2007 IEEE/WIC/ACM International Conference on Intelligent Agent Technology

A Study of an Approach to the Collective Iterative Task Allocation Problem

Christian Guttmann
Clayton School of Information Technology, Monash University
VICTORIA 3800, Australia
christian.guttmann@gmail.com

Iyad Rahwan
British University in Dubai, UAE
(Fellow) University of Edinburgh, UK
irahwan@acm.org

Michael Georgeff
e–Health Research Unit, Monash University
VICTORIA 3800, Australia
michael.georgeff@med.monash.edu.au

Abstract

A major challenge in the field of Multi-Agent Systems is to enable autonomous agents to allocate tasks efficiently. This paper extends previous work on an approach to the collective iterative task allocation problem where a group of agents endeavours to make the best allocations possible over multiple iterations of proposing, selection and learning. We offer an algorithm capturing the main aspects of this approach, and then show analytically and empirically that the agents' estimations of the performance of a task and the type of group decision policy play an important role in the performance of the algorithm.

1 Introduction

Assigning agents to tasks is a challenging problem in the coordination of *Multi-Agent Systems (MAS)*, where each agent is autonomous and has its own unique knowledge of the agents and tasks involved [2]. This problem is prevalent in a wide range of applications, such as network routing, crisis management, logistics, computational grids, and collaborative student support environments [3, 1].

This paper addresses the problem of Collective Iterative Task Allocation (CITA) which involves allocating tasks to teams. We view each team as a unit, and thus are not concerned with the internal coordination within teams. Moreover, we do not assume that agents know the performance of different teams accurately. Instead, we assume each agent has estimations of the performance of different teams. The agents refine their estimations after selecting a particular team and acquiring information of its actual performance (e.g., through observation). Following other work, which argued that estimations of other agents' performance can improve individual agent decision-making [5], we hypothesise that such estimations are useful in making collective decisions about allocating tasks to teams.

Our first question, then, is how to select a team to a given task (a phase we call *selection*) using various individual agents' proposals which are based on their estimations of the performance of different teams (a phase we call *proposing*). To this end, we offer an approach based on group decision policies, where each agent contributes (e.g., through proposing) to the decision as to which team should be selected for a given task (Section 2). After a team is selected using the group decision policy, agents' individual estimations of that team's performance are updated (a phase we call *learning*), and is taken into account in subsequent allocations. We examine two specific policies: the *majority policy* which selects the team proposed by the majority of agents; and the *maximum policy* which selects the team proposed with the highest estimated performance.

Our second question concerns the conditions under which the iterative execution of proposing, selection and learning processes converges to optimal or near optimal allocations. For the purpose of this paper, we consider the case of invariable team performance (i.e., a team's performance is the same every time it performs a particular task).[1] In Section 3, we analyse two aspects of the performance of our approach based on [4]: the *quality* of the final (converged) allocation; and the number of rounds required to converge. We show that these aspects are influenced by the agents' estimations of the performance of teams (e.g., they are optimistic); and the type of group decision policy. For example, we prove that if agents are completely optimistic, then we are guaranteed to converge to an optimal solution. Further, if agents are optimistic then the algorithm will require no greater number of rounds to converge to optimal solutions than an exhaustive approach. Our empirical analysis (Section 4) shows that, under less restrictive condi-

[1] In previous work [6], we presented empirical results in the case of variable team performance. But this paper offers stronger analytical properties about the performance of the algorithm. Also note that in related work [4], the performance of agents is often implicitly assumed to be constant.

0-7695-3027-3/07 $25.00 © 2007 IEEE
DOI 10.1109/IAT.2007.97

tions, the maximum policy converges to better allocations, but when the number of rounds is limited, then the majority policy outperforms the maximum policy.

2 A Formal Framework to Study an Approach to the Collective Iterative Task Allocation (CITA) Problem

We now define the main components of our approach.

Definition 1. *A set of **Tasks** is denoted by* $T = \{t_1, \ldots, t_s\}$ *with* $s = |T|$.

A task defined in T can be assigned to an agent team.

Definition 2. *A set of **Agent Teams** is denoted by* $AT = \{at_1, \ldots, at_p\}$ *with* $p = |AT|$*, where* $at_j \in AT$ *is an individual team.*

Definition 3. *A set of **Agents** is denoted by* $A = \{a_1, \ldots, a_q\}$ *with* $q = |A|$*, where* $a_i \in A$ *is an individual agent.*

The true performance of a team at_j for various tasks is referred to as a team's capability: $C(at_j) = \{V(at_j, t_1), \ldots, V(at_j, t_s)\}$, where $V : AT \times T \rightarrow R^+$ is a value representing the performance of a team for a task (the value 0 corresponds to being unable to perform a task). The capability of a team can only be estimated (a capability is only specified for illustrative purposes and empirical studies). Each agent $a_i \in A$ maintains models M_{a_i} to estimate the capabilities of teams and each agent is able to execute Reasoning Processes RP_{a_i} using these models.

Definition 4. M_{a_i} *are the **Models** maintained by agent* a_i*. Models are expressed by* $M_{a_i} = \{M_{a_i}(at_1), \ldots, M_{a_i}(at_p)\}$ *with* p *being the number of teams in* AT*. A specific model of a team* at_j *is defined by a set of estimations* $M_{a_i}(at_j) = \{\hat{V}_{a_i}(at_j, t_1), \ldots, \hat{V}_{a_i}(at_j, t_s)\}$*, where*

- $t_1, \ldots, t_s$ *are the tasks defined in* T *(Definition 1).*
- at_j *is a team in* AT *(Definition 2).*
- $\hat{V}_{a_i}(at_j, t_k)$ *is an estimation of team* at_j*'s true task performance* $V(at_j, t_k)$*.*

Definition 5. RP_{a_i} *are* a_i*'s **Reasoning Processes**.*

- *For each agent* a_i *and task* t_k*, the* INITIALISE *process returns a set of Models* M_{a_i}*.*
- *For each agent* a_i *and task* t_k*, the* PROPOSE *process returns a proposal defining a team and its estimated performance: proposal*$_{a_i} = \langle at_j, \hat{V}_{a_i}(at_j) \rangle$*.*
- *For each agent* a_i*, task* t_k *and team* at_j*, the* UPDATE *process returns a set of updated models* M'_{a_i} *after it performs task* t_k*.*

We introduce the notion of a group decision policy. Upon receiving a proposal from each agent, the policy P is applied to determine a team for a task.

Definition 6. ***Policy*** P *is denoted by* $P(proposals^A) = at_{selected}$*, where* $at_{selected}$ *is the team selected by the policy* P *and* $proposals^A = \{proposal_{a_1}, \ldots, proposal_{a_q}\}$.

INPUT: Task $t_k \in T$, Teams AT, Agents A, Policy P
OUTPUT: Team $at_j \in AT$

1. ANNOUNCE task $t_k \in T$
2. INITIALISE$_{a_i}(M_{a_i}, t_k)$ $(\forall a_i \in A)$
3. Repeat
(a) $proposals^A = \bigcup_{a_i \in A}$ PROPOSE$_{a_i}(M_{a_i}, t_k)$
(b) $at_{selected}$ = P ($proposals^A$)
(c) UPDATE$_{a_i}(M_{a_i}, performance(at_{selected}, t_k))(\forall a_i \in A)$
4. Until a termination criterion is satisfied

Figure 1. Assignment Algorithm.

Assignment Algorithm: The assignment algorithm presented in Figure 1 uses parameters based on the Definitions 1–6 and works as follows. In step 1, a task is announced to all agents A, and each agent then initialises its models about each team for the announced task (step 2).

The loop (steps 3a–3c) involves proposing teams, selecting a team and learning from the performance of the selected team. This loop is called an *assignment round* r_i with i being the i'th iteration of the loop. In step 3a, each agent proposes a team and its estimated performance for the announced task, and these proposals are stored in $proposals^A$. In step 3b, a trusted agent $a_{trusted}$ uses the $proposals^A$ and the policy P to select a team $at_{selected}$ for the task. After the selected team $at_{selected}$ performs the task, each agent uses the UPDATE process to refine its models (step 3c) based on information of the performance of the selected team: $performance(at_{selected}, t_k)$. An assignment round is repeated until the algorithm satisfies a termination criterion (step 4).

When the algorithm terminates, the output of the algorithm is the team that performed the task better than the teams selected previously. Let $AT^{KnownSoFar} = \{at_i \in AT : at_i = at_{r_t}, r_t \leq r_{current}$, where $r_{current}$ is the current round$\}$. That is, if the algorithm is currently in round $r_{current}$, then $AT^{KnownSoFar}$ is the set of teams that have been selected in rounds $r_1, \ldots, r_{current}$. Let $at_{BestSoFar} = \underset{at_l \in AT^{KnownSoFar}}{\operatorname{argmax}} V(at_l, t_k)$ be the team with the highest true performance of all teams in $AT^{KnownSoFar}$.

3 Analytical Properties

This section reports on our analytical findings pertaining to the solution quality and then the computational requirements of the algorithm (Figure 1).

3.1 Optimal Solution: Complete Optimism

We will first define an agent that does not underestimate the performance of any team in AT.

Definition 7. ***Optimistic Agent.*** *An agent* a_i *is* optimistic *about the performance of a team* at_j *if and only if it does not underestimate the performance of a team* at_j *for a*

given task. That is, the INITIALISE *and* UPDATE *processes (Definition 5) of this agent return models M_{a_i} where $\hat{V}_{a_i}(at_j) \geq V(at_j) : \forall at_j \in AT$.*[2] *In a completely optimistic group of agents, each agent is optimistic about the performance of all teams.*

There are various situations where agents are completely optimistic. For example, assume that agents have to assign a team to run a marathon, but have only observed the running performance of teams for distances shorter than that of a marathon. Assume that teams need more time on average as the distance of a run increases. If agents extrapolate their estimations of the running performance of shorter distances, they will have overestimations of the performance of teams for running a marathon.

The first theorem demonstrates that a completely optimistic group of agents plays an important role in guaranteeing to find an optimal solution with our algorithm. We call this theorem the sufficient condition, because the algorithm finds an optimal solution regardless of the group decision policy used.

Theorem 1. ***Optimality under Complete Optimism: Sufficient Condition.*** *If*

- ***(A)*** *a group of agents A is completely optimistic (Definition 7),*
- ***(B)*** *each agent proposes a team with an estimation higher than $V(at_{BestSoFar})$ (if an agent does not have an estimation of a team higher than $V(at_{BestSoFar})$, then it proposes any team),*
- ***(C)*** *the algorithm does not terminate if $\neg\exists\langle at_j, \hat{V}(at_j)\rangle \in proposals^A$ such that $\hat{V}(at_j) \leq V(at_{BestSoFar})$,*

then if the algorithm terminates it will terminate with an optimal solution.

Proof. The theorem is proven by contradiction, i.e., we assume that if the algorithm terminates, then we have found a suboptimal team $at_{BestSoFar}$.

1. If the team $at_{BestSoFar}$ is not an optimal team, then there must be a team at_*, such that $V(at_*) > V(at_{BestSoFar})$.
2. According to **(C)**, if the algorithm terminates, we know that $\exists\langle at_j, \hat{V}(at_j)\rangle \in proposals^A$ such that $\hat{V}(at_j) \leq V(at_{BestSoFar})$. Assume that $a' \in A$ is the agent that made this proposal.
3. According to **(B)**, we know that $V(at_{BestSoFar}) \geq \hat{V}_{a'}(at_j)$ for all $at_j \in AT$. In particular, we know that $V(at_{BestSoFar}) \geq \hat{V}_{a'}(at_*)$.

According to 1., $V(at_*) > V(at_{BestSoFar})$, and according to 3. $V(at_{BestSoFar}) \geq \hat{V}_{a'}(at_*)$. So, we know that $V(at_*) > \hat{V}_{a'}(at_*)$. However, this is a contradiction, because we assumed that no agent underestimates the performance of any team according to **(A)**. □

[2]For clarity of exposition, from now on we assume a fixed task t_k and remove t_k from the notation (e.g., $\hat{V}_{a_i}(at_j)$ instead of $\hat{V}_{a_i}(at_j, t_k)$).

The power of Theorem 1 is that the algorithm eventually finds an optimal solution despite the decision policy used.

3.2 Defining Two Policies: P_{max} and P_{maj}

We will consider two policies, P_{max} and P_{maj}. To define P_{max}, we denote a set $V^{proposals^A}$ which are the estimations specified in $proposals^A$: $\hat{V}^{proposals^A} = \{\hat{V}_{a_i}(at_j) | \langle at_j, \hat{V}_{a_i}(at_j)\rangle \in proposals^A\}$.

The maximum policy behaves as an optimisation function as it always selects the team proposed with the highest estimated performance.

Definition 8. ***Maximum Policy.*** *$P_{max}(proposals^A) = at_j$, where team at_j is selected from set $AT^{max} = \{at_{max} | \langle at_{max}, \hat{V}_{max}\rangle \in proposals^A$, where $\hat{V}_{max} = \underset{\hat{V}_i \in \hat{V}^{proposals^A}}{argmax}(\hat{V}_i)\}$.*

Another important policy is the majority policy which selects the team that is preferred by the majority of agents. As opposed to the maximum policy, the advantage of the majority policy is that it can prevent selecting a team that has been proposed with an unrealistically high estimation which is far removed from the true performance of that team. To define the policy P_{maj}, we denote a set of the teams specified in $proposals^A$: $AT^{proposals^A} = \{at_j | \langle at_j, \hat{V}_{a_i}(at_j)\rangle \in proposals^A\}$.

Definition 9. ***Majority Policy.*** *$P_{maj}(proposals^A) = at_j$, where team at_j is selected from set $AT^{maj} = \{at_{maj} | \underset{at_{maj} \in AT^{proposals^A}}{\text{argmax}} |\{a_i : \langle at_{maj}, \hat{V}_{a_i}(at_{maj})\rangle \in proposals^A\}|\}$.*

Using Theorem 1 and any of the policies, we know that the algorithm will find an optimal solution.

Corollary 1. ***Optimality with P_{max} and P_{maj}.*** *Under the conditions **(A)** – **(C)** introduced for Theorem 1, we know that using P_{max} and P_{maj} will find with an optimal solution, if the algorithm terminates.*

Proof. The proof follows from Theorem 1. □

3.3 Number of Assignment Rounds

The *computational requirement* is an important feature of the performance of a coordination algorithm [4]. In our study, this requirement is measured by the number of assignment rounds required to reach a solution until a termination criterion is satisfied. Note that the theorems in this section do not rely on the assumption of complete optimism (Definition 7) as is the case for Theorem 1.

Termination Criterion: In this paper, we consider a *termination criterion* that will terminate the

algorithm if all proposed estimations of the performance of the selected team are not greater than $at_{BestSoFar}$. That is, the algorithm terminates if $\forall a_i \in A : \langle at_{selected}, \hat{V}_{a_i}(at_{selected})\rangle \in proposals^A$ such that $\hat{V}_{a_i}(at_{selected}) \leq V(at_{BestSoFar})$. Provided that the same team is not assigned more than once, the algorithm will eventually terminate, because the set of teams is finite (in the worst case after all teams have been selected once). In future research, we will prove optimality of more efficient termination criteria. For example, under complete optimism, we can terminate the algorithm (and find an optimal solution) if only one proposal of the selected team is not greater than $V(at_{BestSoFar})$.

Basic Assumptions: In order to develop proofs of the computational requirement of our algorithm, we make assumptions pertaining to the PROPOSE and UPDATE processes of individual agents (Definition 5). For the PROPOSE process, we assume that agents are *task-rational*, i.e., each agent proposes the team with the highest estimated performance according to its models, $\underset{at_j \in AT}{\operatorname{argmax}} \hat{V}_{a_i}(at_j)$. We also assume that each agent uses an UPDATE process that replaces an estimated value with the value that represents the observed performance ($performance(at_j)$) of the selected team. We assume that the observed performance (after execution) equals the true performance of an agent so that $performance(at_j) = V(at_j)$. If team a at_j has performed a task, then the estimation $\hat{V}'_{a_i}(at_j)$ is updated with team at_j's true performance $V(at_j)$. The UPDATE process does not update models of teams other than at_j.

Exhaustive Procedure: As a benchmark, consider an *exhaustive* procedure which assigns each team at least once until accurate performance estimations of all teams are obtained. This procedure will find the optimal allocation. However, an optimal team could be assigned in the first attempt, but we only know this after all teams have been tested. This section shows that our algorithm finds an optimal solution in no greater number of rounds than the exhaustive procedure.

The following result identifies conditions which determine the number of rounds until the algorithm converges to an optimal solution with P_{max}. Informally, if there exists an agent that estimates the performance of m teams (of n teams) higher than the true performance of an optimal team, and if the performance of an optimal team is higher than the estimated performance of the other $n - m$ teams, then an optimal solution is found in exactly m rounds. Let $AT^* \subseteq AT$ be a set of teams such that $\forall at^*_j \in AT^*, \exists a_i \in A$ with $\hat{V}_{a_i}(at^*_j) \geq V(at')$, where at' is an optimal team.

Theorem 2. ***Maximum Policy: Assignment Rounds.*** *If* $\max_{a_k \in A} \hat{V}_{a_k}(at') \geq V(at')$ *and* $V(at') \geq \max_{a_l \in A, at'' \in |AT - AT^*|} \hat{V}_{a_l}(at'')$, *then* P_{max} *is guaranteed to find an optimal team* at' *in exactly* $|AT^*|$ *rounds.*

Proof. P_{max} selects the team with the highest proposed performance, and we know that $|AT^*|$ teams have higher estimations than the true and estimated performance of an optimal team, but perform a task worse than an optimal team. Each team in AT^* is selected before an optimal team, since $\hat{V}_{a_i}(at_j) > V(at_j), \forall at_j \in AT^*$. Since these teams have a true performance that is lower than the estimated performance of an optimal team, each of them will be selected once and then not be selected again. The other teams $AT^* - AT$ will never be selected as their estimations are lower than that of an optimal team and the teams in AT^*. Thus, exactly $|AT^*|$ teams perform the task, after which the algorithm terminates with an optimal team. □

This theorem implies that in the best case an optimal solution can be found in only one round. The theorem also implies that an optimal solution can be found if the maximum number of rounds is not greater than those required for the exhaustive procedure.

Theorem 3. ***Majority Policy: Assignment Rounds.*** *If the majority of agents estimate the performance of* m *teams (of all* n *teams) higher than the performance of the optimal team, and if the majority of agents estimates the performance of the optimal team higher than its true performance, then* P_{max} *finds the optimal solution in* m *rounds.*

(The proof is omitted due to space limitations.)

If we know that agents are optimistic in a given domain, we can use the assignment algorithm to exploit the above property in reducing the number of rounds required to find an optimal solution. That is, Theorems 2 and 3 enable us to make statements about the computational requirements of our algorithm. For example, the theorems show that if we have prior knowledge of the agents' estimations, then we can determine an acceptable upper bound on the number of reassignments required. Another useful insight is that if we know that the majority of agents is not optimistic about an optimal team (but we have at least one agent that is optimistic about an optimal team), then we are guaranteed to find an optimal team with P_{max}.

4 Empirical Study

This section presents an empirical study to investigate the interaction of agents' initial estimations and the true performance of teams, and the maximum and majority policy. We compare the performance of the assignment algorithm (using the termination criterion introduced in Section 3.3) with benchmark settings which make optimal and random assignments. Finally, we investigate the convergence behaviour of the algorithm for the two policy.

4.1 Experimental parameters

We simulate the assignment algorithm under different simulation settings consisting of four experimental parameters which are assigned a range of values in the simulations.

- **Model Initialisation (MI)** defines four types of agents with different estimations of each team's performance: *Low, Medium, High* and *Mixed-estimating* agents. For the purpose of this paper, we restrict the range of performance values from 0 (worst performance) to 1 (optimal performance). This range is divided evenly for distributions that represents low (mean = 0.25), medium (mean = 0.5) and high-estimating (mean = 0.75) teams.[3] A standard deviation of 0.1 covers approximately 98% of a distribution around its mean before it overlaps with the mean of the other distributions. The distribution for mixed-performing agents has a mean of 0.5 and deviation of 0.25.
- **Capability (C)** defines four types of teams with different capabilities: *Low, Medium, High* and *Mixed-performing* teams. The types of teams are based on the same four types of distributions used for the experimental parameter MI.
- **Policy (P)** defines two types of group decision policies: *maximum* and *majority*. These two group decision policies are guaranteed to converge to an optimal solution if agents are optimistic (Section 3.1).
- **Group Size (GS)** defines the number of agents: *5*, *10*, *20*, *40* and *50*. These values are expected to show a representative trend of our results for varying populations of agents. We assume that agents assign a task to each other $A = AT$.

4.2 Simulation Settings

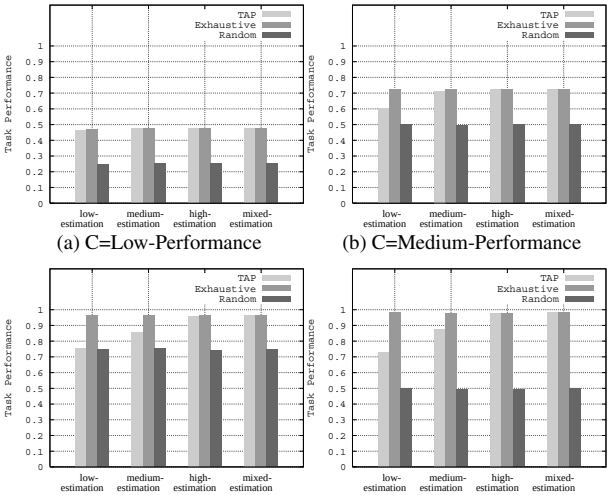
We constructed a TAP[4] setting for each combination of the experimental parameters (Capability (C)×Model Initialisation (MI)×Policy (P)×Group Size (GS)= 4×4×2×5=160).

To evaluate the outcome of a TAP setting, we constructed two benchmark settings, EXHAUSTIVE and RANDOM. Each benchmark setting represents an opposing extreme on a scale that measures the number of assignments (to learn the performance of the selected team) required until a solution is found. At one extreme is a procedure used in the EXHAUSTIVE setting which requires as many assignments as there are teams to find an *optimal solution*. At the other extreme is a procedure in the RANDOM setting that requires no assignment to provide a *random solution*.

4.3 Simulation Run of Settings

The simulation is first populated with a certain number of agents (and consequently, teams) as specified by the parameter group size GS. The agents' models that estimate the performance of teams are initialised as specified by the MI parameter. The simulated true performance of teams is initialised as specified by the C parameter. The performance estimated in the agents' models are likely to be different to the true performance (the capability) of each team. The parameter P determines the type of group decision policy used in a simulation run. The simulation run is completed if the termination criterion is satisfied (Section 3.3).

The following information is stored upon termination of a simulation run. The **solution quality** is measured by the performance of the team that has been computed by the procedure applied in each setting (the measured performance of the team is the value obtained from its simulated capabilities). The **computational requirements** are measured by the number of assignments required to find a solution.



(a) C=Low-Performance (b) C=Medium-Performance
(c) C=High-Performance (d) C=Mixed-Performance

Figure 2. Average performance (P_{max})

4.4 Results and Analysis

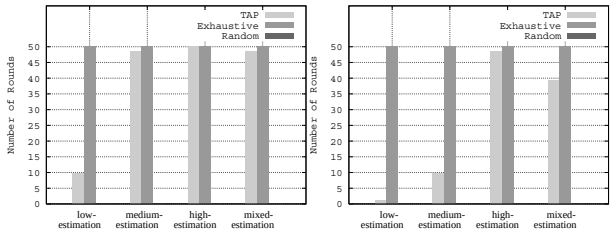
The purpose of this experiment is to examine the performance of the assignment algorithm empirically under different simulation settings (Section 4.1). We have simulated each TAP setting (C×MI×P×GS= 4×4×2×5=160) and each benchmark setting, RANDOM and EXHAUSTIVE. This section only plots results obtained for settings with 50 teams as the results provide patterns similar to those obtained with GS=5, 10, 20 or 40. We haven chosen to only plot the most significant results about the performance of the algorithm.

For each of the 162 simulation settings, the results are averaged over 1000 simulation runs, because these results showed stable and continuous patterns. These results are statistically significant as indicated by a 95% Confidence Interval (CI). The CI was calculated by multiplying 1.96 with the Standard Error (degrees of freedom is 1000).

Results of Benchmark Settings: Figure 2 plots the average performance against model initialisation MI (low, medium, high, and mixed-estimating agents) when the capability C of teams is (a) low, (b) medium, (c) high, and (d) mixed (re-

[3]In future work, we will vary the distribution means to represent other types of agents. For example, *extremely* low or high-estimating agents will be simulated with a mean of 0.1 (low) and 0.9 (high).

[4]The term TAP is based on the main components of our approach: *Tasks, Agents, and Policy*.



(a) C=Low-Performance (b) C=Medium-Performance

(c) C=High-Performance (d) C=Mixed-Performance

Figure 3. Average number of rounds (majority policy), measure for random setting is 0 (not shown in figures)

sults of TAP settings are indicated by light grey bars). As expected, the results obtained from the EXHAUSTIVE setting correspond to an upper benchmark (medium grey bars in Figure 2) and requires as many assignments as there are teams (Figure 3). The simulation results of the RANDOM setting correspond to a lower benchmark which averages the task performance of randomly assigned teams (dark grey bars in Figure 2) and requires no assignments (Figure 3). As seen from Figure 2, the average performance obtained for the EXHAUSTIVE setting is always better than that obtained for the TAP and RANDOM settings.

Solution Quality (Average Task Performance): As seen from Figure 2, settings with high-estimating agents find better solutions than those found in settings where agents have low, medium or mixed estimations. In fact, a more general pattern is that the higher the estimations of agents, the higher the quality of the solution. The reason for this effect is that agents with high estimations are only satisfied with the allocation if the team's performance is higher than their estimations. In contrast, settings with low-estimating agents are satisfied with the performance as soon as they are below their estimations. These results correspond to our theoretical studies (Section 3).

Settings with low-estimating agents find solutions worse than those found by medium, high or mixed-estimating agents. In fact, when teams have a high performance, but the agents' initial estimations are low, the average performance of the best believed teams in the TAP settings is as low as the performance of teams selected in the RANDOM setting, Figure 2(c). The reason is that low-estimating agents are satisfied with a low-performing team if its performance is slightly better than their estimations.

Number of Assignment Rounds: As seen from Figure 3,

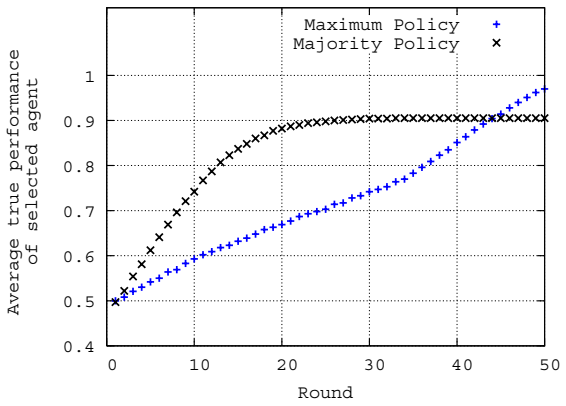


Figure 4. Performance of selected team against the round in which it was selected

TAP settings always require fewer assignment rounds than those required in the EXHAUSTIVE setting, but more than those required in the RANDOM setting. More rounds are required in TAP settings with high-estimating agents than with low or medium-estimating agents. Low-estimating agents require the least number of rounds to find a solution.

Unlike the majority policy, the maximum policy requires more rounds with mixed-estimating agents (MI=mixed-estimation) than with medium-estimating agents (MI=mixed-estimation), Figure 2. Plots for settings with P_{max} are omitted due to space limitations. This observation suggests that the type of distribution used for the MI parameter plays an important role in converging to optimal solutions and should be investigated in future research. This is explained by the fact that the distribution of the parameter MI=mixed-estimation has a larger standard deviation than the distribution used for the parameter MI=medium-estimation. Hence, each agent's estimations are further apart and the maximum policy is likely to explore a larger range of agents than for estimations which are closer.

Convergence Behaviour: Figure 4 shows the average performance measured against the number of assignment rounds (for MI=mixed-estimations and C=mixed-performance). The maximum policy increases the average performance of teams in each round steadily and the performance appears to be linear to the number of rounds. The team proposed by only one agent is selected in each round until no agent believes that any team has a better performance than the currently selected team. In TAP settings where the majority policy is applied the algorithm converges faster to a local maximum than in settings where the maximum policy is applied. This is explained by the effect that the best guesses of the majority of agents is tested and not only the best guess of one agent. As seen in Figure 4, optimistic initial values encourages exploration, an observation commonly made in the single agent reinforcement learning literature [7].

The algorithm converges significantly faster with the majority policy, but eventually finds solutions that are worse than those found with the maximum policy. Similarly, settings with the maximum policy converge significantly slower but eventually find better solutions than those found with the maximum policy.

5 Related Research

Enabling agents to assign tasks to teams autonomously is a long standing problem in the MAS community [2]. Market-driven allocation algorithms assume that each agent knows best its own performance when it bids for a particular task [1]. A well-known market-driven scheme is the Contract Net Protocol (CNP) which is based on a contract metaphor [8]. Each contractor provides information about how well it performs the task in question, and the manager then selects the contractor that makes the highest bid. The CNET protocol focuses on the bidding and rewarding part of making assignments, while our approach is concerned with the use of agents' beliefs of team performance in selecting an optimal team.

Perhaps the best-known agent modelling technique to coordinate agents uses a decision-theoretic approach whereby each agent makes decisions to maximise its own payoff by recursively estimating the payoff of collaborators [5]. A key difference is that agents described in [5] maintain models of how other agents make decisions, and not how well other agents perform tasks. Also, agents in [5] do not make group decisions as done in our research.

It is worth noting that our work differs from existing research on Multi-Agent Reinforcement Learning (MARL) [7]. Most work on MARL focuses on multiple agents that execute tasks individually and use Reinforcement Learning (RL) to coordinate their actions, taking into account various configurations (e.g., if agents can observe each others' actions). In our work, agents jointly select a team which then executes a given task. We are not concerned with the internal coordination within such team, but in its overall performance. Our work is similar to a single agent RL problem, but with a crucial difference. Instead of a *single* agent deciding what *action* to perform based on *its own* changing estimations, we deal with how *multiple* agents jointly decide what *team* to select based on *multiple* estimations. Thus, our group decision policy, combined with the agents' individual estimations of the performance of various teams, together define the way the whole MAS learns. To our knowledge, no existing RL or MARL framework uses this kind of encoding of adaptation through group decision policies such as the majority or maximum policy.

6 Conclusion

This paper presents a formal framework of an approach to the Collective Iterative Task Allocation (CITA) problem, and a theoretical and empirical study of this framework. In the theoretical study, we proved that the algorithm is guaranteed to terminate with an optimal solution if agents are completely optimistic. We verified optimality of two policies in particular: the maximum policy P_{max} (which selects a team with the highest value) and the majority policy P_{maj} (which selects a team that has been proposed most often by agents). We also proved that using estimations of several independent optimistic learners converges the algorithm to an optimal solution in fewer assignment rounds than testing each team once. At best, our algorithm will find an optimal solution in one round and at worst, the algorithm will find an optimal solution in no greater number of rounds than an exhaustive procedure. The empirical analysis extends this theoretical insight and shows that the larger the number of optimistic agents, the better the solution quality. Reducing the number of rounds requires low-estimating agents (but also means that random task allocations provide a similar solution quality). The maximum policy P_{max} should be selected over the majority policy P_{maj} as P_{max} finds better solutions than P_{maj}. However, if the number of assignment rounds is limited or not known, P_{maj} should be selected over P_{max}.

Much future research is required to fully understand the efficiency of our approach. For example, in order to reach optimal solutions fast, we need to identify the conditions under which certain criteria terminate the algorithm as soon as we know that an optimal solution has been found. We also need to consider transaction costs as a means to terminate the algorithm if the potential costs outweigh the expected gain of allocating more teams.

References

[1] Y. Chevaleyre, P. E. Dunne, U. Endriss, J. Lang, M. Lemaître, N. Maudet, J. Padget, S. Phelps, J. A. Rodríguez-Aguilar, and P. Sousa. Issues in multiagent resource allocation. *Informatica*, 30:3–31, 2006.

[2] E. H. Durfee, V. R. Lesser, and D. D. Corkill. Trends in cooperative distributed problem solving. *IEEE Transactions on Knowledge and Data Engineering*, 1(1):63–83, 1989.

[3] L. Garrido, K. Sycara, and R. Brena. Quantifying the utility of building agents models: An experimental study. In *Agents-00/ECML-00 Workshop on Learning Agents*, Barcelona, Spain, 2000.

[4] B. P. Gerkey and M. J. Mataric. A formal analysis and taxonomy of task allocation in multi-robot systems. *Robotics Research*, 23(9):939–954, September 2004.

[5] P. J. Gmytrasiewicz and E. H. Durfee. Rational communication in multi-agent environments. *Autonomous Agents and Multi-Agent Systems*, 4(3):233–272, 2001.

[6] C. Guttmann and I. Zukerman. Agents with limited modeling abilities: Implications on collaborative problem solving. *Journal of CSSE*, 21(3), 2006.

[7] T. Sandholm. Perspectives on Multiagent Learning. *Artificial Intelligence (Special Issue on Multiagent Learning)*, 171:382–391, 2007.

[8] R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, 29(12):1104–1113, 1980.