Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence

Generalized Reaction Functions for Solving Complex-Task Allocation Problems*

Xiaoming Zheng
Facebook, Inc.
Palo Alto, CA 94304
xmzheng@fb.com

Sven Koenig
University of Southern California
Los Angeles, CA 90089-0781
skoenig@usc.edu

Abstract

We study distributed task-allocation problems where cooperative agents need to perform some tasks simultaneously. Examples are multi-agent routing problems where several agents need to visit some targets simultaneously, for example, to move obstacles out of the way cooperatively. In this paper, we first generalize the concept of reaction functions proposed in the literature to characterize the agent costs of performing multiple complex tasks. Second, we show how agents can construct and approximate reaction functions in a distributed way. Third, we show how reaction functions can be used by an auction-like algorithm to allocate tasks to agents. Finally, we show empirically that the team costs of our algorithms are substantially smaller than those of an existing state-of-the-art allocation algorithm for complex tasks.

1 Introduction

We study complex-task allocation problems in a cooperative setting where agents collaborate to minimize the team cost (that is, maximize the team performance) and some tasks need to be performed simultaneously. Our motivating problem is multi-agent routing where the tasks are to visit given targets in the plane. The terrain, the locations of the agents and the locations of the targets are known. Multi-agent routing is a standard problem for robot teams [Koenig *et al.*, 2007; Dias *et al.*, 2006], for example, as part of de-mining, search-and-rescue and taking rock probes on the moon. In this paper, we are interested in the version of multi-agent routing where some targets, called complex targets, need to be visited simultaneously by more than one agent. For example, large fires can only be extinguished with several fire engines, and heavy objects can only be moved with several robots. Thus, allocation algorithms for complex tasks have to solve the following two interrelated subproblems:

- **Subproblem 1:** Each complex target has to be visited by more than one agent. Thus, one has to determine which group of agents should visit a given complex target.
- **Subproblem 2:** Each complex target has to be visited by the group of agents at the same time. Thus, one has to determine when a given group of agents should visit a given complex target.

Solving Subproblem 2 is non-trivial since a group of agents needs to agree on a common visit time. Zheng and Koenig [Zheng and Koenig, 2008] proposed an approach that makes use of reaction functions to allocate complex targets to agents. Reaction functions characterize the costs of agents for visiting a single complex target at any given visit time and thus can be used by a central planner to determine the optimal visit time of a group of agents for visiting an additional complex target. The main drawback of this approach is that the visit times of complex targets cannot be changed once they have been assigned, which results in highly suboptimal task allocations. In this paper, we first generalize the concept of reaction functions to characterize the costs of an agent for visiting multiple complex targets at any given visit times. Then, we show how agents can construct and approximate generalized reaction functions in a distributed way. Third, we show how an auction-like algorithm can use generalized reaction functions to allocate targets to agents. Finally, we show empirically that the team costs of our algorithms can be substantially smaller than those of the allocation algorithm developed in [Zheng and Koenig, 2008].

2 Multi-Agent Routing

We follow [Zheng and Koenig, 2008] to formalize multi-agent routing problems. A multi-agent routing problem consists of a set of agents A and a set of targets X. The number of different agents that need to visit target $x \in X$ simultaneously, called its **coalition size**, is $d(x)$. We call a target x **simple** if $d(x) = 1$ and **complex** otherwise. The set of simple targets X^s and the set of complex targets X^c partition the set of all targets. The group of $d(x)$ different agents that need to visit complex target x at some visit time t is called the **coalition** of the complex target. Each agent in the coalition thus has a **commitment** to visit the complex target x at visit time t, written as $x \leftarrow t$. Each agent $a \in A$ is characterized by a simple target capacity q^s_a and a complex target capacity

*This material is based upon work supported by NSF (while serving at NSF). It is also based upon work supported by ARL/ARO under contract/grant number W911NF-08-1-0468 and ONR in form of a MURI under contract/grant number N00014-09-1-1031. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the sponsoring organizations, agencies or the U.S. government.

q_a^c with the meaning that agent a can visit at most q_a^s simple targets and at most q_a^c complex targets.

An **allocation** of agent a is a triple (X_a^s, X_a^c, C_a), where X_a^s is the set of simple targets assigned to it, X_a^c is the set of complex targets assigned to it, and C_a is the set of its commitments for the complex targets in X_a^c. The **agent cost** $c_a^{agent}(X_a^s, X_a^c, C_a)$ is the smallest sum of travel and wait times needed for agent a to visit all targets assigned to it from its initial location, where it can freely determine when to visit each simple target in X_a^s subject to the restriction that it has to visit all complex targets (if any) at the visit times recorded in its commitment set C_a. (The agent cost is defined to be infinity if agent a cannot satisfy this restriction, $|X_a^s| > q_a^s$ or $|X_a^c| > q_a^c$.)

Our objective is to find a solution with a small team cost, where a solution requires each target $x \in X$ to be assigned to exactly $d(x)$ different agents. All agents in the coalition must have the same commitment for a complex target. The **team cost** of a set of agents A is $\sum_{a \in A} c_a^{agent}(X_a^s, X_a^c, C_a)$ (roughly proportional to the energy needed by the agents for traveling and waiting) for the **MiniSum** team objective and $\max_{a \in A} c_a^{agent}(X_a^s, X_a^c, C_a)$ (the task-completion time) for the **MiniMax** team objective. We use c^{team} as a special operator for either the sum or max operator, depending on the team objective, and write $c_{a \in A}^{team} c_a^{agent}(X_a^s, X_a^c, C_a)$ to make our notation independent of the team objective.

3 Generalized Reaction Functions

In this section, we introduce the concept of (generalized) reaction functions, explain how to construct and use them, and finally describe how to approximate them.

3.1 Concepts

As discussed in the introduction, solving a complex-task allocation problem consists of two parts: which targets should be assigned to which agents (Subproblem 1) and when should the agents visit their assigned targets (Subproblem 2). In order to evaluate a given target assignment, the agents have to determine the visit times of their assigned targets that minimize the team cost (optimal visit times). The reason for introducing reaction functions is to facilitate the determination of the optimal visit times for the complex targets.

Assume that each agent $a \in A$ has been assigned a set of simple targets $X_a^s \subseteq X^s$ and a set of complex targets $X_a^c \subseteq X^c$, let $\mathbf{x}_a = (x_a^1, \ldots, x_a^{n_a})$ be the vector of the complex targets X_a^c (in an arbitrary order), $\mathbf{t}_a = (t_a^1, \ldots, t_a^{n_a})$ be the vector of the corresponding visit times of these complex targets, and let $\mathbf{x}_a \leftarrow \mathbf{t}_a$ be the set of commitments $\{x_a^1 \leftarrow t_a^1, \ldots, x_a^{n_a} \leftarrow t_a^{n_a}\}$. Then, the agent cost of agent a for visiting the complex targets $\mathbf{x}_a$ at visit times $\mathbf{t}_a$ is $c_a^{agent}(X_a^s, X_a^c, \mathbf{x}_a \leftarrow \mathbf{t}_a)$. Let $\mathbf{x} = (x_1, \ldots, x_n)$ be the vector of all complex targets $\cup_{a \in A} X_a^c$ (in an arbitrary order) and $\mathbf{t} = (t_1, \ldots, t_n)$ be the vector of the corresponding visit times of these complex targets. We define a projection function $f_a(\mathbf{v})$ to project any vector $\mathbf{v}$ of complex targets onto the complex targets assigned to agent $a \in A$ in order $(t_a^1, \ldots, t_a^{n_a}$, for example, $\mathbf{x}_a = f_a(\mathbf{x})$ and $\mathbf{t}_a = f_a(\mathbf{t})$. Thus, the optimal visit times of all complex targets can be

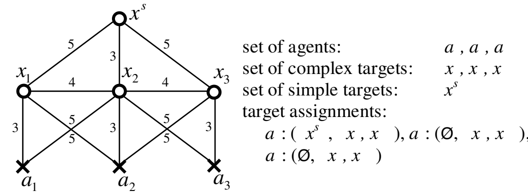


Figure 1: Multi-Agent Routing on a Graph

permutations ($\pi \in \Pi_{a_1}$)	visit orders ($p \in \mathcal{P}_{a_1,\pi}$)	vectors of visit times ($\mathbf{t}_p$)	agent cost (c_p)
(x_1, x_2)	$a_1 \rightsquigarrow x_1 \rightsquigarrow x_2 \rightsquigarrow x^s$	(3, 7)	10
	$a_1 \rightsquigarrow x_1 \rightsquigarrow x^s \rightsquigarrow x_2$	(3, 11)	11
	$a_1 \rightsquigarrow x^s \rightsquigarrow x_1 \rightsquigarrow x_2$	(13, 17)	17
(x_2, x_1)	$a_1 \rightsquigarrow x_2 \rightsquigarrow x_1 \rightsquigarrow x^s$	(5, 9)	14
	$a_1 \rightsquigarrow x_2 \rightsquigarrow x^s \rightsquigarrow x_1$	(5, 13)	13
	$a_1 \rightsquigarrow x^s \rightsquigarrow x_2 \rightsquigarrow x_1$	(11, 15)	15

Table 1: Visit Orders of Agent a_1

determined as

$$\arg\min_{\mathbf{t} \in \mathbb{R}_+^n} c_{a \in A}^{team} c_a^{agent}(X_a^s, X_a^c, \mathbf{x}_a \leftarrow f_a(\mathbf{t}))$$

where $\mathbb{R}_+$ is the set of all non-negative numbers.

For the given target assignment $\{(X_a^s, X_a^c)\}_{a \in A}$, we can construct a bipartite graph of agents A and complex targets X^c where an edge connects an agent a and a complex target x iff x is assigned to a. Two agents are **related** iff they are connected in the graph. This relationship partitions the set of agents A into a set of bundles $B = \{b_1, \ldots, b_l\}$ where each bundle $b \in B$ consists of agents that are related with each other. Since agents in different bundles can determine the visit times of their assigned targets independently, we have $\min_{\mathbf{t} \in \mathbb{R}_+^n} c_{a \in A}^{team} c_a^{agent}(X_a^s, X_a^c, \mathbf{x}_a \leftarrow f_a(\mathbf{t})) = c_{b \in B}^{team}(\min_{\mathbf{t}_b \in \mathbb{R}_+^{n_b}} c_{a \in b}^{team} c_a^{agent}(X_a^s, X_a^c, \mathbf{x}_a \leftarrow f_a(\mathbf{t}_b)))$, where n_b is the number of complex targets assigned to bundle b and $\mathbf{t}_b$ is the vector of the visit times of these complex targets. In other words, we can solve the minimization problem $\min_{\mathbf{t}_b \in \mathbb{R}_+^{n_b}} c_{a \in b}^{team} c_a^{agent}(X_a^s, X_a^c, \mathbf{x}_a \leftarrow f_a(\mathbf{t}_b))$ independently for each bundle $b \in B$.

We now introduce a **reaction function** $\mathcal{F}_a^{X_a^c}(\mathbf{t}_a)$ for each agent a in a bundle $b \in B$ that characterizes its agent cost (= the smallest sum of travel and wait times) of visiting its assigned complex targets X_a^c at any given visit times $\mathbf{t}_a$:

$$\mathcal{F}_a^{X_a^c}(\mathbf{t}_a) := c_a^{agent}(X_a^s, X_a^c, \mathbf{x}_a \leftarrow \mathbf{t}_a),$$

which generalizes the concept of reaction functions proposed in [Zheng and Koenig, 2008]. The optimal visit times $\mathbf{t}_b^*$ for the complex targets assigned to bundle b can thus be determined as follows:

$$\mathbf{t}_b^* = \arg\min_{\mathbf{t}_b \in \mathbb{R}_+^{n_b}} c_{a \in b}^{team} \mathcal{F}_a^{X_a^c}(f_a(\mathbf{t}_b)).$$

3.2 Constructing Reaction Functions

Given a target assignment $\{(X_a^s, X_a^c)\}_{a \in A}$, agent $a \in A$ constructs its reaction function $\mathcal{F}_a^{X_a^c}(\mathbf{t}_a)$ as follows:

1. Agent a constructs all possible visit orders of its assigned targets in $X_a^s \cup X_a^c$. Let $\mathcal{P}_a$ be the set of all these visit orders.
2. Agent a constructs all possible permutations of its assigned complex targets in $X_a^c = \{x_a^1, \ldots, x_a^{n_a}\}$. Let Π_a be the set of all these permutations. For each permutation $\pi = (x_a^{\pi(1)}, \ldots, x_a^{\pi(n_a)})$ in Π_a, let $\mathcal{P}_{a,\pi}$ be the set of the visit orders in $\mathcal{P}_a$ in which complex targets are visited in the order given by permutation π.

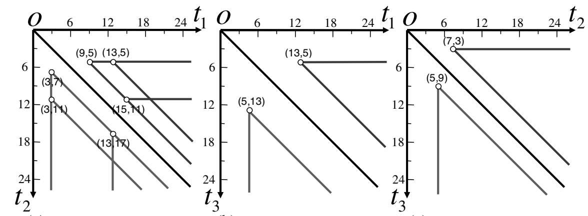
   Consider the multi-agent routing problem with the given target assignment shown in Figure 1, where the agents and targets are located on a graph and the agents can only move along the edges of the graph. The coalition sizes of all three complex targets are 2. Table 1 tabulates the permutations of complex targets and the visit orders of targets that are assigned to agent a_1.
3. For each permutation $\pi \in \Pi_a$ and each visit order $p \in \mathcal{P}_{a,\pi}$, agent a finds the vector of visit times $\mathbf{t}_p = (t_p^{\pi(1)}, \ldots, t_p^{\pi(n_a)})$ of the complex targets in π and calculates the resulting agent cost c_p if it visits its assigned targets in visit order p without waiting at any target. For example, Table 1 tabulates the vectors of visit times and the corresponding agent costs for each possible visit order of agent a_1.
4. For each permutation $\pi \in \Pi_a$ and each visit order $p \in \mathcal{P}_{a,\pi}$, agent a constructs a meta-function $\mathcal{F}_{a,p}^{\pi}(\mathbf{t}_a^{\pi})$, which characterizes the agent cost of agent a if it visits its assigned targets in visit order p, as follows:

   Domain: The domain of this meta-function is the set of all possible visit times $\mathbf{t}_a^{\pi} = (t_a^{\pi(1)}, \ldots, t_a^{\pi(n_a)})$ of the complex targets in $\pi = (x_a^{\pi(1)}, \ldots, x_a^{\pi(n_a)})$ if agent a visits its assigned targets in visit order p and waits only at complex targets. Consider any pair of adjacent complex targets $x_a^{\pi(i-1)} \in \pi$ and $x_a^{\pi(i)} \in \pi$. Their visit times $t_a^{\pi(i-1)}$ and $t_a^{\pi(i)}$ must satisfy $t_a^{\pi(i)} - t_a^{\pi(i-1)} \geq t_p^{\pi(i)} - t_p^{\pi(i-1)}$ since $t_p^{\pi(i)} - t_p^{\pi(i-1)}$ is the travel time of agent a from complex target $x_a^{\pi(i-1)}$ to $x_a^{\pi(i)}$ in visit order p. This constraint can be re-written as $t_a^{\pi(i)} - t_p^{\pi(i)} \geq t_a^{\pi(i-1)} - t_p^{\pi(i-1)}$. Thus, the domain of the meta-function is the set of vectors of visit times $\mathbf{t}_a^{\pi} = (t_a^{\pi(1)}, \ldots, t_a^{\pi(n_a)})$ that satisfy $t_a^{\pi(n_a)} - t_p^{\pi(n_a)} \geq \ldots \geq t_a^{\pi(1)} - t_p^{\pi(1)} \geq 0$. We define a boolean function $\mathscr{B}(\mathbf{t}_a) = \mathbf{true}$ iff $t_a^{n_a} \geq \ldots \geq t_a^1 \geq 0$ for any vector $\mathbf{t}_a$ so that the domain constraint of the meta-function can be simply written as $\mathscr{B}(\mathbf{t}_a^{\pi} - \mathbf{t}_p) = \mathbf{true}$.

   Value: The value of this meta-function is the agent cost of agent a if agent a visits its assigned targets in visit order p and waits only at complex targets, which can be calculated as:

   $$\mathcal{F}_{a,p}^{\pi}(\mathbf{t}_a^{\pi}) = \quad c_p + t_a^{\pi(n_a)} - t_p^{\pi(n_a)} \quad \text{iff } \mathscr{B}(\mathbf{t}_a^{\pi} - \mathbf{t}_p) = \mathbf{true}$$

   where c_p is the travel time of agent a to visit its assigned targets in the order p and $t_a^{\pi(n_a)} - t_p^{\pi(n_a)}$ is the total wait time of agent a at complex targets for the given visit times $\mathbf{t}_a^{\pi}$. For the sake of convenience, we define $\mathcal{F}_{a,p}^{\pi}(\mathbf{t}_a^{\pi}) := \infty$ if $\mathscr{B}(\mathbf{t}_a^{\pi} - \mathbf{t}_p) = \mathbf{false}$.



(a) Reaction Function of a (b) Reaction Function of a (c) Reaction Function of a

Figure 2: Domains of Reaction Functions

5. For each permutation $\pi \in \Pi_a$, agent a constructs a permutation function $\mathcal{F}_a^{\pi}(\mathbf{t}_a^{\pi})$ whose domain consists of the vectors of visit times $\mathbf{t}_a^{\pi}$ of complex targets in π that satisfy the domain constraint $\mathscr{B}(\mathbf{t}_a^{\pi}) = \mathbf{true}$ and whose value is determined as the minimum of meta-functions $\mathcal{F}_{a,p}^{\pi}(\mathbf{t}_a^{\pi})$ for all visit orders $p \in \mathcal{P}_{a,\pi}$:

   $$\mathcal{F}_a^{\pi}(\mathbf{t}_a^{\pi}) = \min_{p \in \mathcal{P}_{a,\pi}} \mathcal{F}_{a,p}^{\pi}(\mathbf{t}_a^{\pi}).$$

6. Finally, the reaction function $\mathcal{F}_a^{X_a^c}(\mathbf{t}_a)$ of agent a is the collection of all permutation functions $\mathcal{F}_a^{\pi}(\mathbf{t}_a^{\pi})$ for each permutation $\pi \in \Pi_a$.

   Consider again the multi-agent routing problem shown in Figure 1. Figure 2 shows the domains of the reaction functions of agents a_1, a_2 and a_3 for the given target assignments, where a meta-function corresponds to a visit order of an agent.

The correctness of our approach is given by the following lemma and theorem.

Lemma 1 *For each permutation $\pi \in \Pi_a$ of the complex targets assigned to agent a, the agent cost (= the smallest sum of travel and wait times) of agent a for visiting its complex targets in the order given by permutation π at any given visit times $\mathbf{t}_a^{\pi}$ is the value of the permutation function $\mathcal{F}_a^{\pi}(\mathbf{t}_a^{\pi})$, namely,* $\min_{p \in \mathcal{P}_{a,\pi}} \mathcal{F}_{a,p}^{\pi}(\mathbf{t}_a^{\pi})$.

Proof: Let c^* be the agent cost of agent a. If $c^* = \infty$, then there does not exist any visit order so that agent a can visit its complex targets in the order given by permutation π at visit times $\mathbf{t}_a^{\pi}$, that is, $\mathscr{B}(\mathbf{t}_a^{\pi} - \mathbf{t}_p) = \mathbf{false}$ for each visit order $p \in \mathcal{P}_{a,\pi}$. Thus, $c^* = \min_{p \in \mathcal{P}_{a,\pi}} \mathcal{F}_{a,p}^{\pi}(\mathbf{t}_a)$. If $c^* \neq \infty$, then let p^* be a visit order of agent a that results in agent cost c^*. We let $\mathbf{w}_a = (w_a^{\pi(1)}, \ldots, w_a^{\pi(n_a)})$ be the vector of total wait times of agent a at complex targets, where $w_a^{\pi(i)}$ is the total wait time of agent a at all complex targets $x_a^{\pi(j)}$ with $1 \leq j \leq i$. Consider the meta-function $\mathcal{F}_{a,p^*}^{\pi}(\mathbf{t}_a^{\pi})$ of agent a for visit order p^*. We have $t_a^{\pi(n_a)} = t_{p^*}^{\pi(n_a)} + w_a^{\pi(n_a)}$ and $\mathcal{F}_{a,p^*}^{\pi}(\mathbf{t}_a^{\pi}) = c_{p^*} + t_a^{\pi(n_a)} - t_{p^*}^{\pi(n_a)} = c_{p^*} + w_a^{\pi(n_a)} = c^*$. Since $p^* \in \mathcal{P}_{a,\pi}$, we have $c^* = \mathcal{F}_{a,p^*}^{\pi}(\mathbf{t}_a^{\pi}) \geq \min_{p \in \mathcal{P}_{a,\pi}} \mathcal{F}_{a,p}^{\pi}(\mathbf{t}_a^{\pi})$. Given that c^* is the smallest sum of travel and wait times of agent a, by definition $\min_{p \in \mathcal{P}_{a,\pi}} \mathcal{F}_{a,p}^{\pi}(\mathbf{t}_a^{\pi})$ cannot be less than c^*. Thus, $c^* = \min_{p \in \mathcal{P}_{a,\pi}} \mathcal{F}_{a,p}^{\pi}(\mathbf{t}_a^{\pi})$. ■

Theorem 1 *The agent cost of agent a for visiting its complex targets $X_a^c = \{x_a^1, \ldots, x_a^{n_a}\}$ at any given visit times $\mathbf{t}_a = (t_a^1, \ldots, t_a^{n_a})$ can be calculated from its reaction function $\mathcal{F}_a^{X_a^c}(\mathbf{t}_a)$.*

permutations	scenarios
	a_1 does not wait at x_1, a_1 does not wait at x_2, a_2 does not wait at x_3
(x_1, x_2, x_3)	a_1 does not wait at x_1, a_1 does not wait at x_2, a_3 does not wait at x_3
(x_1, x_3, x_2)	a_1 does not wait at x_1, a_3 does not wait at x_2, a_2 does not wait at x_3
(x_2, x_1, x_3)	a_1 does not wait at x_1, a_3 does not wait at x_2, a_3 does not wait at x_3
(x_2, x_3, x_1)	a_2 does not wait at x_1, a_1 does not wait at x_2, a_2 does not wait at x_3
(x_3, x_1, x_2)	a_2 does not wait at x_1, a_1 does not wait at x_2, a_3 does not wait at x_3
(x_3, x_2, x_1)	a_2 does not wait at x_1, a_3 does not wait at x_2, a_2 does not wait at x_3
	a_2 does not wait at x_1, a_3 does not wait at x_2, a_3 does not wait at x_3

Table 2: Permutations and Scenarios

Proof: Sort the visit times $\mathbf{t}_a = (t_a^1, \ldots, t_a^{n_a})$ in increasing order to yield $\mathbf{t}_a^\pi = (t_a^{\pi(1)}, \ldots, t_a^{\pi(n_a)})$, which results in permutation $\pi = (x_a^{\pi(1)}, \ldots, x_a^{\pi(n_a)})$. Lemma 1 states that the agent cost of agent a for visiting its complex targets $(x_a^{\pi(1)}, \ldots, x_a^{\pi(n_a)})$ at visit times $\mathbf{t}_a^\pi = (t_a^{\pi(1)}, \ldots, t_a^{\pi(n_a)})$ can be calculated as the value of the permutation function $\mathcal{F}_a^\pi(\mathbf{t}_a^\pi)$, which is also the agent cost of agent a for visiting its complex targets $(x_a^1, \ldots, x_a^{n_a})$ at visit times $\mathbf{t}_a = (t_a^1, \ldots, t_a^{n_a})$. ■

3.3 Determining Optimal Visit Times

Given the reaction function $\mathcal{F}_a^{X_a^c}(\mathbf{t}_a)$ of each agent a in the bundle b of related agents, the optimal visit times of complex targets $\mathbf{x}_b = (x_b^1, \ldots, x_b^{n_b})$ that are assigned to bundle b can be determined as $\arg\min_{\mathbf{t}_b \in \mathbb{R}_+^{n_b}} c_{a\in b}^{team} \mathcal{F}_a^{X_a^c}(f_a(\mathbf{t}_b))$. The minimization is over $\mathbb{R}_+^{n_b}$ (= all possible visit times of the n_b complex targets). The following theorem, however, shows that this is unnecessary.

Theorem 2 *Let T be the set of vectors of visit times $\mathbf{t}_b$ at which the agents in bundle b are able to visit the complex targets $\mathbf{x}_b$ without waiting at any simple target and for each complex target in $\mathbf{x}_b$ there exists at least one agent in its coalition that does not wait at it. Then,*

$$\min_{\mathbf{t}_b \in T} c_{a\in b}^{team} \mathcal{F}_a^{X_a^c}(f_a(\mathbf{t}_b)) = \min_{\mathbf{t}_b \in \mathbb{R}_+^{n_b}} c_{a\in b}^{team} \mathcal{F}_a^{X_a^c}(f_a(\mathbf{t}_b))$$

Proof: Let $\mathbf{t}_b^*$ be the optimal visit times of complex targets $\mathbf{x}_b$. If there exists any simple target at which its assigned agent waits or any complex target at which all agents in its coalition wait, run the following procedure on this target x: First, find the smallest wait time $s_x > 0$ of all agents at target x. Then, let each agent visit target x s_x time units earlier. The resulting team cost does not increase since all agents visit their targets no later than before. Repeat this procedure until agents do not wait at any simple target and for each complex target there exists at least one agent in its coalition that does not wait at it. The resulting visit times belong to set T, and the resulting team cost is no larger than that of the optimal visit times $\mathbf{t}_b^*$. ■

Now we describe how a central planner can construct the set T based on the reaction functions of agents:

1. The central planner constructs all possible permutations π_b of the complex targets in $\mathbf{x}_b$.
2. The central planner constructs all possible scenarios so that, for each complex target in $\mathbf{x}_b$, there is at least one agent in its coalition that does not wait at it.

   For example, Table 2 tabulates all possible permutations and scenarios for the multi-agent routing problem from Figure 1.
3. For each combination of a scenario and a permutation $\pi_b = (x_b^{\pi_b(1)}, \ldots, x_b^{\pi_b(n_b)})$, the central planner generates a search tree for constructing the visit times $\mathbf{t}_b^{\pi_b} = (t_b^{\pi_b(1)}, \ldots, t_b^{\pi_b(n_b)})$ that belong to set T, starting with the root node $(t_b^{\pi_b(1)} \geq 0, \ldots, t_b^{\pi_b(n_b)} \geq t_b^{\pi_b(n_b-1)})$ at depth 0. The depth of the search tree is n_b. Each node at any depth i with $1 \leq i \leq n_b$ has the form $(t_b^{\pi_b(1)}, \ldots, t_b^{\pi_b(i)}, t_b^{\pi_b(i+1)} \geq t_b^{\pi_b(i)}, \ldots, t_b^{\pi_b(n_b)} \geq t_b^{\pi_b(n_b-1)})$, where the visit times $(t_b^{\pi_b(1)}, \ldots, t_b^{\pi_b(i)})$ have been determined from the previous depths. Consider any node $(t_b^{\pi_b(1)}, \ldots, t_b^{\pi_b(i)}, t_b^{\pi_b(i+1)} \geq t_b^{\pi_b(i)}, \ldots, t_b^{\pi_b(n_b)} \geq t_b^{\pi_b(n_b-1)})$ at depth i. We now show how the central planner determines the visit time $t_b^{\pi_b(i+1)}$ at depth $i+1$: Assume that agent a does not wait at complex target $x_b^{\pi_b(i+1)}$ in the given scenario. First, the central planner constructs the permutation $\pi_a = f_a(\pi_b)$ of the complex targets assigned to agent a and finds the complex target $x_a^{\pi_a(j)} \in \pi_a$ that is identical to $x_b^{\pi_b(i+1)} \in \pi_b$. Then, for each visit order $p \in \mathcal{P}_{a,\pi_a}$ of agent a, it determines the visit time $t_a^{\pi_a(j)}$ of complex target $x_a^{\pi_a(j)}$ to be $t_a^{\pi_a(j-1)} + t_p^{\pi_a(j)} - t_p^{\pi_a(j-1)}$, where $t_a^{\pi_a(j-1)}$ is the visit time of complex target $x_a^{\pi_a(j-1)}$ and $t_p^{\pi_a(j)} - t_p^{\pi_a(j-1)}$ is the travel time of agent a from complex target $x_a^{\pi_a(j-1)}$ to $x_a^{\pi_a(j)}$ in visit order p. Finally, it generates one child node in which $t_b^{\pi_b(i+1)} = t_a^{\pi_a(j)}$ and labels the edge from the node in question to the child node with the domain constraint $\mathscr{B}(\mathbf{t}_a^{\pi_a} - \mathbf{t}_p) = \mathbf{true}$ of meta-function $\mathcal{F}_{a,p}^{\pi_a}(\mathbf{t}_a^{\pi_a})$. If the determined visit times in the child node are inconsistent with any domain constraint labeling the edges from the root to the child node, the central planner removes the child node from the search tree. This completes the construction of the search tree. The central planner then adds the vectors of visit times in the leaf nodes of the search tree to set T.

   Consider again the multi-agent routing problem from Figure 1. Assume that the permutation of complex targets is (x_1, x_2, x_3) and the scenario is that a_2 does not wait at x_1, a_1 does not wait at x_2 and a_3 does not wait at x_3. Figure 3 shows the search tree constructed by the central planner, and Figure 4 shows the corresponding meta-functions of agents used to determine the visit times.

3.4 Approximating Reaction Functions

The number of meta-functions in the reaction function of agent a usually depends on the number of visit orders of agent a for visiting its assigned targets, which is usually exponential in the number of its assigned targets. Thus, the computation of the reaction function is time-intensive. Now we show how to approximate reaction functions by considering only a constant number of visit orders, similar to the approach used in [Zheng and Koenig, 2008]:

1. Agent a finds a sufficiently large visit time e so that it can visit all assigned (both simple and complex) tar-

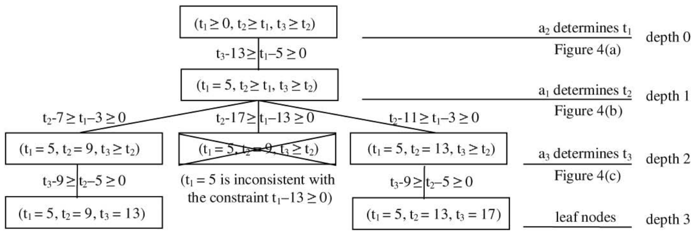


Figure 3: Search Tree

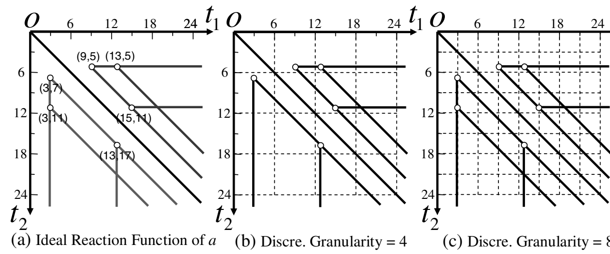


Figure 5: Approximating Reaction Functions of a_1

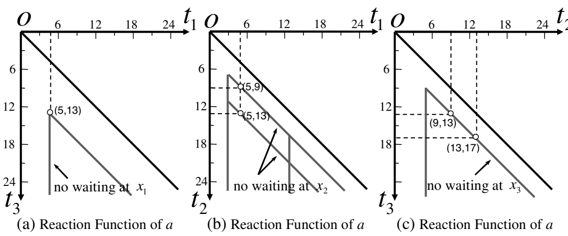


Figure 4: Determining Optimal Visit Times

gets in any visit order before time e. For example, $\sum_{x \in X_a^s \cup X_a^c} 2 \cdot d(a,x) + \epsilon$ could be a simple choice of e, where $d(a,x)$ is the travel time of agent a from its initial location to target $x \in X_a^s \cup X_a^c$ and ϵ is a small positive constant. Then, agent a divides the time interval $[0,e)$ evenly into $\mathcal{G}$ time intervals $[s_j, e_j)$ for a given discretization granularity $\mathcal{G}$. Lastly, agent a constructs all possible vectors of n_a time intervals $([s^1,e^1),\ldots,[s^{n_a},e^{n_a}))$, where the value of each $[s^i,e^i)$ in the vector can be $[s_j,e_j)$ for any $1 \leq j \leq \mathcal{G}$.

2. Agent a solves the following optimization problem for each vector of time intervals $([s^1,e^1),\ldots,[s^{n_a},e^{n_a}))$, namely, to find the visit order p of its targets $X_a^s \cup X_a^c$ with the minimal agent cost that satisfies the following two constraints: 1) it visits complex target x_a^i at time $t_p^i \in [s^i,e^i)$ and 2) it does not wait at any target.[1] (Such visit order may not always exist for any given vector of time intervals.) Let the set $\mathcal{P}'_a$ be the set of the resulting visit orders p found for all possible vectors of time intervals $([s^1,e^1),\ldots,[s^{n_a},e^{n_a}))$.

3. The remaining steps of constructing the approximate reaction function are identical to Steps 2-6 of constructing the reaction function as described in Section "Constructing Reaction Functions" if one replaces P_a with P'_a everywhere.

Approximate reaction functions are identical to the ideal ones if agent a uses an infinite number of time intervals ($\mathcal{G} = \infty$). In general, however, approximate reaction functions may not be identical to the ideal reaction functions. For example, Figure 5 shows two approximate reaction functions obtained with $e = 24$ and different discretization granularities.

[1]This optimization problem is a special case of the NP-hard traveling salesperson problem with time windows [Desrosiers *et al.*, 1995]. We use the Or-opt heuristic [Or, 1976] to solve it approximately.

4 Auctions with Reaction Functions

Now, we describe a greedy auction-like algorithm that makes use of reaction functions to allocate targets to distributed agents in a hill-climbing fashion, similar to [Koenig *et al.*, 2007; Zheng and Koenig, 2008]. The agents are the bidders, and a central planner is the auctioneer. The algorithm consists of multiple rounds to allocate all targets to agents. Initially, all targets are unassigned. During each round, all agents bid on all unassigned targets (bidding phase) and the auctioneer assigns one additional target to a coalition of agents (winner determination phase) so that the team cost increases least (= hill-climbing principle). Consider any round of the algorithm. Assume that the current allocation of agent $a \in A$ is (X_a^s, X_a^c, C_a) and the set of unassigned targets is U. We now explain how the agents bid and how the auctioneer determines the winning bid:

- **Bidding Phase** For each unassigned simple target $x^s \in U$, agent a constructs and bids the following reaction function: (Case 1) If $X_a^c \neq \emptyset$, then it constructs a reaction function $\mathcal{F}_{a,x^s}^{X_a^c}(\mathbf{t}_a) = c_a^{agent}(X_a^s \cup \{x^s\}, X_a^c, \mathbf{x}_a \leftarrow \mathbf{t}_a)$, where $\mathbf{x}_a$ is the vector of complex targets in X_a^c and $\mathbf{t}_a$ is the corresponding vector of visit times; (Case 2) If $X_a^c = \emptyset$, then it constructs a (trivial) reaction function $\mathcal{F}_{a,x^s}^{\emptyset}(\cdot) = c_a^{agent}(X_a^s \cup \{x^s\}, \emptyset, \emptyset)$.

  For each unassigned complex target $x^c \in U$, agent a constructs and bids a reaction function $\mathcal{F}_{a,x^c}^{X_a^c}(\mathbf{t}_a) = c_a^{agent}(X_a^s, X_a^c \cup \{x^c\}, \mathbf{x}_a \leftarrow \mathbf{t}_a)$, where $\mathbf{x}_a$ is the vector of complex targets in $X_a^c \cup \{x^c\}$ and $\mathbf{t}_a$ is the corresponding vector of visit times.

- **Winner Determination Phase** The auctioneer needs to keep records of the following information for each agent $a \in A$: 1) its current allocation (X_a^s, X_a^c, C_a); 2) its current agent cost $c_a^{agent}(X_a^s, X_a^c, C_a)$; and 3) its current reaction function $\mathcal{F}_a^{X_a^c}(\mathbf{t}_a)$ if $X_a^c \neq \emptyset$. Remember that allocation algorithms for complex tasks need to solve two subproblems as described in the introduction:

  Solution of Subproblem 1: First, the auctioneer constructs the set $O(d(x))$ that contains all coalitions of $d(x)$ different agent(s) for visiting each unassigned target $x \in U$. Second, for each coalition $o \in O(d(x))$, the auctioneer constructs the bundle $b(x,o) \subseteq A$ of agents that are related to agents in o after allocating target x to coalition o. Let $\mathbf{x}_{b(x,o)}$ be the vector of all complex targets assigned to bundle $b(x,o)$ and $\mathbf{t}_{b(x,o)}$ be the corresponding vector of visit times. Third, the auctioneer determines the **evaluation cost** $c^{eval}_{x,o}$ of allocating each target $x \in U$ to each coalition $o \in O(d(x))$ and the optimal visit times $\mathbf{t}^*_{b(x,o)}$ for complex targets in $\mathbf{x}_{b(x,o)}$ as explained below in the solution of Subproblem 2. Fourth, the auctioneer determines $(x^*, o^*) = \arg\min_{x \in U, o \in O(d(x))} c^{eval}_{x,o}$. Finally, the auctioneer allocates the target x^* to all agent(s) in o^* and sets the visit times of all complex targets $\mathbf{x}_{b(x^*,o^*)}$ to $\mathbf{t}^*_{b(x^*,o^*)}$.

Solution of Subproblem 2: For each agent $a \in b(x,o)$, the auctioneer constructs the following **evaluation function**:

$$\mathcal{V}^{X^c_a}_a(\mathbf{t}_a) := \begin{cases} \mathcal{F}^{X^c_a}_a(\mathbf{t}_a) - c^{agent}_a(X^s_a, X^c_a, C_a) & \text{(MiniSum)} \\ \mathcal{F}^{X^c_a}_a(\mathbf{t}_a) & \text{(MiniMax)} \end{cases}$$

where the reaction function $\mathcal{F}^{X^c_a}_a(\mathbf{t}_a)$ is either the reaction function $\mathcal{F}^{X^c_a}_{a,x}(\mathbf{t}_a)$ submitted by agent a in the bidding phase if agent $a \in o$, or the reaction function $\mathcal{F}^{X^c_a}_a(\mathbf{t}_a)$ recorded by the auctioneer if agent $a \notin o$. Let $n_{b(o,x)}$ be the number of complex targets in $\mathbf{x}_{b(o,x)}$. Then, the auctioneer determines the evaluation cost $c^{eval}_{o,x}$ to be $\min_{\mathbf{t}_{b(o,x)} \in \mathbb{R}_+^{n_{b(o,x)}}} c^{team}_{a \in b(o,x)} \mathcal{V}^{X^c_a}_a(f_a(\mathbf{t}_{b(o,x)}))$ and $\mathbf{t}^*_{b(o,x)}$ to be the corresponding optimal visit times.

Variants: All simple targets are assigned to agents before any complex target in [Zheng and Koenig, 2008]. However, this is not necessary for our approach with generalized reaction functions since the agents are able to update the visit times of complex targets that have already been assigned to them when a new target is assigned. In order to better evaluate our approach, we use the following three variants of the auction-like algorithm: **V.1 (Mixed)** There is no restriction on the auction-like algorithm. **V.2 (Simple-First)**: All simple targets are assigned before any complex target. **V.3 (Complex-First)**: All complex targets are assigned before any simple target.

5 Experimental Results

We now evaluate the benefits of generalized reaction functions. Since the algorithm ARF introduced in [Zheng and Koenig, 2008] is the only existing allocation algorithm in the literature for multi-agent routing with complex targets, we compare all three variants of the auction-like algorithm introduced in the previous section against it. We use multi-agent routing problems on a known four-neighbor planar grid of size 51×51 with square cells that are either blocked or unblocked. The grid resembles an office environment with walls and doors [Koenig *et al.*, 2007]. Table 3 tabulates the team costs and runtimes of allocation algorithms for solving multi-agent routing problems with 10 agents, 40 simple targets and 5 complex targets. The coalition size of each complex target is 2. Each agent can be assigned at most four simple targets and three complex targets. We report data that has been averaged over 25 instances with randomly generated locations of agents and targets. We make the following observations: First, the team costs of the auction-like algorithms are significantly smaller than those of ARF for both the MiniSum and MiniMax team objectives since agents can update the visit times of their assigned complex targets when a new target is assigned. Second, the runtimes of the auction-like algorithms are still small, although they increase significantly as the discretization granularity $\mathcal{G}$ increases. Third, the team costs and runtimes of the "Mixed" variant are very similar to those of the "Simple-First" variant for the MiniSum team objective. The reason is that simple targets tend to be allocated before complex targets in the "Mixed" variant for the MiniSum team objective since the team cost typically increases less when allocating an additional simple target to some agent than allocating an additional complex target to a coalition of agents.

Allocation Algorithm		MiniSum		MiniMax	
		Team Cost	Runtime	Team Cost	Runtime
ARF [Zheng and Koenig, 2008]		824.16	0.04	162.12	0.03
Mixed	Ideal	717.64	0.38	150.26	0.25
	$\mathcal{G} = 30$	718.44	0.20	150.39	0.19
	$\mathcal{G} = 15$	719.12	0.09	152.60	0.07
Simple-First	Ideal	717.06	0.37	137.18	0.28
	$\mathcal{G} = 30$	717.41	0.24	137.24	0.20
	$\mathcal{G} = 15$	718.16	0.11	137.44	0.12
Complex-First	Ideal	726.48	0.52	151.39	0.43
	$\mathcal{G} = 30$	727.84	0.38	154.40	0.22
	$\mathcal{G} = 15$	733.92	0.20	155.32	0.15

Table 3: Experimental Results

6 Conclusions

We studied how to improve the team performance of allocation algorithms for complex tasks. We first generalized the concept of reaction functions proposed in the literature to characterize the agent costs of performing multiple complex tasks at given times. Second, we showed how agents can construct and approximate reaction functions in a distributed way. Third, we showed how an auction-like algorithm can use reaction functions to allocate tasks to agents. Finally, we showed empirically that the team performance of our algorithm is substantially better than those of an existing state-of-the-art allocation algorithm for complex tasks.

References

[Desrosiers *et al.*, 1995] J. Desrosiers, Y. Dumas, M. Solomon, and F. Soumis. Time constrained routing and scheduling. In *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*, chapter 2, pages 35–139. 1995.

[Dias *et al.*, 2006] M. Dias, R. Zlot, N. Kalra, and A. Stentz. Market-based multirobot coordination: a survey and analysis. *Proceedings of the IEEE, Special Issue on Multirobot Coordination*, 94(7):1257 – 1270, 2006.

[Koenig *et al.*, 2007] S. Koenig, C. Tovey, X. Zheng, and I. Sungur. Sequential bundle-bid single-sale auction algorithms for decentralized control. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1359–1365, 2007.

[Or, 1976] I. Or. *Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking*. PhD thesis, Northwestern University, Evanston, Illinois, 1976.

[Zheng and Koenig, 2008] X. Zheng and S. Koenig. Reaction functions for task allocation to cooperative agents. In *Proceedings of the International Joint Conference on Autonomous Agents and MultiAgent Systems*, pages 559–566, 2008.