An Anytime Heuristic Algorithm for Allocating Many Teams to Many Tasks

Athina Georgara
IIIA-CSIC & Enzyme Advising Group
Barcelona, Spain
ageorg@iiia.csic.es

Juan A. Rodriguez-Aguilar
IIIA-CSIC
Barcelona, Spain
jar@iiia.csic.es

Carles Sierra
IIIA-CSIC
Barcelona, Spain
sierra@iiia.csic.es

Ornella Mich
Fondazione Bruno Kessler
Trento, Italy
mich@fbk.eu

Raman Kazhamiakin
Fondazione Bruno Kessler
Trento, Italy
raman@fbk.eu

Alessio Palmero Approsio
Fondazione Bruno Kessler
Trento, Italy
aprosio@fbk.eu

Jean-Christophe Pazzaglia
SAP
Mougins, France
jean-christophe.pazzaglia@sap.com

ABSTRACT
In many practical applications, we often need to form a team of agents to solve a task, since no agent alone has the full set of required competences or the power to complete the task on time. Here, we address the problem of distributing individuals in teams, with each team being in charge of a specific task. In particular, we formalise the problem and propose a heuristic approach to solve it.

KEYWORDS
Team Formation, Task allocation, Optimisation, Heuristics

1 INTRODUCTION
Many real-world problems require allocating teams of individuals to tasks. For instance, forming teams of robots for search and rescue missions [4], teams of Unmanned Aerial Vehicles (UAVs) for surveillance [9], or grouping students to undertake school projects [3]. Considering the domain of education and motivated by the hard and time-consuming procedure of allocating student teams to school projects or internship programs, we focus on this specific instance of team formation. Currently, teachers and education authorities obtain such allocations mainly by hand, but given the combinatorial nature of the problem, (good) manual allocation requires a large amount of work. That is, manual allocation will very likely not find a good solution given the size of the problem. In this extended abstract, we study the allocation of many teams to many tasks with size constraints, disallowing overlaps. That is, each agent can be part of at most one team, each team can be allocated to at most one task, and each task must be solved by at most one team. Then, we introduce an anytime heuristic approach to match teams to tasks.

2 RELATED WORK
Team formation has received much attention by the AI and MAS communities. Anagnostopoulos et al. in [1] thoroughly study the problem of forming a single team to resolve a single task, and show the employability of several algorithms in large scale communities. Lappas et al. in [8] tackle the problem of finding a single team of experts for a given task in an attempt to minimise the communication cost among the members of the team. Chad et al. [6] add a new dimension to the problem by considering robustness, and focus on finding a single robust team to perform several tasks. Andrejczuck et al. [3] tackle the many teams to single task problem and present algorithms for partitioning a set of agents into equal-size teams in order to resolve the very same task. Capezzuto et al. in [4] tackle the many teams to many tasks team formation problem considering temporal and spacial constraints, and propose an anytime, efficient algorithm. However, compared to the problem we tackle in this paper, the proposed algorithm in [4] provides solutions with overlapping teams.

Regarding the many teams to many tasks team formation problem with no overlaps—i.e., problems for which different teams share no common agents, each team can be allocated to only one task, and each task can be assigned to only one team—we can find a handful of works in the literature. Specifically, we have singled out two works, namely [7, 10], which can be considered as the one most directly related to ours. Although these papers tackle the general many teams to many tasks problem, their version of the problem is essentially different to ours, hence preventing us from conducting

meaningful comparisons. In more detail, [10] propose a branch-
and-bound technique to determine the optimal team size structure
and then they proceed with a brute-force search. On the other
hand, Czatnecki and Dutta [7] propose an algorithm for matching
non-overlapping teams of robots to tasks. Similarly to [10], [7] sets
no constraints on team sizes. However, even if we could ’bypass’
the team size misalignment (by allowing [7] to yield a result, and
use these team sizes in our version), there is yet another essential
difference between [7] and our approach. Our algorithm pursues
to optimise the competence affinity between all teams with their
assigned tasks while targeting at balanced allocations (i.e., all teams
are more or less equally competent for their task). Instead, [7] tar-
gets at finding stable teams such that the agents have no incentive
to unilaterally abandon their current team and task. As such, [7]
and our approach differ notably in their objectives.

3 THE MANY TEAMS TO MANY TASKS
ALLOCATION PROBLEM
In this section we formally describe the many teams to many tasks
allocation problem with no overlaps. To begin with, a competence
corresponds to a specified capability, skill, or knowledge. We as-
sume there is a known, predefined and fixed set of competencies,
denoted by C. A task is characterised by a set of requirements on
agents’ competencies and team size constraints; formally, it is a
tuple (t_id, C, w, s), where t_id is a unique task identifier, C ⊆ C
is the set of required competencies, w : C → (0, 1] is a function
that weighs the importance of each competence, and s ∈ ℕ is the
required team size. An agent a is a tuple (a_id, C’), where a_id is a
unique agent identifier, and C’ ⊆ C is a set of competencies. We
denote with T and A the sets of all tasks and all agents, respectively.
Given τ ∈ T, we denote the set of all size-compliant teams for τ as
Kτ = {K ⊆ A : |K| = sτ}1, where sτ is the team size required by τ.

3.1 Matching Quality
The quality of a size-compliant team when solving a task depends on
the competencies required by the task and those collectively offered
by the team. Given a task and a team, we first shall assign ‘respon-
sibilities’ to all team members, i.e., which required competencies
shall each agent cover/fulfil. As soon as each agent is assigned with
responsibilities, we can determine the quality of each agent against
the task, by comparing their assigned competencies (as responsibili-
ties) with their own ones. Thus, the quality of an agent a ∈ K for
a task τ is given by \( \prod_{c \in \eta_{τ→K}(a)} q_l(c, C_a) \), where \( \eta_{τ→K} : K → 2^{C_τ} \)
is a competence assignment function, and \( q_l : C × 2^{C} → [0, 1] \)
is an evaluation function that measures a quality degree for covering
some competence c ∈ C_τ if the agent has competencies C_a. Note
that given a task and a size-compliant team, assigning responsibili-
ties to agents in a way that maximises the overall team’s quality is
an optimisation problem itself. The quality of a team K for tackling
task τ is given by aggregating the quality of each a ∈ K for its given
responsibilities, i.e., \( \prod_{a \in K} \prod_{c \in \eta_{τ→K}(a)} q_l(c, C_a) \).

3.2 The Optimisation Problem
Finding a good allocation of agents to a collection of tasks is yet
another optimisation problem whose aim is to maximise the overall
matching quality of all teams for their assigned tasks. Specifically,
we need to match each task \( \tau ∈ T \) to at most one team, so that
the matching quality of all pairs team-task is similarly high, i.e.,
the optimum allocation should promote balanced teams instead of
a few high-quality teams together with poor-quality teams [2, 3].
Therefore, we use the product among the matching qualities of all
team-task pairs [5]. Formally, let \( g : T → 2^A \) be an allocation
function that maps each \( \tau ∈ T \) to a size-compliant team \( K ∈ K_τ \),
so that each agent belongs to at most one team. That is, \( g(\tau) \cap g(\tau') = \emptyset \)
for all \( \tau \neq \tau' \). Then, the optimal allocation function is:

\[
g^* = \arg \max \prod_{\tau \in T} \prod_{a \in g(\tau)} \prod_{c \in \eta_{\tau→g(\tau)}(a)} q_l(c, C_a)
\]  (1)

Notice that in each allocation \( g \), for each pair \( g(\tau) \sim \tau \) we consider
the optimal competence assignment function \( \eta_{\tau→g(\tau)}^* \). Thus, for
each team allocation function we need to solve \( |T| \) optimisation
problems (one per task) in order to determine \( \eta_{\tau→g(\tau)}^* \).

4 A HEURISTIC APPROACH
Here we sketch a heuristic approach that allows us to obtain many
team to many tasks allocations in acceptable time. Our approach
comprises a two-step greedy search.

First Step: The first step finds an initial allocation considering a
competence analysis on the agents and the tasks at hand. That is,
for each task we determine whether it is hard to find a high-quality
team based on the task’s requirements and the agents’ competencies.
Thus, for each team allocation function we need to solve \( |T| \) optimisation
problems (one per task) in order to determine \( \eta_{\tau→g(\tau)}^* \).

Second Step: As soon as our algorithm finds an initial allocation,
we improve the allocation by interchanging agents between pairs
of teams following a greedy procedure. That is, for two randomly
selected tasks we optimally re-allocate the agents in their teams in
order to obtain the highest quality. This step is performed as many
times as time allows, always having a feasible allocation (anytime
algorithm).

5 CONCLUSIONS & ONGOING WORK
In this work, we studied a novel type of team formation problem:
the many teams to many tasks allocation problem with no over-
laps. First, we provided the formulation of the problem, and then
we introduced the optimisation problem leading to a solution. Fi-
nally, we sketched an anytime heuristic approach that solves this
optimisation problem. Currently, we are working on a systematic
evaluation of our approach to confirm its effectiveness. Specifically,
we are conducting an empirical analysis both using our approach
in real-world problems and performing an experts’ validation. Our
initial results are promising regarding the quality of the solutions
obtained by the heuristic algorithm.

\(^1\)Note: we use subscript \( a \) to refer to the set of competences and the identifier of an
agent \( a \in A \), and subscript \( \tau \) to refer to the elements of task \( \tau ∈ T \).

\(^2\)Note that for a fixed team allocation, finding the optimal competence assignment for
one pair team-task is independent from finding the optimal competence assignment
for any other pair.
REFERENCES


