How to Form a Task-Oriented Robust Team

Tenda Okimoto
Graduate School of Maritime Sciences, Kobe University
Kobe, Japan
tenda@maritime.kobe-u.ac.jp

Nicolas Schwind
Transdisciplinary Research Integration Center
Tokyo, Japan
schwind@nii.ac.jp

Maxime Clement
The Graduate University for Advanced Studies
Tokyo, Japan
maxime-clement@nii.ac.jp

Tony Ribeiro
The Graduate University for Advanced Studies
Tokyo, Japan
tony_ribeiro@nii.ac.jp

Katsumi Inoue
National Institute of Informatics
Tokyo, Japan
inoue@nii.ac.jp

Pierre Marquis
CRIL - CNRS
Université d’Artois
Lens, France
marquis@cril.univ-artois.fr

ABSTRACT

How to form a team for achieving a given set of tasks is an important issue in multi-agent systems. Task-oriented team formation is the problem of selecting a group of agents, where each agent is characterized by a set of capabilities; the objective is to achieve a given set of tasks, where each task is made precise by a set of capabilities necessary for managing it. Robustness (i.e., the ability to reach the goal even if some agents break down) is an expected property of a team. In this paper, the focus is laid on the Task-Oriented Robust Team Formation (TORTF) problem. A formal framework is defined and some decision and optimization problems for TORTF are pointed out. The computational complexity of TORTF is then identified. Interestingly, TORTF does not prove more computationally demanding than the task-efficient team formation problem, i.e., robustness is in some sense “for free”. In order to solve these TORTF problems, two algorithms, ART (Algorithm for Robust Team) for the decision problem and AORT (Algorithm for Optimal Robust Team) for bi-objective constraint optimization problems, are presented and evaluated on a number of benchmarks.

Categories and Subject Descriptors
I.2.11 [Distributed Artificial Intelligence]: Multi-agent Systems

General Terms
Theory

Keywords
Team Formation, Robustness, Complexity Analysis, Bi-Objective Constraint Optimization

1. INTRODUCTION

Task-oriented team formation is the problem of forming the best possible team to accomplish some tasks of interest, given some limited resources. This problem is a key issue for many applications related to multi-agent cooperation, e.g., RoboCup rescue team [15], Unmanned Aerial Vehicles (UAVs) operations [9], team formation in social networks [15], and online soccer prediction games [21].

In the following, we are interested in the robustness issue for task-oriented team formation. Let us start with a motivating example. Assume that you are a project leader. There are a set of tasks to be achieved and a set of available agents, where each agent has different skills to achieve the tasks; the cost for hiring an agent typically varies with its capabilities, and you have a limited budget $c$. Your objective is to select a team (i.e., a subset of agents), which is able to achieve the tasks of interest. Considering a team with all agents is enough to determine efficiently whether the tasks will be achievable. But the great team does not meet the budget constraint in the general case: one looks for a $c$-costly team (i.e., a team which can accomplish the goal, but for an expense upper bounded by the limited budget $c$) or even for a team which minimizes the global expense.

Furthermore, what happens if some of the team members fall sick once the team has been formed? Clearly enough, some tasks might not be achieved and it becomes possible that the whole project ends in failure. This is obviously unexpected. In order to be able to manage the case when such failures occur, an approach consists in addressing the robustness issue for task-oriented team formation at the team design step. This is the main purpose of our work.

In this paper, a formal framework for the Task-Oriented Robust Team Formation (TORTF) problem is first defined and some decision and optimization problems for TORTF are pointed out. A team is viewed as $k$-robust (for a given non-negative integer $k$) if removing any $k$ agents from it leads to a remaining team which can still accomplish the given tasks. For the decision problem, the aim is to compute (when it exists) one $c$-costly and $k$-robust team, for a given cost $c$ and robustness $k$. For optimization problems, one can be interested in optimizing the robustness of the team, while keeping its cost below the given budget. Dually, one can also fix the robustness and try to find a cheapest
In this section, the problem of task-oriented robust team formation is formally defined. Both the decision problem (finding out a team which is "sufficiently" robust and cheap) and the optimization problem (finding out every team which is optimally robust and/or cheap) are considered. Also, the computational complexity of TORTF is identified.

**Definition 1. (Team formation problem description)** A team formation problem description is a tuple $TF = \langle A, P, f, \alpha \rangle$ where $A = \{a_1, a_2, ..., a_n\}$ is a set of agents, $P = \{p_1, p_2, ..., p_m\}$ is a set of tasks, $f: 2^T \rightarrow \mathbb{N}$ is a cost function, and $\alpha$ is a mapping from $A$ to $2^P$. Both $f$ and $\alpha$ are supposed to be computable in polynomial time. A set of agents $T \subseteq A$ is called a team, and a set of tasks $G \subseteq P$ is said to be a goal.

We first recall two standard properties of expected teams, namely team cost and team efficiency, that both apply to any team $T \subseteq A$. These two properties are defined as follows.

**Definition 2. (Team affordability)** Let $TF = \langle A, P, f, \alpha \rangle$ be a team formation problem description. Given a team $T \subseteq A$ and a non-negative integer $c$, $T$ is said to be $c$-costly if the cost of $T$ is less than $c$: $f(T) \leq c$.

For simplicity, in this paper, we assume that the cost of a team is given by the sum of the costs of each agent $a_i$ of the team $T$, i.e., $f(T) = \sum_{a_i \in T} f(a_i)$.

**Definition 3. (Team efficiency)** Let $TF = \langle A, P, f, \alpha \rangle$ be a team formation problem description. Given a team $T \subseteq A$ and a goal $G \subseteq P$, $T$ is said to be efficient with respect to $G$ if $T$ can accomplish $G$: $G \subseteq \bigcup_{a_i \in T} a(a_i)$.

**Example 1. (Team affordability and efficiency)** Consider the following TORTF instance: let $TF = \langle \{a_1, a_2, ..., a_k\}, \{p_1, p_2, ..., p_l\}, f, \alpha \rangle$ be a team formation description and $G = \{p_1, p_3\}$ be a goal. We set the cost $c = 8$. Table 1 shows a set of accomplishable tasks and the cost of each agent of $A$. We assume that a set of tasks is given by a mapping $\alpha$ (e.g., $\alpha(a_1) = \{p_1, p_2\}$, agent $a_1$ can accomplish the tasks $p_1$, and $p_2$, and the cost is provided by $f$ (e.g., $f(a_1) = 4$)). Consider a team $T = \{a_2, a_3\}$. Since $f(T') = f(a_2) + f(a_3) = 8$, $T'$ is $8$-costly. Also, since $G = \{p_1, p_3\} \subseteq \alpha(T') = \{p_1, p_2, p_3\}$, $T'$ is efficient w.r.t. $G$. 

In this section, the problem of task-oriented robust team formation is formally defined. Both the decision problem (finding out a team which is "sufficiently" robust and cheap) and the optimization problem (finding out every team which is optimally robust and/or cheap) are considered. Also, the computational complexity of TORTF is identified. 

The rest of the paper is organized as follows. In the next section, our framework for the Task-Oriented Robust Team Formation TORTF problem is introduced and some decision and optimization problems for TORTF are provided. The computational complexity of TORTF is then identified. The next section presents the algorithm for TORTF. Afterwards, some empirical results are provided. Just before the concluding section, some related works are discussed.
Table 1: Accomplishable tasks and cost of each agent.

<table>
<thead>
<tr>
<th>agent</th>
<th>accomplishable tasks</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>a₁</td>
<td>{p₁, p₂}</td>
<td>4</td>
</tr>
<tr>
<td>a₂</td>
<td>{p₁, p₃}</td>
<td>3</td>
</tr>
<tr>
<td>a₃</td>
<td>{p₁, p₂, p₃}</td>
<td>5</td>
</tr>
<tr>
<td>a₄</td>
<td>{p₂, p₃}</td>
<td>2</td>
</tr>
<tr>
<td>a₅</td>
<td>{p₁, p₂, p₃, p₄}</td>
<td>9</td>
</tr>
<tr>
<td>a₆</td>
<td>{p₅}</td>
<td>1</td>
</tr>
</tbody>
</table>

These two properties are standard ones. When considered together, they require the cost for a team to be kept under a certain threshold while covering the given set of tasks. The corresponding decision problem, namely Task Efficient Team Formation (TETF), is defined as follows:

**Definition 4.** (TETF)  
- **Input:** A team formation problem description TF = ⟨A, P, f, α⟩, a non-negative integer c and a goal G ⊆ P.  
- **Question:** Does there exist a team T ⊆ A such that T is c-costly and efficient w.r.t. G?

We get the following complexity result:

**Proposition 1.** TETF is NP-complete.

**Proof.** Let TF = ⟨A, P, f, α⟩, G ⊆ P and c, k ≥ 0. To check whether there exists a team T ⊆ A that is c-costly and efficient w.r.t. G, it is enough to guess a team T ⊆ A and check in polynomial time that T is c-costly and efficient w.r.t. G. Therefore, TETF ∈ NP. Let us prove now that TETF is NP-hard. We consider the following polynomial reduction from the well-known NP-hard problem SET COVER [42]: given a collection C of subsets of a finite set S and a non-negative integer k, does C contain a cover for S of size c or less, i.e., a subset C′ ⊆ C with |C′| ≤ c such that every element of S belongs to at least one member of C′? Let COV = (C, S, c) where C = {C₁, C₂, . . . , Cₙ} is a collection of subsets of a finite set S = {p₁, p₂, . . . , pₘ} and c be a non-negative integer. We associate with COV in polynomial time the tuple (TF, G, c) where TF is the team formation problem TF = ⟨A, S, f, α⟩ defined as A = {a₁, a₂, . . . , aₙ}, P = S, for every T ⊆ A, f(T) = |T| and for every aᵢ ∈ A, α(aᵢ) = Cᵢ, and G = P (i.e., G = S). Additionally, we associate with every set C′ ⊆ C the team T(C′) ⊆ A defined as T(C′) = {aᵢ ∈ A | Cᵢ ⊆ C′}. Now, let C′ ⊆ C. On the one hand, C′ has a size of c or less if and only if |T(C′)| ≤ c if and only if f(T(C′)) ≤ c and if and only if T(C′) is c-costly. On the other hand, C′ is a cover for S if and only if S ⊆ ∪Cᵢ∈C′ Cᵢ if and only if for every G ⊆ ∪aᵢ∈T(C′) α(aᵢ), if and only if T(C′) is efficient w.r.t. G. Therefore, C contains a cover for S of size c or less if and only if there is a team T ⊆ A that is c-costly and efficient w.r.t. G. This shows that TETF is NP-hard, thus TETF is NP-complete.

The induced optimization problem can be expressed as follows: given a team formation problem description, find a subset of agents of minimal cost that is efficient w.r.t. a given goal. This problem is NP-hard, since the associated decision problem is NP-hard as well.

Robustness can now be defined in formal terms as follows:

**Definition 5.** (Team robustness) Let TF = ⟨A, P, f, α⟩ be a team formation problem description. Given a team T ⊆ A, a goal G ⊆ P and a non-negative integer k, T is said to be k-robust w.r.t. G if for every set of agents T′ ⊆ T, such that |T′| ≤ k, the team T ∪ T′ is efficient w.r.t. G.

This property is a generalization of the property of team efficiency (Definition 3), considered at different strength degrees, depending on the choice of the value k. This will be more salient given the following observations:

**Observation 1.** Let T ⊆ A and G ⊆ P. T is efficient w.r.t. G if and only if it is 0-robust w.r.t. G.

To complete this observation, let us point out that team efficiency is the weakest version of team robustness, and that the property of k-robustness is monotonic when the values of k vary:

**Observation 2.** Let T ⊆ A, G ⊆ P and k > 0. If T is k-robust w.r.t. G, then T is (k − 1)-robust w.r.t. G.

Accordingly, the notion of k-robustness is non-trivial only when k takes its value within the set {0, . . . , |T| − 1}. From Observations 2 and 3, for any given team T ⊆ A and a non-empty goal G, we can conclude that when T is efficient w.r.t. G, there exists a unique, highest value k ∈ {0, . . . , |T| − 1} such that T is k-robust. We call this value the “degree of robustness” of a team w.r.t. G.

**Definition 6.** (Degree of robustness) Let TF = ⟨A, P, f, α⟩ be a team formation problem description. Given a team T ⊆ A and a goal G ⊆ P, the degree of robustness of T w.r.t. G, denoted deg₉(T) is defined as ∞ if T is not efficient w.r.t. G, and by deg₉(T) = max{k ∈ {0, . . . , |T| − 1} | T is k-robust w.r.t. G} otherwise.

When the robustness issue is added on top of this problem, the resulting problem cannot become computationally easier than the TETF problem since it requires to solve a TETF problem and also to check the k-robustness of the obtained solution (team). However, we show that the complexity of the resulting problem does not increase.

First, we show that for a given team T ⊆ A and a goal G ⊆ P, whether T is k-robust w.r.t. G can be decided in polynomial time. Before proving it, let us first introduce the following notation: for every task pᵢ ∈ P and every team T ⊆ A, let pᵢ(T) be the set of agents from T that can perform the task pᵢ, i.e., pᵢ(T) = {aᵢ ∈ T | pᵢ ∈ α(aᵢ)}. The following proposition holds:

1Please note that Observation 3 holds for non-empty sets of tasks only; indeed k-robustness w.r.t. an empty set of tasks would be trivially satisfied for any team T. However, this specific case can be ignored since we initially assumed goals to be non-empty sets of tasks in Definition 7.
PROPOSITION 2. Given a team $T \subseteq A$, a non-negative integer $k$ and a goal $G \subseteq P$, $T$ is $k$-robust w.r.t. $G$ if and only if for every task $p_i \in G$, we have: $|p_i(T)| > k$.

Proof. Let $T \subseteq A$, $k$ be a non-negative integer and $G \subseteq P$. (If Part) Assume $T$ is not $k$-robust w.r.t. $G$. By Definition 5, there exists a subset $T' \subseteq T$ of agents such that $|T'| \leq k$ and such that the team $T \setminus T'$ is $G$-efficient. Then from Definition 3, there exists a task $p_i \in G$ such that $p_i \notin \bigcup_{a_i \in T'} \alpha(a_i)$, or equivalently, that $p_i \notin \alpha(a_i)$ for every $a_i \in T \setminus T'$, that is, $|p_i(T \setminus T')| = 0$. Yet $|T| \leq k$, so $|p_i(T')| \leq k$, thus we get that $|p_i(T)| \leq k$.

(Only If Part) Assume that there exists a task $p_i \in G$ such that $|p_i(T)| \leq k$. Let $T' \subseteq T$ defined as $T' = p_i(T)$. Then by definition of $p_i(T', T)$, we have $p_i \notin \bigcup_{a_i \in T} \alpha(a_i)$, thus from Definition 3, $T \setminus T'$ is not efficient w.r.t. $G$. Moreover, since $T' = p_i(T)$ and $|p_i(T)| \leq k$, we have $|T'| \leq k$. Hence, from Definition 5, $T \setminus T'$ is not $k$-robust w.r.t. $G$. □

As a direct consequence, the degree of robustness of a team w.r.t. a goal can be computed in polynomial time:

COROLLARY 1. Given a team $T \subseteq A$, and a goal $G \subseteq P$, we have $deg_c(T) = \min\{|p_i(T)| - 1 \mid p_i \in G\}.$

We can now generalize the decision problem TETF to the Task-Oriented Robust Team Formation (TORTF) problem. TORTF is formally defined as follows:

DEFINITION 7. (TORTF)

• Input: A team formation problem description $TF = \langle A, P, f, a \rangle$, two non-negative integers $c, k$ and a goal $G \subseteq P$.

• Question: Does there exist a team $T \subseteq A$ such that $T$ is $c$-costly and $k$-robust w.r.t. $G$?

From Observation 1 and Proposition 1, TORTF is a generalization of TETF, thus it is an NP-hard problem. However, from Proposition 2, since it turns out that TORTF is not harder than TETF, the following corollary holds.

COROLLARY 2. TORTF is NP-complete.

EXAMPLE 2. (Team robustness) We consider the same TORTF instance as in Example 1: let $TF = \langle \{a_1, a_2, \ldots, a_6\}, \{p_1, p_2, \ldots, p_6\}, f, a \rangle$, $T = \{p_1, p_2, \ldots, p_6\}$ be a team and $G = \{p_1, p_2\}$ be a goal. We set the cost $c$ and the degree of robustness $k$ to $c = 8$ and $k = 1$. Consider a team $T' = \{a_2, a_3\}$. Since $\alpha(T' \setminus \{a_j\}) = \alpha(a_3) = \{p_1, p_2, p_3\} \supseteq G$ and $\alpha(T \setminus \{a_j\}) = \alpha(a_2) = \{p_1, p_3\} = G$, $T'$ is 1-robust w.r.t. $G$. This means that $T'$ can accomplish a goal $G$, even if any agent (i.e., $k = 1$), is removed from $T'$. Similarly, $T'' = \{a_2, a_3\}$ is the team that is $k$-costly (see example 1) and 1-robust. From Observation 3, the degree of robustness of this team is one.

Beyond the decision problem TORTF, several constraint optimization problems for TORTF are meaningful. Mainly, one can be interested in optimizing the degree of robustness of the team, while keeping its cost below the given budget. Dually, one can also fix the minimal degree of robustness which is expected, and try to point out a cheapest team meeting the robustness requirement.² We can also view

²These problems can be represented as Constraint Optimization Problems [28] and solved using existing COP solvers.

the TORTF problem as a bi-objective constraint optimization problem, and be interested in computing Pareto optimal (i.e., non-dominated) robust teams. Clearly, all those optimization problems are NP-hard, since the associated decision problem is NP-hard as well. In the following, we focus on the bi-objective constraint optimization problem, only. To be more precise, we are interested in computing all the Pareto optimal solutions of a TORTF problem.

DEFINITION 8. (Dominance) Let $TF = \langle A, P, f, a \rangle$ be a team formation problem description, $G \subseteq P$ be a goal and $T, T' \subseteq A$ be two teams. $T$ dominates $T'$ if and only if $deg_c(T) \geq deg_c(T')$ and $f(T) < f(T')$, or $deg_g(T) > deg_g(T')$ and $f(T) \leq f(T')$.

DEFINITION 9. (Pareto optimality) Let $TF = \langle A, P, f, a \rangle$ be a team formation problem description, $G \subseteq P$ be a goal. A team $T \subseteq A$ which is efficient w.r.t. $G$ is a Pareto optimal robust team (also called a "trade-off" team in the following) if no team $T' \subseteq A$ that is efficient w.r.t. $G$ dominates $T$.

In order to solve it, our problem is modeled as a Multi-Objective Constraint Optimization Problem (MO-COP) [20, 25] (the extension of mono-objective Constraint Optimization Problem (COP) [7, 28] to multi-criteria decision making). A COP consists of a set of variables, and a value assignment of those variables is sought in such a way that the sum of the resulting costs is optimized. In our framework, each agent $a_i$ is identified by a variable $x_i$. It takes its value from a finite, discrete domain [join, not join], expressing whether the agent will participate or not to the team. A team is a set of agents who choose the value join. In case all agents choose not join, the (empty) team cannot achieve any tasks and its cost is 0. Deriving a trade-off team consists in finding a value assignment to all agents so that the cost of the team is minimized and the degree of robustness of the team is maximized. Compared to typical MO-COPs (i.e., problems with more than two objectives), the number of Pareto optimal solutions (trade-off teams) of TORTF problem is "small", since (i) our problem is a bi-objective COP³, and (ii) the number of trade-off teams is bounded by the number of agents $|A|$, i.e., for each $k$, there exists at most one Pareto optimal robust team and $k$ is bounded by $|A|$, while the number of Pareto optimal solutions in MO-COPs is exponential in the number of agents, i.e., every possible assignment can be Pareto optimal solution in the worst case.

EXAMPLE 3. (Bi-objective constraint optimization problem) We consider the same instance as in Example 1, but the goal is changed to $G = \{p_1\}$. The purpose is now to find the set of trade-off teams w.r.t. the cost $c$ of the team and the number of removal agents $k$. Table 2 shows all Pareto optimal robust teams. There are three teams with one agent that can accomplish the goal, i.e., $T_1 = \{a_1\}$, $T_2 = \{a_3\}$, and $T_3 = \{a_2\}$ (see Table 1). Since the costs of $T_1$, $T_2$ and $T_3$ are $f(T_1) = 2$, $f(T_2) = 5$ and $f(T_3) = 3$, and all teams are 0-robust (from Observation 3), $T_1$ dominates $T_2$ and $T_3$, i.e., the cost of $T_1$ ($f(T_1) = 2$) is smaller than $5 (= f(T_2))$ and $3 (= f(T_3))$. Let $v_i$ be the vector of values for $T_i$, where $1 \leq i \leq 3$. The following three bi-objective vectors are obtained: $v_1 = (2, 0)$, $v_2 = (5, 1)$, and $v_3 = (10, 2)$ (the first

³In general, the number of Pareto optimal solutions becomes larger when the number of objectives increases.
Table 2: All trade-off teams for bi-objective constraint optimization problem.

<table>
<thead>
<tr>
<th>team</th>
<th>agents</th>
<th>accomplishable tasks</th>
<th>cost</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₁</td>
<td>{a₁}</td>
<td>[p₁,p₂]</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>T₂</td>
<td>{a₂,a₃}</td>
<td>[p₁,p₂,p₃]</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>T₃</td>
<td>{a₂,a₃,a₄}</td>
<td>[p₁,p₂,p₃,p₄]</td>
<td>10</td>
<td>2</td>
</tr>
</tbody>
</table>

Algorithm 1: AORT (and ART)

1: INPUT : a team formation problem description TF and non-negative integers maxCost and k, (plus a Boolean value Decision when decision problems are to be handled).
2: OUTPUT : the set of all trade-off teams PF
3: PF ← ∅ // a set of teams
4: T ← ∅ // a set of agents
5: solve (1, T, PF, TF, maxCost, k, Decision) // start with the first agent of the ordering
6: Return PF

Algorithm 2 solve (N, T, PF, TF, maxCost, k, Decision)

1: INPUT : a non-negative integer N, a team T, a set of teams PF, a team formation problem description TF, two non-negative integers maxCost and k (plus an additional Boolean value Decision when decision problems are to be handled).
2: // 1) Check whether all agents has been assigned
3: if N > |A| then
4: if T is not efficient then
5: Return
6: end if
7: Decision then return {T} // for decision problem
8: end if
9: if T is not dominated by any element of PF then
10: Remove all teams dominated by T from PF
11: PF ← PF ∪ {T}
12: end if
13: Return
14: end if
15: // 2) Assign agent N
16: a: the Nth agent of A
17: T ← T ∪ {a}
18: // 3) Check the affordability
19: if cost(T) > maxCost then
20: solve (N + 1, T \ {a}, PF, TF, maxCost, k)
21: Return
22: end if
23: // 4) Check the efficiency
24: MinAgent := 0
25: if T is not efficient then
26: MinAgent ← MinAgent + k + 1
27: end if
28: if T is kT-robust and kT < k then // kT is the degree of robustness of the current team
29: MinAgent ← MinAgent + (k − kT)
30: end if
31: if (|A| − N) < MinAgent then
32: T ← T \ {a}
33: Return
34: end if
35: // 5) Check the dominance
36: Maxk := (|A| − N)
37: if T is kT-robust then
38: Maxk ← Maxk + kT
39: end if
40: for all team T’ of PF do
41: if cost(T’') < cost(T) and kT’ ≤ Maxk then
42: solve(N + 1, T \ {a}, PF, TF, maxCost, k)
43: Return
44: end if
45: end for
46: // 6) Continue the search with the next agent
47: solve(N + 1, T, PF, TF, maxCost, k)
48: solve(N + 1, T \ {a}, PF, TF, maxCost, k)
49: Return

cordinate represents the cost of each team and the second coordinate gives the degree of robustness). Clearly, no solu-
tion (team) is dominated by another one, hence \{T₁, T₂, T₃\} is the set of Pareto optimal robust (trade-off) teams of this bi-objective constraint optimization problem.

3. ALGORITHMS FOR TASK-ORIENTED ROBUST TEAM FORMATION

In this section, in order to solve decision and optimization problems, we present a branch and bound search-based algorithm, which is a widely used technique for solving MO-COPs [20]. When the decision problem is considered, the algorithm is referred to as ART; this algorithm aims at computing one “satisfying” solution, i.e., a c-costly and k-robust team where c and k are given. When bi-objective constraint optimization problems are considered, the algorithm is referred to as AORT; this algorithm computes every Pareto optimal robust team (each of them corresponds to a trade-off between cost c and degree of robustness k). We mainly describe how AORT works, since ART can be viewed as a by-product of AORT.

Algorithms 1 and 2 give the pseudo-codes for AORT (and also ART). Initialization is made in Algorithm 1 where the input is a team formation problem description TF, two non-negative integers maxCost and k, and the expected output is the set of all trade-off teams PF (lines 1 and 2). In this algorithm, we assume that a variable ordering (corresponding to an ordering over agents) is provided. The algorithm starts the search with the first agent w.r.t. this ordering (line 5). The search is based on a recursive call to solve, detailed in Algorithm 2. This algorithm successively considers each agent as part of the team (then out of the team), building partial teams until either one of the bounding functions is unsatisfied or a full assignment is reached.

Let us first describe the optimization case where we consider parameters k = 0 and maxCost = ∞. Function solve takes an integer N as parameter and assigns the N-th agent of A to the team T (line 17 in Algorithm 2). It then checks the affordability, efficiency and dominance of the partial team. The team T is not affordable if its cost is superior to maxCost. In this case, the partial team T is ignored and the algorithm continues the search without considering the N-th agent as part of the team (line 20). For ensuring effi-
ciency (line 23-34), the algorithm checks if the robustness $k$ that is expected is reachable, based on the current robustness $k_T$ of the team $T$ and the remaining agents that can be added. In case $k$ is not reachable, the current partial team $T'$ can be given up. Finally, the algorithm checks if $T$ is not dominated by another team in $PF$ (where the teams found previously are stored) (line 35-45). To do this, the maximum possible robustness (denoted $Maxk$) that $T$ can reach is computed by adding the current robustness $k_T$ to the number of remaining agents that can be added ($|A| - N$). The algorithm then checks if there exists a team $T'$ in $PF$ that has a cost lower than the one of $T$ and for a degree of robustness higher or equal to $Maxk$. If such team $T'$ exists, then $T$ can be given up (line 42).

If the three conditions (i.e., affordability, efficiency and dominance) are satisfied, the search considers the $N$-th agent as part of team (line 47) and then not part of it (line 48). When the last agent (w.r.t. the ordering) has been assigned, an assignment to all agents is obtained (line 3). If $T$ is efficient and not Pareto dominated by another team in $PF$, then $T$ is added to $PF$ (line 11) and the solutions of $PF$ dominated by $T$ are removed.

When the bi-objective constraint optimization problem with parameters $k = 0$ and $maxCost = \infty$ is considered, every team $T$ will pass the affordability and efficiency checks (line 18 and 23). However, when the decision problem is considered instead, all checks must be performed and the algorithm must stop once the first efficient solution is found. We can easily obtain $ART$ for the decision problem by removing the dominance check (line 35) and giving the fixed $maxCost$ and $k$. Clearly enough, the time complexities of $ART$ and $AORT$ are exponential in the number of agents.

**Example 4. (AORT)** We explain how $AORT$ works on Example 1. The input is a TORTF instance with $A = \{a_1, a_2, \ldots, a_6\}$ and $G = \{p_1, p_2, \ldots, p_6\}$. In order to find each Pareto optimal solution, every possible team in $A$ must be considered; for each of them, its cost as well its $k$-robustness must be computed. During the search, a vector of solutions is maintained for each possible $k$ and is updated whenever a new Pareto optimal solution is found. For example, assume that the assignment $A_1 = \{a_1, a_2, a_3, a_4, a_5, a_6\}$ has been found as a possible solution for $k = 1$ with $cost = 24$. When the assignment $A'_1 = \{a_1, a_2, a_3, a_6\}$ leading to $k = 1$ and $cost = 17$ is considered, the complete assignment $A_1$ can be replaced by $A'_1$ since, for the same value of $k$, $A'_1$ has an inferior cost. For $k = 0$, if $A_0 = \{a_1, a_4, a_5\}$ with $cost = 15$ has been previously found, and the assignment $A'_0 = \{a_1, a_4, a_6\}$ with $cost = 7$ is considered, $A_0$ can be replaced by $A'_0$ since, for the same value of $k$, $A'_0$ has an inferior cost. Finally, the algorithm returns two solutions $A'_0$ and $A'_1$ that are the cost minimal teams for $k = 0$ and $k = 1$ respectively.

**Example 5. (ART)** We now explain how ART works, using again Example 1 where the set of agents $A = \{a_2, a_3, a_4\}$ is considered, only. We want to determine if there exists a team $T$ that realizes $G = \{p_3\}$ so that $T$ is 7-costly and 1-robust. ART starts with the empty set $\emptyset$ and tries adding agents until either it finds a 7-costly and 1-robust team or the team cost exceeds $7$. Starting from the empty set $\emptyset$, the first agent $a_2$ is added and thus the algorithm checks the team $\{a_2\}$. This team accomplishes $\{p_3\}$ but it is not 1-robust. The search continues by adding the next agent $a_3$. The team $\{a_2, a_3\}$ which is 1-robust but not 7-costly ($cost = 8$), is obtained. The algorithm does not add any further agent to the team $\{a_2, a_3\}$. Indeed, this team is not 7-costly and since adding agents can only increase the cost, no supersets of $\{a_2, a_3\}$ are 7-costly. The algorithm then backtracks to the team $\{a_2\}$ and adds agent $\{a_4\}$. The team $\{a_2, a_4\}$ is obtained: this team accomplishes the tasks, is 1-robust and 7-costly ($cost = 5$). Thus ART stops the search and outputs $\{a_2, a_4\}$.

### 4. SOME EMPIRICAL RESULTS

In this section, the performances of $ART$ and $AORT$ on a number of benchmarks are reported. The empirical protocol we considered is as follows. The number of accomplishable tasks per agent has been chosen uniformly at random within range $[1...10]$, considering a uniform distribution, for defining goals $G$. For bi-objective constraint optimization problems for TORTF, the domain of each variable is of size two, i.e., $\{join, not\ join\}$. We present some representative results. Each data point in the graphs and tables is an average value over 100 instances. $ART$ and $AORT$ have been implemented in Python; experiments have been carried out on a One Core Computer running at 2.6GHz with 12GB

\[\text{Figure 1: ART with heuristic h1.}\]

\[\text{Figure 2: ART with heuristic h2.}\]

\[\text{Figure 3: ART with heuristic h3.}\]

\[\text{Table: Comparison of ART and AORT.}\]
Figures 1-3 present the results obtained by ART RAM. For assessing the performances of AORT, we can observe the shape of the curve (easy-hard-easy transition) in graphs, which is well-known as “phase transition behavior” in CSP [10]. For example, the peak (where the required runtime is maximum) occurs around 6000 for the costs and 9 for the degree of robustness in Fig.1. We call such a peak the critical area. In the critical area, we get the case where the search space is not greatly reduced (the cost is not low enough and the degree of robustness is not high enough) and also the case where finding a solution is the most difficult (the cost is not high enough to have a team with many agents and the degree of robustness is not low enough to be able to reach it with only a few agents). Also, we can see that before and after the critical area ART can find a solution quickly, i.e., it can easily find a team for the given cost c and the degree of robustness k (e.g., c is around 6000 and k is 4 in Fig.1), or it can easily find that there exists no team for the given c and k (e.g., c is around 6000 and k is 14 in Fig.1). Moreover, when we compare the effect of variable-ordering heuristics, h3 outperforms h1 and h2. The average runtime of ART with h3 at the critical area is 4.2s (Fig.3), while they are 5.3s and 7.5s for ART with h1 and h2 (Fig.1 and Fig.2).

We also compared the performances of AORT with those of a naive (brute-force) algorithm (without pruning) for several numbers of agents and goal tasks. Table 3 shows the number of trade-off teams (i.e., Pareto optimal solutions) obtained by these algorithms. As we expected, for all results, i.e., $|G| = 3, 5, 7, 9$ and 11, the number of trade-off teams increases slightly with the number of agents. Thus, for $|G| = 3$, there exist in average 3.7 trade-off teams for 10 agents, 7.6 trade-off teams for 20 agents, and 12.1 trade-off teams for 30 agents. Also, we can observe that the number of trade-off teams becomes smaller, when the number of goal tasks increases. Thus, when the number of agents is 30, the average number of trade-off teams for $|G| = 3$ is 12.1, while there exist 10.6 trade-off teams for $|G| = 11$. The results can be explained by the fact that the number of teams depends heavily on the maximal degree of robustness (denoted $k_{max}$) of the problem under consideration. For example, if the team containing all agents is 8-robust, we can expect to find a 9-robust team (one per possible k between 0 and $k_{max}$). Since a wide cost range per agent and a uniform distribution model have been considered in our experiments the cases when several teams exist for the same k and the same cost are avoided. Thus, increasing the number of agents increases the potential $k_{max}$, while increasing the number of tasks makes robustness more difficult to be achieved and decreases $k_{max}$.

Note that finding all trade-off teams of a TORTF problem is generally easier than solving a bi-objective COP. In TORTF problem, the number of solutions is bounded by the number of agents, i.e., there exist at most $|A| - 1$ trade-off teams where A is a set of agents. On the other hand, in a bi-objective COP, the number of Pareto optimal solutions is often exponential in the number of agents (i.e., all assignments are Pareto optimal solutions in the worst case).

Table 4 gives the average runtimes of AORT and of the naive algorithm. The results of AORT with $h_3$ and $h_1$ are reported. The results between brackets indicate the results of AORT with $h_1$. The naive algorithm utilizes $h_1$ for agent ordering. In all cases (i.e. $|G| = 3, 5, 7, 9$ and 11), we can observe that the difference between the results of AORT and those of the naive algorithm becomes larger, when the number of agents increases. For example, in case $|G| = 3$, the average runtime of AORT is 0.001s for 10 agents, 0.8s for 20 agents and 266s for 30 agents, while the corresponding runtimes of the naive algorithm are respectively 0.01s, 3.2s and 3175s. For $|G| = 11$, the average runtime of AORT is 0.01s for 10 agents, 1.5s for 20 agents and 592s for 30 agents, while they are 0.01s, 4.3s and 421s for the naive algorithm. Also, when the number of agents is at least 25, we can observe that the difference between the results of AORT and those of the naive algorithm increases with the number of goal tasks. Moreover, the effect of variable-ordering heuristics (i.e., $h_1$ and $h_3$) becomes significant, when the number of agents becomes larger. Thus, for $|G| = 11$, the average runtime of AORT with $h_3$ is 592s for 30 agents, while it is 734s for AORT with $h_1$.

In summary, as we expected, these experimental results reveal that (i) there exists the easy-hard-easy transition for

Table 3: Average number of trade-off teams obtained by AORT (and also Naive) for $|G| = 3, 5, 7, 9, 11$.  

<table>
<thead>
<tr>
<th># agents \</th>
<th>Naive</th>
<th>AORT</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>G</td>
<td>= 3$</td>
</tr>
<tr>
<td>10</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>15</td>
<td>0.1</td>
<td>0.11</td>
</tr>
<tr>
<td>20</td>
<td>3.2</td>
<td>3.5</td>
</tr>
<tr>
<td>25</td>
<td>94.6</td>
<td>103.5</td>
</tr>
<tr>
<td>30</td>
<td>3175</td>
<td>3412</td>
</tr>
</tbody>
</table>
decision problems, and (ii) for bi-objective constraint optimization problems, the number of trade-off teams increases slightly with the number of agents. Additionally, we examined the effect of agent-ordering heuristics. For decision problems, we compared the performances of ART with three agent-ordering heuristics (i.e., random ordering h1, skill-based ordering h2 and cost-based ordering h3) and observed that h3 outperforms h1 and h2. For optimization problems, we also observed that h3 is more effective than h1, when the number of agents increases.

5. OTHER RELATED WORK

Many works have been devoted so far to the problem of forming teams. Classically, a set of tasks and a set of agents are given; each agent has some capabilities and some skills are required to achieve each task. The team performance is often viewed as the set of all capabilities of its members.

Thus, Nair et al. [23] worked on forming a team with the maximum expected value so that the team has all required skills to accomplish the tasks of interest. Vidal et al. [29] focused on task-oriented domain problems and showed how the benefits of teaming and selflessness arise in this domain. Bachrach et al. [4] introduced coalitional skill games, where the aim is to make a coalition among agents so that it can cover a set of required skills for a given task (team efficiency).

Other works considered e.g., the optimal joint action with a new ad-hoc agent [1] where agents compute their actions based on the observations of their teammates, configuration of a network of agents [8], and minimal coordination cost for a single task [15]. Liemhetcharat et al. [16, 18] considered synergetic effects among agents, and introduced a (weighted) synergy graph model to capture interactions among agents in a team. Marcolino et al. [19] focused on the diversity of a team and showed that a diverse team can overcome a uniform team. In this work, the authors provided optimal voting rules for selecting a diverse team. This property is also important for team formation, when dynamic changes are considered, e.g., some agents break down because of the unexpected accident and injury. We plan to investigate the relationship between diverse and robust teams.

In the context of robot team formation, Kaminka et al. [11] introduced a behavior-based teamwork architecture that automates collaboration in physical robots. Liemhetcharat et al. [17] considered configurable robots that are composed of modules, e.g., motors and sensors; he focused on the probability of module failures of each robot and considered how to form a multi-robot team that is robust to failures.

Coalition Structure Generation (CSG) [3, 24, 27] involves partitioning a set of agents into groups (called coalitions) so that the sum of the values of all coalitions is maximized. A partition is called a coalition structure. In CSG, the value of a coalition is given by a black box function. It is well-known that finding an optimal coalition structure is NP-hard. Indeed, the decision problem associated with CSG is equivalent to the complete set partition problem [30]. The purpose is different from the one of team formation; indeed, the objective of coal formation is to select a team (a subset of agents), which can achieve the tasks of interest, while the aim of CSG is to find an optimal partition of all agents.

None of those works actually considers the robustness issue for team formation and multi-objective setting, making them quite different from the present work.

Related to our work are also task-allocation problems [14, 31], which involve deciding how to assign a set of tasks to a set of agents. The robustness issue has been considered for these problems, e.g., in [2, 5]. In [2], Ali et al. investigated how to determine a resource allocation so that the robustness of desired system features against perturbations is maximized. This research addressed the design of a robustness metric for resource allocations. In [5], Choi et al. worked on task allocation to coordinate a fleet of autonomous vehicles and presented decentralized task-allocation algorithms that provides conflict-free solutions independent of inconsistencies in Situational Awareness (i.e., robustness to inconsistent SA). Our notion of robustness is consistent with these works: some goal must still be accomplished even when some agents break down, i.e., the formed team is robust against some potential perturbations. However, our work differs from task-allocation problems in the sense that the agents forming a team are not effectively assigned to a specific task. In fact, they are associated with a set of tasks which they have the skill for. Therefore, our work can be used as an upstream step of a task-allocation problem. Making such a link between these two frameworks is worth being considered for further research, where the dependencies between both notions of robustness will be investigated.

6. CONCLUSION

How to form a team for achieving a given set of tasks is an important issue in multi-agent systems. Task-oriented team formation is the problem of forming the best possible team to achieve some tasks of interest, given some limited resources. This paper investigated the robustness issue for task-oriented team formation.

The contribution of this paper is mainly twofold:

- The concept of task-oriented robust team has been first defined and studied. Especially, the issue of computing a robust, yet affordable team has been investigated from a computational point of view. While robustness generalizes the usual notion of team efficiency, this generalization does not lead to a computational shift.

- Two algorithms for solving the TORTF problem have been provided and evaluated. ART for solving a decision problem which aims at computing one c-costly and k-robust team, for given cost c and robustness k. AORT for solving a bi-objective constraint optimization problem which aims at computing every Pareto optimal robust solution. Experiments showed that (i) an easy-hard-easy phase transition pattern can be observed for decision problems, and (ii) for bi-objective constraint optimization problems, the number of trade-off teams increases slightly with the number of agents.

As a perspective for further research, we plan to develop some efficient heuristics and algorithms (based on Russian Doll Search [26]) for solving TORTF problems. Also, we intend to apply our approach to some real-world problems, especially rescue team formation, nurse scheduling problem and fault tolerant system design. Furthermore, we plan to extend our model to a dynamic setting in which goal tasks change with time. An objective will be to develop an algorithm that reconstructs the team after each change, and to apply it to a distributed robot team reconfiguration problem [6].
REFERENCES


