning problems, which involve more tasks, robots, and logical objects, can be solved.

Acknowledgments

This work was partially supported by NSF grant #1723924.

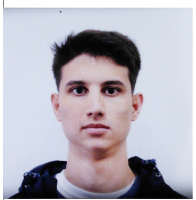
References

[1] M. Agarwal, N. Agrawal, S. Sharma, L. Vig and N. Kumar, Parallel multi-objective multi-robot coalition formation, *Expert Systems with Applications* **42**(21) (Nov 2015), 7797–7811.
[2] R. Alterovitz, S. Koenig and M. Likhachev, Robot planning in the real world: Research challenges and opportunities, *AI Magazine* **37**(2) (2016), 76–84.
[3] S. Arnborg, D.G. Corneil and A. Proskurowski, Complexity of finding embeddings in a k-tree, *SIAM Journal on Algebraic Discrete Methods* **8**(2) (Apr 1987), 277–284.
[4] D. Borrajo, Multi-agent planning by plan reuse, In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems*, May 2013, pp. 1141–1142.
[5] R.I. Brafman and C. Domshlak, From one to many: Planning for loosely coupled multi-agent systems, In *Proceedings of the International Conference on Automated Planning and Scheduling*, 2008, pp. 28–35.
[6] R.I. Brafman and C. Domshlak, On the complexity of planning for agent teams and its implications for single agent planning, *Artificial Intelligence* **198** (May 2013), 52–71.
[7] D. Bryce, S. Gao, D.J. Musliner and R.P. Goldman, SMT-based nonlinear PDDL + planning, In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2015, pp. 3247–3253.
[8] J. Buehler and M. Pagnucco, A framework for task planning in heterogeneous multi robot systems based on robot capabilities, In *Proceedings of the AAAI Conference on Artificial Intelligence*, AAAI Press, 2014, pp. 2527–2533.

[9] M. Cashmore, A.I. Coles, B. Cserna, E. Karpas, D. Magazzeni and W. Ruml, Temporal planning while the clock ticks, In *Proceedings of the International Conference on Automated Planning and Scheduling*, June 2018, pp. 39–46.
[10] A.J. Coles, A.I. Coles, M. Fox and D. Long, POPF2: A forward-chaining partial order planner, In *The International Planning Competition*, 2011, pp. 65–70.
[11] A.J. Coles, A.I. Coles, M. Fox and D. Long, COLIN: Planning with continuous linear numeric change, *Journal of Artificial Intelligence Research* **44** (May 2012), 1–96.
[12] A. Coman and H. Munoz-Avila, Generating diverse plans using quantitative and qualitative plan distance metrics, In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2011, pp. 946–951.
[13] J.S. Cox and E.H. Durfee, Efficient and distributable methods for solving the multiagent plan coordination problem, *Multiagent and Grid Systems* **5**(4) (Dec 2009), 373–408.
[14] M. Crosby, R.P.A. Petrick, F. Rovida and V. Krueger, Integrating mission and task planning in an industrial robotics framework, In *Proceedings of the International Conference on Automated Planning and Scheduling*, June 2017, pp. 471–479.
[15] M. Crosby, M. Rovatsos and R.P.A. Petrick, Automated agent decomposition for classical planning, In *Proceedings of the International Conference on Automated Planning and Scheduling*, June 2013, pp. 46–54.
[16] M. de Weerdt and B. Clement, Introduction to planning in multiagent systems, *Multiagent and Grid Systems* **5**(4) (Dec 2009), 345–355.
[17] Y. Dimopoulos, M.A. Hashmi and P. Moraitis, mu-SATPLAN: Multi-agent planning as satisfiability, *Knowledge-Based Systems*, 2012, pp. 54–62.
[18] A. Dukeman and J.A. Adams, Hybrid mission planning with coalition formation, *Journal of Autonomous Agents and Multi-Agent Systems* **31**(6) (Nov 2017), 1424–1466.
[19] K. Erol, D.S. Nau and V. Subrahmanian, On the complexity of domain-independent planning, In *Proceedings of the National Conference on Artificial Intelligence*, 1992, pp. 381–386.
[20] P. Eyerich, R. Mattmuller and G. Roger, Using the context-enhanced additive heuristic for temporal and numeric planning, In E. Prassler, R. Bischoff, W. Burgard, R. Haschke, M. Hagele, G. Lawitzky, B. Nebel, P. Ploger, U. Reiser and M. Zollner, editors, *Towards Service Robots for Everyday Environments*, Springer, 2012, pp. 49–64.
[21] B.P. Gerkey and M.J. Mataric, Sold!: Auction methods for multirobot coordination, *IEEE Transactions on Robotics and Automation* **18**(5) (Oct 2002), 758–768.
[22] M. Ghallab, D.S. Nau and P. Traverso, *Automated Planning: Theory and Practice*, Elsevier, 2004.
[23] M. Ghallab, D.S. Nau and P. Traverso, *Automated Planning and Acting*, Cambridge University Press, 2016.
[24] N. Gupta and D.S. Nau, On the complexity of blocks-world planning, *Artificial Intelligence* **56**(2–3) (1992), 223–254.
[25] J. Hoffmann, FF: The fast-forward planning system, *AI magazine* **22**(3) (2001), 57–62.
[26] S. Jimenez and A. Jonsson, Temporal planning with required concurrency using classical planning, In *Proceedings of the International Conference on Automated Planning and Scheduling*, 2015, pp. 129–137.
[27] G. Marcon dos Santos and J.A. Adams, Task fusion heuristics for coalition formation and planning, In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, Stockholm, Sweden, July 2018, pp. 2198–2200.
[28] G. Marcon dos Santos and J.A. Adams, Optimal temporal plan merging, In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, Auckland, New Zealand, May 2020.
[29] L. Mudrova, B. Lacerda and N. Hawes, Partial order temporal plan merging for mobile robot tasks, In *Proceedings of the European Conference on Artificial Intelligence*, Aug 2016, pp. 1537–1545.
[30] T.A. Nguyen, M. Do, A.E. Gerevini, I. Serina, B. Srivastava and S. Kambhampati, Generating diverse plans to handle unknown and partially known user preferences, *Artificial Intelligence* **190** (Oct 2012), 1–31.
[31] F.A. Oliehoek and C. Amato, *A concise introduction to decentralized POMDPs*, Springer, 2016.
[32] T. Pereira, N. Luis, A. Moreira, D. Borrajo, M. Veloso and S. Fernandez, Heterogeneous multi-agent planning using actuation maps, In *IEEE International Conference on Autonomous Robot Systems and Competitions*, Apr 2018, pp. 219–224. IEEE.
[33] M. Quigley, K. Conley, B.P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler and A.Y. Ng, ROS: An open-source robot operating system, In *Workshops at the IEEE International Conference on Robotics and Automation*, 2009.
[34] M.F. Rankooh and G. Ghassem-Sani, ITSAT: An efficient SAT-based temporal planner, *Journal of Artificial Intelligence Research* **53** (July 2015), 541–6.
[35] S. Russell and P. Norvig, *Artificial intelligence: A modern approach*, (3rd Edition) Pearson Education, 2009.
[36] S.D. Sen and J.A. Adams, An influence diagram based multi-criteria decision making framework for multirobot coalition formation, *Autonomous Agents and Multi-Agent Systems* **29**(6) (Nov. 2015), 1061–1090.
[37] T.C. Service and J.A. Adams, Coalition formation for task allocation: Theory and algorithms, *Autonomous Agents and Multi-Agent Systems* **22**(2) (2011), 225–248.
[38] O. Shehory and S. Kraus, Methods for task allocation via agent coalition formation, *Artificial Intelligence* **101**(1-2) (1998), 165–200.

[39] S. Sreedharan, Y. Zhang and S. Kambhampati, A first multi-agent planner for required cooperation, In *Proceedings of the Competition of Distributed and Multi-Agent Planners*, 2015, pp. 17–20.
[40] B. Srivastava, S. Kambhampati, T.A. Nguyen, M.B. Do, A. Gerevini and I. Serina, Domain independent approaches for finding diverse plans, In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2007, pp. 2016–2022.
[41] A. Torralba, V. Alcazar, D. Borrajo, P. Kissmann and S. Edelkamp, SymBA*: A symbolic bidirectional A* planner, In *The International Planning Competition*, 2014, pp. 105–108.
[42] A. Torreno, E. Onaindia, A. Komenda and M. Stolba, Cooperative multi-agent planning, *ACM Computing Surveys* **50**(6) (Nov 2017), 1–32.
[43] L. Vig and J.A. Adams, Market-based multi-robot coalition formation, In M. Gini and R. Voyles, editors, *Distributed Autonomous Robotic Systems 7*, 2006, pp. 227–236. Springer.
[44] L. Vig and J.A. Adams, Multi-robot coalition formation, *IEEE Transactions on Robotics* **22**(4) (2006), 637–649.
[45] M. Wooldridge and N. Jennings, Intelligent agents: Theory and practice, *Knowledge Engineering Review* **10**(2) (1995), 115–152.
[46] Y. Zhang, S. Sreedharan and S. Kambhampati, Capability models and their applications in planning, In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, May 2015, pp. 1151–1159.
[47] E. Zitzler and L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, *IEEE Transactions on Evolutionary Computation* **3**(4) (1999), 257–271.

Authors' Bios



Gilberto Marcon dos Santos is a PhD candidate at the Collaborative Robotics and Intelligent Systems (CoRIS) Institute, Oregon State University (OSU). His current research applies coordination algorithms to improve domain-independent planning and solve multi-robot problems. He graduated in Brazil from the Universidade Federal de Goias in 2016 with a BS in Computer Engineering and an MS in Electrical and Computer Engineering.

Dr. Julie A. Adams is the Associate Director of Research in the Collaborative Robotics and Intelligent Systems Institute, Professor of Computer Science in the School of Electrical Engineering and Computer Science, and Professor (courtesy) of Mechanical and Industrial Engineering in the School of Mechanical, Industrial and Manufacturing Engineering at Oregon State University. She was the founder of the Human-Machine Teaming Laboratory at Vanderbilt University, prior to moving the laboratory to Oregon State. Dr. Adams received her M.S. and Ph.D. degrees in Computer and Information Sciences from the University of Pennsylvania and her B.S. in Computer Science and B.B.E. in Accounting from Siena College.