Building Contrastive Explanations for Multi-Agent Team Formation[*]

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

ABSTRACT

As more and more hard and complex procedures are being automated with the aid of artificial intelligence, the need for humans to understand the rationale behind AI decisions becomes imperative. Adequate explanations for decisions made by an intelligent system do not just help describing how the system works, they also earn users' trust. In this work we focus on a general methodology for justifying why certain teams are formed and others are not by a team formation algorithm (TFA). Specifically, we introduce an algorithm that wraps up any existing TFA and builds justifications regarding the teams formed by such TFA. This is done without modifying the TFA in any way. Our algorithm offers users a collection of commonly-asked questions within a team formation scenario and builds justifications as contrastive explanations. We also report on an empirical evaluation to determine the quality of the explanations provided by our algorithm.

KEYWORDS

Explainable AI, Explainability, Team Formation

ACM Reference Format:
Athina Georgara, Juan A. Rodriguez-Aguilar, and Carles Sierra. 2022. Building Contrastive Explanations for Multi-Agent Team Formation. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Online, May 9–13, 2022*, IFAAMAS, 9 pages.

1 INTRODUCTION

In an era where artificial intelligence can be practically found in any system, it is more and more common that people make decisions guided by the suggestions and recommendations of some intelligent system. As these systems support everyday life's decisions they unavoidably make people curious about their functionality. Often, users question the rationale of the AI system in use, bearing a feeling of 'distrust' with machines. Explainable artificial intelligence (XAI) [27] aims at alleviating this distrust by providing answers to questions like "How does this machine learning model operate?", or "Why did the AI system reached this decision?". Over the past years, XAI has attracted much attention with a main focus on explaining classification and machine learning (ML) models [2, 12, 16, 28, 33–35]. Apart from general ML models, recommender systems are the main type of algorithms for which explanations are needed [23, 26, 37], explainability is specially critical for recommender systems as they need to earn the trust of users so that they accept the recommendations provided.

Recently, Kraus et al. [24] argued about the importance of explaining decisions in multiagent environments (xMASE), an area that has received little attention so far. As described by the authors in [24], providing explanations in a multiagent context is rather challenging. In particular, xMASE requires: (1) identifying the technical reasons for a decision, (2) adapting the answer to the different agents' preferences, and (3) deciding what kind of information can be revealed considering privacy and fairness issues. Moreover, [24] also states that XAI usually ignores the explanations' social nature, or 'misses' the human factor during the explanation evaluation, which are essential aspects in the context of xMASE.

As discussed by [24], there is just a handful of recent research works tackling xMASE in a few application domains. Among them, [29] introduces a system that recommends a sharing policy (i.e., a policy for sharing posts in online social networks) within multiuser environments that can justify its proposed sharing policies. Another work in the context of multiagent systems proposes an explanation scheme for the *multiagent path finding* problem that justifies the agents' routes [3]. A further MAS contribution builds explanations to justify winners' selections in voting settings [10].

In this paper, we turn our attention to a well-studied multiagent system problem, that of *team formation*. There is a plethora of works that tackle this problem (e.g. [5, 6, 14, 15, 25]) and investigate different versions of it. The interest is justified as there are many domain applications relevant to the team formation problem (e.g. forming and coordinating working groups within companies [9, 21], grouping students to undertake school projects [6], etc.) Despite the interest, to the best of our knowledge, the problem of providing explanations to team formation decisions has not been addressed yet. In this paper we try to make headway in this matter. Thus, we propose a general method that allows to build justifications for the decisions of team formation algorithms *without modifying them*. Our method *wraps* a team formation algorithm at hand and calls it, possibly several times, to build *contrastive explanations* [27]. Such type of explanations are motivated by the fact that people expect explanations justifying the decision taken compared to another decision (that was not taken). In particular, our method focuses on building explanations for team formation algorithms following a general model of team formation for task allocation (based on [6]) that we introduce in this paper. More precisely, we make the following contributions:
(1) We introduce a novel, general algorithm for building contrastive explanations in the context of team formation; without modifying the team formation algorithm at hand.
(2) We identify a collection of query templates that allow to challenge the decisions of a TFA from multiple perspectives.

[*]Research supported by projects AI4EU (H2020-825619), TAILOR (H2020-952215), 2019DI17, Humane-AI-Net (H2020-952026), Crowd4SDG (H2020-872944), and grant PID2019-104156GB-I00 funded by MCIN/AEI/10.13039/501100011033.

(3) We detail how our justification algorithm translates queries posed by a questioner into constraints and how these are handled by the code wrapping the TFA. Furthermore, we analyse the varying cost of explanations depending on each type of query.
(4) We propose an explanation building technique that presents different points of view of an explanation.
(5) Based on novel evaluation metrics that we introduce, we empirically evaluate the quality of explanations and show that, despite the complexity of the scenario, they are easy to understand, requiring just the reading level of a high-school student.

The paper is structured as follows. In Section 2 we present the general team formation model for task allocations. Section 3 outlines our novel methodology for building explanations. In Section 4 we identify query templates that are relevant to team formation, and in Sections 5 and 6 we show how to handle the queries. Namely, how to translate and incorporate them into an *extended* version of the original team formation problem that must be solved to build explanations. Section 7 describes the building and tailoring of contrastive explanation, while Section 8 conducts a systematic evaluation of the quality of the explanations we generate. Finally, Section 9 concludes the paper and discusses future work.

2 A GENERAL TEAM FORMATION PROBLEM FOR TASK ALLOCATION

In this section we define a general team formation problem that generalises previous proposals. In particular, the team formation problems defined in [6, 11, 17, 18].

Let $A = \{a_1, a_2, \cdots, a_n\}$ be a set of agents with $|A| = n$, and $T = \{\tau_1, \tau_2, \cdots, \tau_m\}$ be a set of tasks with $|T| = m$. A team is a subset of agents in A. Teams of agents are built with the purpose that its agents jointly tackle one or several of the tasks in T. A team formation algorithm (TFA for short) forms such teams. More precisely, a TFA returns a team allocation function that assigns teams to tasks, denoted by g. In the literature we find several variants of team formation problems, and therefore different TFAs. That is, there are TFAs that find a single team to tackle a single task [4, 5, 25]; TFAs that find a single team to tackle multiple tasks [14]; TFAs that find multiple teams to tackle a single task [6]; and TFAs that find multiple teams to tackle many different tasks [8, 11, 15, 17, 31].

Typically, a TFA forms teams based on a number of desirable properties (e.g. agents with specific skills [4, 6, 7, 25], agents whose location is close to the task's location [11], etc.). The satisfaction of such properties determines how good a team is for tackling a task. In what follows, we refer to such properties as *attributes.* Formally, we represent attribute i as an evaluating function $f_i : 2^A \times T \rightarrow \mathbb{R}$, which determines how good is a team for a task from i's perspective. We define the adequacy of matching team K with task τ with respect to a set of attributes as an *oracle* function:

$$u(K, \tau, F) \in \mathbb{R}^+ \cup \{0\} \tag{1}$$

where $F = \{f_1, f_2, \cdots, f_r\}$ is a set of attribute functions.[1] As such, a TFA does not compute the quality of a team for a task, instead it consults with the oracle u to obtain such information. In fact, any TFA is driven by the quality value of a whole team allocation as specified by g. Thus, the quality of allocation g is naturally an aggregation over the values $u(g(\tau), \tau, F)$ for all $\tau \in T$, i.e., the quality of g is given by $v(g) = \mathcal{F}_{\tau\in T}\, u\big(g(\tau), \tau, F\big)$, where $\mathcal{F}$ is some aggregating function. Moreover, a TFA may be able to handle some constraints imposed by the Team Formation Problem (TFP) at hand. Such constraints may refer to: whether an agent can participate in multiple teams, and if so what is the maximum workload per agent; acceptable team sizes; whether every agent must be part of at least one team; etc. Notice though that not all TFAs deal with constraints [15, 25]. A TFA that does not handle constraints cannot be used if the solution must respect certain constraints, some TFAs can handle only a specific type of constraint (e.g., each agent must be assigned to exactly one task) [4, 5, 31], others can solve a rich variety if constraints [6, 11, 17]. With this in mind, we define our generalised model of team formation problem for task allocation, as follows:

Definition 2.1 (Team Formation Problem for Task Allocation (TFP-TA)). A Team Formation Problem for Task Allocation p is represented by a tuple $\langle A, T, F, u, C\rangle$, where $A = \{a_1, \cdots, a_n\}$ is a set of agents; $T = \{\tau_1, \cdots, \tau_m\}$ is a set of tasks; $F = \{f_1, f_2, \cdots, f_r\}$ is a set of attribute functions; u is the oracle determining the suitability of a team $K \subseteq 2^A$ for a task $\tau \in T$ across all attributes in F; and $C = \{c_1, c_2, \cdots, c_s\}$ is a possibly empty set of linear constraints (with the understanding that a problem with $C = \{\emptyset\}$ is equivalent to an unconstrained problem).

There are many instances and variations of TFP-TAs, and therefore there are several corresponding TFAs [8, 11, 15, 17, 31]. In this paper, we propose how to build explanations for problem instances of the TFP-TA defined above, without focusing on any particular TFA. In other words, given an instance of a TFP-TA we can use any available TFA that can solve the problem, and we will be able to build contrastive explanations for a rich set of questions about the solution found. And this without modifying the TFA. Let us now present an example which we will be using throughout the paper.

Example 2.2. In a university class a professor must divide their 20 students into teams of size 4 for them to work on their semester projects. Each semester project shall cover a different topic of the course. The professor offers five projects: building a pathfinder agent-squad to explore unknown planets in a simulation *(pathfinder)*, building competitive agents to play chess using reinforcement and Q learning *(chess)*, building agents to solve SUDOKU problems using probabilistic inference and probabilistic graphical models *(sudoku)*, building trading agents to play the game "Shelters of Catan (SoC)" *(trading)*, and building competitive agents simulating electrical power producers and consumers *(energy)*. While every student must be part of exactly one team, ideally the professor would like to compose teams such that: (i) the members of each team have complementary skills to tackle their assigned semester project; (ii) the members of each team have diverse personalities; and (iii) students' preferences over projects are satisfied as much as possible. In this case, the TFP-TA would be $p = \langle A, T, F, u, C\rangle$, where:
$A = \{a_1, a_2, a_3 \cdots, a_{20}\}$ is a set of 20 students;
$T =$ {pathfinder, chess, sudoku, trading, energy};
$F = \{f_{\text{skills}}, f_{\text{pers}}, f_{\text{sat}}\}$ are the desired attribute's since the value of a team working on a semester project depends on the students'

[1]Using an oracle function is common in several team formation algorithms, e.g., [8, 11, 30].

complementary skills for the project (f_{skills}), their balanced personalities (f_{pers}), and their interests in the project (f_{sat});
u is the oracle determining the quality of a team for a given task; and
C = {team size is 4, each student takes part in a single team} are the constraints to fulfil.

3 THE JUSTIFICATION ALGORITHM

In this section we outline the algorithm wrapper to justify team membership and assignments of teams to tasks. Given a TFP-TA p and a TFA that can solve it we first need to *wrap* the TFA so that it can now deal with the extra constraints C_q associated with a particular question q in order to build contrastive explanations. We will explain later on how this wrapper function is built. For now, let us note the TFA algorithm with its u function changed as TFA($\tilde{u}$) and see the definition of a Query-Compliant TFP-TA.

Definition 3.1 (Query-Compliant TFP-TA (QTFP-TA)). Given a Team Formation Problem for Task Allocation $p = \langle A, T, F, u, C \rangle$, and a question q we define a Query-Compliant TFP-TA (QTFP-TA) as $p' = \langle A, T, F, u, C, \tilde{u}, C_q \rangle$ where $\tilde{u}$ is the wrapping function of u that satisfies the constraints C_q derived from question q.

Our justification algorithm follows the steps illustrated in Figure 1 and described below:

(1) The TFA algorithm solves problem p and yields a team allocation g.
(2) A questioner makes a query q regarding team allocation g.
(3) Query q is translated into a set of query-constraints C_q.
(4) A problem transformation process combines the original problem p with the query constraints in C_q to produce a *Query-Compliant* TFP-TA p'.
(5) The TFA($\tilde{u}$) solves the QTFP-TA p' and outputs a query-compliant team allocation g'.
(6) A builder of contrastive explanations compares the original team allocation (g) with the query-compliant allocation (g') to analyse their differences and generate explanations.
(7) Finally, the contrastive explanations are tailored, to highlight different perspectives, and passed back to the questioner.

Next we go through the steps above in some more detail to explain the whole justification algorithm. However, further details are left for Sections 4-7. To begin with, given a TFP-TA $p = \langle A, T, F, u, C \rangle$, the TFA solves the problem and produces a team allocation g. Thereafter, a questioner questions the allocation g. That is, a questioner may ask, for instance, why a particular team was assigned to a task, or why a questioner-made team was not assigned to the task. Following Example 2.2, consider that the TFA outputs the team allocation depicted in the second column of Table 1, the one labelled as *Original.*

The professor studies the allocation, and, acting as questioner, places a question regarding g: for instance, "Why is student a_{15} assigned to the pathfinder project?". In Section 4 we introduce a number of query templates that focus on team formation problems. Our justification algorithm deals with such questions by providing contrastive explanations [27]. Contrastive explanations are based on findings in the philosophical and cognitive sciences literature indicating that people are not interested in the causes leading to a particular outcome (in our case an allocation) *per se*, but, on the contrary, they are interested in the causes that explain a non-occurring outcome. In other words, people are interested (and also tend to give) explanations regarding questions of the type "Why X instead of Y?". For instance, in our running example, the professor is interested in the causes that led TFA assign student a_{15} to the pathfinder project instead of assigning them to a different project.

Semester Project	Original (g)	Query-Compliant (g')
pathfinder	$\{a_1, a_6, a_7, a_{15}\}$	$\{a_1, a_6, \boldsymbol{a_8}, \boldsymbol{a_{18}}\}$
chess	$\{a_3, a_8, a_{13}, a_{18}\}$	$\{a_3, \boldsymbol{a_7}, a_{13}, \boldsymbol{a_{16}}\}$
sudoku	$\{a_2, a_{11}, a_{17}, a_{19}\}$	$\{a_2, a_{11}, a_{17}, a_{19}\}$
trading	$\{a_4, a_9, a_{14}, a_{20}\}$	$\{a_4, a_9, a_{14}, a_{20}\}$
energy	$\{a_5, a_{10}, a_{12}, a_{16}\}$	$\{a_5, a_{10}, a_{12}, \boldsymbol{a_{15}}\}$

Table 1: Original (g) vs Query-complaint (g') Allocation

To build such a contrastive explanation, we need to compute the *differences* between the original allocation produced by the TFA and another, alternative allocation, determined by the particular question of the questioner, produced by the TFA($\tilde{u}$). Thus, given a question q made by the questioner, the justification algorithm processes this question and proceeds on building an alternative, *query-compliant allocation.* This procedure is implemented by steps 3-5 in Figure 1. First, our justification algorithm translates the question posed into a set of constraints C_q. For instance, in our example, question "Why is student a_{15} assigned to the pathfinder project?" is translated into one constraint that *forbids* student a_{15} to be assigned to the pathfinder project. Thorough details about how to translate queries follow in Section 5. This problem p' is a *query-compliant* TFP-TA, that is, p' is an extension of the original problem p with the query-constraints C_q and the wrapper function $\tilde{u}$.

Back to our example, problem p' would split the 20 students into teams of size 4, though constrained to assigning each team to exactly one semester project, and ensuring that every student participates in exactly one team (likewise the original problem p). However, p' must consider an extra constraint: "student a_{15} must not be assigned to the pathfinder project". The QTFP-TA p' is then solved by TFA($\tilde{u}$). The output is a query-compliant allocation g' that respects the query constraints. For instance, in the allocation depicted along the third column in Table 1, student a_{15} is assigned to the energy project. Note that the query-compliant allocation is the best allocation that TFA($\tilde{u}$) could find, while fulfilling the query-constraints. Once our justification algorithm gets the original and the query-compliant allocations (g and g' respectively), it is ready to build a contrastive explanation by computing the differences between the two.

As soon as the contrastive explanation builder finds these differences (step 6 in Figure 1), the generated explanations go through a *tailoring process.* This process highlights different points of view. Specifically, as we discuss in Section 7, we focus on different levels of abstraction in order to provide the questioner with a suitable explanation. In our example, our justification algorithm would generate three different types of explanations: *(a)* one referring specifically to student a_{15}, *(b)* one referring to teams and tasks, and *(c)* one referring to the allocation as a whole.

The following sections detail the main processes of the algorithm: Section 4 identifies a number of meaningful questions regarding team formation, Section 5 shows the translation process of these

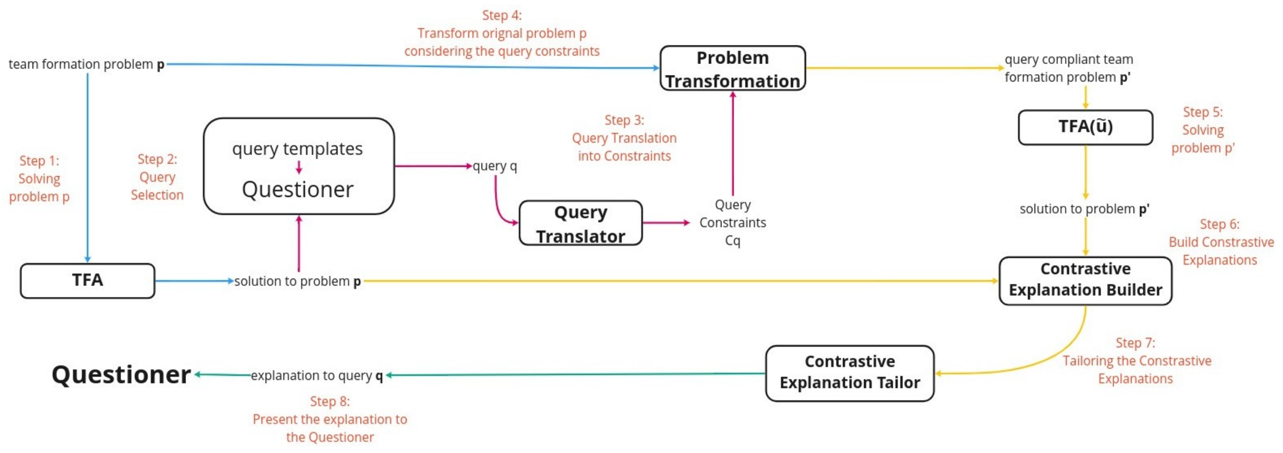
Step 4:
Transform orignal problem p considering the query constraints
team formation problem p
Problem Transformation
query compliant team formation problem p'
Step 3:
Query Translation into Constraints
Step 5:
Solving problem p'
TFA(ũ)
Step 1:
Solving problem p
Step 2:
Query Selection
query templates
Questioner
query q
Query Translator
Query Constraints Cq
solution to problem p'
Step 6:
Build Constrastive Explanations
TFA
solution to problem p
Contrastive Explanation Builder
Step 7:
Tailoring the Constrastive Explanations
Questioner
explanation to query q
Contrastive Explanation Tailor
Step 8:
Present the explanation to the Questioner

Figure 1: The Justification Algorithm.

queries into query constraints, Section 6 transforms the initial TFP-TA to accommodate such constraints, and Section 7 elaborates on how to build contrastive explanations.

4 IDENTIFYING USER QUERIES

Given a team formation scenario, there are several types of questions that a questioner may ask regarding the allocation computed as the solution of a team formation problem. Here we identify a collection of questions that are intuitive, meaningful, and cover, in our opinion, the main points of interests regarding team formation scenarios. In Table 2 we list the collected questions as query templates. There, we distinguish two types of queries:

Collaboration Queries question the established collaboration between agents. Thus, they question the teams formed while disregarding their assignments. Queries of this type consider the teams formed in a given allocation, focusing on the complete team (see queries Q8, Q9 in Table 2), or on individual agents (see queries Q10, Q11 in Table 2). This type of queries also include questions about the participation of specific agents in a team (see queries Q7, Q12 and Q13 in Table 2).

Assignment Queries challenge the assignment of tasks to teams and individuals. This type of queries concern the assignment of a complete team to a specific task (see queries Q3, Q4 in Table 2), or the assignment of certain agents to a specific task (see queries Q1, Q2, Q5 and Q6 in Table 2).

Following our running example 2.2, question "Why is student a_{15} assigned to the pathfinder project?" is an assignment query questioning the assignment of student a_{15} to a specific project, the pathfinder project. Question "Why is team $K = \{a_1, a_{10}, a_{15}, a_{20}\}$ not assigned to the SUDOKU project?" is also an assignment query questioning the assignment of the complete team K to a specific project, and therefore the assignment of each student in K to that project. Alternatively, question "Why is student a_3 in team $K = \{a_3, a_8, a_{13}, a_{18}\}$?" is a collaboration query questioning the participation of student a_3 in team K, and therefore the collaboration of student a_3 with each of the students in team K.

After identifying these relevant team formation query templates, in the following section we show how to translate each of these queries into query constraints.

5 QUERY TRANSLATION

As mentioned in Section 3, we translate the queries posed by a questioner into constraints in order to compute an alternative allocation based on the query, and thereafter build a contrastive explanation. Complying with a query constraint is necessary to compute an alternative allocation corresponding to a 'what-if' scenario, an alternative scenario. Query constraints are *hard constraints*, and must be met when solving the new team formation problem that arises by imposing the query constraints. Looking at the query templates in Table 2, we observe that a (collaboration or assignment) query poses: *(i)* why a specific collaboration or assignment is established in a given allocation, *or (ii)* why a specific collaboration or assignment is not established. Such types of queries are then translated into one of two types of constraints:

- *veto constraints* that capture queries that question a established team or assignment, and
- *enforcement constraints* that capture queries regarding why a team or assignment did not occur.

Consider the question asked by the professor in our running example "Why is student a_{15} assigned to the pathfinder project?", which is an instance of query template Q1. To provide an explanation, we must find an allocation for the alternative what-if scenario: "What would happen if student a_{15} were not assigned to the pathfinder project?". By adding the appropriate constraint, we can compute alternative allocations suitable for answering the query posed by the professor. Thus, in general, when a query template targets at a team or an assignment established in the original allocation, it translates into veto constraints on the team or assignment. This is necessary to describe a meaningful what-if scenario, and therefore build meaningful contrastive explanations.

Alternatively, when a query template targets a team or assignment that did not occur in the original allocation, the query translates into enforcement constraints so that the query-compliant

Code	Query Template	Query Type	Query Constraints	Constraints per *query-compliant* Allocations	Number of *query-compliant* Allocations
Q1	Why is agent a_i assigned to task τ?	Assignment	VETO assigning a_i to τ	1	1
Q2	Why is agent a_i not assigned to task τ?	Assignment	ENFORCE assigning a_i to τ	1	1
Q3	Why is team $K = \{a_1, \cdots a_{\lvert K \rvert}\}$ assigned to task τ?	Assignment	VETO assigning a_1 in τ **OR** VETO assigning a_2 in τ **OR** ... VETO assigning $a_{\lvert K \rvert}$ in τ	1	|K|
Q4	Why is team $K = \{a_1, \cdots a_{\lvert K \rvert}\}$ not assigned to task τ?	Assignment	ENFORCE assigning a_1 in τ **AND** ENFORCE assigning a_2 in τ **AND** ... ENFORCE assigning $a_{\lvert K \rvert}$ in τ	1	|K|
Q5	Why is agent a_i assigned to task τ, instead of a_j?	Assignment	VETO assigning a_i in τ **AND** ENFORCE assigning a_j in τ	2	1
Q6	Why is agent a_i assigned to task τ, while a_j is not?	Assignment	VETO assigning a_i in τ **OR** ENFORCE assigning a_j in τ	1	2
Q7	Why is agent a_i in team $K = \{a_1, \cdots, a_{\lvert K \rvert}\}$?	Collaboration	VETO collaborating a_i with a_1 **OR** VETO collaborating a_i with a_2 **OR** ... VETO collaborating a_i with $a_{\lvert K \rvert}$	1	|K|-1
Q8	Why is team $K = \{a_1, \cdots, a_{\lvert K \rvert}\}$ formed?	Collaboration	VETO collaborating a_1 with a_2 **OR** VETO collaborating a_1 with a_3 **OR** ... VETO collaborating a_1 with $a_{\lvert K \rvert}$ **OR** ... VETO collaborating $a_{\lvert K \rvert -1}$ with $a_{\lvert K \rvert}$	1	$\binom{\lvert K \rvert}{2}$
Q9	Why is team $K = \{a_1, \cdots, a_{\lvert K \rvert}\}$ not formed?	Collaboration	ENFORCE collaborating a_1 with a_2 **AND** ENFORCE collaborating a_1 with a_3 **AND** ... ENFORCE collaborating a_1 with $a_{\lvert K \rvert}$ **AND** ... ENFORCE collaborating $a_{\lvert K \rvert -1}$ with $a_{\lvert K \rvert}$	1	$\binom{\lvert K \rvert}{2}$
Q10	Why are a_i and a_j in the same team?	Collaboration	VETO collaborating a_i with a_j	1	1
Q11	Why are a_i and a_j not in the same team?	Collaboration	ENFORCE collaborating a_i with a_j	1	1
Q12	Why is agent a_i in team $K = \{a_1, \cdots, a_{\lvert K \rvert}\}$, instead of agent a_x?	Collaboration	ENFORCE collaborating a_x with a_1 **AND** ... ENFORCE collaborating a_x with a_{i-1} **AND** ENFORCE collaborating a_x with a_{i+1} **AND** ... ENFORCE collaborating a_x with $a_{\lvert K \rvert}$ **AND** VETO collaborating a_x with a_i **AND** VETO collaborating a_x with a_y, $\forall a_y \notin K$	|A|	1
Q13	Why is agent a_i in team $K = \{a_1, \cdots, a_{\lvert K \rvert}\}$, instead of agent a_x?	Collaboration	ENFORCE collaborating a_x with a_1 **AND** ... ENFORCE collaborating a_x with a_{i-1} **AND** ENFORCE collaborating a_x with a_{i+1} **AND** ... ENFORCE collaborating a_x with $a_{\lvert K \rvert}$ **OR**	|K|-1	2
			VETO collaborating a_x with a_i **AND** VETO collaborating a_x with a_y, $\forall a_y \notin K$	|A|-|K|-1	

Table 2: Query templates for a team formation problem.

allocation contains the team or the assignment described in the query. For instance, consider the question "Why is team $K = [\{a_1, a_3, a_5, a_7\}$ not formed", as an instance of query template Q9. The what-if scenario to consider would be: "What would happen if team K was formed?". Hence, replying to this query demands to compute an allocation that enforces the formation of team K.

Depending on the query template, and more precisely on how many agents are involved in a query, the queries are translated into *(a)* a single constraint, or *(b)* a set of constraints. Queries translated into a single constraint are referred to as *simple queries*, and involve a pair ⟨agent, task⟩ (Q1,Q2) or two agents (Q10,Q11). Regarding the queries translated into a set of constraints, we refer to them as *complex queries*, and we differentiate three translation patterns for them: *(i)* conjunction of constraints (Q4, Q5, Q9, Q12), which deal with a team formation or assignment that did not occur; *(ii)* disjunction of constraints (Q3, Q6, Q7, Q8), which focus on an established team formation or assignment; and *(iii)* disjunction of conjunctions (Q13), which consider both an established team and one that did not occur.

6 PROBLEM TRANSFORMATION

In this section we show how to transform an original TFA by wrapping its oracle function u to obtain TFA($\tilde{u}$). The problem transformation process is based on defining the wrapper function $\tilde{u}$ and thus change the evaluation function that determines the quality of the assignment of a team to a task, depending on the query constraints. In more detail, given a query q, we incorporate its query-constraints C_q in the oracle function by imposing large penalties when a team-task pair violates a query constraint. What a penalty is depends on the function that aggregates the quality of each team-task allocation. For instance, if the aggregation is the product of the individual qualities then the penalty would be zero, when the query constraint.

If the aggregation is based on an addition of qualities then the penalty would be a large negative value. In the equations below we are assuming we have a product aggregation as an example. In other words, the new, query-compliant problem p' is given by $\langle A, T, F, u, C, \tilde{u}, C_q \rangle$ where

$$\tilde{u}(K, \tau, F, C_q) = \psi(K, \tau, C_q) \cdot u(K, \tau, F) \tag{2}$$

and

$$\psi(K, \tau, C_q) = \begin{cases} 0, & \text{if } \langle K, \tau \rangle \text{ violates a constraint in } C_q \\ 1, & \text{otherwise} \end{cases} \tag{3}$$

Thus, when TFA($\tilde{u}$) is solving problem p', allocations that violate a query constraint are penalised and therefore avoided, since the allocation's quality depends on the quality of all teams for their assigned tasks—i.e., $v(g) = \mathcal{F}_{\tau \in T}\, \tilde{u}\Big(g(\tau), \tau, F\Big)$. Hence, an allocation ($g'$) that assigns team K to task τ (i.e., $g'(\tau) = K$) that violates some query constraint is penalised, since $\tilde{u}(K, \tau, F, C_q) =$ "*Large Penalty*". The wrapper $\tilde{u}$ simply needs to *(i)* check whether the constraints are violated for each of the pairs $\langle g(\tau), \tau \rangle$, and then *(ii)* multiply its suitability $u(g(\tau), \tau, F)$ by a large penalty if there is a violation, or by 1 otherwise.

Now we analyse how to check whether an assignment violates a constraint. There are two types of constraints: assignment and collaboration constraints, which result from translating assignment and collaboration queries respectively. On the one hand, let $c_a = \langle t, a_i, \tau_j \rangle$ represent an assignment constraint of type t (veto or enforcement), over an agent a_i and a task τ_j. We say that an assignment $\langle K, \tau \rangle$ violates c_a if:

(1) $t =$ veto **and** $a_i \in K$ **and** $\tau_j = \tau$, **or**
(2) $t =$ enforce **and**
  (a) $a_i \in K$ **and** $\tau_j \neq \tau$ **or**
  (b) $a_i \notin K$ **and** $\tau_j = \tau$

On the other hand, let $c_c = \langle t', a_k, a_l \rangle$ represent a collaboration constraint of type t' over agents a_k and a_l. We say that an assignment $\langle K, \tau \rangle$ violates c_c if:

(1) $t' =$ veto **and** $\{a_k, a_l\} \subseteq K$, **or**
(2) $t' =$ enforce **and**
  (a) $a_k \in K$ **and** $a_l \notin K$, **or**
  (b) $a_k \notin K$ **and** $a_l \in K$

Therefore, notice that checking the violation of constraints in C_q is not expensive, since it involves verifying the membership of elements to sets and checking for set inclusion.

Computational cost per query template. Depending on the query template used by the questioner the number of 'what-if' team formation scenarios to consider varies. For example, consider the question "Why is student a_{15} assigned to the pathfinder project?" (instance of query template Q1). There is only one 'what-if' scenario deriving from the question that we should consider to build an explanation: "What if student a_{15} was not assigned to the pathfinder project?". However, consider the question "Why is student a_3 assigned to the chess project, while student a_{14} is not?" (instance of query template Q6). In this case, we should consider two different 'what-if' scenarios to build an explanation: "What if student a_3 was not assigned to the chess project?", and "What if student a_{14} was assigned to the chess project?".

In general, each 'what-if' scenario requires the computation of a new allocation, namely the running of TFA($\tilde{u}$). As such, the number of 'what-if' scenarios that a query template requires to consider determines the number of times that our justification algorithm needs to translate the original TFP-TA p into a QTFP-TA p', and hence the number of *runs* of the TFA($\tilde{u}$) (one per query-compliant problem). The last column in Table 2 shows the number of query-compliant allocations (computed by different runs of the TFA($\tilde{u}$)) that are required to build a contrastive explanation per query template. We differentiate query templates that lead to: (i) one what-if scenario (Q1, Q2, Q5, Q10, Q11 and Q12); (ii) two what-if scenarios (Q6,Q13); and (iii) a number of what-if scenarios that is either linear or quadratic to the number of agents appearing in the query (queries Q3, Q4, Q7 and Q8, Q9 respectively). Note that complex queries translated into a disjunction of constraints lead to multiple what-if scenarios. Therefore, such queries require the computation of more than query-compliant allocations. However, with the exception of query template Q13, each one of the different query-compliant allocations of these queries need to respect a single constraint.

	Attribute function f_i
Individual View	$\Delta f_i^{IV} = \frac{f_i(g(\tau),\tau)\vert_{a \in g(\tau)} - f_i(g'(\tau'),\tau')\vert_{a \in g'(\tau')}}{f_i(g(\tau),\tau)\vert_{a \in g(\tau)}}$
Local View	$\Delta f_i^{LV} = \frac{f_i(g(\tau),\tau) - f_i(g'(\tau),\tau)}{f_i(g'(\tau),\tau)},\ \forall \tau \in T$
Global View	$\Delta f_i^{GV} = \frac{\mathcal{F}_{\tau \in T}\, f_i(g(\tau),\tau) - \mathcal{F}_{\tau \in T}\, f_i(g'(\tau),\tau)}{\mathcal{F}_{\tau \in T}\, f_i(g(\tau),\tau)}$

Table 3: Relative Differences per Explanation View

7 BUILDING CONTRASTIVE EXPLANATIONS

This section describes how to build and tailor contrastive explanations to answer a question about team formation.

As outlined in Section 3, a contrastive explanation compares some original allocation (g) with a query-compliant one (g'), the latter one resulting from some 'what-if' scenario posed by the query. The explanation compiles the differences that make the original allocation preferable to the query-compliant one. The comparison between the original and the query-compliant allocation is done along three levels of abstraction (individual, local, and global), which we call *explanation views.* Each explanation view corresponds to a different point of view that we can adopt to justify the reasons why the original allocation is better than the query-compliant one: the *individual view (IV)* focuses on the agents referred to within the query; the *local view (LV)* focuses on the individual tasks and their assigned teams; and the *global view (GV)* focuses on the overall quality of the allocations.

For each view we compute the *relative differences* between the original allocation and the query-compliant one, as shown in Table 3. Such differences serve to quantify the gains or losses of one allocation with respect to the other. That is, we compute $\Delta f_i^{IV}, \Delta f_i^{LV}, \Delta f_i^{GV}$ for each attribute function $f_i \in F$. Moreover, we compute relative differences for oracle u in IV and LV, and aggregation function $\mathcal{F}$ in GV. Notice that in this step we need access on *(a)* the oracle function u, and (b) the aggregating function $\mathcal{F}$. However, the oracle function is accessible by our justification algorithm (through the wrapper $\tilde{u}$); while the aggregating function is known for a given TFA (again, our wrapper must be aware of $\mathcal{F}$

in order to impose the proper penalty). Moreover, in order to compute the relative differences for the individual view, we make the following assumption: there is a way to compute the contribution of an agent a to its team $g(\tau)$ (denoted as $|_{a \in g(\tau)}$) with respect to: *(i)* each attribute function f_i, and *(ii)* the value yielded by the oracle u. Even though this may seem a strong assumption, it is common in the literature to employ oracles that allow such computation (e.g. [6, 15]). In our empirical evaluation, we illustrate the use of one general oracle that allows to compute relative differences as defined in Table 3.

Back to our example, consider the question "Why is student a_{15} assigned to the pathfinder project?", along with the allocations g and g' in Table 1. The explanation builder would generate an explanation containing one explanation per view as follows: "If student a_{15} was not assigned to the pathfinder project then...
(IV) "Student a_{15} would have had to take part in the energy project, for which they are less skilled."
(LV) "40% of the tasks would have been assigned to dramatically less-skilled (up to 98.9%) teams and 20% of the tasks would have been assigned to less-compatible (up to 60.1%) teams, while 40% of the tasks would have been assigned to equally skilled, compatible and satisfied teams."
(GV) "The overall matching of teams to tasks would be 99.98% less-skilled, 53.62% less-diverse in terms of personality, and 7.37% less-satisfying. Thus, the alternative allocation would be 36.08% less-suitable considering all attributes".

The explanation example above is built by first computing the relative differences, and then filling out an *explanation template* in natural language. In general, the explanation template is common to all queries—changes only for the IV—and, due to space limitations, can be found in the supplementary material.

8 EVALUATING EXPLANATIONS

This section evaluates the quality of the explanations that our algorithm generates. First, Section 8.1 specifies the instance of the TFP-TA selected for our experiments, along with the TFA of choice. Then, Section 8.2 introduces our evaluation metrics for explanations, some of them adapted from metrics used in the ML literature. Finally, Section 8.3 reports the results of our evaluation.

8.1 Team formation problem and algorithm

We choose the team formation problem of forming non-overlapping teams that tackle one task each, and in turn each task is tackled by only one team. This is a relevant team formation problem in the MAS literature appearing, for instance, in [15, 17, 31]). We choose the same attributes as described in our running example: skills, personality, and agents' preferences over tasks ($\{f_{\text{skills}}, f_{\text{pers}}, f_{\text{sat}}\}$). Each task specifies the skills required by a team to perform it. Moreover, here we adopt the typical team-size constraint per task (e.g. [6, 17]). When assigning a team to a task, and similarly to the competence assignment in [6], we must also decide the required skills covered by each team member. As oracle u that yields suitability, we define: $u(K, \tau, F) = \sum_{f_i \in F} w_i \cdot f_i(K, \tau)$. Note that using a linear combination of multiple attributes to form a scalar function is a commonly used technique [22]. As aggregating function $\mathcal{F}$ we use the product ($\mathcal{F} \triangleq \prod$), since, as claimed by [6, 7], a product promotes "balanced" team allocations —according to [13], it favours both increasing the overall team allocation utility and reducing the differences in individual team allocation quality.

We encode our TFP-TA as a linear program (LP) to subsequently use an LP solver as team formation algorithm, similarly to [6, 18]. Next we describe such LP encoding. Let $\mathcal{K} = \{K_1, K_2, \cdots, K_\Xi\} \equiv 2^{|A|} \setminus \{\emptyset\}$ be a set containing all possible teams that can be formed in A. We use a decision variable $x_\tau^\xi \in \{0, 1\}$ that indicates whether team $K_\xi \in \mathcal{K}$ is assigned to task $\tau \in T$ or not. Moreover let d be an $|A| \times |\mathcal{K}|$ matrix, where $d_{i,\xi}$ is equal to 1 if agent a_i is part of team K_ξ (i.e., $a_i \in K_\xi$), and $d_{i,\xi} = 0$, otherwise. The objective function we use is the function $u(K, \tau, F)$ when we solve the original TFP-TA p, and the wrapper function $\tilde{u}(K, \tau, F, C_q)$ when we solve a QTFP-TA p'. Thus the LPs are:

$$\max \sum_{\substack{K_\xi, \tau \text{ s.t.} \\ g(\tau) \equiv K_\xi}} x_\tau^\xi \cdot \log u(K_\xi, \tau, F) \tag{4}$$

and

$$\max \sum_{\substack{K_\xi, \tau \text{ s.t.} \\ g(\tau) \equiv K_\xi}} x_\tau^\xi \cdot \log \tilde{u}(K_\xi, \tau, F, C_q) \tag{5}$$

subject to *(i)*$\sum_{K_\xi \in \mathcal{K} \cap \mathcal{K}_\tau} x_\tau^\xi = 1 \, \forall \tau \in T$, and *(ii)* $\sum_{\tau \in T} \sum_{K_\xi \in \mathcal{K} \cap \mathcal{K}_\tau} x_\tau^\xi \cdot d_{i,\xi} = 1 \, \forall a_i \in A$, where $\mathcal{K}_\tau \subseteq \mathcal{K}$ is the set of size-compliant teams for task τ. Note that equations *(i)* and *(ii)* encode the constraints C of the initial problem (i.e. each task requires one team of a specific size, each agent belongs to exactly one team, each team tackles exactly one task, and each task is assigned to exactly one team).

8.2 Evaluation Metrics

To evaluate the quality of explanations we have singled out a number of *offline* evaluation metrics, aligned with the existing literature.
Number of attributes (NOA). This is a commonly-used metric in many explainable models. Indeed, since most XAI models attempt to give insights on the functionality and the rationale of a black box (e.g., ML models), it is common to use the number of features used to justify a decision as a quality index for explanations [32, 37]. This is the case even for non-ML explanatory systems. For instance, when justifying elections winners in [10], the number of axioms that backs up the elected winner is used to evaluate the quality of an explanation. The number of attributes can be also seen as the number of causes [27] displayed in an explanation, with the general guideline that good explanations are the simple ones containing a relatively small number of causes. Here, we consider the number of attributes that exhibit a utility decrease in the query-compliant allocation compared to the original allocation.
Mean explainability precision (MEP) [1, 28]. Explainability precision—resembling the corresponding metric in information retrieval, classification, and ML in general— is the proportion of explainable items in a list relative to the total number of items. Regarding team formation, we measure MEP in terms of the percentage of agents, tasks, and attributes for which we can justify why the alternative query-compliant allocation is worse than the initial allocation.
Gunning Fog readability (GFR) index [20]. Since we produce explanations in natural language, we also use the *Gunning Fog readability (GFR)* index [20], a well-known readability metric in

Query	Average Number of Attributes | Standard Dev.			Mean Explainable Precision			Gunning-Fog Reading Index		
	Individual View	Local View	Global View	Individual View	Local View	Global View	Individual View	Local View	Global View
Q1	1 | 0	2.2 | 0.8	2.7 | 0.55	65%	40%	67.5%	8.37	8.78	10.61
Q2	1 | 0	2.20 | 0.74	3.00 | 0.71	70%	42.5%	73.75%	9.1	8.77	10.84
Q7	0.91 | 0.20	1.88 | 0.51	2.82 | 0.55	55%	37.17%	70.42%	7.47	8.64	10.63
Q8	0.91 | 0.22	1.86 | 0.51	2.84 | 0.53	55.41%	36.58%	71.04%	7.25	8.51	10.45
Q10	1.06 | 0.23	1.95 | 0.59	2.80 | 0.68	50%	38.5%	70%	8.4	8.76	10.88
Q11	1.18 | 0.58	2.45 | 0.86	2.90 | 0.62	35%	49.5%	72.5%	9.34	8.99	11.8
Q3	0.91 | 0.14	1.87 | 0.47	2.75 | 0.50	91.25%	37.75%	68.75%	7.08	8.53	10.33
Q4	1.55 | 0.49	2.8 | 0.4	3.35 | 0.57	75%	80%	83.75%	12.4	9.05	11.53
Q5	1.29 | 0.45	2.85 | 0.65	3.05 | 0.67	47.5%	54.5%	76.25%	9.89	8.87	11.42
Q6	0.58 | 0.18	2.58 | 0.48	3.03 | 0.43	35%	41.25%	75.63%	8.3	8.76	10.9
Q9	1.41 | 0.49	3.35 | 0.47	3.2 | 0.5	61.67%	73.5%	80%	10.2	8.91	10.79
Q12	1.60 | 0.49	2.20 | 0.92	3.30 | 0.55	17.49%	39.5%	82.5%	14.62	8.75	12.32
Q13	1.18 | 0.39	2.73 | 0.51	3.20 | 0.66	48.87%	47.5%	80.62%	9.82	8.78	11.54

Table 4: Average number of attributes, mean explainable precision, and average Gunning Fog reading index used per explanation per query template.

the literature. The GFR considers: *(a)* the proportion of "complex words" relative to the total number of words; and *(b)* the number of words per sentence. The resulting score indicates the reading level by grade needed to comprehend the text.

8.3 Results

Our empirical evaluation employs 20 synthetic instances of the TFP-TA in Section 8.1 with 10 tasks and $\sim$ 26.5 agents each. In supplementary material we fully detail our datasets. Note that we can easily compute relative differences for IV, since the oracle u in Section 8.1 allows to compute the contribution to a team per agent at low computational cost. We solved each one of the problem instances with an LP solver to obtain their *original* allocations.

After that, we generated one query for each query template and for each original allocation so that each allocation was artificially *challenged* by one query of each one of the 13 types. We generated queries depending on the query template as follows. First, we selected an original allocation. Second, we randomly selected a team (K) in the allocation. Third, we built one query per query template by randomly selecting: agents from K (for templates Q1, Q7, Q10); agents from $A \setminus K$ (templates Q2, Q10); one agent from K and one from $A \setminus K$ (templates Q5, Q9, Q11, Q12, Q13).

Finally, we handed the queries to our algorithm to compute explanations. We evaluated their quality using the metrics in Section 8.2. Table 4 compiles our results per metric grouped per query template, which we discuss below comparing explanation views.

Average number of attributes. The first three columns in Table 4 illustrate the average number of attributes used to justify why the original allocation is preferable to the query-compliant one. That is, these are the average number of attributes per explanation that: *(a)* exhibit a relative gain, and *(b)* are part of the textual explanation. The results tell us that the individual view uses less attributes than the local view, which in turn uses less attributes than the global view when building an explanation for each query. Moreover, explanations of simple queries (Q1,Q2, Q7,Q8, Q10 or Q11) use less attributes compared to more complex ones.

Mean explainable precision. According to Table 4, the GV reaches higher MEP (at least 67%) than the LV and the IV, i.e., using the GV we can easily justify why the original allocation is preferred to the query-compliant one. This is because the TFA's goal is to optimise the overall allocation function; and hence, the GV is in line with the algorithm's point of view. The LV reaches low MEP (below 50%), since there are many tasks that are not being affected at all by the query, and therefore cannot justify why one allocation is preferred to the other. Finally, the IV exhibits a wide variability in MEP, since the IV is highly query-dependent. Thus, the GV is more precise for justifying team formation.

Gunning Fog readability index. In Table 4, we observe that 92% (36 out of 39) of our explanations achieve scores between 8 (reading level of eighth grade) and 12 (reading level of a high school senior student), and 6% are very close to 12. Thus, we conclude that our explanations are easy to read and comprehend. In general we see that simple queries achieve a low GFR index ($\sim$ 8.36), while complex queries achieve higher GFR index ($\sim$ 10.33). Notably, the explanations for both the LV and GV exhibit a 'stable' reading index (the LV index is always $\sim$ 8.8, the GV index is $\sim$ 11). This is because, regardless of the query template, the LV always considers the same number of tasks (10 in all problem instances), and the GV considers a constant number of attributes. On the contrary the readability score for the IV is query-dependent.

9 CONCLUSIONS & FUTURE WORK

In this work we turn to explainable AI within multiagent systems, and specifically to explanations regarding team formation. Here we introduced a generic procedure for providing justifications based on contrastive explanations, that wraps existing team formation algorithms, with *no modifications needed.* We build on the notion of *facts and foils* [27], and justify the teams formed by exploring 'what-if' scenarios. We described the steps we need to follow in order to *(i)* express a 'what-if' scenario, *(ii)* reach a formation of teams respecting this scenario; and generate an explanation that discredits the 'what-if' scenario compared to the original teams formed. Moreover, we identified a number of intuitive query templates that widely cover questions regarding team formation. Finally, we conducted a preliminary, empirical evaluation on the quality of our generated explanations over synthetic data. Our results showed that our explanations are easy to read, they are simple by using a small number of causes (attributes), and they exhibit an acceptable mean explainable precision. As future work, regarding our tailoring process, we intend to lift the assumption that we can easily compute relative differences in the IV, by using e.g., the Labour Union game payoffs [19] or the Shapley values[36]. Moreover, since our preliminary results are promising, we should proceed towards evaluating our explanations by humans in a real-world scenario.

REFERENCES

[1] Behnoush Abdollahi and Olfa Nasraoui. 2017. Using Explainability for Constrained Matrix Factorization. In *Proceedings of the Eleventh ACM Conference on Recommender Systems* (Como, Italy) *(RecSys '17)*. Association for Computing Machinery, New York, NY, USA, 79–83. https://doi.org/10.1145/3109859.3109913

[2] Amina Adadi and Mohammed Berrada. 2018. Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access* 6 (2018), 52138–52160. https://doi.org/10.1109/ACCESS.2018.2870052

[3] Shaull Almagor and Morteza Lahijanian. 2020. Explainable Multi Agent Path Finding *(AAMAS '20)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 34–42.

[4] Aris Anagnostopoulos, Luca Becchetti, Carlos Castillo, Aristides Gionis, and Stefano Leonardi. 2010. Power in Unity: Forming Teams in Large-Scale Community Systems. *International Conference on Information and Knowledge Management, Proceedings*, 599–608. https://doi.org/10.1145/1871437.1871515

[5] Aris Anagnostopoulos, Luca Becchetti, Carlos Castillo, Aristides Gionis, and Stefano Leonardi. 2012. Online Team Formation in Social Networks. In *Proceedings of the 21st International Conference on World Wide Web* (Lyon, France) *(WWW '12)*. Association for Computing Machinery, New York, NY, USA, 839–848. https://doi.org/10.1145/2187836.2187950

[6] Ewa Andrejczuk, Rita Berger, Juan A. Rodríguez-Aguilar, Carles Sierra, and Víctor Marín-Puchades. 2018. The composition and formation of effective teams: computer science meets organizational psychology. *Knowledge Eng. Review* 33 (2018), e17. https://doi.org/10.1017/S026988891800019X

[7] Ewa Andrejczuk, Filippo Bistaffa, Christian Blum, Juan A. Rodríguez-Aguilar, and Carles Sierra. 2019. Synergistic team composition: A computational approach to foster diversity in teams. *Knowledge-Based Systems* 182, 104799 (10/2019 2019). https://doi.org/10.1016/j.knosys.2019.06.007

[8] Yoram Bachrach, Reshef Meir, Kyomin Jung, and Pushmeet Kohli. 2010. Coalitional Structure Generation in Skill Games. *24th AAAI Conference on Arti-cial Intelligence (AAAI)* 2.

[9] P. Ballesteros-Pérez, Ma. C. González-Cruz, and M. Fernández-Diego. 2012. Human resource allocation management in multiple projects using sociometric techniques. *International Journal of Project Management* 30, 8 (2012), 901–913.

[10] Arthur Boixel and Ulle Endriss. 2020. Automated Justification of Collective Decisions via Constraint Solving. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems* (Auckland, New Zealand) *(AAMAS '20)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 168–176.

[11] Luca Capezzuto, Danesh Tarapore, and Sarvapali D. Ramchurn. 2020. Anytime and Efficient Coalition Formation with Spatial and Temporal Constraints. arXiv:2003.13806 [cs.MA]

[12] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. 2019. Machine Learning Interpretability: A Survey on Methods and Metrics. *Electronics* 8, 8 (2019). https://doi.org/10.3390/electronics8080832

[13] Yann Chevaleyre, Paul E Dunne, Ulle Endriss, Jerome Lang, Michel Lemaitre, Nicolas Maudet, Julian Padget, Steve Phelps, Juan A Rodrguez-Aguilar, and Paulo Sousa. 2006. Issues in Multiagent Resource Allocation. *Informatica* 30 (2006), 3–31.

[14] Chad Crawford, Zenefa Rahaman, and Sandip Sen. 2016. Evaluating the Efficiency of Robust Team Formation Algorithms. In *Autonomous Agents and Multiagent Systems*, Nardine Osman and Carles Sierra (Eds.). Springer International Publishing, Cham, 14–29.

[15] E. Czatnecki and A. Dutta. 2019. Hedonic Coalition Formation for Task Allocation with Heterogeneous Robots. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*. 1024–1029.

[16] Filip Karlo Došilović, Mario Brčić, and Nikica Hlupić. 2018. Explainable artificial intelligence: A survey. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. 0210–0215. https://doi.org/10.23919/MIPRO.2018.8400040

[17] Athina Georgara, Juan A. Rodríguez-Aguilar, and Carles Sierra. 2021. *Towards a Competence-Based Approach to Allocate Teams to Tasks.* International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1504–1506.

[18] Athina Georgara, Carles Sierra, and Juan A. Rodríguez-Aguilar. 2020. TAIP: an anytime algorithm for allocating student teams to internship programs. arXiv:2005.09331 [cs.AI]

[19] Sreenivas Gollapudi, Kostas Kollias, Debmalya Panigrahi, and Venetia Pliatsika. 2017. Profit Sharing and Efficiency in Utility Games. In *25th Annual European Symposium on Algorithms (ESA 2017) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 87)*, Kirk Pruhs and Christian Sohler (Eds.). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 43:1–43:14. https://doi.org/10.4230/LIPIcs.ESA.2017.43

[20] R. Gunning. 1952. *The Technique of Clear Writing.* McGraw-Hill. https://books.google.gr/books?id=ofI0AAAAMAAJ

[21] Jimmy H. Gutiérrez, César A. Astudillo, Pablo Ballesteros-Pérez, Daniel Mora-Melià, and Alfredo Candia-Véjar. 2016. The Multiple Team Formation Problem Using Sociometry. *Comput. Oper. Res.* 75, C (Nov. 2016), 150–162.

[22] J. Horn, N. Nafpliotis, and D.E. Goldberg. 1994. A niched Pareto genetic algorithm for multiobjective optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence.* 82–87 vol.1. https://doi.org/10.1109/ICEC.1994.350037

[23] Akiva Kleinerman, Ariel Rosenfeld, and Sarit Kraus. 2018. Providing Explanations for Recommendations in Reciprocal Environments. In *Proceedings of the 12th ACM Conference on Recommender Systems* (Vancouver, British Columbia, Canada) *(RecSys '18)*. Association for Computing Machinery, New York, NY, USA, 22–30. https://doi.org/10.1145/3240323.3240362

[24] Sarit Kraus, Amos Azaria, Jelena Fiosina, Maike Greve, Noam Hazon, Lutz Kolbe, Tim-Benjamin Lembcke, Jorg Muller, Sören Schleibaum, and Mark Vollrath. 2020. AI for Explaining Decisions in Multi-Agent Environments. *Proceedings of the AAAI Conference on Artificial Intelligence* 34 (04 2020), 13534–13538. https://doi.org/10.1609/aaai.v34i09.7077

[25] Theodoros Lappas, Kun Liu, and Evimaria Terzi. 2009. Finding a Team of Experts in Social Networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Paris, France) *(KDD '09)*. Association for Computing Machinery, New York, NY, USA, 467–476. https://doi.org/10.1145/1557019.1557074

[26] Zachary C. Lipton. 2018. The Mythos of Model Interpretability: In Machine Learning, the Concept of Interpretability is Both Important and Slippery. *Queue* 16, 3 (June 2018), 31–57. https://doi.org/10.1145/3236386.3241340

[27] Tim Miller. 2018. Explanation in Artificial Intelligence: Insights from the Social Sciences. arXiv:1706.07269 [cs.AI]

[28] Sina Mohseni, Niloofar Zarei, and Eric D. Ragan. 2020. A Multidisciplinary Survey and Framework for Design and Evaluation of Explainable AI Systems. arXiv:1811.11839 [cs.HC]

[29] Francesca Mosca and Jose M. Such. 2021. *ELVIRA: An Explainable Agent for Value and Utility-Driven Multiuser Privacy.* International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 916–924.

[30] Sameera S. Ponda, Luke B. Johnson, Alborz Geramifard, and Jonathan P. How. 2015. *Cooperative Mission Planning for Multi-UAV Teams.* Springer Netherlands, Dordrecht, 1447–1490. https://doi.org/10.1007/978-90-481-9707-1_16

[31] Fredrik Präntare and Fredrik Heintz. 2020. An anytime algorithm for optimal simultaneous coalition structure generation and assignment. *Autonomous Agents and Multi-Agent Systems* 34, 1 (03 Mar 2020), 29. https://doi.org/10.1007/s10458-020-09450-1

[32] Avi Rosenfeld. 2021. *Better Metrics for Evaluating Explainable Artificial Intelligence.* International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 45–50.

[33] Avi Rosenfeld and Ariella Richardson. 2019. Explainability in human–agent systems. *Autonomous Agents and Multi-Agent Systems* 33, 6 (01 Nov 2019), 673–705. https://doi.org/10.1007/s10458-019-09408-y

[34] Wojciech Samek and Klaus-Robert Müller. 2019. *Towards Explainable Artificial Intelligence.* Springer International Publishing, Cham, 5–22. https://doi.org/10.1007/978-3-030-28954-6_1

[35] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. 2017. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296* (2017).

[36] L.S. Shapley. 1953. A Value for n-Person Games. In *Contributions to the Theory of Games II*, H. Kuhn and A.W. Tucker (Eds.). Princeton University Press, Princeton, 307–317.

[37] Yongfeng Zhang and Xu Chen. 2020. Explainable Recommendation: A Survey and New Perspectives. *Foundations and Trends® in Information Retrieval* 14, 1 (2020), 1–101. https://doi.org/10.1561/1500000066