The International Journal of
Robotics Research

A comprehensive taxonomy for multi-robot task allocation

G. Ayorkor Korsah, Anthony Stentz and M. Bernardine Dias

Published by:

$SAGE

http://www.sagepublications.com

On behalf of:

Multimedia Archives

Additional services and information for *The International Journal of Robotics Research* can be found at:

Email Alerts: http://ijr.sagepub.com/cgi/alerts

Subscriptions: http://ijr.sagepub.com/subscriptions

Reprints: http://www.sagepub.com/journalsReprints.nav

Permissions: http://www.sagepub.com/journalsPermissions.nav

Citations: http://ijr.sagepub.com/content/32/12/1495.refs.html

>> Version of Record - Oct 22, 2013

What is This?

A comprehensive taxonomy for multi-robot task allocation

G. Ayorkor Korsah[1,2], Anthony Stentz[2] and M. Bernardine Dias[2]

Abstract

Task allocation is an important aspect of many multi-robot systems. The features and complexity of multi-robot task allocation (MRTA) problems are dictated by the requirements of the particular domain under consideration. These problems can range from those involving instantaneous distribution of simple, independent tasks among members of a homogenous team, to those requiring the time-extended scheduling of complex interrelated multi-step tasks for members of a heterogenous team related by several constraints. The existing widely used taxonomy for task allocation in multi-robot systems was designed for problems with independent tasks and does not deal with problems with interrelated utilities and constraints. While that taxonomy was a ground-breaking contribution to the MRTA literature, a survey of recent work in MRTA reveals that it is no longer a sufficient taxonomy, due to the increasing importance of interrelated utilities and constraints in realistic MRTA problems under consideration. Thus, in this paper, we present a new, comprehensive taxonomy, iTax, that explicitly takes into consideration the issues of interrelated utilities and constraints. Our taxonomy maps categories of MRTA problems to existing mathematical models from combinatorial optimization and operations research, and hence draws important parallels between robotics and these fields.

Keywords

multi-robot coordination, task allocation, taxonomy

1. Introduction

Task allocation in a multi-robot system is the problem of determining which robots should execute which tasks in order to achieve the overall system goals. Its purpose is coordinated team behaviour. In some systems, such as some biologically inspired robotic systems, coordinated team behavior emerges as a result of local interactions between members of a team and with the environment. This is referred to as implicit or *emergent* (Gerkey, 2003) coordination. We are interested instead in explicit or *intentional* (Parker, 1998) cooperation in which tasks are explicitly assigned to a robot or sub-team of robots, a problem described as multi-robot task allocation (MRTA).

MRTA problems of various forms are the subject of a growing body of research. To help organize this work and identify the theoretical foundations of what they describe as largely *ad hoc* approaches, Gerkey and Matarić (2004) proposed a taxonomy for MRTA problems. This taxonomy, which is now widely used, provides a common vocabulary for describing MRTA problems. It is, however, limited in scope. It is described by its authors as restricted to systems with independent tasks, and as such excludes a large collection of problems in the widely growing body of multi-robot coordination work in which there are interrelated task

utilities and constraints. For example, the basic multi-robot routing problem (Lagoudakis et al., 2005) in which a team of robots visits a set of locations with routes that optimize criteria such as travel distance or time, is outside the space covered by Gerkey and Matarić's taxonomy. Also excluded are many more complicated problems involving constraints between tasks and/or robots.

In this paper, which is based on part of a doctoral dissertation (Korsah, 2011), we propose a more complete taxonomy, named *iTax*, that explicitly handles the issues of interrelated utilities and constraints and as such is applicable to a much larger space of important task allocation problems. In describing each class in our taxonomy, we give examples of existing work in the MRTA literature addressing problems in that class. The descriptions also indicate the computational complexity of problems in each class, and identify well-known problems and mathematical models from the

[1] Ashesi University College, Berekuso, Eastern Region, Ghana
[2] Carnegie Mellon University Pittsburgh, PA, USA

Corresponding author:
G. Ayorkor Korsah, Ashesi University College, PMB CT3, Cantonments, Accra, Ghana.
Email: ayorkor@alumni.cmu.edu

combinatorial optimization and operations research literature that exemplify the problem class. The goal in doing this is to point out relationships between similar problems addressed in different fields. This serves to identify mathematical models that apply to these problems and, thus, could potentially be useful in the analysis of solution approaches in robotics. Thus, the key contributions of this paper are (i) a comprehensive taxonomy of MRTA problems that highlights a key dimension, namely the degree of interrelatedness of agent–task utilities and constraints, that differentiates these problems into difficulty classes, and (ii) an identification of relevant mathematical models, where they exist, exemplifying each problem class.

In this work, we are concerned with teams that include robots but may optionally include humans or non-robotic vehicles. We consider these collectively to be *embodied mobile agents*, but shall simply refer to them as *agents*. For consistency, we shall not, however, change Gerkey and Matarić's acronyms referring to *single-robot* (SR) tasks and *multi-robot* (MR) tasks, with the understanding that the term *robot* in this context generalizes to the embodied mobile agents under consideration in this work.

The rest of this paper is organized as follows. We will first summarize, in Section 2, the existing taxonomy proposed by Gerkey and Matarić. Section 3 presents relevant concepts and terminology for the new taxonomy, the high-level features of which are presented in Section 4. Section 5 gives detailed descriptions of each class in the taxonomy. Finally, Section 6 summarizes the paper.

2. Background: Gerkey and Matarić's taxonomy

Gerkey and Matarić (2004) categorize MRTA problems along three axes. The first axis, single-task (ST) robots versus multi-task (MT) robots, distinguishes between problems in which each robot can execute only one task at a time and problems in which some robots can execute multiple tasks simultaneously. The second axis, SR tasks versus MR tasks, distinguishes between problems in which each task requires exactly one robot to achieve it and problems in which some tasks may require multiple robots. The third axis, instantaneous assignment (IA) versus time-extended assignment (TA), distinguishes between problems concerned with instantaneous allocation of tasks to robots with no planning for future allocations and problems concerned with both current and future allocations, meaning that each robot is allocated several tasks which must be executed according to a given schedule. This taxonomy is illustrated in Figure 1.

In presenting their taxonomy for MRTA problems, Gerkey and Matarić point out that the ST-SR-IA problem is an instance of the optimal assignment problem in combinatorial optimization and is the only problem in this space that can be solved in polynomial time. The remaining problems are all strongly NP-hard. They describe the
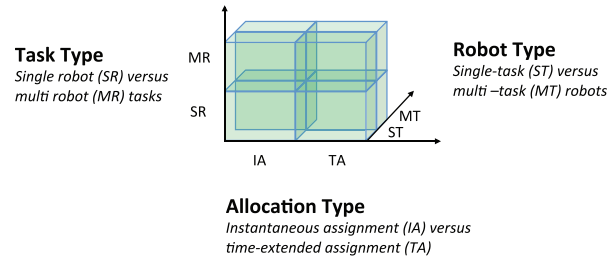

Task Type
Single robot (SR) versus multi robot (MR) tasks

Robot Type
Single-task (ST) versus multi –task (MT) robots

Allocation Type
Instantaneous assignment (IA) versus time-extended assignment (TA)

Fig. 1. Visual representation of the three axes of Gerkey and Matarić's taxonomy.

ST-SR-TA problem, which involves determining a schedule of tasks for each robot, as an instance of a machine scheduling problem. The ST-MR-IA problem is significantly harder and is also referred to as *coalition formation*. Expressed as the problem of dividing or partitioning the set of robots into non-overlapping sub-teams to perform the given tasks, this problem is mathematically equivalent to the well-known *set-partitioning problem* in combinatorial optimization. They explain that the less-common MT-SR-IA problem is mathematically equivalent to the ST-MR-IA problem, with the roles of tasks and robots reversed. The ST-MR-TA problem involves both coalition-formation and scheduling. It is mathematically equivalent to the less common MT-SR-TA problem. In the MT-MR-IA problem, the goal is to try to compute a coalition of robots to perform each task, where a given robot may be simultaneously assigned to more than one coalition (that is, a robot may work on more than one task). This problem can be expressed as an instance of the *set-covering problem* in combinatorial optimization. It is distinguished from the set-partitioning problem in that the subsets of robots need not be disjoint. Finally, Gerkey and Matarić assert that the MT-MR-TA problem is an extremely difficult problem that can be thought of as an instance of a scheduling problem with multiprocessor tasks and multipurpose machines. (We will, however, explain in Section 5.3 why we disagree with this analogy.)

While it was a ground-breaking contribution to the MR literature, Gerkey and Matarić explain that a key limitation of their taxonomy is that it does not capture problems with interrelated utilities and task constraints. For example, notably excluded are multi-robot routing problems which can be modeled as multiple traveling salesman problems (m-TSP), in which the robots have to visit multiple locations to perform spatially distributed tasks, and the utility function is related to routing costs. In such domains, there are synergies between tasks that are close together, and the total utility to a robot that performs these clustered tasks is not equal to the sum of its utilities for performing them individually. Such problems are common in robotics and so it is essential to develop a taxonomy that includes them. Before presenting the new task allocation taxonomy, we discuss several relevant concepts in the next section.

3. Relevant concepts and terminology

3.1. Tasks and task decomposition

We distinguish between various types of tasks that can be performed by agents. Intuitively, some tasks comprise a single action that can be performed by a single agent and these are described as *elemental* or *atomic* tasks. Other tasks can be broken up or *decomposed* into multiple steps or subtasks, and these are referred to as *compound* tasks, provided that there is a single fixed way of decomposing the task into subtasks. Different parts of a compound task may be allocated to different agents. Alternatively, the different parts of a compound task may need to be performed by the same agent, in which case it is described as a *decomposable simple task*. Finally, a *complex* task is one for which there are multiple possible ways of decomposing the task, and which can be allocated to multiple agents. These task types are illustrated in Figure 2. More formally, we adopt the following terminology proposed by Zlot (2006):

Decomposition and decomposability. A task t is *decomposable* if it can be represented as a set of subtasks σ_t for which satisfying some specified combination (ρ_t) of subtasks in σ_t satisfies t. The combination of subtasks that satisfy t can be represented by a set of relationships ρ, that may include constraints between subtasks or rules about which or how many subtasks are required. The pair (σ_t,ρ_t) is also called a *decomposition* of t. The term *decomposition* can also be used to refer to the process of decomposing a task.

Multiple decomposability. A task t is *multiply decomposable* if there is more than one possible decomposition of t.

Elemental task. An *elemental* (or *atomic*) task is a task that is not decomposable.

Decomposable simple task. A decomposable simple task is a task that can be decomposed into elemental or decomposable simple subtasks, provided that there exists no decomposition of the task that is multi-[agent-]allocatable.

Simple task. A simple task is either an elemental task or a decomposable simple task.

Compound task. A *compound task* t is a task that can be decomposed into a set of simple or compound subtasks with the requirement that there is exactly one fixed full decomposition for t (i.e., a compound task may not have any multiply decomposable tasks at any decomposition step).

Complex task. A *complex task* is a multiply decomposable task for which there exists at least one decomposition that is a set of multi-[agent-]-allocatable subtasks. Each subtask in a complex task's decomposition may be simple, compound, or complex.

From these definitions, it can be seen that a key difference between compound and complex tasks is that the optimal decomposition for compound tasks can be determined *prior* to task allocation, whereas for complex tasks, it is not known prior to task allocation which of the possible decompositions is optimal. Thus, a complete algorithm for allocating compound tasks can optimally decompose these into simple tasks prior to task allocation whereas a complete algorithm for allocating complex tasks would need to explore the various possible task decompositions concurrently with task allocation. In addition to answering the basic task allocation question of *"who does what?"*, an algorithm for allocating complex tasks also needs to answer the question *"which simple tasks should be executed (or which decomposition should be used)?"*. The space of possible allocations for a multi-agent task allocation problem with simple or compound tasks is exponential in the number of agents and tasks. The space of possible allocations for the same problem with complex tasks is exponentially larger than this (Zlot, 2006).

3.2. Constraints

Constraints in a task allocation problem are potentially arbitrary functions that restrict the space of feasible solutions to the problem. For example, capability constraints may define which robots are capable of performing which tasks. Capacity constraints can define how many tasks a given robot can perform at a time. Simultaneity constraints can specify that two tasks must be performed at the same time, while non-overlapping constraints may specify that they must *not* be performed at the same time, and precedence constraints may specify that one task must be performed before another. In problems where tasks have a choice of locations at which they can be performed, proximity constraints may specify that two tasks must be performed less (or greater) than a specified distance from each other.

3.3. Relationship between task decomposition and inter-task constraints

For compound tasks, task allocation can be preceded by task decomposition, during which a compound task is broken up into several simple tasks. To be equivalent to the original compound task, these simple tasks might need to be related by constraints such as simultaneity or precedence constraints. The simple tasks might be allocated to different robots, but the constraints between the tasks ensure that the robots work together appropriately. Thus, there is a close relationship between the issue of task decomposition and the issue of inter-task constraints: a problem with independent compound tasks, unrelated by constraints, may be equivalent to a problem with simple tasks related by inter-task constraints. Thus, some problems can be expressed in multiple ways. The taxonomy presented in this paper naturally accommodates this observation by focussing primarily on the degree of interrelatedness of agent–task utilities
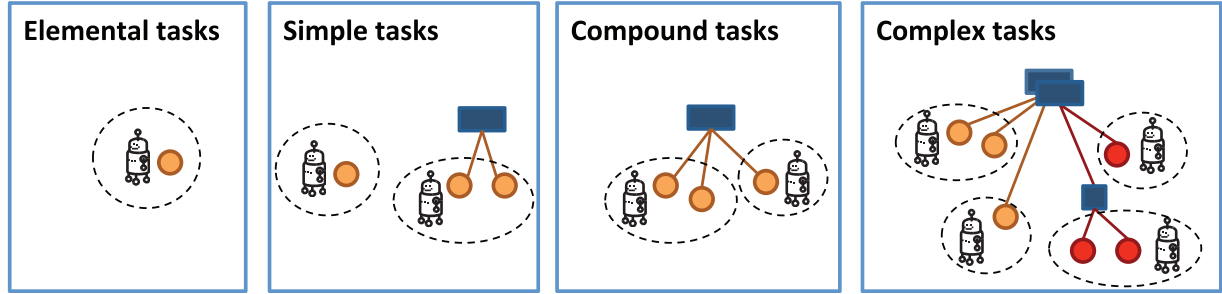
Fig. 2. Illustration of Zlot's task types. Dotted circles indicate examples of potential valid allocations of tasks to robots. Shaded circles represent elemental tasks while shaded rectangles represent decomposable tasks, whose decomposition into elemental tasks is illustrated by a tree-like structure. The superimposed trees in the rightmost figure illustrates multiple possible ways of decomposing the example complex task.

rather than the structure of how the problem is expressed. In this case, our example problem, regardless of which formulation is used, will be put in the high-level category of problems with cross-schedule dependencies.

Although some problems might explicitly deal with complex tasks, as in Zlot's work (Zlot, 2006), there are other problems for which complex tasks might exist implicitly. Consider a problem with a set of simple tasks that are related by constraints, such that there is a choice of which constraints should be satisfied, and part of the problem is to determine which constraints should be satisfied, in order to optimize the overall solution. For example, task A may need to be preceded either by tasks B_1 and B_2 or by tasks C_1, C_2, and C_3. Each of these potential pre-requisite tasks may be performed by a different agent. In a process opposite to task decomposition, we can compose these simple tasks into a complex task, $\mathcal{A}$, with two possible decompositions. One decomposition comprises tasks B_1 and B_2 followed by task A, and the other decomposition comprises tasks C_1, C_2, and C_3, followed by task A. Thus, although the complex task was not explicitly defined in the problem definition, we can see that a complex task implicitly exists.

3.4. Utility

As an optimization problem, task allocation seeks to determine a feasible assignment of tasks to agents that optimizes some objective, which can be described as a utility function. Here, we adapt Gerkey and Matarić's (2004) definition of the utility of an agent for a task to allow both positive and negative utilities.

Given a robot r and a task t, if r is capable of executing t, then one can define, on some standardized scale, Q_{rt} and C_{rt} as the quality and cost, respectively, expected to result from the execution of t by r. This results in a combined, utility measure:

$$U_{rt} = \begin{cases} Q_{rt} - C_{rt} & \text{if } r \text{ is capable of executing } t \\ -\infty & \text{otherwise} \end{cases}$$

For some problems, an agent's utility for performing a task is independent of its utility for performing any other task. In other problems, this is not true. Consider, for example, a problem where there are a number of items or "treasures" scattered in the environment, and there are a number of robots at different starting locations in the environment. The team of robots is tasked with collecting each treasure in the environment and bringing it back to the starting location of the robot that picks up the treasure. Suppose the robots are identical and can each carry one treasure at a time. For this scenario we could define Q_{rt} as a fixed reward for each treasure that is picked up, and C_{rt} as a cost proportional to the distance from a robot's starting location to a task location and back again. Because a robot can carry only one treasure at a time, it must return to its starting location after every pick-up. Thus, the utility of the robot for performing a given task is independent of all other agent–task utilities. Taking all of the agent–task utilities into consideration, the global optimal solution to this task allocation problem would allocate each task to its closest robot. Suppose, however, that each robot is capable of carrying multiple treasures at a time. Suppose further that the distance between two particular treasures, T_1 and T_2, is smaller than the distance from either treasure to the starting location of robot R_1. In this case, the robot R_1's cost to pick up T_1 will be less if it is already assigned to pick up T_2 than if it is not. This is because it can travel directly from the location of T_2 to the location of T_1. Thus, the utilities of R_1 for tasks T_1 and T_2 are not independent.

To formalize this notion of interrelated utilities, and thus lay the groundwork for our new taxonomy, we can generalize the above definition of utility to encompass not only single agents and tasks, but also subsets of agents and tasks. Let $\mathcal{R}$ represent a subset of agents in the team, such that $|\mathcal{R}| \geq 1$, and similarly $\mathcal{T}$ represent a subset of tasks in the problem such that $|\mathcal{T}| \geq 1$. We can then define a utility measure for a subteam of agents and a subset of tasks:

$$U_{\mathcal{R}\mathcal{T}} = \begin{cases} Q_{\mathcal{R}\mathcal{T}} - C_{\mathcal{R}\mathcal{T}} & \text{if subteam } \mathcal{R} \text{ is capable of} \\ & \text{executing task subset } \mathcal{T} \\ -\infty & \text{otherwise} \end{cases}$$

(1)

Furthermore, for each subteam of agents, $\mathcal{R}$ and subset of tasks, $\mathcal{T}$, we can implicitly define an *effective utility*, ${}^{e}U_{rt}^{RT}$, for an agent $r \in \mathcal{R}$ and task $t \in \mathcal{T}$ such that

$$U_{\mathcal{R}\mathcal{T}} = \sum_{r \in \mathcal{R}} \sum_{t \in \mathcal{T}} {}^{e}U_{rt}^{RT} \qquad (2)$$

We can then indicate that for a problem with *independent* utilities,

$$U_{\mathcal{R}\mathcal{T}} = \sum_{r \in \mathcal{R}} \sum_{t \in \mathcal{T}} U_{rt} \quad \text{or} \quad {}^{e}U_{rt}^{RT} = U_{rt} \qquad (3)$$

And for a problem with *interrelated* utilities,

$$U_{\mathcal{R}\mathcal{T}} \neq \sum_{r \in \mathcal{R}} \sum_{t \in \mathcal{T}} U_{rt} \quad \text{or} \quad {}^{e}U_{rt}^{RT} \neq U_{rt} \qquad (4)$$

If the subset of agents and the subset of tasks have a synergistic relationship, then

$$U_{\mathcal{R}\mathcal{T}} > \sum_{r \in \mathcal{R}} \sum_{t \in \mathcal{T}} U_{rt} \quad \text{or} \quad {}^{e}U_{rt}^{RT} > U_{rt} \qquad (5)$$

It should be noted that task allocation problems are not always stated as utility maximization problems: they are also commonly expressed as cost minimization problems, which do not attempt to combine quality and cost on a standardized scale. A cost minimization problem can be converted to a utility maximization problem with some adjustments regarding whether all tasks must be performed or whether the solution can determine that some tasks are too costly to perform. As such, the results discussed in this paper apply equally to task allocation problems formulated as cost minimization problems.

3.5. Relationship between utilities and constraints

Utilities can be thought of as real-valued functions of relevant problem features, whereas constraints are binary-valued functions of relevant problem features. They are thus related, although not identical concepts. Interrelated utilities and constraints both have an impact on the degree of interdependence between the agents as well as tasks in a problem. Our proposed taxonomy explicitly considers this degree of interdependence.

4. iTax: a taxonomy addressing interrelated utilities and constraints

We propose a new MRTA taxonomy called iTax, which is based on the recognition that a key distinguishing factor between different types of MRTA problems is the degree of interdependence of agent–task utilities in the problem. Based on the definitions in Section 3, and recalling that MRTA involves the combined problems of task decomposition, task allocation, and scheduling, we realize that

the degree of interdependence of agent–task utilities determines how coupled the MRTA problem is, and is a strong determining factor of problem difficulty.

We thus propose a two-level taxonomy in which the first, primary, level comprises a single dimension defining the degree of interdependence of agent–task utilities. The second level optionally provides further descriptive information about the problem configuration, utilizing Gerkey and Mataric's taxonomy. We represent the degree of interdependence with a single categorical variable with four possible values, listed below and illustrated in Figure 3.

No Dependencies (ND). These are task allocation problems with simple or compound tasks that have **independent** agent–task utilities. That is, the effective utility of an agent for a task does not depend on any other tasks or agents in the system. In these problems, the task decomposition problem is decoupled from the task allocation problem, and there is no schedule optimization problem since the order in which an agent performs its assigned tasks does not affect its utility. We will see that problems in this class can be cast as linear assignment problems (Votaw and Orden, 1952) and are solvable in polynomial time.

In-schedule Dependencies (ID). These are task allocation problems with simple or compound tasks for which the agent–task utilities have *intra*-schedule dependencies. That is, the effective utility of an agent for a task depends on what other tasks that agent is performing. Constraints may exist between tasks on a single agent's schedule, or might affect the overall schedule of the agent. In these problems, the task decomposition problem is decoupled from the task allocation problem, and the scheduling optimization problems of individual agents are decoupled from each other. We will see that problems in this class are NP-hard, as are the problems in the remaining classes.

Cross-schedule Dependencies (XD). These are task allocation problems with simple or compound tasks for which the agent–task utilities have *inter*-schedule dependencies (in addition to any in-schedule dependencies). That is, the effective utility of an agent for a task depends not only on its own schedule but also on the schedules of other agents in the system. For this class, allowable dependencies are "simple" dependencies in that the task decomposition can be optimally pre-determined prior to task allocation. In other words, task decomposition is decoupled from task allocation. Constraints may exist between the schedules of different agents; that is, the schedule optimization problems of the individual agents cannot be decoupled from each other.

Complex Dependencies (CD). These are task allocation problems for which the agent–task utilities have *inter*-schedule dependencies for *complex* tasks (in addition to any in-schedule and cross-schedule dependencies for simple or compound tasks). That is, the effective utility of an agent for a task depends on the schedules of other agents in the system in a manner that is determined by the particular task decomposition that is ultimately chosen. Thus,
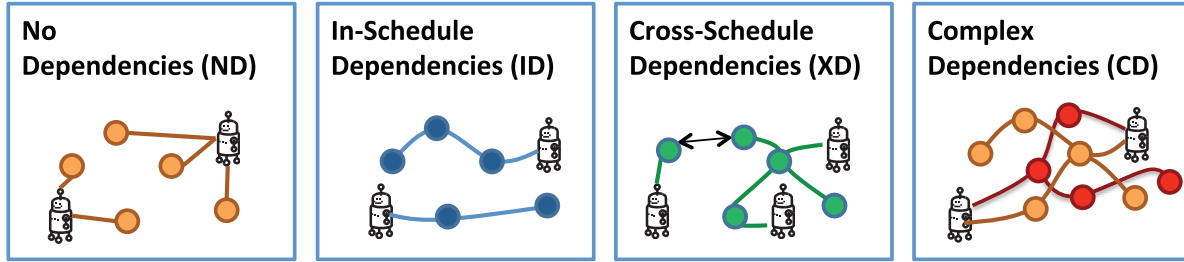
Fig. 3. Examples illustrating the four high-level categories of the new taxonomy. Shaded circles represent tasks and solid lines represent agent routes. Arrows between tasks indicate constraints. The superimposed routes in the rightmost figure illustrate multiple possible task decompositions.
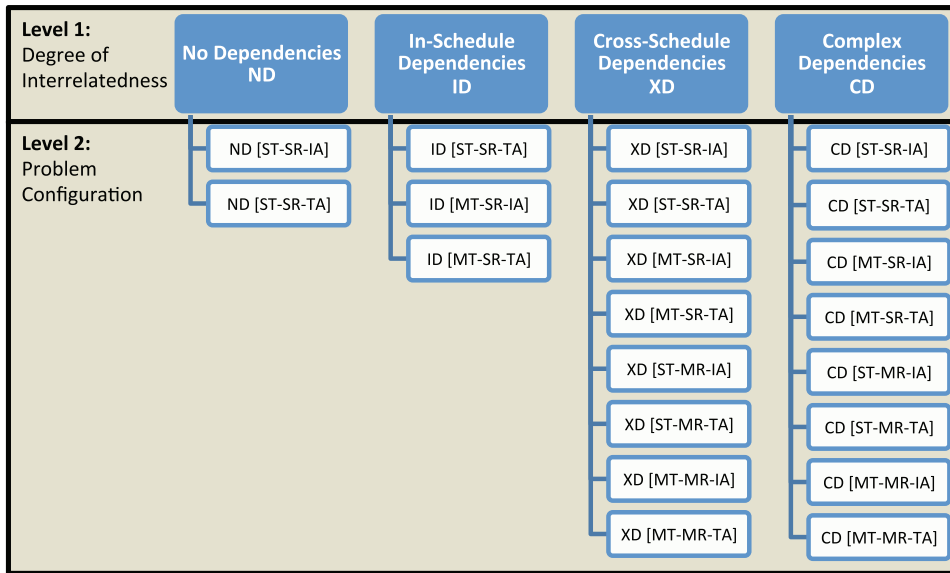

Fig. 4. iTax: a two-level task allocation taxonomy.

the task decomposition problem is coupled with the task allocation problem: the optimal task decomposition must be determined concurrently with task allocation. Furthermore, constraints may exist between the schedules of different agents; that is, the schedule optimization problems of the individual agents also cannot be decoupled from each other.

Figure 4 illustrates the overall two-level taxonomy, which is described in detail in the next section. Level 1 of this taxonomy comprises the four categories above. For a finer-grained classification, this level 1 designation can be optionally followed, in square braces, by a level 2 designation given by Gerkey and Matarić's taxonomy. For example the label **XD [ST-SR-TA]** refers to the category of problems with cross-schedule dependencies (XD) and for which we need to compute a time-extended assignment (TA) of single-agent tasks (SR) to single-task agents (ST). Figure 4 illustrates that the proposed taxonomy does not contain subcategories corresponding to the full cross-product between the level 1 and level 2 designations. Rather, some of the potential subcategories are not meaningful and thus not

included in the new taxonomy, while other subcategories are very rare. For example, in the category of problems with completely independent agent–task utilities according to our definition (the ND class), the key subcategories are the ST-SR-IA and ST-SR-TA subcategories. Most problems with multi-task robots (MT) and/or multi-robot tasks (MR) have some form of interrelated utilities and so fall under one or more of the ID, XD and CD classes in our new taxonomy.

5. Details of the proposed taxonomy

We now elaborate on each class in the taxonomy, highlighting relevant mathematical models as well as example MRTA solution approaches.

5.1. No Dependencies class (ND)

For problems in the ND class, the effective utility of an agent for a task depends only on the agent and the task.

Any constraints in the problem can involve a single agent, a single task, or a single agent–task pair. A common example in this class is a problem in which the utility function is based on agent capabilities or proximity to a task. Most problems in the ND class have single-task agents (ST) and single-agent tasks (SR). Problems with multi-task agents (MT) can only be included in this class if there are no limits on how many tasks (or on which particular tasks) an agent can execute simultaneously. In most cases, even if an agent can execute multiple tasks at once, its capabilities and resources will place limits on how many tasks, or which tasks, it can execute simultaneously. Thus, the agent's effective utility for a given task will depend on what other tasks are also assigned to it. There would be, as such, in-schedule dependencies, placing the problem in the ID class. Similarly, problems with multi-agent tasks (MR) also cannot in general be included in this class because if a task requires multiple agents, then the effective utility of a given agent for that task depends on what other agents are assigned to it. MR tasks thus, in general, give rise to cross-schedule dependencies, placing such problems in the XD class. In discussing the ND class of problems, we thus focus our attention on the two subcategories ND [ST-SR-IA] and ND [ST-SR-TA].

5.1.1. ND [ST-SR-IA] *Mathematical model*.

The ND [ST-SR-IA] subcategory of problems captures the one-to-one assignment of independent single-agent tasks to independent single-task agents. As described previously (Gerkey and Matarić, 2004), it can be represented by the linear assignment problem (Votaw and Orden, 1952) from the combinatorial optimization literature:

Maximize

$$\sum_{i \in N} \sum_{j \in M} u_{ij} x_{ij} \qquad (6)$$

Subject to:

$$\begin{aligned}
\sum_{i \in N} x_{ij} &= 1 &\forall i \in N \\
\sum_{j \in M} x_{ij} &= 1 &\forall j \in M \\
x_{ij} &\in \{0, 1\}
\end{aligned} \qquad (7)$$

where N is the set of agents, and M is the set of tasks.

The linear assignment problem can be solved in polynomial time with algorithms such as the Hungarian algorithm (Kuhn, 1955). For a feasible solution to this problem, the number of agents, $|N|$, must be equal to the number of tasks, $|M|$. An imbalance in the number of robots and tasks can be fixed by including "dummy" agents or tasks as needed. These dummy agents (or tasks) must have very low utility values with respect to all tasks (or agents) in the system. Furthermore, the utility values, u_{ij}, can be defined so as to accommodate agent–task constraints such as capability

constraints. For example, if an agent is not capable of performing a task, it can be assigned a large negative utility for that task.

MRTA solution approaches. Several approaches to MRTA address the ND [ST-SR-IA] problem. A few examples are work by Vail and Veloso (2003) using potential fields, Gerkey and Matarić (2002) using auction methods, and Simmons et al. (2000) also using auctions. In their paper presenting the first MRTA taxonomy (Gerkey and Matarić, 2004), Gerkey and Matarić point out that several multi-robot systems, particularly in the robot soccer domain, solve an *iterated assignment* variant of this problem: that is, a problem in which the instantaneous assignment problem must be repeatedly solved at some frequency. In robot soccer, this enables the roles of team members to be dynamically determined as the game progresses. Gerkey and Matarić (2004) present a detailed discussion of ST-SR-IA problems with no dependencies, so we will not elaborate further on this class.

5.1.2. ND [ST-SR-TA]

In the time extended version of the problem, each robot can be assigned more than one task, and a time-extended schedule of tasks must be built for each robot. This may be because there are more known tasks than robots, or simply to allow solutions where some robots perform multiple tasks while others do nothing. Because there are no in-schedule dependencies, the order in which a given agent performs its assigned tasks does not affect the overall utility or objective function. The example discussed earlier in which robots need to pick up several treasures, returning to their starting locations after picking up each item and assuming there is no time deadline for task execution, falls into the ND [ST-SR-TA] category. The version of the problem in which the robots can carry multiple items at a time and so need not return to their starting locations after picking up each item, does not fall in this category because of the existence of in-schedule dependencies.

Mathematical model. Because the agent–task utilities are independent, the ND [ST-SR-TA] problem can be reformulated as a linear assignment problem and as such can also be solved in polynomial time. For a trivial, albeit inefficient, reformulation, create $(|M| - 1)$ additional "clone" agents for each agent in N, so that the total number of agents is $|N||M|$. The agent–task utilities for each clone of agent i are equal to those for agent i. Then, create as many dummy tasks as are needed to ensure that the number of tasks is equal to the total number of agents (both real agents and clones). The utility of any agent for any of the dummy tasks is set very low (e.g. a large negative number). When this reformulated linear assignment problem is solved, any task that is assigned to a clone of agent i can be considered as assigned to agent i. In the solution, dummy tasks assigned to an agent are ignored, and thus each real agent i can end up with as few as zero or as many as M tasks.

It is important to note a difference between our description of this class of problems and that in Gerkey and Matarić's original taxonomy. They describe the ST-SR-TA problem as an instance of the NP-hard class of scheduling problems, represented in standard scheduling notation as $R||\sum w_j C_j$, in which "the robots execute tasks in parallel (R) and the optimization criterion is the weighted sum of execution costs ($\sum w_j C_j$)" (Gerkey and Matarić, 2004). We point out that in the scheduling literature, the objective function $\sum w_j C_j$ actually represents the weighted sum of *task completion* (or *finishing*) *times* (Brucker, 2001) and not execution costs. Since the task completion time depends on what tasks are scheduled earlier on the same machine, the fact that the objective or utility function depends on task completion times implies that this scheduling problem actually has in-schedule dependencies, and as such falls in the ID class, discussed in the next section.

5.2. In-schedule Dependencies class (ID)

For the ID class of problems, the effective utility of an agent for a task depends on what other tasks are assigned to the agent. This commonly arises in time-extended task allocation problems in which utility functions involve routing costs or task completion times. In these domains, the utility of an agent for a task depends on tasks that occur earlier in the agent's schedule. In-schedule dependencies also arise in cases where a robot is capable of executing more than one task at a time. Constraints on an agent's resources or capabilities might limit the number of tasks the agent can perform at a time, and might affect the execution quality or time for tasks it executes concurrently. For example, a robot cannot simultaneously travel to point A on one side of a room and point B on the opposite side of the room (assuming that the robot is small compared to the size of the room). It may, however, be able to monitor a location that falls within its camera's field of view, while simultaneously navigating to point A.

The ID class of problems does not include any problems of the single-task agent, single-agent task, instantaneous assignment (ST-SR-IA) subclass because by definition, agents in this subclass cannot be assigned more than one task and so cannot have in-schedule dependencies. Furthermore, any problems that involve multi-agent tasks, although they might have in-schedule dependencies, in general also have cross-schedule dependencies and fall under the XD class. Thus, we shall discuss only three subcategories under the ID class: ID [ST-SR-TA], ID [MT-SR-IA], and ID [MT-SR-TA]. Despite not having many subcategories, this is an important class of problems that captures many realistic MRTA scenarios.

There are several well-known combinatorial optimization problems that exemplify the ID class. These include the generalized assignment problem (Shmoys and Tardos, 1993; Savelsbergh, 1997), several machine scheduling problems (Brucker, 2001), the m-TSP (Bektas, 2006)

and several forms of the vehicle routing problem (VRP) (Toth and Vigo, 2001). While the linear assignment problem that exemplifies the ND class can be solved in polynomial time, these exemplifying problems for the ID class are all strongly NP-hard (with the exception of some special cases of machine scheduling problems, but these special cases generally do not correspond well to general MRTA problems). Thus, the ID class of problems represents a fundamentally more difficult class than the ND class.

5.2.1. ID [ST-SR-TA] **Mathematical models.** Consider our treasure-gathering scenario in which the robots can carry one treasure at a time and so must return to their starting location after picking up each treasure. Suppose further that each robot has a time limit within which it must complete its tasks. The execution time for a given task depends on the robot's speed and the distance of the treasure from the robot's location. We can thus specify an execution time for each (robot, task) combination. Whether or not a given robot can execute a given task depends on its time limit and what other tasks are in its schedule. This is one of the simplest cases of in-schedule dependencies, and it can be represented by the generalized assignment problem (Shmoys and Tardos, 1993), interpreting the side constraints in the mathematical formulation of this problem as time constraints.

In the generalized assignment problem, each robot can be assigned more than one task, but a side constraint, often interpreted as a "budget" or time constraint, limits the number of tasks that it can be assigned. Representing the utility of a robot $i \in N$ for a task $j \in M$ as u_{ij}, the execution time for task j by robot i as t_{ij}, and the time limit for robot i as T_i, we can express the generalized assignment problem as follows:

Maximize

$$\sum_{i \in N} \sum_{j \in M} u_{ij} x_{ij} \tag{8}$$

Subject to:

$$\begin{aligned} \sum_{j \in M} t_{ij} x_{ij} &\leq T_i & \forall i \in N \\ \sum_{i \in N} x_{ij} &\leq 1 & \forall j \in M \\ x_{ij} &\in \{0, 1\} & \forall i \in N, \forall j \in M \end{aligned} \tag{9}$$

Note that, while the linear assignment problem, which was used to model problems in the ND class, can be solved in polynomial time, the generalized assignment problem above is NP-hard.

Other than the generalized assignment problem, there are other well-known problems in the combinatorial optimization literature that can model ID [ST-SR-TA] problems. Suppose that our treasure-gathering robots were not required to pick up the treasure to bring home, but instead simply had to visit the treasure location, take a picture of it, and transmit this picture to a supervisor. The robots no

longer have to return to the start location after visiting each treasure, but can move from one treasure location directly to another. If the utility function is related to routing costs, this is another example of a problem with ID. Assuming that it is possible to travel from each task location to every other task location (a fully connected graph), it can be represented by a variant of the m-TSP (Bektas, 2006). The standard well-known Traveling Salesman Problem (TSP) finds a minimum-cost tour for a salesman residing in one city to visit all specified cities once before returning home without going through any city twice. The m-TSP generalizes the TSP to multiple salesmen who collectively must visit all of the cities such that each city is visited exactly once. With the salesmen all starting out at different locations, this is also called the Multi-Depot Multiple Traveling Salesman Problem. Variants of the TSP and the m-TSP that involve finding paths rather than tours are sometimes called the Traveling Salesman *Path* Problem (Lam and Newman, 2008) and the Multiple Traveling Salesman *Path* Problem (Zlot, 2006), respectively. It is these "path" variants that are often more relevant to robotics routing problems.

If our treasure-gathering robots were again required to pick up the treasures, but this time had a finite capacity such that they could carry a limited number of treasures at a time, we could represent this task allocation problem as a capacitated vehicle routing problem (CVRP) (Toth and Vigo, 2001), or more specifically a Multi-Depot Capacitated Vehicle Routing Problem, since each robot starts out at a different location. VRPs are a general problem class that address the transportation of passengers or the distribution of goods between depots and final users (Toth and Vigo, 2001). Solving a VRP involves determining a set of routes, each performed by a single vehicle that starts and ends at its own depot, such that all customer requirements are met, all operational constraints are satisfied, and the global transportation cost is minimized. In general, problems of this class can be expressed as integer or mixed integer programming problems that involve the minimization of some objective function subject to several constraints. In the most basic version of the VRP, the CVRP, all vehicles originate from the same depot and all customer requests or demands are known in advance. The only constraints imposed are vehicle capacity constraints ensuring that a vehicle does not hold more passengers or goods than it can carry.

One final example of a mathematical model from combinatorial optimization that can represent some problems in the ID [ST-SR-TA] class is the problem mentioned earlier of scheduling tasks on "unrelated" (i.e. heterogeneous) machines to minimize the weighted sum of completion times. This problem is represented by $R||\sum w_j C_j$ in the standard scheduling classification scheme in which $\alpha|\beta|\gamma$ represents a scheduling problem whose *machine environment* is represented by α, *job characteristics* are represented by β, and *optimality criterion* is represented by γ (Brucker, 2001). In this example, the *machine environment* is R, which is the notation for unrelated parallel machines. The *job characteristics* field is empty, and the *optimality criterion* is represented by the objective function $\sum w_j C_j$. Note that mathematical models for machine scheduling problems often do not apply directly to task allocation problems for embodied mobile agents because they do not account for the travel time required for spatially distributed tasks. Accounting for this travel time would be equivalent to specifying non-uniform task-order-dependent set-up times before each task, which significantly complicates the scheduling problem.

MRTA solution approaches. There are several examples of work in the multi-robot coordination literature that address the ID [ST-SR-TA] class of problems. Some approaches, particularly earlier approaches, leverage centralized solution methods developed for solving the TSP and m-TSP. For example, the GRAMMPS mission planner (Brummit and Stentz, 1998) uses exhaustive and randomized search (simulated annealing) to plan for a mission that is defined in terms of TSP and m-TSP components.

Melvin et al. (2007) address a multi-robot routing problem with rewards and disjoint time windows. For the special case with homogenous robots and singleton time windows, they convert the problem to a minimum-cost network flow problem which can be efficiently solved. For the more general case, they develop a mixed-integer mathematical model that bears some resemblance to models for the m-TSP. They do not solve this model directly, however, but instead develop an auction-based approach which uses repeated single-item auctions to allocate targets to agents.

Auction or market-based approaches have become widely used for solving MRTA problems since their distributed nature are particularly suited to distributed robot teams. TraderBots (Dias, 2004) is a market-based architecture for multi-robot coordination in which agents hold auctions and submit bids to determine task allocation. The system enables computation of a time-extended allocation of tasks to agents since each agent internally maintains a current schedule of tasks that it is committed to, and computes bids with respect to this schedule. The system thus explicitly takes into consideration in-schedule dependencies. Agents also periodically try to auction tasks in their current schedules that they have not begun executing. This allows the solution process to escape some local minima and find good solutions. TraderBots is designed to be a flexible architecture which allows the solution of several types of problems through customizable bidding functions and auction mechanisms such as clustered auctions and auction trees. It provides no optimality bounds or guarantees. The proof-of-concept problem addressed by Dias (2004) was a distributed sensing problem in which the team had to visit a collection of points. This problem is essentially a Multi-depot Multiple Traveling Salesman Path Problem as described earlier. In response to task auctions, agents bid

their incremental cost to insert the new task into their current schedule, plus a percentage of their expected profit for executing the task, where that percentage could be zero.

Berhault et al. (2003) address a similar exploration task in which members of the robotic team need to visit a number of predetermined target points in the environment. They also use a market mechanism, and their approach to handling in-schedule dependencies is to use combinatorial auctions, rather than single-item auctions. In combinatorial auctions, multiple tasks are auctioned at a time, and the agents bid on bundles of tasks. Berhault et al. (2003) experiment with several bidding strategies, all of which explicitly consider in-schedule dependencies by bidding an agent's _surplus_, that is overall profit minus overall cost, for each bundle.

For solving the same ID [ST-SR-TA] multi-agent routing problem, Koenig et al. (2007) find a balance between single item auctions and combinatorial auctions by designing what they describe as sequential bundle-bid single-sale auctions. In this approach, during each auction round, all agents bid on selected non-empty bundles up to a specified maximum bundle size, k. The auctioneer then assigns exactly k additional tasks to agents, either to the same or to different agents. Auction rounds are repeated until all tasks have been allocated. The contribution of their approach is a reduction in the complexity of the winner determination algorithm, relative to that for combinatorial auctions.

Lagoudakis et al. (2005) also address market-based approaches to multi-robot routing, this time with a focus on contributing a theoretical analysis of the performance of auction methods for solving this problem. They study three possible objective functions: minimizing the sum of robot path costs (MINISUM), minimizing the maximum robot path cost (MINIMAX), and minimizing the average robot path cost (MINIAVE). They determine appropriate bidding rules for each of these objective functions and prove approximation bounds for using auction methods to solve the problem.

The approaches described above do not represent an exhaustive list, but a sample of the approaches taken in the multi-robot coordination literature. Many variations of these approaches and algorithms have been explored.

_5.2.2. ID [MT-SR-IA]_ The ID [MT-SR-IA] subcategory represents problems for which there is an instantaneous allocation of a set of tasks to a robot, which must then execute these tasks concurrently. That is, each task requires only one agent but an agent can potentially perform more than one task at a time. Problems in this subclass can theoretically be represented by the generalized assignment problem, this time interpreting the side constraints as capacity constraints (Savelsbergh, 1997) (instead of as time constraints as we did in the previous section). The capacity constraints represent the fact that no realistic embodied agent can execute an unlimited number of tasks at once.

We know of no MRTA work that falls in this category, but include this category for completeness. We will later see some work that includes multi-task robots (MT) in the context of coalition formation or multi-robot tasks (MR).

_5.2.3. ID [MT-SR-TA]_ In the ID [MT-SR-TA] subclass of problems, we are tasked with determining a time-extended assignment of single-agent tasks to multi-task agents. Although we are not aware of any mathematical models to represent a general case of this problem, some variants of the VRP (Toth and Vigo, 2001) can be considered as falling in this category. For example, pick-up and delivery problems (PDPs) and dial-a-ride problems (DARPs) are particular subclasses of VRPs that deal with the transportation of packages and people respectively from given pick-up locations to given drop-off locations (Cordeau and Laporte, 2007; Desaulniers et al., 2001). The vehicles can carry multiple packages or people at a time, and so if we consider the duration of a task to be from when a person/package is picked up at the pick up location to when it is dropped off at the drop-off location, then the vehicle can clearly execute multiple tasks at a time, subject to its capacity constraints. PDP and DARP models can thus be used to represent ID [MT-SR-TA] problems in which tasks have fixed locations for their beginning and ending but are flexible in terms of what happens in between. Transportation tasks clearly fall into this category. However, a monitoring task for which an agent must stay within view of a given point for the duration of the task would not fall into this category and would need additional constraints on the location of the robot between the start and end of the task. Again, we are not aware of multi-robot coordination work in this category.

5.3. Cross-schedule Dependencies class (XD)

Problems in the XD class involve allocating simple or compound tasks in domains where the effective utility of a robot for a task depends not only on its own schedule, but also on the schedules of other robots. There are two common cases where this arises. In the first case, two or more single-agent tasks which can be allocated to different agents are related by inter-task constraints such as proximity, precedence, and simultaneity constraints. In the second case, there are multi-agent tasks each of which need to be allocated to a subset of the agents, resulting in a coalition formation problem. A key difference between the class of problems with in-schedule dependencies and this class with cross-schedule dependencies is that given an allocation of tasks to agents, agents can, in the former case, independently optimize their individual schedules, whereas in the latter case they cannot do so without coordinating with each other.

5.3.1. Cross-schedule Dependencies in problems with single-robot tasks: XD [ST-SR-IA], XD [ST-SR-TA], XD [MT-SR-IA] and XD [MT-SR-TA] The simplest type of problem with XD is when we need to perform an instantaneous assignment of single-agent tasks, some of which are related by inter-task constraints, to single-task agents (XD

[ST-SR-IA]). Consider a variation on our treasure-gathering scenario in which the treasures must be deposited at one of two holding bins, instead of being transported to the robots' starting locations. The choice of which bin to use for each treasure is made in such a way as to minimize the objective function, which might be the total distance traveled. If we specify that particular pairs of treasures which happen to be co-located in the environment must end up in the same bin even if they are picked up by different agents, this results in cross-schedule dependencies linking the actions of two different agents.

Similar cross-schedule dependencies can arise due to inter-task constraints when computing a time-extended assignment of tasks to robots (XD [ST-SR-TA]). In our treasure-gathering scenario, if some treasures are co-located such that they are stacked on each other, then the treasure stacked on top will need to be moved before the treasure that is underneath. Since each of these tasks might be assigned to a different robot, this precedence constraint between the two tasks can result in cross-schedule dependencies.

Mathematical models. There are a few mathematical models from the combinatorial optimization literature that capture the notion of cross-schedule dependencies for problems with single-agent tasks. For the instantaneous case, we can consider a further generalization of the assignment problem in which there are joint, rather than per-agent, side constraints (Mazzola and Neebe, 1986). In the model below, N is the set of agents, M is the set of tasks, and K is the set of joint side constraints.

Maximize
$$\sum_{i \in N} \sum_{j \in M} u_{ij} x_{ij} \qquad (10)$$

Subject to:
$$\sum_{i \in N} \sum_{j \in M} t_{ij} x_{ij} \leq T_k \qquad \forall k \in K$$
$$\sum_{i \in N} x_{ij} \leq 1 \qquad \forall j \in M \qquad (11)$$
$$x_{ij} \in \{0, 1\} \quad \forall i \in N, \forall j \in M$$

For the time-extended case, the problem of machine scheduling with precedence constraints on unrelated machines to minimize weighted sum of completion times ($R|prec|\sum w_j C_j$) (Brucker, 2001; Lenstra and Rinnooy Kan, 1978), falls into this category. Mathematical models have also been proposed for VRPs with simultaneity and/or precedence constraints (Bredström and Rönnqvist, 2007, 2008; Larsen et al., 2009; Rasmussen et al., 2010), and as discussed earlier, these models are better suited to our task allocation scenario than are the machine scheduling models, since routing times and costs are captured in these models.

MRTA solution approaches. There are a few MRTA approaches that support cross-schedule dependencies. The

M+ system (Botelho and Alami, 1999) performs task allocation with a market system that does iterated instantaneous assignment. It supports precedence constraints by allowing negotiation only on *executable* tasks, defined as tasks whose antecedents have already been achieved.

MacKenzie (2003) supports constraints between tasks using a variant of a market-based economy. In this approach, an auctioneer puts up several tasks, which have constraints between them, for auction. The agents then submit for each task not single bids, but rather costs expressed as functions of constrained variables such as location and time. Given the discretized cost functions submitted by each agent, the auctioneer then uses a cost minimization algorithm to determine which agent each task should be awarded to and the values of the constrained variables. Although the time at which a given task is to be executed may be set based on ordering constraints between tasks, this method supports only instantaneous assignment of tasks to agents: each agent is assigned only one task to execute, and the method cannot support determining a schedule of tasks for each agent.

Chien et al. (2000) address a robot routing problem corresponding to a geological scenario in which a team of rovers must perform a set of distributed science goals. In addition to individual resource constraints for each rover, there are cross-schedule constraints resulting from the need to access shared resources, such as a lander that can receive data from only one rover at time. They present three different approaches to this problem. The first uses the centralized ASPEN planner which uses several heuristic algorithms (such as an iterative repair algorithm combined with heuristics for m-TSP problems) to compute a conflict-free schedule for the team. The second approach uses a centralized goal allocation (with equal division of shared resources) followed by decentralized detailed planning and scheduling by each rover using the ASPEN planner. The third approach is an auction-based approach in which the individual rovers use the ASPEN planner to generate bids.

Lemaire et al. (2004) support simple ordering constraints between tasks in the form of "Task x must take place n seconds before task y". This is done in a simple way by first auctioning one task to a robot, designated the "master", that determines the start time for that task. This is then used to fix the start time of the other task, which is then auctioned to another robot designated the "slave". The "master" and "slave" robots now have a relationship that lasts the duration of the execution of the plans. During this period, they maintain communication in case dynamic changes in the environment require the tasks to be rescheduled or reallocated to other robots. This method cannot support arbitrary ordering constraints.

Mosteo et al. (2008) solve the problem of multi-robot routing while maintaining connectivity among team members, given that mobile robots have a limited communication range. This problem has cross-schedule dependencies as the schedule of an individual robot is coupled with

the schedules of other members of the team in a way that ensures that the team retains connectivity for communication. The authors propose several task allocation approaches ranging from greedy to auction-based approaches, in conjunction with a navigation mechanism that solely addresses the connectivity problem.

*5.3.2. Cross-schedule Dependencies in problems with multi-robot tasks (e.g. coalition formation): XD [ST-MR-IA], XD [ST-MR-TA], XD [MT-MR-IA], XD [MT-MR-TA] **Mathematical models**.* Identifying a subset of robots to perform a multi-robot task is equivalent to the problem of coalition formation, which has received a significant amount of interest in the multi-robot coordination literature. For an instantaneous assignment of tasks to coalitions where each robot can perform only one task at a time (i.e. can be a member of only one coalition) (XD [ST-MR-IA]), this is equivalent to the set-partitioning problem (Balas and Padberg, 1976) in combinatorial optimization. When each robot can perform multiple tasks simultaneously (i.e. be a member of multiple coalitions simultaneously) (XD [MT-MR-IA]), it is a set-covering problem (Balas and Padberg, 1976).

The time-extended assignment version of the problem in which each robot can only perform one task at a time but can be part of different coalitions over time (XD [ST-MR-TA]) bears some similarity to the Multi-mode Multi-Processor Machine Scheduling Problem (Brucker, 2001; Bianco et al., 1999). In a multi-processor machine schedule problem, each task requires one or more processors at a time, and the specific processors it needs are identified in the problem. In a *multi-mode* multi-processor problem, the specific processors are not identified; rather, there are a number of possible modes (each corresponding to a particular subset of processors) and the problem is to both assign a mode and to schedule the task operations. This is similar, in our problem, to deciding which subset of agents should perform the task, and then scheduling the task.

The XD [ST-MR-TA] is also addressed in recent work by Ramchurn et al. (2010). They present a mixed-integer programming formulation of what they describe as the Coalition Formation with Spatial and Temporal Constraints problem (CFSTP). They also present anytime heuristics to solve this problem.

We know of no existing mathematical models that capture the XD [MT-MR-TA] subcategory of problems in which we compute a time-extended allocation for a set of tasks that require multiple agents and for agents that can perform multiple tasks concurrently. We should note that Gerkey and Matarić (2004) assert that the MT-MR-TA problem is an instance of a scheduling problem with multiprocessor tasks and multipurpose machines:

$$MPTmMPMn || \sum w_j Cj$$

We argue that this is not the most appropriate analogy, however, for the following reason. In the scheduling literature, a multipurpose machine is defined as a machine that is capable of performing a subset of the tasks (in problems with heterogeneous tasks). This is as opposed to the typical machine scheduling scenarios where, on one extreme, each processor is considered capable of performing all of the tasks (assuming homogenous tasks) and on the other extreme, each task must be performed on a specific processor (Brucker, 2001). Thus, the term "multipurpose" processor/machine does *not* indicate that the machine is able to perform multiple tasks simultaneously, as interpreted by Gerkey and Matarić. Also, the case of "unrelated" machines, which we have already discussed, is equivalent to the case of "unrelated, multi-purpose machines" (Brucker, 2001, Chapter 10). Thus, multipurpose machines correspond, in our problem, to a heterogenous team of agents, rather than to agents that can perform multiple tasks simultaneously (MT).

MRTA solution approaches. There is a great deal of work in the multi-robot coordination literature that addresses the coalition formation problem. For example, Shehory and Kraus (1995) address an instantaneous assignment problem with multi-agent tasks and single-task agents (XD [ST-MR-IA]) in which goods of various sizes and weights need to be transported. Some goods can be transported by a single agent, but others require multiple agents to work together. For example, a crane might be needed to lift a heavy object onto a truck for transportation. Thus, agents might need to form coalitions to perform some of the tasks. The authors propose a greedy, distributed, anytime set-partitioning algorithm to solve this problem. The requirements of a given task are represented by a vector of required capabilities, and each coalition similarly has a vector of available capabilities. The coalition value for a task is the joint utility the coalition can reach for cooperating to perform a task. The first stage in the algorithm is a distributed computation of coalition values, in which each agent communicates with potential team members and commits to compute the values of a subset of coalitions of which it could be a member. The next stage involves iteratively deciding upon preferred coalitions, forming them, and removing the tasks and team members involved in those coalitions from further consideration. The authors then extend this work to a distributed set-covering algorithm to solve a XD [MT-MR-IA] problem in which agents can contribute their capabilities to more than one task at a time, thus resulting in overlapping coalitions (Shehory and Kraus, 1998). In this latter work, they also address a version of the problem with precedence constraints by ensuring that when a task is selected for execution, coalitions are simultaneously formed to perform any pending predecessors of that task.

Vig and Adams (2006) adapt the Shehory and Kraus's coalition formation algorithm (developed for disembodied agents) to be more suitable for the multi-robot domain by

reducing the required communication, discouraging imbalanced coalitions, and additionally constraining the capability vector to specify which of the required capabilities must appear together on a single robot versus on different robots in the coalition. They apply the adapted algorithm to a XD [ST-MR-IA] multi-robot coalition formation problem which disallows overlapping coalitions.

Guerrero and Oliver (2003) address an XD [ST-MR-IA] coalition formation problem with an auction-like mechanism in which a robot that discovers a task becomes its leader and holds an auction to engage other robots in a coalition to perform the task. Lin and Zheng (2005) describe an auction mechanism with *combinatorial bids* for coalition formation to perform a task. They define robot and task capability vectors. A robot serving as the "manager" of a task announces the task. Interested agents then submit bids specifying their capability vectors. The manager decides on a subset of the agents to award the task to, and informs them via a task pre-award message. The selected agents then communicate among themselves to form what the authors describe as a "bidding combination", and communicate their acceptance of the award to the manager who in turn responds with the task allocation. The authors do not give details on how the manager decides which subset of agents to award the task to.

Shiroma and Campos (2009) propose the CoMutaR framework for task allocation with share-restricted resources. The problem addressed is an XD [MT-MR-IA] problem in which some tasks require multiple robots, and a robot can perform multiple tasks simultaneously, subject to constraints on its share-restricted resources such as its communication link, its processor, and its position. This is achieved via the concept of a robot *action* that can accomplish a task while making use of resources on the current robots, or other robots. Multiple actions, addressing different tasks, can simultaneously run on one robot. Coalitions are formed by sending *queries* for the data and resources that the action needs. The solution process uses a single-round auction which has two stages. In the first stage, the auction for a task is opened up, and each action capable of performing the task sends out queries for its required inputs, resulting in the formation of potential coalitions which then bid for the task. In the second stage, the auctioneer determines and announces a winner.

Koes et al. (2006) addresses a time-extended coalition formation problem for robots that can perform one task at time (XD [ST-MR-TA]). They represent the coordination problem as a constraint optimization problem with a mixed-integer linear program (MILP) formulation. They then develop a solution approach named COCoA (Constraint Optimization Coordination Architecture). The approach iteratively combines the use of a commercial linear programming solver (CPLEX) with a heuristic method that produces a solution that is used as a starting step for CPLEX.

In our own work (Korsah, 2011; Korsah et al., 2012), we also address a time-extended task allocation, routing and scheduling problem in which each agent can perform one task at a time and a given compound task might require the action of multiple agents (XD [ST-MR-TA]). Furthermore, the problem includes several cross-schedule constraints and utility dependencies. These include cross-schedule temporal constraints in the form of inter-task precedence, synchronization and non-overlapping constraints. They also include cross-schedule location-related constraints, namely inter-task proximity and location capacity constraints. Finally, there are costs for delays or waiting time in an agent's schedule due to temporal constraints with another agent. We represent this challenging coordination problem with a set-partitioning MILP formulation, and present the *xTeam* planner comprising a custom branch-and-price (Barnhart et al., 1998) algorithm for its solution.

5.4. Complex Dependencies (CD)

The CD class of problems involves task allocation for *complex* tasks in domains where the effective utility of an agent for a task depends on the schedules of other agents. Recall that complex tasks have multiple possible decompositions, at least one of which can be allocated to multiple agents (Zlot, 2006). As such, allocating complex tasks involves answering the question of *which* set of subtasks should be allocated (i.e. which decomposition should be used) in addition to the standard task allocation and scheduling questions of *who* should perform each task, and *when*. As described previously, complex tasks might exist explicitly in the problem description, or implicitly as sets of simple tasks that can be composed into complex tasks due to the existence of choices of constraints. These two sources of complex tasks result in two natural groups of problems with complex dependencies. The first group are problems which have single-agent tasks (SR) but which are related by disjunctions of constraints such that we can compose complex tasks. The second group are problems with multi-agent tasks (MR) that are complex tasks.

We know of no well-known problems or mathematical models in the combinatorial optimization literature that capture this problem class. There are, however, a few examples of approaches in the MRTA literature that address problems in this class.

5.4.1. Complex Dependencies in problems with single-robot tasks: CD [ST-SR-IA], CD [ST-SR-TA], CD [MT-SR-IA] and CD [MT-SR-TA]
Jones et al. (2009) address the problem of time-extended multi-robot coordination for domains with "intra-path" constraints. This is exemplified with a disaster-response problem in which a number of fire tasks need to be assigned to fire-truck robots. There are however, piles of debris on various roads, blocking some of the routes that the fire trucks must take to reach the

Table 1. Summary of the two-level task allocation taxonomy, with exemplifying problems and models from combinatorial optimization, vehicle routing, scheduling, and coalition formation.

Level 2: Problem Configuration	Level 1: Degree of interrelatedness			
	No Dependencies (ND)	In-schedule Dependencies (ID)	Cross-schedule Dependencies (XD)	Complex Dependencies (CD)
ST-SR-IA	Linear sum assignment problem (LSAP) (Votaw and Orden, 1952)	*(shaded)*	Assignment problem with side constraints (APSC) (Mazzola and Neebe, 1986)	
ST-SR-TA	Can be reformulated as Linear sum assignment problem (LSAP)	Generalized assignment problem (interpreting constraints as time limits) (Shmoys and Tardos, 1993), Scheduling on unrelated machines to minimize weighted sum of completion times $(R||\sum w_jC_j)$ (Brucker, 2011; Bruno et al., 1974), Multiple Traveling Salesman Problem (m-TSP) (Bektas, 2006), Vehicle Routing Problem (VRP) (Toth and Vigo, 2001)	Scheduling, with precedence constraints, on unrelated machines to minimize weighted sum of completion times $(R|prec|\sum w_jC_j)$ (Brucker, 2001; Lenstra and Rinnooy Kan, 1978), Vehicle Routing Problems with precedence or synchronization constraints (Bredström and Rönnqvist, 2007, 2008)	
MT-SR-IA	*(shaded)*	Generalized assignment problem (interpreting constraints as capacity limits) (Savelsbergh, 1997)	Modified version of generalized assignment problem (interpreting constraints as capacity limits)	
MT-SR-TA	*(shaded)*	*(shaded)*		
ST-MR-IA	*(shaded)*	*(shaded)*		
ST-MR-TA	*(shaded)*	*(shaded)*	Set Partitioning Problem (Balas and Padberg, 1976) Multi-mode Multi-processor Task Scheduling (Brucker, 2001; Bianco et al., 1999),	
MT-MR-IA			Coalition Formation with Spatial and Temporal Constraints (CFSTP) (Ramchurn et al., 2010)	
MT-MR-TA			Set Covering Problem (Balas and Padberg, 1976)	

Table 2. Summary of the two-level task allocation taxonomy, with some example problems and solution approaches from the MRTA literature.

Level 1: Degree of Interrelatedness / Level 2: Problem Configuration	No Dependencies (ND)	In-schedule Dependencies (ID)	Cross-schedule Dependencies (XD)	Complex Dependencies (CD)
ST-SR-IA	Vail and Veloso (2003), Gerkey and Mataric (2002), Simmons et al. (2000)		Botelho and Alami (1999) (M+), MacKenzie (2003)	Jones (2009)
ST-SR-TA		Brummit and Stentz (1998) (GRAMMPS), Melvin et al. (2007), Dias (2004) (Trader Bots), Berhault et al. (2003), Koenig et al. (2007) Lagoudakis et al. (2005)	Chien et al. (2000), Lemaire et al. (2004)	
MT-SR-IA				
MT-SR-TA				
ST-MR-IA			Shehory and Kraus (1995), Vig and Adams (2006), Guerrero and Oliver (2003), Lin and Zheng (2005) Koes et al. (2006) Shiroma and Campos (2009) (CoMutaR)	Parker and Tang (2006) (ASyMTRe)
ST-MR-TA				
MT-MR-IA			Koes et al. (2006) Shiroma and Campos (2009) (CoMutaR)	Zlot (2006)
MT-MR-TA				

fires. These piles of debris can be cleared by bulldozer robots. Clearly, not all of the piles of debris need to be cleared; it would be sufficient to clear only those along the routes that will be taken by the fire trucks if these routes were known. However, the cost of each route, and hence the choice of route, for the fire trucks depends in turn on which piles of debris are cleared. In the most basic case when each fire requires only one fire truck, and each pile of debris is cleared by only one bulldozer, this problem is the CD [ST-SR-TA] class. Its solution must simultaneously determine not only an allocation of fires to fire trucks, but also the paths that the fire trucks should take to reach the fires and which bulldozers should be assigned to clear debris along these routes. Jones et al. (2009) apply two different approaches to this complex task allocation problem. The first uses tiered auctions along with clustering and opportunistic path planning to perform a bounded search of possible time-extended schedules and allocations. The second method uses a genetic algorithm. A more complicated version of Jones et al.'s disaster-response problem, in which multiple fire trucks may work on one fire or multiple bulldozers may cooperate to clear one pile debris, can be classified in the CD [ST-MR-TA] category, described in the next section.

5.4.2. Complex Dependencies in problems with multi-robot tasks: CD [ST-MR-IA], CD [ST-MR-TA], CD [MT-MR-IA] and CD [MT-MR-TA] Parker and Tang (2006) present a method for coalition formation through a process they describe as automated task solution synthesis. This work involves building a solution to a task by dynamically connecting a network of *schemas* that reside on individual robots. Schemas are defined by inputs and output ports, a local variable list, and a behavior. Given the information types of the inputs and outputs of various schemas, the schemas can be automatically connected to produce the desired behavior. Thus, different possible schema configurations represent different possible ways of achieving a task, and the tasks in this problem can be thought of as complex tasks since they have multiple possible decompositions. The problem addressed in this work can thus be classified as a CD [ST-MR-IA] problem. The solution method presented, called ASyMTRe, greedily searches through the space of potential schema configurations to find a solution. In the distributed version of the algorithm, ASyMTRe-D, each robot decides what information it needs and requests this information from others.

Zlot (2006) addresses the problem of time-extended task allocation for explicitly-defined complex tasks (CD [ST-MR-TA]). For this purpose, TraderBots (Dias, 2004) is extended to enable agents to auction and bid on *task trees*, rather than simple tasks. A task tree represents a possible decomposition of a task. When a task tree is auctioned, robots can bid on either the auctioneer's decomposition of the task, or their own decomposition. They can also bid on selected profitable nodes of the tree, rather than all of them.

Once all the bids come in, the auctioneer's winner determination algorithm then decides which set of minimally satisfying nodes from the tree result in the lowest cost team solution.

6. Summary

We have presented a new task allocation taxonomy based on the degree of interrelatedness between agent–task utilities. The new taxonomy, iTax, is significantly more comprehensive than the current *de facto* standard MRTA taxonomy, and provides a useful way of categorizing task allocation problems in a manner that is strongly related to problem difficulty. It groups task allocation problems into four natural classes that relate to the problem complexity. Problems in the ND class can generally be modeled by the linear assignment problem, and solved in polynomial time. Problems in the other classes are generally NP-hard. For problems in the ID class, the schedules of individual agents can be optimized independently of each other. For problems in the XD class, schedule optimization requires coordination between agents. Finally, the CD class requires task decomposition and task allocation to be performed simultaneously.

For each problem class, we present exemplifying problems and mathematical models from the combinatorial optimization literature. These are summarized in Table 1. We also identified example problems and solution approaches in the multi-robot coordination literature, summarized in Table 2. In both tables, grayed-out cells represent non-existent categories in the taxonomy. Empty cells indicate categories for which examples from the literature have not been identified.

References

Balas E and Padberg MW (1976) Set partitioning: A survey. *SIAM Review* 18(4): 710–760.

Barnhart C, Johnson EL, Nemhauser GL, Savelsbergh MWP and Vance PH (1998) Branch-and-price: Column generation for solving huge integer programs. *Operations Research* 46: 316–329.

Bektas T (2006) The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega* 34(3): 209–219.

Berhault M, Huang H, Keskinocak P, et al. (2003) Robot exploration with combinatorial auctions. In: *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003 (IROS 2003)*, volume 2. pp. 1957–1962.

Bianco L, Dell'Olmo P, Giordani S and Speranza MG (1999) Minimizing makespan in a multimode multiprocessor shop scheduling problem. *Naval Research Logistics* 46: 893–911.

Botelho SC and Alami R (1999) M+: a scheme for multirobot cooperation through negotiated task allocation and achievement. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1234–1239.

Bredström D and Rönnqvist M (2007) A branch and price algorithm for the combined vehicle routing and scheduling problem with synchronization constraints. Discussion Paper Number FOR7, Norwegian School of Economics and Business Administration, Department of Finance and Management Science.

Bredström D and Rönnqvist M (2008) Combined vehicle routing and scheduling with temporal precedence and synchronization constraints. *European Journal of Operations Research* 191: 19–31.

Brucker P (2001) *Scheduling Algorithms*, 3rd edition. Secaucus, NJ: Springer-Verlag.

Brummit B and Stentz A (1998) GRAMMPS: A generalized mission planner for multiple mobile robots. In: *Proceedings of the IEEE International Conference on Robotics and Automation*.

Bruno J, E G Coffman J and Sethi R (1974) Scheduling independent tasks to reduce mean finishing time. *Communications of the ACM* 17: 382–387. DOI: 10.1145/361011.361064.

Chien S, Barrett A, Estlin T and Rabideau G (2000) A comparison of coordinated planning methods for cooperating rovers. In: *Proceedings of the Fourth International Conference on Autonomous Agents (AGENTS '00)*. New York: ACM Press, pp. 100–101. DOI: 10.1145/336595.337057.

Cordeau JF and Laporte G (2007) The dial-a-ride problem: models and algorithms. *Annals of Operations Research* 153(1): 29–46.

Desaulniers G, Desrosiers J, Erdmann A, Solomon MM and Soumis F (2001) VRP with pickup and delivery. In: *The Vehicle Routing Problem*. Philadelphia, PA: Society for Industrial and Applied Mathematics, pp. 225–242.

Dias MB (2004) *TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments*. Ph.D. thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Gerkey BP (2003) *On Multi-Robot Task Allocation*. Ph.D. thesis, University of Southern California Computer Science Department, Los Angeles, CA.

Gerkey BP and Matarić MJ (2002) Sold!: auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation* 18(5): 758–768.

Gerkey BP and Matarić MJ (2004) A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research* 23(9): 939–954.

Guerrero J and Oliver G (2003) Multi-robot task allocation strategies using auction-like mechanisms. *Artificial Intelligence Research and Development in Frontiers in Artificial Intelligence and Application* 100: 111–122.

Jones E, Dias MB and Stentz AT (2009) Time-extended multirobot coordination for domains with intra-path constraints. In: *Robotics: Science and Systems (RSS)*.

Jones EG (2009) *Multi-Robot Coordination in Domains with Intra-path Constraints*. Ph.D. thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Koenig S, Tovey C, Zheng X and Sungur I (2007) Sequential bundle-bid single-sale auction algorithms for decentralized control. In: *Proceedings of the 20th International Joint Conference on Artifical Intelligence*. San Francisco, CA: Morgan Kaufmann, pp. 1359–1365.

Koes M, Nourbakhsh I and Sycara K (2006) Constraint optimization coordination architecture for search and rescue robotics. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2006*, pp. 3977–3982.

Korsah GA (2011) *Exploring Bounded Optimal Coordination for Heterogeneous Teams with Cross-schedule Dependencies*. Ph.D. thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Korsah GA, Kannan B, Browning B, Stentz A and Dias MB (2012) xBots: An approach to generating and executing optimal multi-robot plan with cross-schedule dependencies. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2012*.

Kuhn HW (1955) The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly* 2: 83–97.

Lagoudakis MG, Markakis E, Kempe D, et al. (2005) Auction-based multi-robot routing. In: *Proceedings of Robotics: Science and Systems*, Cambridge, MA.

Lam F and Newman A (2008) Traveling salesman path problems. *Mathematical Programming* 113: 39–59. DOI: 10.1007/s10107-006-0046-8.

Larsen J, Dohn A and Rasmussen MS (2009) *The Vehicle Routing Problem with Time Windows and Temporal Dependencies*. Technical Report 1.2009, DTU Management Engineering.

Lemaire T, Alami R and Lacroix S (2004) A distributed tasks allocation scheme in multi-UAV context. In: *Proceedings of the 2004 IEEE International Conference on Robotics and Automation, 2004 (ICRA 2004)*.

Lenstra JK and Rinnooy Kan AHG (1978) Complexity of scheduling under precedence constraints. *Operations Research* 26(1): 22–35. DOI: 10.1287/opre.26.1.22.

Lin L and Zheng Z (2005) Combinatorial bids based multi-robot task allocation method. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005 (ICRA 2005)*, pp. 1145–1150.

MacKenzie DC (2003) Collaborative tasking of tightly constrained multi-robot missions. In: Schultz AC, Parker LE and Schneider FE (eds.) *Multi-Robot Systems: From Swarms to Intelligent Automata (Proceedings Second International Workshop on Multi-Robot Systems)*, volume 2. Washington, DC: Kluwer Academic Publishers, pp. 39–50.

Mazzola JB and Neebe AW (1986) Resource-constrained assignment scheduling. *Operations Research* 34: 560–572. DOI: 10.1287/opre.34.4.560.

Melvin J, Keskinocak P, Koenig S, Tovey CA and Ozkaya BY (2007) Multi-robot routing with rewards and disjoint time windows. In: *Intelligent Robots and Systems (IROS) 2007*, pp. 2332–2337.

Mosteo A, Montano L and Lagoudakis MG (2008) Multi-robot routing under limited communication range. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, Pasadena, CA, pp. 1531–1536.

Parker LE (1998) Alliance: An architecture for fault tolerant multi-robot cooperation. *IEEE Transactions on Robotics and Automation* 14: 220–240.

Parker LE and Tang F (2006) Building multirobot coalitions through automated task solution synthesis. *Proceedings of the IEEE* 94(7): 1289–1305. DOI: 10.1109/JPROC.2006.876933.

Ramchurn SD, Polukarov M, Farinelli A, Truong C and Jennings NR (2010) Coalition formation with spatial and temporal constraints. In: *Proceedings of the 9th International Conference*

on Autonomous Agents and Multiagent Systems (AAMAS '10), Vol. 3. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, pp. 1181–1188.

Rasmussen MS, Justesen T, Dohn A and Larsen J (2010) *The Home Care Crew Scheduling Problem: Preference-based Visit Clustering and Temporal Dependencies*. Technical Report 11.2010, DTU Management Engineering.

Savelsbergh M (1997) A branch-and-price algorithm for the generalized assignment problem. *Operations Research* 45(6): 831–841. DOI: 10.1287/opre.45.6.831.

Shehory O and Kraus S (1995) Task allocation via coalition formation among autonomous agents. In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Vol. 1. San Francisco, CA: Morgan Kaufmann, pp. 655–661.

Shehory O and Kraus S (1998) Methods for task allocation via agent coalition formation. *Artificial Intelligence* 101(1–2): 165–200. DOI: 10.1016/S0004-3702(98)00045-9.

Shiroma P and Campos M (2009) Comutar: A framework for multi-robot coordination and task allocation. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009 (IROS 2009)*, pp. 4817–4824. DOI: 10.1109/IROS.2009.5354166.

Shmoys DB and Tardos E (1993) An approximation algorithm for the generalized assignment problem. *Mathematical Programming* 62: 461–474. DOI: 10.1007/BF01585178.

Simmons RG, Apfelbaum D, Burgard W, et al. (2000) Coordination for multi-robot exploration and mapping. In: *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*. AAAI Press/The MIT Press, pp. 852–858.

Toth P and Vigo D (2001) An overview of vehicle routing problems. In: *The Vehicle Routing Problem*. Philadelphia, PA: Society for Industrial and Applied Mathematics, pp. 1–26.

Vail D and Veloso M (2003) Multi-robot dynamic role assignment and coordination through shared potential fields. In: Schultz A, Parker L and Schneider F (eds.) *Multi-Robot Systems*. Dordrecht: Kluwer.

Vig L and Adams JA (2006) Multi-robot coalition formation. *IEEE Transactions on Robotics* 22(4): 637–649. DOI: 10.1109/TRO.2006.878948.

Votaw D Jr and Orden A (1952) The personnel assignment problem. In: *Symposium on Linear Inequalities and Programming*. Washington, DC: Planning Research Division, Comptroller, Headquarters U.S. Air Force, pp. 155–163.

Zlot RM (2006) *An Auction-Based Approach to Complex Task Allocation for Multirobot Teams*. Ph.D. thesis, Robotics Institute, Carnegie Mellon University.