2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

# Adaptive Multi-Robot Coordination: A Game-Theoretic Perspective

Gal A. Kaminka, Dan Erusalimchik, and Sarit Kraus
Computer Science Department
Bar Ilan University, Israel

***Abstract*— Multi-robot systems researchers have been investigating adaptive coordination methods for improving spatial coordination in teams. Such methods adapt the coordination method to the dynamic changes in density of the robots. Unfortunately, while their empirical success is evident, none of these methods has been understood in the context of existing formal work on multi-robot learning. This paper presents a reinforcement-learning approach to coordination algorithm selection, which is not only shown to work well in experiments, but is also analytically grounded. We present a reward function (*Effectiveness Index*, EI), that reduces time and resources spent coordinating, and maximizes the time between conflicts that require coordination. It does this by measuring *the resource-spending velocity*. We empirically show its success in simulations of multi-robot foraging. In addition, we analytically explore the reasons that EI works well. We show that under some assumptions, spatial coordination opportunities can be modeled as matrix games in which the payoffs are directly a function of EI estimates. The use of reinforcement learning leads to robots maximizing their EI rewards in equilibrium. This work is a step towards bridging the gap between the theoretical study of interactions, and their use in multi-robot coordination.**

## I. INTRODUCTION

Multi-robot systems researchers have been investigating distributed coordination methods for improving spatial coordination in teams [7], [15], [14]. Such methods attempt to resolve spatial conflicts between team-members, e.g., by dynamic setting of right-of-way priorities [17], [20], territorial separation [16], [5], [10], or role-based priorities [12]. It is accepted that no one method is always best [6], [14], and that all methods reach a point where adding robots to the group (i.e., increasing the density of the robots in space) reduces overall productivity [16], [15].

There is thus growing interest in adaptive coordination approaches, in which each robot adapts the coordination method to the dynamic changes in density. For instance, Zuluaga and Vaughan adjust the right-away priorities based on the amount of local effort (or investment) by team-members [20]. Rosenfeld et al. [14] advocated allowing each robot to individually switch coordination methods to reduce its own estimated resource costs. In general, these adaptive methods have demonstrated much success in multiple domains of interest.

Unfortunately, while their empirical success is evident, none of these methods have ever been analytically proven to work, nor understood in the context of existing formal work on multi-robot learning and adaptation. As a result, their optimality and the appropriate conditions for their use remain open questions. Put simply, they pose a puzzle: These are methods that work well in practice—both in simulations and with real robots—but the reasons for their success remain elusive.

This paper presents a reinforcement-learning approach to coordination algorithm selection, which is not only shown to work well in experiments, but also explored analytically. The reward function used as the basis for the learning is called *Effectiveness Index* (EI). The key idea in EI is to reduce time and resources spent coordinating, and maximize the time between conflicts that require coordination. It does this by measuring *the resource-spending velocity* (the resource "burn rate"). The use of reinforcement learning minimizes this velocity.

We empirically and analytically evaluate the use of EI. We empirically show that EI succeeds in improving multi-robot coordination in simulated multi-robot foraging. We then analytically explore the reasons and assumptions underlying this success. We formalize foraging as extensive-form games. We show that under some assumptions, these games can be modeled as matrix games in which the payoffs to the robots are unknown, but are directly a function of EI estimates. The use of reinforcement learning leads to robots maximizing their EI rewards in equilibrium. We believe that this work represents a step towards bridging the gap between the theoretical study of interactions (via game theory), and their use to explain and inform multi-robot coordination.

## II. RELATED WORK

Earlier work on adaptation based on coordination effort is closely related. Vaughan et al. [17] presented a method called *aggression* for dynamic coordination. When robots come too close to each other, each of the robots chooses an aggression level; the robot with the lower level concedes its position, preventing a collision. Later, Zuluaga and Vaughan [20] have shown that choosing aggression level proportional to the robot's task investment can further improve overall system performance. In contrast to this work, our method is based on measuring the robot's investment in coordination. In addition, we cast adaptive coordination as a reinforcement-learning problem.

Rosenfeld et al. [14] presented the Combined Coordination Cost (*CCC*) method that adapts the selection of coordination methods by robots; however, it ignores the gains accumulated from long periods of no coordination needs, in contrast to our work. Similarly to our work, the adaptation is stateless, i.e., has no mapping from world state to actions/methods. Instead, the CCC is estimated at any given point, and once it passes pre-learned (learned offline) thresholds, it causes dynamic re-selection of the coordination methods by each individual robot, attempting to minimize the CCC. In contrast, all our learning and adaption is done on-line.

Most investigations of reinforcement learning in multi-robot settings have focused on improving the learning mechanisms (e.g., modifying the basic Q-learning algorithm), and utilized task-specific reward functions. We briefly discuss these below. Two recent surveys are provided in [19], [8].

Matarić [11] discusses several techniques for using rewards in multi-robot Q-learning: A local performance-based reward, a global performance-based reward, and a heuristic strategy referred to as shaped reinforcement; it combines rewards based on local rewards, global rewards and coordination interference of the robots. Balch [2] reports on using reinforcement learning in individual robot behavior selection. The rewards for the selection were carefully selected for each domain and application, in contrast to our work. In contrast to these investigations, we explore a domain-independent reward function, based on minimizing resource use, and use them in selecting between coordination methods, rather than task behaviors.

Wolpert et al. [18] developed the COIN reinforcement-learning framework. Each agent's reward function is based on *wonderful life utility*, the difference between the group utility with the agent, and without it. Similarly to these our study focuses on the reward function, rather than the learning algorithm; and similarly, we focus on functions that are *aligned* with global group utility. However, our work differs in several ways. First, we distinguish utility due to coordination, from utility due to task execution. Second, our reward function distinguishes also the time spent coordinating and time spent executing the task.

## III. Limiting Resource Spending

We first cast the problem of selecting coordination algorithms as a reinforcement learning problem (Section III-A). We then introduce the effective index (EI) reward function in Section III-B.

### A. Coordination Algorithm Selection

Multilateral coordination prevents and resolves conflicts among robots in a multi-robot system (MRS). Such conflicts can emerge as results for shared resource (e.g., space), or as a result of violation of joint decisions by team-members. Many distributed coordination algorithms (protocols) have been proposed and explored by MRS researchers [5], [12], [16], [17]. Not one method is good for all cases and group sizes [14]. However, deciding on a coordination method for use is not a trivial task, as the effectiveness of coordination methods in a given context is not known in advance.

We focus here on loosely-coupled application scenarios where coordination is triggered by conflict situations, identified through some mechanism (we assume that such a mechanism exists, though it may differ between domains; most researchers simply use a pending collision as a trigger). Thus the normal routine of a robot's operation is to carry out its primary task, until it is interrupted by an occurring or potentially-occurring conflict with another robot, which must be resolved by a coordination algorithm. Each such interruption is called *a conflict event*. The event triggers a coordination algorithm to handle the conflict. Once it successfully finishes, the robots involved go back to their primary task.

There are common themes that run through all these tasks: (i) loose coordination between the robots (i.e., only occasional need for spatial or temporal coordination); (ii) a cooperative task (the robots seek to maximize group utility); and (iii) the task is bound in time. We refer to these tasks as *LCT tasks* (Loose-coordination, Cooperative, Timed tasks).

Example LCT tasks include multi-robot foraging, search and exploration, and making deliveries. For instance, in multi-robot foraging, robots execute their individual roles (seeking pucks and retrieving them) without any a-priori coordination. When they become too close to each other, they need to spatially coordinate. The robot all contribute to the team goal, of maximizing the number of pucks retrieved. Moreover, they have limited time to do this. In multi-robot exploration, execution follows a similar pattern: robots spread around, avoiding each other or coordinating as needed (e.g., to decide who is to explore a newly-discovered area); they have the goal of completely exploring a new area as quickly as possible.

Let $A = \{\ldots, a_i, \ldots\}, 1 \le i \le N$ be a group of $N$ robots, cooperating on a group task that started at time 0 (arbitrarily) lasts up-to time $T$ ($A$ starts working and stops working on the task together). We denote by $T_i = \{c_{i,j}\}, 0 \le j \le K_i$ the set of conflict events for robot $i$, where $c_{i,j}$ marks the time of the beginning of each conflict.

The time between the beginning of a conflict event $j$, and up until the next event, the interval $I_{i,j} = [c_{i,j}, c_{i,j+1})$, can be broken into two conceptual periods: The *active* interval $I^a_{i,j} = [c_{i,j}, t_{i,j})$ (for some $c_{i,j} < t_{i,j} < c_{i,j+1}$) in which the robot was actively investing resources in coordination, and the *passive* interval $I^p_{i,j} = [t_{i,j}, c_{i,j+1})$ in which the robot no longer requires investing in coordination; from its perspective the conflict event has been successfully handled, and it is back to carrying out its task. By definition $I_{i,j} = I^a_{i,j} + I^p_{i,j}$. We define the *total active time* as $I^a = \sum_i \sum_j I^a_{i,j}$ and the *total passive time* as $I^p = \sum_i \sum_j I^p_{i,j}$.

Our research focuses on a case where the robot has a nonempty set $M$ of coordination algorithms to select from. The choice of a specific coordination method $\alpha \in M$ for a given conflict event $c_{i,j}$ may effect the active and passive intervals $I^a_{i,j}, I^p_{i,j}$ (and possibly, other conflicts; see next section). To denote this dependency we use $I^a_{i,j}(\alpha)$,$I^p_{i,j}(\alpha)$ as active and passive intervals (respectively), due to using coordination method $\alpha$. Figure 1 illustrates this notation.

We define the problem of decentralized coordination algorithm selection in terms of reinforcement learning. We assume each robot tries to maximize its own reward by selecting a coordination method $\alpha$. Typically, reward functions are given, and indeed most previous work focuses on learning algorithms that use the reward functions as

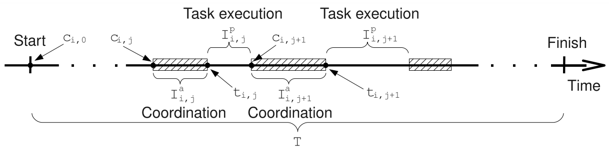



Fig. 1. Illustration of task time-line, from the robots' perspective. Task execution is occasionally interrupted by the requirement to spend resources on coordination.

efficiently as possible. Instead, we assume a very simple Q-Learning variant, and instead focus on defining a reward function (see below).

### B. Effectiveness Index

We call the proposed general reward for coordination *Effectiveness Index* (EI). Its domain independence is based on its using three intrinsic (rather than extrinsic) factors in its computation; these factors depend only on internal computation or measurement, rather than environment responses.

**III-B.1 The cost of coordinating.** The first factor we consider is the cost of internal resources (other than time) used by the chosen method. This is especially important in physical robots, where battery life and power are a concern.

We denote by $C_i^C$ the total cost of coordination, of robot $i$. It can be broken into the costs spent on resolving all conflicts $C_i^C = \sum_j C_{i,j}^C$. $C_{i,j}^C$ is similar to other measures suggested previously, but excludes the cost of time and resources spent before the conflict (unlike [14]), and is limited to only considering individual intrinsic resources (unlike [20]).

Let us use a cost function $cost_i(\alpha, t)$ to represent the costs due to using coordination method $\alpha \in M$ at any time $t$ during the lifetime of the robot. The function is not necessarily known to us a-priori (and indeed, in this research, is not).

Using the function $cost_i(\alpha, t)$ we define the $C_{i,j}^C$ of a particular event of robot $i$ at time $c_{i,j}$:

$$\begin{aligned} C_{i,j}^C(\alpha) &= \int_{c_{i,j}}^{t_{i,j}} cost_i(\alpha,t)\ \mathrm{d}t + \int_{t_{i,j}}^{c_{i,j+1}} cost_i(\alpha,t)\ \mathrm{d}t \\ &= \int_{c_{i,j}}^{t_{i,j}} cost_i(\alpha,t)\ \mathrm{d}t \end{aligned} \tag{1}$$

$C_{i,j}^C$ is defined as the cost of applying the coordination algorithm during the active interval $[c_{i,j}, t_{i,j})$ and the passive interval $[t_{i,j}, c_{i,j+1})$. However, the coordination costs during the passive interval are zero by definition.

**III-B.2 The time spent coordinating.** The main goal of a coordination algorithm is to reach a (joint) decision that allows all involved robots to continue their primary activity. Therefore, the sooner the robot returns to its main task, the less time is spent on coordination, and likely, the robot can finish its task more quickly. Thus, smaller $I_i^a$ is better. Note that this is true regardless of the use of other resources (which are measured by $C_i^C$). Even if somehow other resources were free, effective coordination would minimize conflict-resolution time.

We thus define the *Active Coordination Cost* (ACC) function for robot $i$ and method $\alpha$ at time $c_{i,j}$, that considers the *active time* in the calculation of coordination resources cost:

$$ACC_{i,j}(\alpha) \equiv I_{i,j}^a(\alpha) + C_{i,j}^C(\alpha) \tag{2}$$

**III-B.3 The frequency of coordinating.** If there are frequent interruptions to the robot's task in order to coordinate, even if short-lived and inexpensive, this would delay the robot. We assume (and the results show) that good coordination decisions lead to long durations of non-interrupted work by the robot. Therefore, the frequency of coordination method's use is not less important than the time spent on conflict resolving. Thus, larger $I_{i,j}^p$ is better.

We thus want to balance the total active coordination cost $ACC_i = \sum_j ACC_{i,j}$ against the frequency of coordination. We want to balance short-lived, infrequent calls to an expensive coordination method against somewhat more frequent calls to a cheaper coordination method.

We therefore define the Effectiveness Index of robot $i$, of conflict $j$, due to using coordination method $\alpha \in M$ as follows:

$$EI_{i,j}(\alpha) \equiv \frac{ACC_{i,j}(\alpha)}{I_{i,j}^a(\alpha) + I_{i,j}^p(\alpha)} = \frac{I_{i,j}^a(\alpha) + C_{i,j}^C(\alpha)}{I_{i,j}^a(\alpha) + I_{i,j}^p(\alpha)} \tag{3}$$

That is, the effectiveness index (EI) of a coordination method $\alpha$ during this event is the velocity by which it spends resources during its execution, amortized by how long a period in which no conflict occurs. Since greater EI signifies greater costs, we typically put a negation sign in front of the EI, to signify that greater velocity is worse; we seek to minimize resource spending velocity.

In this paper we use the simple single-state Q-learning algorithm to estimate the EI values from the robot's individual perspective. The learning algorithm we use is stateless:

$$Q_t(a) = Q_{t-1}(a) + \rho(R_t(a) - \gamma Q_{t-1}(a))$$

where $\rho$ is the learning speed factor, $\gamma$ is a factor of discounting, and $\beta$ is an exploration rate.

## IV. Experiments in Multi-Robot Foraging

We now turn to briefly survey a subset of experiment results, in simulated foraging, supporting the use of EI in multi-robot team tasks. Due to lack of space, we only provide representative results.

Foraging is a canonical task in multi-robot systems research. Here, robots locate target items (pucks) within the work area, and deliver them to a goal region. As was the case in Rosenfeld et al.'s work [14], we used the TeamBots simulator [1] to run experiments. Teambots simulated the activity of groups of Nomad N150 robots in a foraging area that measured approximately 5 by 5 meters. We used a total of 40 target pucks, 20 of which were stationary within the search area, and 20 moved randomly. For each group, we measured how many pucks were delivered to the goal region by groups of 3,5,15,25,35,39 robots within 10 and 20 minutes. We averaged the results of 16–30 trials in each group-size configuration with the robots being placed at random initial positions for each run. Thus, each experiment simulated for each method a total of about 100 trials of 10 and 20 minute intervals.

We compare the EI method with random coordination algorithm selection (RND), and to the method of Rosenfeld et al. (ACIM) (which uses offline learning [14]). Each of these selection methods selects between three types of coordination methods ($\alpha$), described also in [14]: Noise (which essentially allows the robots to collide, but increases their motion uncertainty to try to escape collisions), Aggression [17] (where one robot backs away, while the other moves forward), and Repel, in which robots move away (variable distance) to avoid an impending collision.

Figures 2(a)–2(c) show a subset of results. In all, the X axis marks the group size, and the Y axis marks the number of pucks collected. Figure 2(a) shows that given no resource limitations, the EI method is as good as ACIM (and Repel) which provides the best results, though it has not used prior off-line learning. Figure 2(b) shows the advantage of EI over ACIM when resource costs apply. Here, when ACIM takes fuel costs into account, it performs well. But when it does not, its performance is very low. On the other hand, EI with fuel costs and without perform well. Finally, Figure 2(c) shows how ACIM and EI respond to unknown costs. Here, both EI and ACIM take fuel costs into account, but the actual fuel costs are greater. EI provides significantly better performance in these settings (1-tailed t-test, $p = 0.0027$).

## V. Why Does EI Work?

We now turn to discuss the use of EI as a reward function, from an analytical perspective. We are interested in exploring the conditions under-which we expect EI to be effective. There are common themes that run through all the tasks in which EI has been successful: (i) loose coordination between the robots (i.e., only occasional need for spatial coordination); (ii) a cooperative task (the robots seek to maximize group utility); and (iii) the task is bound in time. We refer to these tasks as *LCT tasks* (Loose-coordination, Cooperative, Timed tasks).

For instance, in foraging, we see that robots execute their individual roles (seeking pucks and retrieving them) essentially without any a-priori coordination. When they become too close to each other, they need to spatially coordinate. The robot all contribute to the team goal, of maximizing the number of pucks retrieved. Moreover, they have limited time to do this. Incidentally, they also have finite number of pucks, which break some of the assumptions we make below. We shall come back to this.

Computing optimal plans of execution for tasks such as foraging is purely a theoretical exercise in the current state of the art. In practice, determining detailed trajectories for multiple robots in continuous space, with all of the uncertainties involved (e.g., pucks slipping from robots' grips, motion and sensing uncertainty), is infeasible. Much more so, when we add the a-priori selection of coordination methods in different points in time. We therefore seek alternative models with which to analytically explore LCT tasks.

### A. LCT Tasks as Extensive-Form Games

We turn to game theory to represent LCT tasks. As we have already noted, each individual robot's perspective is that its task execution is occasionally interrupted, requiring the application of some coordination method in order to resolve a spatial conflict, to get back to task execution. Assume for simplicity of the discussion that we limit ourselves to two robots, and that whenever they are in conflict, they are both aware of it, and they both enter the conflict at the same time. This is a strong assumption, as in actuality, most often LCT tasks often involve more than two robots. We address this assumption later in this section.

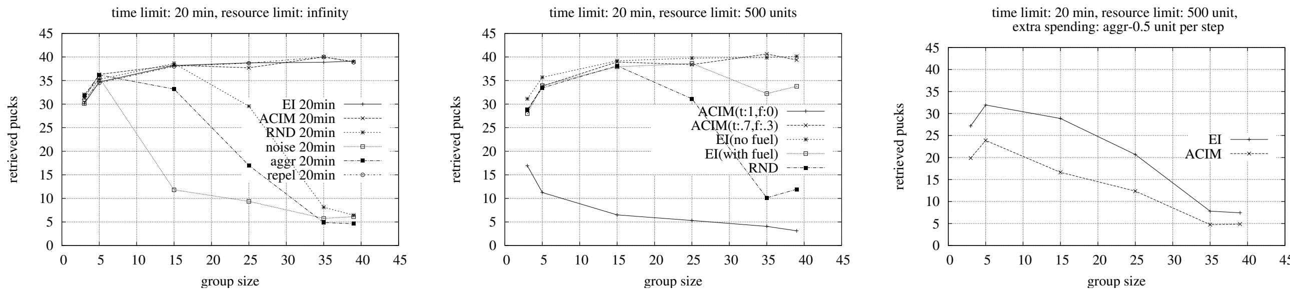
At first glance, it may seem possible to model LCT tasks as a series of single-shot games (i.e., repeating games), where in each game the actions available to each robot consist of the coordination methods available to it. The joint selection of methods by the two robots creates a combination of methods which solves the conflict (at least temporarily). The payoffs for the two robots include the pucks collected in the time between games, minus the cost of resources (including time) spent making and executing the selected methods. The fact that there exists a time limit to the LCT task in question can be modeled as a given finite horizon.

However, finite-horizon repeating games are not a good model for LCT tasks. In particular, the methods selected by the robots in one point in time affect the payoffs (and costs) at a later point in time. First, the choice of coordination methods at time $t$ affects the time of the next conflict. One coordination method may be very costly, yet reduce the likelihood that the robots get into conflict again; another method may be cheap, but cause the robots to come into conflict often. Second, the robots change the environment in which they operate during the time they are carrying out their tasks, and thus change future payoffs. For instance, robots collect pucks during their task execution time, and often collect those nearest the goal area first. Thus their payoff (in terms of pucks collected) from games later in the sequence is lower than from games earlier on.

We thus utilize a model of LCT tasks as extensive-form games. The initial node of the game tree lies at the time of the first conflict, $c_{i,1}$, and the choices of the first robot at this time lead to children of this node. As the two robots act simultaneously, these children also occur at time $c_{i,1}$. Also, note that the selections of the robots are not observable to each other[1]. The game tree is illustrated in in Figure 3.

Following each simultaneous choice of methods by the robots, the chosen combination of coordination methods is executed (during coordination time $I^a_{i,j}$), and this is followed by a period of task execution $I^p_{i,j}$. The game ends when total time $T$ runs out. The payoffs to the robots are then given as the number of pucks retrieved, minus the cost of resources spent on the task. Terminal nodes may appear anywhere in

[1]This is true in all communication-less coordination methods, used in previous work [17], [14]. When used with communication-based coordination method, this restriction may be removed. It might also be possible to relax this restriction if robots could infer each others' choices post-factum.




(a) $T = 20$, no resource limits. (b) $T = 20$, severe fuel limits. (c) $T = 20$, resource cost unknown.

Fig. 2. Results from the TeamBots foraging domain.

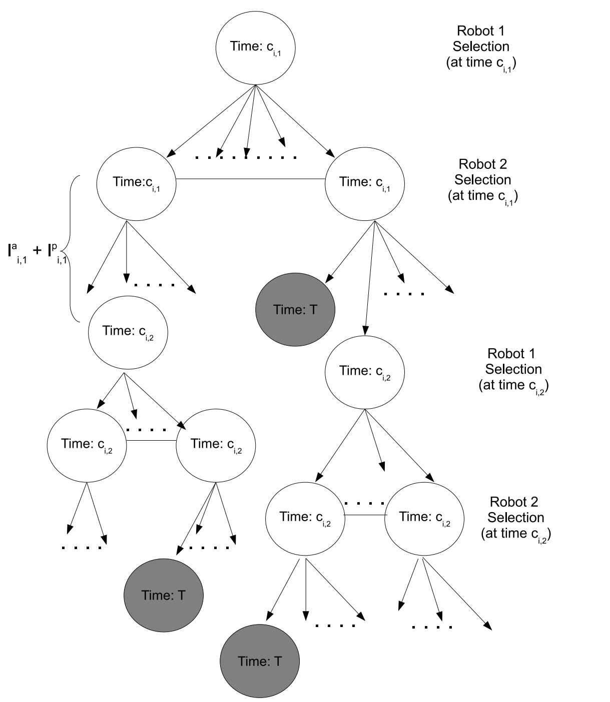



Fig. 3. An illustration of the extensive-form game tree for an LCT task. Conflict times are denoted in the nodes. Terminal nodes (total time=$T$) are dark. Note that the second conflict $c_{i,2}$ may occur at different absolute times depending on the choices of the robots at time $c_{i,1}$.

the game tree, as some selections of the robots lead to less conflicts, and thus greater opportunity for task execution.

Under ideal—and purely theoretical—conditions the robots would know the payoffs awaiting them in each terminal node, and would thus be able to, in principle, compute a game-playing strategy that would maximize the team's utility. To do this, the robots would need to know the times spent resolving conflicts and executing the task, and would also need to know (in advance) the gains achieved during each task-execution period. Even ignoring the gains, and assuming that maximizing task-execution time $\sum_i \sum_j I^p_{i,j}$ is sufficient, the robots would be required to know all conflict resolution times in advance. This is clearly impractical, as it requires predicting all possible conflicts (hundreds in a typical foraging task), their durations and effects.

### *B. Modeling LCT Tasks as a Matrix Game*

We thus make a simplifying assumption, that all effects of coordination method selections remain fixed, regardless of where they occur. In other words, we assume that the joint execution of a specific combination of selected coordination methods will always cost the same (in time and resources), regardless of the time in which the conflict occurred. Moreover, the assumption also implies that we assume that the task-execution time (and associated gains)—which depends on the methods selected—will also remain fixed. We state this formally:

**Assumption 1.** Let $\alpha$ be a coordination method, selected by robot $i$. We assume that for any $0 \leq j, k \leq K_i$, these hold:

$$I^a_{i,j}(\alpha) = I^a_{i,k}(\alpha), \quad I^p_{i,j}(\alpha) = I^p_{i,k}(\alpha), \quad C^C_{i,j}(\alpha) = C^C_{i,k}(\alpha)$$

This strong assumption achieves a key reduction in the complexity of the model, but gets us farther from the reality of LCT multi-robot tasks. However, the resulting model provides an intuition as to why and when EI works. In Section V-D we examine the assumptions of the model and their relation to the reality of the experiments.

The duration of coordination method execution ($I^a_i$), and the duration of the subsequent conflict-free task-execution ($I^p_i$), are fixed; they now depend only on the method selected, rather than also on the time of the selection. Thus a path through the game tree can now be compressed. For each combination of selected coordination method, we can simply multiply the costs and gains from using this combination, by the number of conflicts that will take place if it is selected.

Thus we can reduce the game tree into a matrix game, where $K_{i,j}$ is the number of conflicts occurring within total time $T$ that results from the first robot selecting $\alpha_i$, and the second robot selecting $\alpha_j$. $U_{i,j}$ is the utility gained from this choice. This utility is defined as:

$$\begin{aligned} U_{i,j} \equiv \quad & [gain(I^p_i(\alpha_i) + gain(I^p_j(\alpha_j))] \\ & - [C^C_i(\alpha_i) + C^C_j(\alpha_j)] \end{aligned} \tag{4}$$

where we use (for robot $i$) the notation $gain(I^p_i(\alpha_i))$ to denote the gains achieved by robot $i$ during the task execution time $I^p_i(\alpha_i)$. Note that we treat these gains as being a function of a time duration only, rather than the method $\alpha$, which only affect the time duration. Underlying this is an assumption that the coordination method choice affect utility (e.g., the pucks acquired) only indirectly, by affecting the time available for task execution. We assume further that gains monotonically increase with time. Maximizing the time available, maximizes the gains.

Table I is an example matrix game for two robots, each selecting between two coordination methods. Note however that in general, there are $N$ robots and $|M|$ methods available to each.

| | $\alpha_1^2$ | $\alpha_2^2$ |
|---|---|---|
| $\alpha_1^1$ | $K_{1,1}U_{1,1}$ | $K_{1,2}U_{1,2}$ |
| $\alpha_2^1$ | $K_{2,1}U_{2,1}$ | $K_{2,2}U_{2,2}$ |

TABLE I

LCT TASK AS A MATRIX GAME, REDUCED FROM THE LCT GAME TREE BY ASSUMPTION 1. ENTRIES HOLD TEAM PAYOFFS.

Note that the robots do not have access to the selections of the other robots, and thus for them, the game matrix does not have a single common payoff, but individual payoffs. These are represented in each cell by rewriting $K_{i,j}U_{i,j}$ as $K_{i,j}u_i(\alpha_i), K_{i,j}u_j(\alpha_j)$, where

$$u_k(\alpha_k) \equiv gain(I_k^p(\alpha_k)) - C_k^C(\alpha_k).$$

This results in the revised matrix game (Table II).

| | $\alpha_1^2$ | $\alpha_2^2$ |
|---|---|---|
| $\alpha_1^1$ | $K_{1,1}^1 u_1(\alpha_1^1), K_{1,1}^2 u_1(\alpha_1^2)$ | $K_{1,2}^1 u_1(\alpha_1^1), K_{1,2}^2 u_2(\alpha_2^2)$ |
| $\alpha_2^1$ | $K_{2,1}^1 u_2(\alpha_2^1), K_{2,1}^2 u_1(\alpha_1^2)$ | $K_{2,2}^1 u_2(\alpha_2^1), K_{2,2}^2 u_2(\alpha_2^2)$ |

TABLE II

AN LCT TASK AS A MATRIX GAME, WITH INDIVIDUAL PAYOFFS.

The number of conflicts $K_{i,j}$ is really the total time $T$, divided by the duration of each conflict cycle, i.e., $I^a + I^p$. Thus the individual payoff entries for robot $l$ selecting method $k$ can be rewritten as $\frac{T}{I_l^a(\alpha_k)+I_l^p(\alpha_k)}u_l$.

Let us now consider these individual payoffs. The payoff for an individual robot $l$ which selected $\alpha$ is:

$$\frac{T[gain(I_l^p(\alpha)) - c(I_l^a(\alpha))]}{I_l^a(\alpha) + I_l^p(\alpha)} \propto \frac{I_l^p(\alpha) - c(I_l^a(\alpha))}{I_l^a(\alpha) + I_l^p(\alpha)} \tag{5}$$

This step require some explanation. First, of course, since for all entries in the matrix $T$ is constant, dividing by $T$ maintains the proportionality. Furthermore, the proportion will hold only under certain restrictions on the nature of the function $gain()$, but we believe these restrictions hold for many gain functions in practice. For instance, the step holds whenever $gain()$ is linear with a coefficient greater than 1. Now:

$$\frac{I_l^p(\alpha) - c(I_l^a(\alpha))}{I_l^a(\alpha) + I_l^p(\alpha)} = \frac{I_l^p(\alpha) + [I_l^a(\alpha) - I_l^a(\alpha)] - c(I_l^a(\alpha))}{I_l^a(\alpha) + I_l^p(\alpha)} \tag{6}$$

$$= 1 - EI_l(\alpha) \tag{7}$$

$$\propto -EI_l(\alpha) \tag{8}$$

Thus the game matrix above (Table II) is analytically shown to be equivalent to the following matrix (Table III). Here, each robot seeks to minimize its own individual EI payoff (maximize its -EI payoff). If robots minimize their individual EI payoffs, and assuming that their equilibrium is Hicks optimal (i.e., the sum of payoffs is maximal), then solving this game matrix is equivalent to maximizing group utility.

| | $\alpha_1^2$ | $\alpha_2^2$ |
|---|---|---|
| $\alpha_1^1$ | $-EI_1(\alpha_1^1), -EI_2(\alpha_1^2)$ | $-EI_1(\alpha_1^1), -EI_2(\alpha_2^2)$ |
| $\alpha_2^1$ | $-EI_1(\alpha_2^1), -EI_2(\alpha_1^2)$ | $-EI_2(\alpha_2^1), -EI(\alpha_2^2)$ |

TABLE III

LCT TASK AS AN EI MATRIX GAME.

### C. Learning Payoffs in LCT Matrix Games

Unfortunately, when the robots first begin their task, they do not know the payoffs, and thus rely on the reinforcement learning framework to converge to appropriate EI values. Of course, it is known that Q-learning does not, in the general case, converge to equilibrium in 2-player repeated games [3], [19], [8]. However, there are a number of features that hold for the EI game matrix *in the domains we study*, which makes the specific situation special.

Most importantly, the games that take place here are *not* between two players. Rather, the process is more akin to randomized anonymous matching in economics and evolutionary game theory. In this process, pairs of players are randomly selected, and they do not know their opponents' identity (and thus do not know whether they have met the same opponents before).

Indeed, this last quality is crucial in understanding why our use of EI works. It turns out that there exists work in economics that shows that under such settings, using simple reinforcement learning techniques (in our case, stateless Q-learning) causes *the population* to converge to Nash equilibrium, even if mixed [9]. Thus rather than having any individual agent converge to the mixed Nash equilibrium, the population as a whole converges to it, i.e., the number of agents selecting a specific policy is proportional to their target probabilities under the mixed Nash equilibrium.

There remains the question of why do agents converge to the maximal payoff Nash equilibrium. We again turn to economics literature, which shows that for coordination games—including even the difficult Prisoner's Dilemma game—agents in repeated randomized matching settings tend to converge to the Pareto-efficient solution [4], [13]. However, these works typically assume public knowledge of some kind, which is absent in our domain. Thus we leave this as a conjecture.

### D. Revisiting the EI Experiments

Armed with the analytically-motivated intuition as to why EI works, we now go back to re-examine the experiment results. In general, there are of course differences between the analytical intuitions and assumptions and the use of EI in a reinforcement learning context: (i) the values learned our approximations of the EI values, which cannot be known with certainty; (ii) the assumptions allowing reduction of the LCT extensive-form game tree to a game matrix do not hold in practice; and (iii) even the assumptions underlying the extensive-form game tree (e.g., that robots start their conflict at the same time, or that their gains depend only on time available for task execution) are incorrect. We examine specific lessons below.

We begin with the teambots simulation experiments, where EI was highly successful, and was also demonstrated to be robust to unknown costs. Despite the fact that the domain cannot be reduced to the matrix game form, it turns out that some of the assumptions are approximately satisfied, which explain the success of EI here.

First, the fact that about half the pucks moved randomly helped spread them around the arena even after many pucks were collected. Thus the gains expected later in the task were closer to the gains at the beginning to the task, than it would have been had all pucks been immobile (in which case pucks closer to base are collected first, resulting in higher productivity in the beginning).

Second, the size of the arena, compared to the size of the robots, was such that the robots did not need to converge to one optimal combination of selection methods: Different zones in the arena required different combinations. In principle, this should have challenged the approach, as the stateless learning algorithm cannot reason about the robots being in different states (zones). However, as the robots moved between areas fairly slowly, they were able to adapt to the conditions in new zones, essentially forgetting earlier EI values. This is a benefit of the stateless algorithm.

## VI. SUMMARY

This paper examined in depth a novel reward function for cooperative settings, called Effectiveness Index (EI). EI estimates the resource spending velocity of a robot, due to its efforts spent on coordination. By minimizing EI, robots dedicate more time to the task, and are thus capable of improving their team utility. We used EI as a reward function for selecting between coordination methods, by reinforcement-learning. This technique was shown to work well in two foraging domains. The experiments explore the scope of the technique, its successes and limitations. In addition, we have formally explored multi-robot tasks for which EI is intended. We have shown that under some assumptions, EI emerges analytically from a game-theoretic look at the coordination in these tasks. We believe that this work represents a step towards bridging the gap between theoretical investigations of interactions, and their use to inform real-world multi-robot system design. Improved results can be achieved by extending both the theory underlying the use of EI, and the learning algorithms in which it is used.

**Acknowledgments.** We gratefully acknowledge the support of ISF grant #1357/07. We thank K. Ushi, Vladimir and Galina Erusalimchik for their support.

## REFERENCES

[1] T. Balch. www.teambots.org, 2000.
[2] T. R. Balch. Integrating learning with motor schema-based control for a robot soccer team. In *RoboCup*, pages 483–491, 1997.
[3] M. Bowling and M. Veloso. An analysis of stochastic game theory for multiagent reinforcement learning. Technical Report CMU-CS-00-165, Computer Science Department, Carnegie Mellon University, 2000.
[4] G. Ellison. Cooperation in the prisoner's dilemma with anonymous random matching. *The Review of Economic Studies*, 61(3):567–588, July 1994.
[5] M. Fontan and M. Matarić. Territorial multi-robot task division. *IEEE Transactions of Robotics and Automation*, 14(5):815–822, 1998.
[6] J. R. Galbraith. *Designing Complex Organizations*. Addison-Wesley Longman Publishing Co., Inc., 1973.
[7] D. Goldberg and M. Matarić. Design and evaluation of robust behavior-based controllers for distributed multi-robot collection tasks. In *Robot Teams: From Diversity to Polymorphism*, pages 315–344, 2001.
[8] P. J. Hoen, K. Tuyls, L. Panait, S. Luke, and J. A. L. Poutré. An overview of cooperative and competitive multiagent learning. In K. Tuyls, P. J. Hoen, K. Verbeeck, and S. Sen, editors, *First International Workshop on Learning and Adaption in Multi-Agent Systems*, volume 3898 of *Lecture Notes in Computer Science*, pages 1–46. Springer, 2006.
[9] E. Hopkins. Learning, matching, and aggregation. *Games and Economic Behavior*, 26:79–110, 1999.
[10] M. Jager and B. Nebel. Dynamic decentralized area partitioning for cooperating cleaning robots. In *ICRA 2002*, pages 3577–3582, 2002.
[11] M. J. Matarić. Reinforcement learning in the multi-robot domain. *Auton. Robots*, 4(1):73–83, 1997.
[12] E. Ostergaard, G. Sukhatme, and M. Matarić. Emergent bucket brigading. In *Agents-01*, pages 29–30, 2001.
[13] A. J. Robsona and F. Vega-Redondob. Efficient equilibrium selection in evolutionary games with random matching. *Journal of Economic Theory*, 70(1):65–92, July 1996.
[14] A. Rosenfeld, G. A. Kaminka, S. Kraus, and O. Shehory. A study of mechanisms for improving robotic group performance. *AIJ*, 172(6–7):633–655, 2008.
[15] P. Rybski, A. Larson, M. Lindahl, and M. Gini. Performance evaluation of multiple robots in a search and retrieval task. In *Proc. of the Workshop on Artificial Intelligence and Manufacturing*, pages 153–160, Albuquerque, NM, August 1998.
[16] M. Schneider-Fontan and M. Matarić. A study of territoriality: The role of critical mass in adaptive task division. In P. Maes, M. Matarić, J.-A. Meyer, J. Pollack, and S. Wilson, editors, *From Animals to Animats IV*, pages 553–561. MIT Press, 1996.
[17] R. Vaughan, K. Støy, G. Sukhatme, and M. Matarić. Go ahead, make my day: robot conflict resolution by aggressive competition. In *Proceedings of the 6th int. conf. on the Simulation of Adaptive Behavior*, Paris, France, 2000.
[18] D. H. Wolpert and K. Tumer. Collective intelligence, data routing and braess' paradox. *JAIR*, 16:359–387, 2002.
[19] E. Yang and D. Gu. Multiagent reinforcement learning for multi-robot systems: A survey. Technical Report CSM-404, University of Essex, 2004.
[20] M. Zuluaga and R. Vaughan. Reducing spatial interference in robot teams by local-investment aggression. In *IROS*, Edmonton, Alberta, August 2005.