An ideal team is more than a team of ideal agents*

Can Kurtan, Pınar Yolum, and Mehdi Dastani

Utrecht University, The Netherlands
{a.c.kurtan, p.yolum, m.m.dastani}@uu.nl

The problem of forming teams has been a central question in many disciplines. Put simply, the problem is to form a set of agents with required capabilities so that they can perform a task together. In the domain of service composition, this would correspond to a set of service providers, each performing a single service, which overall yields a composition of the desired service. In the domain of query answering, this would correspond to data sources that can each answer part of a given question. In the domain of rescue operations, this would correspond to a set of human and robot agents that work to help civilians.

The term team may refer to a set of agents only or a set of agents and their specific subtasks assignments. Following this, here we refer to teams as the set of agents and their assigned subtasks. Assigning subtasks to individual agents is typically modeled as an optimization problem, where coverage of the subtasks is maximized, or the number of agents involved is minimized. Earlier approaches model team formation with three important assumptions: (i) the overall task can be divided into independent subtasks, (ii) agents' capabilities as to what subtask they can do is known or computed easily and (iii) the agents' capabilities of performing a subtask is binary (e.g., no variation in the quality of the subtask performance). Even under these assumptions, the problem of assigning subtasks to individual agents is known to be NP-hard.

In many domains, these assumptions do not hold. Consider the following simple example from a query answering domain, where the question is to find the title, author and summary of books. Three agents separately provide book names, book authors, and book summaries. An answer to the question can be found by using the information of these agents if they provide title, author and summary information of the same books. The individual performance of an agent depends on which subquery the agent is assigned. For example, one agent provides three book titles and a thousand book summaries, while another agent provides a thousand book titles and three book summaries. The number of books returned in the answer depends on which agents are assigned to which subqueries. Hence, building a team based on finding best performing individuals does not always yield the best performing team.

This paper summarizes our work [1] that addresses the above challenges: building teams for tasks whose subtasks are dependent to each other and the performance of agents is affected by the assignments. We represent the interplay

* This work was done in the context of the Golden Agents project, funded by the Netherlands Organization of Science NWO – Large Investments program.

between possible assignments of the agents using *expertise graph*, which stores information on how well agents perform tasks individually and how well they can support each other on common or different subtasks. It represents agent-capability pairs as nodes and pair-wise co-performances as edges. An edge between two nodes can be one of the three types: *c*-edge denoting how well the agent can carry out two subtasks, *j*-edge denoting two agents working on two different tasks, and *s*-edge denoting how well one agent supports the other on the same subtask.

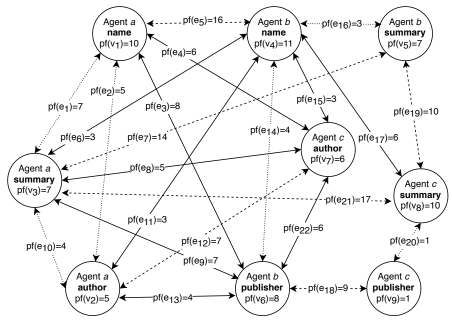

Fig. 1. An example expertise graph

We develop two algorithms: *one shot team building* and *iterative team building* algorithms. Our algorithms use the expertise graph to approximate how likely the agents are to perform well in dependent subtasks. We provide graph metrics, such as cooperativeness and versatility, to use as heuristics. The one shot team building algorithm selects assignments that have the highest value for a given metric. It is similar to the traditional algorithms that maximize one specific property of a team, such as communication. The iterative team building algorithm starts with an existing team for a task, which might be generated with a tool in hand, and improves the team to yield a better team performance in an iterative manner. It replaces an assigned agent with another agent that has better local cooperativeness in that specific team. The algorithm uses performance estimation functions to decide when to iterate to find a better team or to stop.

We demonstrate the workings of our algorithms in a query answering multiagent system, where agents are data providers and tasks are queries. We evaluate the algorithms in an experimental setup and compare the performance of the algorithms. We see that for smaller task sizes, the teams built by the one shot algorithm obtain better results. However, for larger task sizes, the iterative algorithm outperforms the one shot algorithm consistently. This shows that when the task is large, building a team by just adding "ideal" agents does not capture the relations among them. It is necessary to consider how the team will perform as a whole and update the team when an agent does not fit the team. Our work differs from other approaches in that we consider how well two agents work together on a given subtask as well as the possible degradation of performance when multiple subtasks are performed by a single agent. Further, we can build better performing teams by improving existing teams incrementally.

References

1. Kurtan, C., Yolum, P., Dastani, M.: An ideal team is more than a team of ideal agents. In: Proceedings of the 24th European Conference on Artificial Intelligence (ECAI). pp. 43–50 (2020)