Autonomous Agents and Multi-Agent Systems (2020) 34:10
https://doi.org/10.1007/s10458-019-09433-x

Multi-objective multi-agent decision making: a utility-based analysis and survey

Roxana Rădulescu[1] · Patrick Mannion[2] · Diederik M. Roijers[1,3,4] · Ann Nowé[1]

Published online: 9 December 2019

Abstract
The majority of multi-agent system implementations aim to optimise agents' policies with respect to a single objective, despite the fact that many real-world problem domains are inherently multi-objective in nature. Multi-objective multi-agent systems (MOMAS) explicitly consider the possible trade-offs between conflicting objective functions. We argue that, in MOMAS, such compromises should be analysed on the basis of the utility that these compromises have for the users of a system. As is standard in multi-objective optimisation, we model the user utility using utility functions that map value or return vectors to scalar values. This approach naturally leads to two different optimisation criteria: expected scalarised returns (ESR) and scalarised expected returns (SER). We develop a new taxonomy which classifies multi-objective multi-agent decision making settings, on the basis of the reward structures, and which and how utility functions are applied. This allows us to offer a structured view of the field, to clearly delineate the current state-of-the-art in multi-objective multi-agent decision making approaches and to identify promising directions for future research. Starting from the execution phase, in which the selected policies are applied and the utility for the users is attained, we analyse which solution concepts apply to the different settings in our taxonomy. Furthermore, we define and discuss these solution concepts under both ESR and SER optimisation criteria. We conclude with a summary of our main findings and a discussion of many promising future research directions in multi-objective multi-agent systems.

✉ Roxana Rădulescu
roxana.radulescu@vub.be

Patrick Mannion
patrick.mannion@nuigalway.ie

Diederik M. Roijers
diederik.yamamoto-roijers@hu.nl

Ann Nowé
ann.nowe@vub.be

1 Artificial Intelligence Lab, Vrije Universiteit Brussel, Brussel, Belgium

2 School of Computer Science, National University of Ireland Galway, Galway, Ireland

3 Institute of ICT, HU University of Applied Sciences Utrecht, Utrecht, The Netherlands

4 Computational Intelligence, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands

Keywords Multi-agent systems · Multi-objective decision making · Multi-objective optimisation criteria · Solution concepts · Reinforcement learning

1 Introduction

A multi-agent system (MAS) features multiple agents deployed into a common environment. This is an inherently distributed paradigm, which benefits from scalability (agents can be added as required) and fault tolerance (the failure of any one agent does not imply failure of the whole system). The agents within a MAS may act cooperatively, competitively, or may exhibit a mixture of these behaviours [173,183]. Recent works have surveyed multi-agent decision making from a variety of different perspectives, including: the evolutionary dynamics of multi-agent learning [20], non-stationarity in multi-agent environments [66], modelling other agents in MAS [3], and deep reinforcement learning in MAS [67,109]. In this survey, we focus exclusively on multi-objective approaches to decision making in MAS, a perspective that has not yet been addressed in preceding works.

The majority of MAS implementations aim to optimise agent's policies with respect to a single objective, despite the fact that many real world problems are inherently multi-objective in nature. Single-objective approaches seek to find a single policy to a problem, whereas in reality a system may have multiple possibly conflicting objectives. Multi-objective optimisation (MOO) [40] approaches consider these possibly conflicting objectives explicitly. In multi-objective multi-agent systems (MOMAS) the reward signal for each agent is a vector, where each component represents the performance on a different objective. By taking a multi-objective perspective on decision making problems, complex trade-offs can be managed; e.g., when selecting energy sources for electricity generation, there is an inherent trade-off between using cheap sources of energy which damage the environment, versus using renewable energy sources which are more expensive but better for the environment [91]. Such trade-offs appear in a wide range of domains such as urban transportation [12,38], aviation [58,190], management of natural resources [43,100] and robotics [32,119]; these are all domains where multi-objective multi-agent approaches could confer huge benefits.

Compromises between competing objectives should be made on the basis of the utility that these compromises have for the users. In other words, if we can define a utility function that maps the vector value of a compromise solution to a scalar utility—called a *utility* or *scalarisation* function—then we can derive what to optimise [133], and how to measure the quality of solutions [193]. In some rare cases, we might even be able to apply the utility function a priori, and try to solve the decision problem as a single-objective problem. However, as it is known from single-agent multi-objective decision making [132], it is often impossible, undesirable, or unfeasible to perform such a priori scalarisation. For example, if the utility function is non-linear, this typically renders a priori scalarisation and subsequent single-objective solution methods intractable. Moreover, while trying to find compromise policies, i.e., while the agents are planning or learning, the utility function is often unknown or uncertain. In such cases, it is often desirable to construct a so-called *coverage set*, a set of solutions that has at least one optimal policy with respect to every possible utility function that a user might have.

The utility-based approach naturally leads to two different optimisation criteria for agents in a MOMAS: expected scalarised returns (ESR) and scalarised expected returns (SER). In the former, the users derive their utility from single roll-outs of the policy, while in the latter, the utility is derived from the expected outcomes, i.e., the mean over multiple roll-outs. To

date, the differences between the SER and ESR approaches have received little attention in multi-agent settings, despite having received some attention in single-agent settings (see for e.g. [131,132]). Consequently, the implications of choosing either ESR or SER as the optimisation criterion for a MOMAS are currently not well-understood.

In single-agent or fully cooperative multi-agent settings [133], it is typically assumed that there is one, possibly unknown, utility function that determines the possibly unknown preferences of the users, and that the users are interested in the utility of the expected vector-valued returns (i.e., the SER optimality criterion). In the execution phase all agents will ultimately pursue the best utility with respect to this single function. Therefore, the coverage set can be derived from everything that is known about this utility function, and the types of policies allowed. For example, for deterministic stationary policies and possibly non-linear utility functions, a coverage set is a so-called Pareto front of deterministic stationary policies. That is the set of policies that are not Pareto-dominated, i.e., for which there is no other deterministic stationary policy that has a better or equal value for all objectives and is better in at least one objective. A Pareto-undominated policy is also called Pareto-optimal. Another well-known coverage set is a *convex coverage set (CCS)*, which is a coverage set with respect to all possible linear utility functions. Incidentally, in single agent or fully cooperative settings a Pareto-coverage set for stochastic policies can be constructed from a CCS of deterministic stationary policies [132,168]. We discuss related work on single-objective and fully cooperative multi-agent systems in Sect. 2.2.

In multi-agent settings however, the situation can become much more complex. While in fully cooperative multi-agent systems, the agents are assumed to all receive the same team rewards, the individual reward vectors received by agents may be different in general multi-agent settings. We review various models with different assumptions regarding the reward functions, as well as observability and statefulness (i.e., whether or not the model involves a sequential decision making problem with multiple states), in Sect. 3. Then, we consider settings where individual agents value objectives according to their own preferences, i.e., where each agent can have their own utility function (even if the reward vector for the agents is the same, as for e.g., in Example 2 in Sect. 4). This has important consequences for what constitutes a solution set. We build a taxonomy of what constitutes a solution for a multi-objective multi-agent decision problem based on modelling assumptions, utility functions, and optimisation criteria, by analysing what happens at execution time in Sect. 4. We note that many of the different settings we identify in Sect. 4 are under-explored in the current literature, and offer examples of decision problems that would merit further investigation for each part of our taxonomy.

Using our taxonomy, we review the literature on multi-objective multi-agent decision problems in terms of optimal solution sets (Sect. 5), and solution methods (Sect. 6). Finally, we discuss what we consider to be the key open problems in this new and exciting field (Sect. 7).

1.1 Survey methodology

To identify potentially relevant papers for this survey we primarily used search engines (e.g. Google Scholar), as well as less formal methods such as climbing up citation trees, alerts from indexing services and social media, and considering recommendations from colleagues. Of the initial set of papers, we included only works that are directly relevant to multi-objective multi-agent decision making (theory and/or applications).

Table 1 General formats for single and multi-objective normal form games

		Player Y: A	Player Y: B
Player X	A	$(x_{A,A}, y_{A,A})$	$(x_{A,B}, y_{A,B})$
	B	$(x_{B,A}, y_{B,A})$	$(x_{B,B}, y_{B,B})$

(a) **Single-Objective Normal Form Game** with **scalar payoffs** x and y

		Player Y: A	Player Y: B
Player X	A	$(\mathbf{x}_{A,A}, \mathbf{y}_{A,A})$	$(\mathbf{x}_{A,B}, \mathbf{y}_{A,B})$
	B	$(\mathbf{x}_{B,A}, \mathbf{y}_{B,A})$	$(\mathbf{x}_{B,B}, \mathbf{y}_{B,B})$

(b) **Multi-Objective Normal Form Game** with **vector payoffs** $\mathbf{x}$ and $\mathbf{y}$

A and *B* represent different actions which are available to the agents, and each agent *X* and *Y* receives a payoff depending on which combination of actions was selected

For search engines we used search strings such as `("multi-objective" OR "multi-criteria") AND ("multi-agent" OR "game")`. Unfortunately, these terms are often used for settings other than that which is the focus of this survey, so many of the results returned by such queries are not relevant. A large body of work on multi-objective meta-heuristic algorithms (e.g. genetic/evolutionary algorithms/particle swarm optimisation/ant colony optimisation etc.) uses the term "multi-agent" in a different sense, where each agent represents a centralised solution to an entire optimisation problem, rather than a distinct component of a system which can make decisions autonomously using its own policy (which is our focus in this survey).

1.2 Motivating example

As a motivating example for adopting a multi-objective perspective on multi-agent decision making, we introduce a Multi-Objective Normal Form Game (MONFG) which is called the Commuting MONFG (Example 1). By contrast to the usual Single-Objective Normal Form Game (SONFG) format, which is common in the literature, in a MONFG the agents receive payoffs in vector rather than scalar format after selecting their actions. This difference is illustrated in Table 1a and b.

Example 1 In the Commuting MONFG, two agents wish to commute from a common origin to the same destination. There are two transportation options available: travel by taxi or travel by train. If both agents choose the taxi option, they may split the cost equally between them. If they both choose to travel by train, they must each purchase their own ticket individually. If one chooses to travel by taxi and the other chooses to travel by train, they must also pay their own fares individually. A train ticket is cheaper than a taxi fare (even when agents share a taxi ride); however, the taxi journey takes less time than the train journey. The expected travel time and cost for each mode of transport is listed in Table 2. The individual or local vector payoffs for each agent are shown in Table 3. Note that the values in the matrix are negative as this is a minimisation problem for both objectives (commuters in general do not want to spend any additional time or money on their commute above what is necessary).

From an utility-based perspective, each commuter in Example 1 will try to balance these conflicting objectives such that his/her derived utility is maximised. Each commuter can of

Table 2 Cost and travel time for different modes of transport in the Commuting MONFG

	cost	time
taxi	20	10
train	5	30

Table 3 Individual (local) payoff matrix for the Commuting MONFG

		Player Y	
		taxi	train
Player X	taxi	$([-10,-10],[-10,-10])$	$([-20,-10],[-5,-30])$
	train	$([-5,-30],[-20,-10])$	$([-5,-30],[-5,-30])$

course have a different utility function depending on how each objective is valued. Furthermore, depending on when each commuter evaluates the commute (e.g., on a monthly basis or after each trip), the two different optimisation criteria come into play: ESR or SER. For example, for some commuters travelling costs should be maintained within a certain budget every month, while still being on time at least 75% of the time. This requires the use of the scalarised expected returns for the month. For other commuters the time component might be crucial and they cannot be late on any given day, thus imposing the expected scalarised return criterion.

2 Background

Before we address the specifics of multi-objective multi-agent systems, and how to define optimal solutions for the Commuting MONFG in the previous section, we first introduce relevant background work on multi-agent decision theory, multi-objective decision making, optimisation criteria and utility functions which is necessary to understand the material covered later in this article.

2.1 Multi-agent decision theory

Multi-agent systems appeared as a natural paradigm for modelling numerous real-world problems (e.g., health-care [69], smart grid management [77,103], traffic [63], and Internet of Things [33]) as they lend themselves perfectly to the idea of large distributed systems. They combine several disciplines ranging from artificial intelligence, software engineering, economics to social sciences [183]. We are mostly interested here in autonomous intelligent systems, where multiple agents are deployed in the same environment and are faced with a series of decision making problems.

Multi-agent decision making problems can often be modelled as a stochastic (or Markov) game (SG) [149,173]. A stochastic game can be formally defined as a tuple: $M = (S, \mathcal{A}, T, \mathcal{R})$, with $n \geq 2$ agents, where:

- S is the system state space
- $\mathcal{A} = A_1 \times \cdots \times A_n$ is the set of joint actions, A_i is the action set of agent i
- $T : S \times \mathcal{A} \times S \to [0, 1]$ is a probabilistic transition function
- $\mathcal{R} = R_1 \times \cdots \times R_n$ are the environment reward functions, where $R_i : S \times \mathcal{A} \times S \to \mathbb{R}$ is the reward function of agent i

At every timestep, the environment emits a joint state $\boldsymbol{s} = \langle s_1, \ldots, s_n \rangle$, where $\boldsymbol{s} \in S$. Depending on the system, the agents can fully observe this state, or can only observe a local view s_i, in which case the problem becomes decentralised and therefore partially observable [15,17]. Notice that the reward received by an agent depends on the joint action taken by all the agents in the environment, not just on her own action.

However, the SG is not the most general model. The stochastic game model can be further generalised to a partially observable stochastic game (POSG) [64,178] to include the possibility that the agents do not have full access to the environment state. In this case, each agent receives an observation and maintains a belief, i.e., a vector specifying the probabilities of being in each possible system state. Because the issue of partial observability is orthogonal to the existence of multiple objectives, but does make the model significantly more complex, we will restrict ourselves to fully observable models in this article. Note however that the solution concepts presented here generalise to partially observable environments as well.

The behaviour of an agent is defined by its policy $\pi_i : S \times A_i \rightarrow [0, 1]$, meaning that given a state, actions are selected according to a certain probability distribution. In the discounted infinite-horizon case, an agent's goal is to find a policy π_i which maximises the expected discounted long-term reward:

$$V^{\pi_i} = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_{i,t} \mid \boldsymbol{\pi}, \mu_0\right] \tag{1}$$

where $\boldsymbol{\pi}$ is the joint policy of the agents acting in the environment, μ_0 is the distribution over initial states s_0, γ is the discount factor and $r_{i,t} = R_i(s_t, \mathbf{a}_t, s_{t+1})$ is the reward obtained by agent i at timestep t, for the joint action $\mathbf{a}_t \in \mathcal{A}$, at state $s_t \in S$ and transitioning to the next state $s_{t+1} \in S$.

Learning in multi-agent systems is considered a vital component, as environments are often characterised by high complexity and stochasticity, meaning that optimal behaviours are often impossible to achieve using pre-programmed approaches [5]. However, we note that transitioning from single- to multi-agent learning is not straightforward. Building an intelligent distributed system is a notoriously difficult problem as it involves dealing with non-stationarity, limited resource sharing, and often requires coordination or overcoming conflicting goals [148]. As a learning paradigm, we mainly consider reinforcement learning (RL) [159], but we will also discuss approaches from related fields such as game theory, planning or negotiation.

Multi-agent decision making is a multifaceted problem that can be explored through the lens of many fields and from different perspectives (e.g., system versus agent point of view). But perhaps the most important distinction we observe in multi-agent learning problems concerns the *definition of the reward function*, establishing the nature of the task at hand, e.g., cooperative or competitive. The literature usually distinguishes between three different settings [30]:

- *cooperative*, where the reward function is the same for all agents: $R_1 = \cdots = R_n = R$. Examples of this setting include domains where all agents work together to optimise the performance of a larger system, such as urban traffic control [93], air traffic control [36] and electricity generator scheduling [96].
- *competitive*, where any win for one agent implies a loss for another. Some competitive settings are zero-sum. Examples of this setting include fully competitive games such as Backgammon [165] and Go [154].
- *mixed*, where no restriction is imposed on the reward function definitions. Mixed games may incorporate elements of both cooperation and competition. Examples of this setting

include games with opposing teams of agents, such as RoboCup soccer [78] and Starcraft II [172].

This classification of multi-agent decision making problems is also reflected in the taxonomy we define in Sect. 4, where we consider a separation between cooperative settings (i.e., team reward) and competitive/mixed setting (i.e., individual reward).

Discrete, tabular representations are the simplest way for agents to store information that they have learned (e.g., policies, environment models, or action values in the case of model-free RL agents). When information is stored discretely, each additional feature tracked in the state leads to an exponential growth in the number of state-action pair values that must be stored [159]. This problem is commonly referred to in the literature as the "curse of dimensionality", a term originally coined by Bellman [16]. While this rarely occurs in simple environments, it may lead to an intractable learning task in complex real-world domains due to memory and/or computational constraints. Learning over a large state-action space is possible, but may take an unacceptably long time to learn useful policies.

Function approximation methods, such as artificial neural networks (ANNs), may be employed to represent policies, environment models or action values. These methods allow one to handle higher dimensional inputs, as well as allowing generalisation between similar observations and actions. Agents using function approximation can also potentially deal with continuous observation and/or action spaces. Recent advances in single and multi-agent RL make use of deep ANNs as function approximators; this emerging paradigm is known as deep reinforcement learning (DRL). For further information on recent multi-agent DRL methods, the interested reader is referred to recent survey articles [67,109].

The issue of scalability and dealing with large state-action spaces in multi-agent systems has been approached numerous times in the literature. A common theme is exploring the idea that numerous multi-agent systems are characterised by sparse interactions between agents and that one can leverage the underlying structure of the problem to deal with issues such as scalability and non-stationarity [66]. For example, De Hauwere [39] proposes a layered approach, starting from a single-agent representation and expanding it only when it is necessary to factor in other agents. In cooperative settings, pursuing a similar idea, one can exploit loose couplings, i.e., each agent's actions will only affect a smaller subset of the system [14,62,80,146]. Another approach is to build influence models that allow agents to identify what are the important components that should be factored into their reasoning process [15,34,114]. Finally, one can identify smaller components inside a complex setting, solve each task individually, and then use transfer planning [113] to go back to the original problem, mitigating issues such as non-stationarity [121].

On a related note, regardless of the setting one is considering, MAS designers often have the possibility to shape or modify the reward function in order to influence the type of behaviour the agents will learn. Furthermore, a problem often encountered in multi-agent systems is the multi-agent credit assignment. In order to address these elements, one can consider various reward structures that have the role of guiding the agents towards certain types of behaviour and also of trying to offer a more informative view on the task at hand. The global and local rewards are two classical reward structures considered in multi-agent reinforcement learning [37,180]. Under the *global reward* paradigm all agents receive the same numerical signal, thus sharing the same goals and being encouraged to develop a cooperative behaviour. However, the global reward does not address the credit assignment problem, as it makes it hard for individual learners to distinguish their contribution to the state of the environment. In contrast, the *local reward* does provide learners with information about the part of the environment they are involved in, but it also encourages selfish behaviour, as each agent is

trying to optimise their own local signal. *Difference rewards* [181] mitigate the previously mentioned issues, by aligning the agents' reward signals with the system's interests, while also allowing each learner to distinguish its own contribution to the performance of the system. The idea of using counterfactual signals to allow agents to reason about their contribution also inspired further work in multi-agent DRL [55,157].

Another dimension to characterise multi-agent systems is represented by the degree of decentralisation. The planning/learning phase and the execution phase may be either (partially) centralised or fully decentralised. The paradigm of *centralised training with decentralised execution* represents a middle ground between fully centralised and decentralised settings often used in cooperative or mixed settings [53,54,81,86,112,144,156]. The aim here is to enrich and aid the training/learning phase with extra information shared between the agents, however during the policy execution phase, the agents act in a fully decentralised manner.

2.2 Multi-objective decision-making

Single-objective decision making requires the existence of a single scalar reward function that agents can observe. The goal for the agents is then to find a policy that maximises the expected sum of these scalar rewards. However, most real-world problems do not adhere to this requirement. Specifically, there are typically multiple objectives that agents should care about. For example, consider the cost and time objectives of our transportation example (Example 1) in Sect. 1.2.

When there are multiple objectives, one might in some special cases still be able to use single-objective methods by using *a priori scalarisation*. Specifically, if there exists a function that maps every possible outcome, i.e., a vector with policy values in each objective, which captures the exact preferences that the user(s) might have with respect to all these possible outcomes of the decision problem; this function is known a priori; and can be applied to the decision problem in such a way that the resulting single-objective problem remains tractable; then single-objective methods could still be used to solve the problem. However, often such a priori scalarisation is either impossible, infeasible or undesirable. Roijers et al. [132] identify three use-case scenarios for multi-objective decision making, shown in Fig. 1, in which this is indeed the case.

In the *unknown weights scenario*, or more precisely the *unknown utility function scenario* (Fig. 1a), a priori scalarisation is undesirable, as the utility that the user is able to get from the alternatives is too uncertain, or even unknown at the moment when planning or learning must occur. For example, when the objectives correspond to things that can be purchased or sold at an open market, but due to the complexity of the planning problem the prices can change significantly before planning or learning is complete. In such cases, it is desirable to compute a coverage set in order to be able to respond as quickly as possible whenever the available information about the market prices is updated.

In the *decision support scenario* (Fig. 1b), a priori scalarisation is infeasible or impossible, as a utility function that corresponds to the preferences of the user is never known explicitly. For example, consider a decision on the medical treatment of a serious illness. This decision problem has objectives such as maximising the probability of being cured and minimising the side effects. However, it is very difficult for a patient to articulate an exact utility function that makes all hypothetical trade-offs between these objectives a priori. In such cases it is therefore highly preferable to create a set containing the available possibly optimal alternatives, and present this set to the user. The decision support scenario thus proceeds almost identically

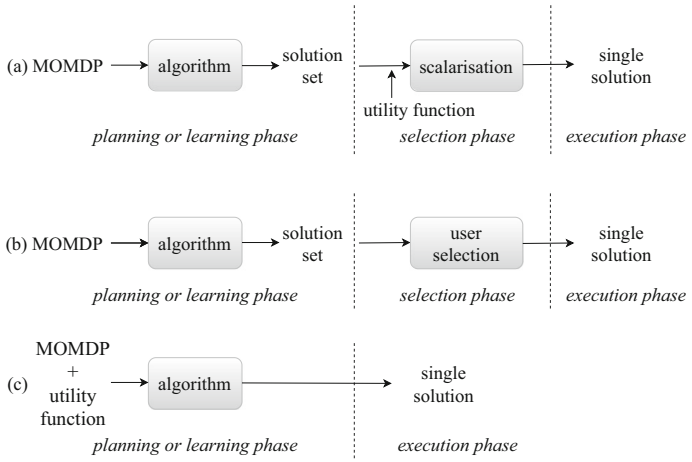

Fig. 1 Use-case scenarios for multi-objective decision making without a priori scalarisation [132]. In the *unknown weights scenario* (**a**), the utility function is not known during the planning or learning phase, thus it is desirable to compute a solution set in order to respond appropriately when the information becomes available. In the *decision support scenario* (**b**), the utility function is never explicitly known, thus it is best to provide a set of optimal alternatives and leave the user to make the final decision. Finally, in the *known weights scenario* (**c**), applying a priori scalarisation can lead to an intractable problem when having to deal with nonlinear utility functions that lead to single objective MDPs with non-additive returns

to the unknown weights scenario. The only difference is that in the selection phase, the user selects a policy from the coverage set according to her arbitrary preferences, rather than explicit scalarisation according to given weights.

Finally, in the *known weights scenario* or *known utility function scenario* (Fig. 1c), a priori scalarisation would in principle be possible, as an exact utility function is available before planning or learning. However, even if this is indeed the case, it can still be undesirable to do so, because performing a priori scalarisation can lead to an intractable problem.

Key to all of these use cases is the notion of *user utility*. Indeed, we argue that for any (multi-objective) decision making problem, the agent should always aim to maximise the user's utility. Specifically, following the work of Roijers et al. [132], we take the *utility-based approach* to multi-objective decision making. In short, this means that the ultimate goal is to maximise *user utility*, and that what constitutes a solution to a multi-objective decision problem should be derived from what is known about the user utility. User utility is characterised by a *utility function u* that maps vector-valued (expected) returns to a scalar value. We first discuss one optimisation criterion, SER, which will be discussed in the next section to give an impression of how an optimal solution set in multi-objective problems can be constructed. In SER, it is the value vector $\boldsymbol{V}^{\pi_i}$, i.e., the expected vector-valued return of policy π_i, that is projected to a scalar value:

$$V^{\pi_i} = u(\boldsymbol{V}^{\pi_i}) \tag{2}$$

In order to derive the optimal solution set—which is what a planning or learning algorithm should output—one should start at the back of the use-case scenario, i.e., the execution phase, and work back, through the selection phase, until a specification of the optimal output of the algorithm is reached. As shown in Fig. 2, in single-agent multi-objective decision making, the execution phase is straight-forward. The agent uses its policy π to interact with the environment, which leads to a value vector $\mathbf{V}^{\pi}$. Under SER (see next section) the

Fig. 2 The execution phase for single-agent multi-objective decision making. From a utility-based perspective, in order to derive the optimal solution set one should start at the back of the use-case scenario, i.e., the execution phase, and work back, through the selection phase, until a specification of the optimal output of the algorithm is reached

utility function u is applied to $\mathbf{V}^{\pi}$. This means that in the selection phase, the policy that maximises $u(\mathbf{V}^{\pi})$, must be available, which brings us to the selection phase. In a known weights scenario, this is trivial, as u is known, so let us focus on the decision support and unknown weights scenarios. In both these scenarios, u is (implicitly or explicitly) applied to a set of alternative value vectors, leading to the maximising policy from a set of alternatives to be chosen.[1] Because in the unknown weights and decision support scenarios, u is at least partially unknown when the agent needs to plan or learn, the planning or learning algorithm should output a set of alternative policies, that for every possible u that a user might have (subject to what is already known at the beginning of the planning or learning phase), contains at least one optimal policy. This is called a *coverage set* (see Sect. 5.2 for a formal definition).

In multi-agent settings, the execution phase is much less straight-forward. In fact, there are different settings, all with their own uses, that lead to very different schemas for the execution phase. Therefore, after discussing the various multi-agent multi-objective decision-theoretic problem settings in Sect. 3, we conduct a thorough analysis of the execution phase for multi-agent multi-objective decision making in Sect. 4. Before going into different choices with respect to the execution phase in multi-agent settings however, we must first discuss another, perhaps even more fundamental choice: when to apply the utility function.

2.3 Multi-objective optimisation criteria

In the previous section, we have seen examples of how single agents might deal with multiple objectives. This is often motivated from the perspective that users may have unknown or uncertain preferences with respect to these objectives. In multi-agent systems, there is another key motivation for explicitly using multi-objective problem formulations. Specifically, individual objectives are typically formulated as clearly measurable desirable properties of a solution which all agents can agree upon (e.g., minus the travel time in minutes it takes to go from one place to another, and minus the costs in euros of getting there in Example 1 in Sect. 1.2). In other words, the rewards for each objective are properties of the environment. The individual utilities of the agents on the other hand are a property of (the user(s) associated with) an agent. Hence, uncertainty in the rewards for each objective and uncertainty about the utility function for each agent are distinctly different properties. Agents may have uncertainty about the utilities of their users, as in a single-agent setting, but also, agents may attempt to hide information about their utility function, if this is information that may be exploitable or otherwise private information. Furthermore, they may simply be unable to communicate information about their utility function in a format that the other agents can understand.

In general, when agents in both single- and multi-agent systems consider multiple conflicting objectives, they should balance these in such a way that the user utility derived from the outcome of a decision problem is maximised. This is known as the utility-based approach

[1] Note that we are assuming here that there is a small discrete set of alternatives, and that this maximisation can explicitly be computed in reasonable time. If this is not the case, for example if their set of alternatives is continuous, the user can be assisted in selecting a good policy using specific algorithms designed for the selection phase [194]. However, in such cases optimality can typically not be guaranteed.

[132,133,193]. Following this approach, we assume that there exists a utility function that maps a vector with a value for each of the d objectives to a scalar utility: $u : \mathbb{R}^d \rightarrow \mathbb{R}$. In a single-agent system, this depends only on the environment and the return vectors that may be obtainable from interacting with the environment. In a multi-system, this also depends on the utility function of the other agents, and how they may adjust their behaviour accordingly while interacting with other agents.

We recall that the value function vector is defined similarly to Eq. 1, as the expected discounted long-term reward:

$$\mathbf{V}^{\pi} = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathbf{r}_t \mid \pi, \mu_0\right]$$

where μ_0 is the distribution over initial states, π is the agent's policy, γ is the discount factor and $\mathbf{r}_t$ is the reward vector received for each of the objectives at timestep t.

When deciding what to optimise in a multi-objective decision making problem, we thus need to apply this utility function to the vector-valued outcomes of the decision problem in some way. There are two choices for how to do this [132,133]. Computing the expected value of the payoffs of a policy first and then applying the utility function, leads to the *scalarised expected returns (SER)* optimisation criterion, i.e.,

$$V_u^{\pi} = u\left(\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathbf{r}_t \mid \pi, \mu_0\right]\right) \tag{3}$$

where V_u^{π} is the return derived by the agent from the vector $\mathbf{V}^{\pi}$. SER is employed in most of the multi-objective planning and reinforcement learning literature [169,177]. Alternatively, the utility function can be applied before computing the expectation, leading to the *expected scalarised returns (ESR)* optimisation criterion [131], i.e.,

$$V_u^{\pi} = \mathbb{E}\left[u\left(\sum_{t=0}^{\infty} \gamma^t \mathbf{r}_t\right) \mid \pi, \mu_0\right] \tag{4}$$

ESR is employed in most of the game theory literature on multi-objective games [23,24,87,150]. Which of these criteria should be considered best depends on how we are interested in evaluating the outcome of a policy. SER is the correct criterion if we want to execute a policy multiple times, and it is the average return over multiple executions that determines the agent's utility. ESR is the correct formulation if the return of a single policy execution is what is important to the agent.

For example, when selecting a medical treatment that will only be carried out once for a single patient, ESR would be the correct criterion to choose as it is the outcome of a single policy execution which is relevant to that patient [131]. Even if a policy is to be executed multiple times, optimising based on the outcome of a single policy execution might be desirable in certain cases. For example, commuting to work is an activity that happens daily; even if the average commute time and comfort levels are acceptable, the utility of these average outcomes (SER) may be substantially different than the average of the utilities of each outcome (ESR) [131]. Therefore, the selection of optimisation criterion is an important part of the process when specifying a multi-objective decision making problem. We note that the difference between SER and ESR is especially key in multi-agent systems as it may not only alter the solutions, but even whether solutions are guaranteed to exist or not (see e.g., Sect. 5.3.2).

2.4 Utility functions

Linear combinations are a widely used canonical example of a utility function:

$$u(\mathbf{r}) = \sum_{d \in D} w_d r_d \tag{5}$$

where D is the set of objectives, $\mathbf{w}$ is a weight vector,[2] $w_d \in [0, 1]$ is the weight for objective d and r_d is the component for objective d of some reward vector $\mathbf{r}$. Interestingly, for such linear utility functions, there is no difference between SER and ESR, as the inner product with the weight vector distributes over the expectation.

Non-linear, discontinuous utility functions may arise in the case where it is important for an agent to achieve a minimum payoff on one of the objectives; such a utility function may look like the following:

$$u(\mathbf{r}) = \begin{cases} r_{t_d} & \text{if} \quad r_d \geq t_d \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where r_d represents the component of $\mathbf{r}$ for objective d, t_d is the required threshold value for d, and r_{t_d} is the reward for reaching the threshold value on d. In general we are interested in the class of all monotonically increasing (possibly non-linear) utility functions.

Definition 1 A scalarisation function u is *monotonically increasing* if:

$$\left(\forall o,\ V_o^{\pi} \geq V_o^{\pi'}\right) \Rightarrow u(\mathbf{V}^{\pi}) \geq u(\mathbf{V}^{\pi'}). \tag{7}$$

This means that if for all objectives, the value for that objective under policy π is greater than or equal than the value for that same objective under policy π', then policy π yields equal or higher utility than policy π'. This is a rather minimal assumption to make, as it translates to: we always want more of each objective. Non-linear utility functions may yield different optimal policies under SER and ESR [131], due to the fact that a non-linear operation may not return the same result when applied to a reward vector before or after the expectation (see Eqns. 3 and 4 above).

Utility functions may not always be known *a priori* and/or may not be easy to define depending on the setting. For example, in the *decision support scenario* [132] it may not be possible for users to specify utility functions directly; instead users may be asked to provide their preferences by scoring or ranking different possible outcomes. After the preference elicitation process is complete, users' responses may then be used to model their utility functions [25,194].

3 Modelling multi-objective multi-agent settings

In this section, we discuss how multi-objective problems with multiple agents can be modelled. We discuss the multi-objective stochastic game, and partially observable stochastic game models, as the most general models, and then show which additional assumptions can be made to arrive at more restricted models.

[2] A vector whose coordinates are all non-negative and sum up to 1.

3.1 The multi-objective stochastic game model

As a framework for defining multi-objective multi-agent decision making settings we will use the Multi-Objective Stochastic Game (MOSG). We formally define a MOSG as a tuple $M = (S, \mathcal{A}, T, \mathcal{R})$, with $n \geq 2$ agents and $d \geq 2$ objectives, where:

- S state space
- $\mathcal{A} = A_1 \times \cdots \times A_n$ set of joint actions, A_i is the action set of agent i
- $T : S \times \mathcal{A} \times S \rightarrow [0, 1]$ probabilistic transition function
- $\mathcal{R} = \mathbf{R}_1 \times \cdots \times \mathbf{R}_n$ reward functions, $\mathbf{R}_i : S \times \mathcal{A} \times S \rightarrow \mathbb{R}^d$ is the vectorial reward function of agent i for each of the d objectives

Furthermore, the same as in the stochastic game case, the MOSG can be extended to also incorporate partial observability. We can thus also define a multi-objective partially observable stochastic game (MOPOSG), where agents do not have access anymore to the full state of the environment. In this situation, agents receive observations from the environment and have to maintain beliefs over possible states. While, for the scope of this work, it is sufficient to consider the MOSG model, we will build our categorisations having the MOPOSG model as the most general case.

An agent behaves according to a policy $\pi_i : S \times A_i \rightarrow [0, 1]$, meaning that given a state, actions are selected according to a certain probability distribution. Optimising π_i is equivalent to maximising the expected discounted long-term reward:

$$\boldsymbol{V}^{\pi_i} = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathbf{R}_i(s_t, \mathbf{a}_t, s_{t+1}) \mid \boldsymbol{\pi}, \mu_0\right] \tag{8}$$

where $\boldsymbol{\pi}$ is the joint policy of the agents acting in the environment, μ_0 is the distribution over initial states s_0, γ is the discount factor and $\mathbf{R}_i(s_t, \mathbf{a}_t, s_{t+1})$ is the vectorial reward obtained by agent i for the joint action $\mathbf{a}_t \in \mathcal{A}$, at state $s_t \in S$. We also note that it is also possible to extend this framework to include the case in which the discount factor is different for each agent i by replacing γ with γ_i.

The value function is also vectorial, $\boldsymbol{V}^{\pi_i} \in \mathbb{R}^d$. We consider that each agent also has an individual utility function u_i to project $\boldsymbol{V}^{\pi_i}$ to a scalar value, as described in Sect. 2.4.

Starting from this model, we will further develop a taxonomy in Sect. 4 focusing on the utility and reward axis. Furthermore, we show how the approaches found in the literature can be mapped to this model by limiting various dimensions such as states, observability, individual rewards, or utilities.

3.2 Special case models

The MOPOSG model is general enough to encompass a wide range of multi-objective multi-agent decision making settings;[3] consequently, many prior decision making models may be viewed as special cases of a MOPOSG. By restricting certain degrees of freedom in the MOPOSG model, one can derive many commonly used decision making models from the single-agent and multi-agent literature, as well as the single-objective and multi-objective

[3] By definition, multi-objective models are a super-class of the corresponding single-objective models. In this survey however, we focus on models with $d \geq 2$, as our aim is to specifically look into multi-objective settings, where the utility functions is a meaningful construct that specifies the importance of different objectives for a given agent. The same remark holds true for multi-agent models.

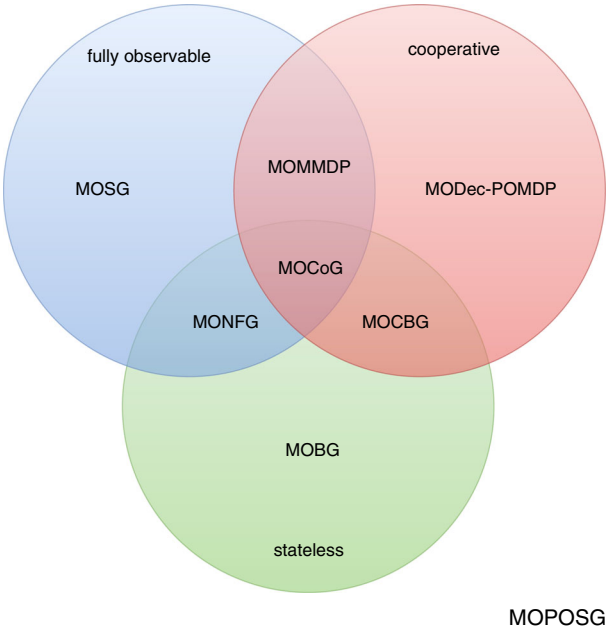

Fig. 3 The MOPOSG is a general model, which encompasses many other common decision making models. The abbreviations in the Venn diagram stand for: multi-objective partially observable stochastic game (MOPOSG), multi-objective stochastic game (MOSG), multi-objective decentralised partially observable Markov Decision Process (MODec-POMDP), multi-objective Bayesian game (MOBG), multi-objective multi-agent Markov Decision Process (MOMMDP), multi-objective normal form game (MONFG), multi-objective collaborative Bayesian game (MOCBG), and multi-objective coordination graph (MOCoG). Please note that these models all correspond to single-objective models, for which the only difference is that they have only one objective. The names of these single-objective models are obtained by dropping "multi-objective", and "MO" from their abbreviations

literature; e.g. by setting the number of agents $n = 1$ and the number of objectives $d = 1$ in a MOPOSG, one may obtain a traditional POMDP.

Figure 3 outlines the relationship between the MOPOSG model and many other common multi-objective multi-agent decision making models, along three axes: (i) observability—the *fully observable* property characterises an environment in which the agents have access to the full state information; (ii) cooperativeness—a *cooperative* task is characterised by all the agents sharing the same reward function and working together towards optimising the performance of a larger system; (iii) statefulness—a *stateless* environment is characterised by only having one state, thus the agents are not required to keep track of this information in their learning or planning process.

Table 4 summarises other common decision making models, and outlines which degrees of freedom of the MOPOSG model must be restricted to derive each other model. We hope that readers will be able to use this as a reference, so they can easily identify ways in which problem settings and algorithms from the single-objective literature could be extended/reanalysed from a multi-objective perspective. Furthermore, it should be possible to easily spot methods developed specifically for multi-objective models which could be applied to the corresponding single-objective model.

Table 4 Summary of which degrees of freedom must be restricted to derive common decision making models from the MOPOSG model

		Model	d	n	$\|S\|$	observability
multi-objective	multi-agent	MOPOSG	≥ 2	≥ 2		
		MOSG	≥ 2	≥ 2		full
		MODec-POMDP	≥ 2	≥ 2		
		MOMMDP	≥ 2	≥ 2		full
		MOCoG	≥ 2	≥ 2	1	full
		MOBG	≥ 2	≥ 2	1	
		MOCBG	≥ 2	≥ 2	1	
		MONFG	≥ 2	≥ 2	1	full
		MOMG	≥ 2	≥ 2	1	full
	single-agent	MOPOMDP	≥ 2	1		
		MOMDP	≥ 2	1		full
		MO Multi-armed bandit	≥ 2	1	1	full
single-objective	multi-agent	POSG	1	≥ 2		
		SG	1	≥ 2		full
		Dec-POMDP	1	≥ 2		
		MMDP	1	≥ 2		full
		CoG	1	≥ 2	1	full
		BG	1	≥ 2	1	
		CBG	1	≥ 2	1	
		NFG	1	≥ 2	1	full
		MG	1	≥ 2	1	full
	single-agent	POMDP	1	1		
		MDP	1	1		full
		Multi-armed bandit	1	1	1	full

Here d is the number of objectives, n is the number of agents and $|S|$ is the size of the environment's state space. Blank cells indicate no restriction, whereas numeric values indicate a required parameter setting. (See Fig. 3 for a list of the abbreviations)

4 The execution phase

As mentioned in Sect. 3, multi-objective multi-agent models are typically named according to assumptions about observability, whether the problem is sequential, and the structure of the reward function. These are indeed important distinctions. However, following the utility-based approach [132], this information is not sufficient to determine what constitutes a solution for such a problem. Specifically, we should aim to optimise the utility of the user(s). In single-agent multi-objective problems, we can typically assume that at execution time we aim to optimise the utility of a single user with a single utility function.[4] The shape of the utility function, in conjunction with the allowed policy space, can be used to derive the optimal solution set that a multi-objective decision-theoretic algorithm should produce.

In multi-agent settings, the situation is more complex than in single-agent settings. Particularly, each individual agent can represent one or more distinct users. In other words, the utility function may vary per agent:

Example 2 Consider a group of friends deciding where to go on holiday, who outsource the decision making to a group of agents (one agent per friend). The objectives they agree on are minimising costs, minimising the distance from the hotel to the beach, maximising the expected number of hours of sun, and maximising the number of museums and other points of cultural interest within a 20km radius.

[4] Or multiple users whose utility functions can be aggregated in an overall aggregated utility function.

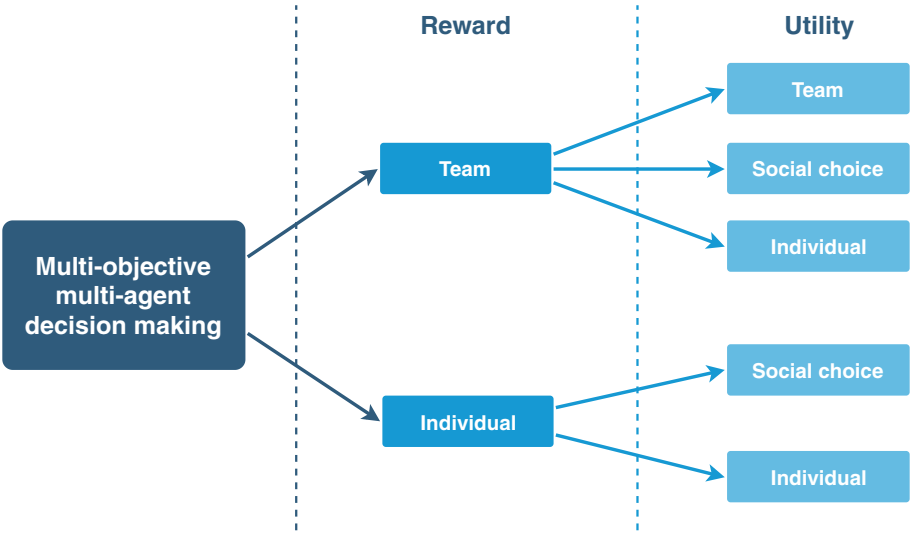

Fig. 4 Taxonomy of multi-objective multi-agent decision making settings. We present an in-depth description of each category in Sects. 4.1 and 4.2

In Example 2, after a decision is reached, every friend will get the same (expected) returns vector. However, each friend may have a different utility for each possible vector—in fact this is the entire reason that this decision problem may be hard. Furthermore, it depends on which perspective we take, as the algorithm designers. In the example, we have taken the perspective of the individual users, but we could also take the perspective of an external observer that wants the outcome to be fair (for whatever definition of fair), i.e., wants to optimise some form of social welfare.

We propose a taxonomy based on the *reward* as well as the *utility* functions. We distinguish between two types of reward functions: a *team reward*, in which each agent receives the same value or return vector for executing the policy, and *individual rewards* in which each agent receives a different value/return vector. Furthermore, we make a distinction in three types of *utility*—more or less orthogonally to the types of rewards—i.e., *team utility*, which is what happens when all the agents serve the same interest, e.g., when they all work for a single company or are on the same football team; *social choice utility*, when we are interested in optimising the overall social welfare across all agents; and *individual utility*, which is what happens if each agent serves a different agenda and just tries to optimise for that. This results in the taxonomy provided in Fig. 4. We further note that the utility functions may be applied according to the ESR or SER criteria (Sect. 2.3 for every setting.

We note that in the taxonomy, the team reward and team utility setting could be translated to a single-agent setting, by flattening out the multi-agent aspect. Specifically, we could define a single agent that would control the actions of all other agents, i.e., one agent choosing its actions from the entire joint action space. As such, the solution concepts from the single-agent multi-objective literature apply [132]. However, the problem can still be significantly harder than a single-agent problem, due to the size of the joint action space, as we discuss in Sect. 4.1.1.

Furthermore, we note that the individual rewards with a team utility setting is not realistic; even if the utility function of all the individual agents would be the same (i.e., the agents have

the same opinion about what is important), that would still lead to different individual utilities due to different input (expected) return vectors. Hence, even when the utility functions are identical, we treat these as *individual utilities*.

In the remainder of this section, we discuss each of the remaining settings in our taxonomy in more detail. In Sect. 5 we discuss the various solution concepts that apply to these settings (see Fig. 10 for an overview).

4.1 Team reward

First, we consider the top row of Fig. 4, *team reward*. In this setting each agent receives the same reward vectors, $\mathbf{R}_1 = \cdots = \mathbf{R}_n = \mathbf{R}$. As a result, the (expected) return vector is the same for each agent when a given joint policy is executed. This is for example the case in *multi-objective multi-agent Markov decision processes (MOMMDPs)* [130, Section 5.2.1], as we discussed in the previous section.

At first glance, this may appear to be a fully cooperative setting. However, this depends on how much the individual agents value their (expected) cumulative reward vectors, i.e., on the utility function of each agent. We distinguish between three cases: team utility, individual utility, and social choice with respect to individual utilities.

4.1.1 Team reward team utility

Perhaps due to its relative simplicity, a commonly encountered case in the multi-objective multi-agent decision-theoretic planning and reinforcement learning literature is the team reward with team utility setting, i.e., all the agents together aim to strive for a single maximum utility, under SER,

$$V^* = \max_{\pi} u(\boldsymbol{V}^{\pi}),$$

or under ESR:

$$V^* = \max_{\pi} \mathbb{E}[u(\boldsymbol{\rho})|\pi, \mu_0],$$

where $\boldsymbol{\rho} = \sum_{t=0}^{\infty} \gamma^t \mathbf{r_t}$. The utility function u (including its parametrisation) may or may not be known to the agents. This is a truly fully cooperative setting. For example, imagine a company that aims to be environmentally responsible, while maximising profits. The reward functions with respect to environmental impact and profit are company-wide, and as such the same for all agents (i.e., employees) in the company. Furthermore, the utility derived from the objectives is also company-based, and all agents can be assumed to be optimising the company's utility.

The single-agent setting and the team-reward team-utility multi-agent setting are mathematically rather similar (e.g., compare Figs. 2, 3, 4 and 5). In fact, the only difference is that there is not a single agent that takes one action each timestep, but multiple agents that each take an action each timestep. Therefore, the optimal solution sets, i.e., coverage sets, can be derived from the same information as in single-agent multi-objective settings (see Sect. 2), and the same types of solution methods apply.

Even though techniques similar to single-agent multi-objective settings can be used to solve multi-agent multi-objective settings, multi-agent multi-objective problems are much more complex than their single-agent counterparts. Specifically, the number of possible joint actions increases exponentially in the number of agents, leading to a much larger policy

Fig. 5 The execution phase for the Team Reward Team Utility setting. This figure depicts the SER optimality criterion, where the expected values (i.e., the average over many executions of the policies) will be input to u. Under ESR the input to u would be $\boldsymbol{\rho}$, i.e., the returns for an individual roll-out

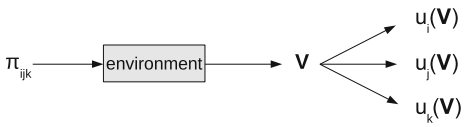

Fig. 6 The execution phase for the Team Reward Individual Utility setting. This figure depicts the SER optimality criterion, where the expected values (i.e., the average over many executions of the policies) will be input to u. Under ESR the input to u would be $\boldsymbol{\rho}$, i.e., the returns for an individual roll-out. Here the notation i, j, k represents different agents in the system, and each agent has its own individual utility function

space. In turn, in cases where the utility function is unknown during planning or learning this leads to much larger coverage sets.

To keep multi-objective multi-agent planning and reinforcement learning tractable in these settings, it is key to exploit so called *loose couplings* [62,79], i.e., each agent's actions only directly affect a subset of the other agents. Loose couplings can be expressed using a factorised reward function. Such a factorised reward function can be visually represented as a graphical model known as a *coordination graph* in the multi-agent literature. The single-shot setting—the multi-objective coordination graph (MO-CoG)—is one of the most well-studied models in the multi-objective multi-agent literature [41,44,45,97,98,133,134,137,140,141,179, etc.]. Exploiting loose couplings also plays an important role in sequential multi-objective multi-agent settings [130,146].

We discuss the solution concepts for this setting in Sect. 5.2.

4.1.2 Team reward individual utility

When a group of agents receives a single shared reward vector, that does not mean that all agents value that reward equally. For example, imagine that you are playing a massive multiplayer online role playing game (MMORPG), and you set out on a quest with teammates. You will play multiple quests with the same team, so you are interested in the expected returns rather than the returns of a single quest (SER). The expected value of doing a quest in terms of experience points, currency and gear is the same for each member of the team, but for different players each of these objectives may be more or less important. Therefore, even when the team gets team rewards for all quests, the members of the team may prefer different quests. This is because mathematically, each agent tries to optimise its own utility via the team value of a joint policy:

$$V_i^{\boldsymbol{\pi}} = u_i(\boldsymbol{V}^{\boldsymbol{\pi}}),$$

under SER, or,

$$V_i^{\boldsymbol{\pi}} = \mathbb{E}[u_i(\boldsymbol{\rho})|\boldsymbol{\pi}, \mu_0],$$

under ESR, where $\boldsymbol{\pi}$ is the joint team policy. This leads to the execution phase depicted in Fig. 6.

The existence of individual utilities immediately poses a problem for the agents. Each agent can only control a small part of the joint policy, i.e., its own actions, and a lack of coordination

may lead to a very bad policy for all agents. In other words, an agent cannot simply maximise its utility by changing its own policy without taking the policies, and policy changes, of the other agents into account. Therefore, in the selection phase—immediately preceding the execution phase—it is vitally important to coordinate, and agree on a joint policy.

There are two main ways to go about this. Firstly, let us view the game-theoretic perspective, in which we aim to find a joint policy that is in some sense *stable*, i.e., agents do not have an incentive to deviate from the joint policy. Stable solutions come in many different levels of strictness [35], from core stability, to Nash equilibria, to individual stability. Particularly challenging in this respect is how to figure out what the individual preferences are. When agents do not or cannot divulge their individual utility functions a priori, for example because it would be hard or even impossible to specify this utility function exactly, algorithms that aim to find stable outcomes must learn about the individual utility functions of the agents to learn whether a joint policy is stable or not [70].

Finding a stable joint policy in the planning or learning phase may seem to mitigate the need for an extensive selection phase; as no agent will have an incentive to deviate from it, deviations should not happen. There are however two problems that could still arise. Firstly, if there are multiple possible stable solutions, the agents still need to agree on which of these to pick. Secondly, in repeated interaction settings, an agent could be spiteful,[5] i.e., aim to be as disruptive to the elected stable solution as possible, in order to strengthen its hand the next time a stable solution must be selected.

Secondly, there is the negotiation perspective [48,167], i.e., agents will try to hammer out a deal on which policy they will jointly execute.[6] This has the advantage that even non-stable solutions—that may offer better utility for all agents than the stable ones—could be selected, as long as the agents are obligated to follow through. For example, the Automated Negotiating Agents Competition (ANAC) [74] considers three-agent negotiations in which agents negotiate about possible alternative outcomes. When each alternative is associated with its own vector corresponding to different objectives, the agents will know that some outcomes are Pareto-dominated, and should therefore be excluded from consideration, but for the solutions that are in the Pareto coverage set, different agents may have different preferences. In general, the outcome of such a negotiation should thus be a "deal" between the agents about which alternative joint policy from the coverage set to execute.

Finally, we note that there is a special case of the team rewards and individual utilities, in which the number of objectives is equal to the number of agents, and the utility function of each agent would just be the value of the objective corresponding to that agent. This special case may seem identical to the single-objective multi-agent case with individual rewards, but there is in fact a significant difference. Specifically, it reflects the situation in which the agents can care about the rewards of the other agents, and can make (a priori or a posteriori) agreements on which division of rewards is admissible. In other words, it can be used to model various degrees of altruism. At a very minimum, the agents could all exclude Pareto-dominated solutions, leading to the situation in which agents always prefer to help the other agents to increase their rewards, as long as it does not cost them anything. A bit more

[5] Spite can evolve in a population through strategies such as bullying. A spiteful behaviour describes a strategy through which a player will choose to harm others, even at the expense of incurring a cost, given that in the long term this will prove beneficial. This is due to the fact that fitness metrics have a comparative nature [171].

[6] One might argue that it may be theoretically possible to create much larger MO(PO)SGs from a simpler multi-objective multi-agent problems by including the meta-interactions necessary for negotiation to the problem, and try to solve the problem using a general-purpose MO(PO)SG solver. However, this is likely not to be a fruitful approach, as such a large MO(PO)SG may well be intractable for general-purpose solvers for MO(PO)SGs.

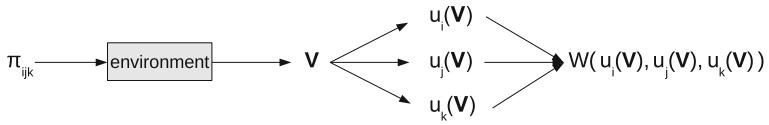

Fig. 7 The execution phase for the Team Reward and Social Welfare with Respect to Individual Utilities setting. Please note that the social welfare can depend *both* on the utilities of the agents *and* the value/return vector. This figure depicts the SER optimality criterion, where the expected values (i.e., the average over many executions of the policies) will be input to u. Under ESR the input to u would be $\boldsymbol{\rho}$, i.e., the returns for an individual roll-out. Here the notation i, j, k represents different agents in the system, and each agent has its own individual utility function

drastically, the agents could agree to exclude a joint policy from consideration if another policy exists in which the total sum of the values for each objective/agent is at least the same, but is more fairly distributed over the agents. This leads to the solution concept of Lorenz optimality [60], which we will discuss in Sect. 5.2.4.

4.1.3 Team reward and social welfare with respect to individual utilities

In the individual rewards setting, it is hard to predict, let alone optimise for, utility. This is because the agents have different agendas, leading to complex behavioural dynamics, in which agents react to each other's behaviours. This process may not converge to stable solutions. Furthermore, the individual utility functions may not be common knowledge.

A different perspective on this problem is to take a step back from the self-interested agents and optimising for their individual utilities, and instead look at what would be a *desirable outcome*. For example, we can focus on what would be socially favourable by the agents. Once we have decided on what would be desirable, we can define *social welfare* as a *social choice function*, corresponding to the desirability of each outcome, and construct a system of payments that will make the joint policy converge to the desired outcome. This is known as mechanism design [173, Ch. 6]. For example, looking at the massive multiplayer online role playing game (MMORPG) example of the previous section, the social welfare perspective would take a team perspective, determine what is a socially desirable outcome for the team given the individual utilities of the individual agents, and aim to obtain such desirable outcomes. This can for example be achieved through incentivising agents to "take one for the team", i.e., taking different actions from those that would serve their individual utilities best. In artificial systems these incentives are often assumed to be numerical, e.g., monetary incentives. In an MMORPG, for human players, these incentives are usually much more soft and social in nature, e.g., one's reputation in a guild.

It is important to note that the social welfare function can depend both on the value or return vector, as well as the individual utilities of the agents, as illustrated in Fig. 7. For example, in traffic, social welfare may depend on the pollution levels, as well as fairness between different vehicles in terms of their total expected time that they have to wait for traffic lights.

In mechanism design, the challenge is to formulate the system of payments in such a way that the agents will be non-manipulable, i.e., do not have an incentive to lie about their preferences. If this succeeds, the agents will report their preferences truthfully, and from a planning perspective, the decision-problem becomes fully cooperative, i.e., aiming to collectively optimise the social choice function.

For multi-objective decision problems, the social welfare perspective can for example be used by governments to control the parameters of tenders, to balance the different objectives

for projects. For example, in a traffic network maintenance planning setting [147], the balancing of traffic delays and costs can be made a posteriori, by computing a convex coverage set for a cooperative multi-objective multi-agent MDP [136], because a non-manipulable mechanism exists for every different weighting of the objectives. While mechanism design methods are very powerful, they do pose challenges. Specifically, they typically require (near-)optimal policies to be guaranteed, and they require agents to articulate their preferences exactly, in order for the mechanism to be non-manipulable. The first condition poses restrictions on the type of planning methods than can be used; which is particularly important in highly complex sequential decision problems. The second condition poses restrictions on the way the utility functions can be accessed. We discuss the implications of this in Sect. 5.6.

4.2 Individual rewards

Up until now, we have considered situations in which all agents have the same vector input to the utility function, $\mathbf{V}$ under SER and $\boldsymbol{\rho}$ under ESR, but may have separate individual utilities, $u_i(\mathbf{V})$ or $u_i(\boldsymbol{\rho})$, with respect to this vector. We now consider situations in which the rewards, and therefore (expected) return vectors, are different for the individual agents.

First, we note that we consider only two settings for individual rewards: individual utilities and social choice. This is because when individual rewards are received, even if the utility functions for all agents are the same, the resulting utilities are still individual, and the interest of the agents may still be opposed.

We observe that individual reward settings may seem similar to the team reward but individual utilities settings, regarding the fact that ultimately the joint policies will be selected on the utilities of the individual agents. However, whether the value (or return) vectors are identical or not, can have a profound impact on how complex it is to solve the decision problem. Specifically, a joint policy can often be excluded from consideration if all agents agree that executing a different policy would be better for all agents.[7] When the rewards are shared, all agents will share the same joint policy outcomes and will thus always agree on whether a policy is Pareto-dominated or not. When the rewards are individual however, a joint policy can be the only Pareto-optimal policy (in terms of value or return vectors) for one agent, while it is dominated for another. In other words, settings with individual rewards are considerably more difficult to solve.

4.2.1 Individual reward individual utility

First let us consider the completely self-interested setting of agents with individual rewards and utilities. This results in the execution phase depicted in Fig. 8. For example, imagine a traffic scenario, individual drivers may want to minimise their individual travel time, their individual fuel costs, and their probabilities of getting into an accident. However, given a joint policy for all the drivers, each driver receives different rewards in terms of these objectives. Furthermore, some drivers may care more about minimising their traffic time, while others care more about minimising fuel costs, i.e., they have different utility functions. Note that this may make the behaviour of other drivers more unpredictable (possibly leading to misunderstands, if not irritations).

For example Fernandez et al. [51] study cooperative games, in which coalitions of agents are formed that can obtain rewards in different objectives, and then divide the value of these

[7] Note that this is not a sufficient condition in multi-agent settings though, as there may be equilibria that are Pareto-dominated.

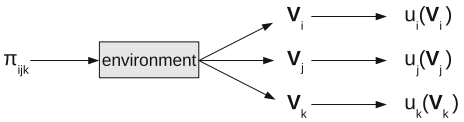

Fig. 8 The execution phase for the Individual Reward Individual Utility setting. This figure depicts the SER optimality criterion, where the expected values for each agent (i.e., the average over many executions of the policies) will be input to u. Under ESR the input to u would be $\boldsymbol{\rho}_i$, i.e., the returns for an individual roll-out for an individual agent. Here the notation i, j, k represents different agents in the system, and each agent has its own individual utility function and its own individual value vector

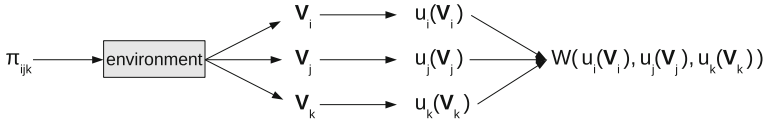

Fig. 9 The execution phase for the Individual Reward and Social Welfare with Respect to Individual Utilities setting. Please note that the social welfare can depend *both* on the utilities of the agents *and* the value/return vectors for each agent. This figure depicts the SER optimality criterion, where the expected values for each agent (i.e., the average over many executions of the policies) will be input to u. Under ESR the input to u would be $\boldsymbol{\rho}_i$, i.e., the returns for an individual roll-out for an individual agent

objectives amongst themselves, leading to individual rewards. Subsequently, they consider what information regarding the utility functions of the agents is available, and whether stable coalitions can be found given this information.

Because the individual rewards and individual utilities setting is highly complex, it is vitally important to exploit all the available information there is regarding the utility functions of the agents. For example, consider the situation in which all individual agents have the same utility function [51,163], but it is not a priori clear what this utility function is, or the utility function is not fixed. This could be the case if the objectives correspond to resources that can be sold on an open market. Because these prices can vary (possibly rapidly) over time, the agents will need to adjust their policies according to the latest possible price information. A multi-objective multi-agent model with individual rewards and individual utilities, may then help to predict how the agents will respond to changing prices.

In general, the individual utility functions may be different for each agent, and various degrees of knowledge may exist about their shape or properties. In such settings, it may be hard to produce a sufficiently compact set of possibly viable joint policies to choose from or negotiate with. In this case, we suspect that interactive approaches [70], in which more information about the utility functions is actively pursued by querying the agents while planning or learning to limit the set of viable alternatives, will play an important role in future research.

4.2.2 Individual reward and social choice with respect to individual utilities

Finally, let us consider the individual rewards and utilities, from the perspective of social choice. This leads to the situation in Fig. 9, in which agents obtain individual value or return vectors, value these according to individual utilities, which are then weighed up, together with the individual value or return vectors, through a social welfare function.

As in the team reward setting, it is important to note that the social welfare function can depend both on the individual value or return vectors, as well as the individual utilities of the agents. For example, in auctions [120], social welfare may depend on attributes of the winning bid(s), as well as a fair outcome in terms of payments to the individual agents, that

together with the costs the agents need to support to execute their bids if chosen, typically determine the individual utilities.

As in the team reward but individual utilities case, we aim to find a mechanism, i.e., a social welfare function, that forces agents to be truthful about their utility functions, such that the joint policy can be optimised with respect to a notion of social welfare. An interesting—but to our knowledge unexplored—aspect would be to investigate, in the case when individual reward vectors are common knowledge, but the preferences are (partially) unknown, whether such mechanisms could still be established, possibly through active querying to obtain information about the individual utility functions.

5 Solution concepts

In this section we introduce some of the main solution concepts which are featured in MAS and multi-objective optimisation research, as well as explaining how they relate to the scenarios described in our taxonomy above.

In the context of MAS, it is difficult to identify what constitutes an optimal behaviour, as the agents' strategies are interrelated, each decision depending on the choices of the others. For this reason, we usually try to determine interesting groups of outcomes (i.e., solution concepts), which allow the system can reach some form of equilibrium. Figure 10 provides an overview of which of these solution concepts are relevant to each of the five settings in our multi-agent decision making taxonomy.

5.1 Policies

We introduce a few preliminary definitions regarding types of behaviour agents can learn, depending on the action selection strategy given a certain state or on whether or not time plays any role in the policy definition.

A *deterministic* (or *pure*) policy is one where the same action a is always selected for a given state s (i.e., $Pr(a|s) = 1$). A *stochastic* policy is one where actions in a given state are selected according to a probability distribution (i.e., $Pr(a|s) \in [0, 1], \forall a \in A$). The output of a *stationary* policy depends only on state, not on time. The output of a *non-stationary* policy may vary with both state and time.

A deterministic environment or system is one where the transition function is deterministic, i.e., the system always transitions to the same next state, for a given system state and joint action. While in single-objective decision problems it is often sufficient to take only deterministic stationary policies into account, it is known that in multi-objective decision problems stochastic or non-stationary policies can lead to better utility [132,177] both under SER and ESR [131].

A *mixture* policy [151,168] is a stochastic combination of deterministic policies (referred to as base policies). This technique has been used in single agent multi-objective settings to combine two or more deterministic Pareto optimal policies to satisfy a user's preferences. Mixing happens *inter-episode* only, rather than *intra-episode*. Vamplew et al. [168] note that switching between base policies during an episode will likely result in erratic and sub-optimal behaviour. Therefore, one of the available base policies is selected probabilistically at the beginning of each episode and followed for the entire episode duration. The aim is to determine mixture probabilities, which on average, after a large number of runs, will yield the desired long-term average return on each objective.

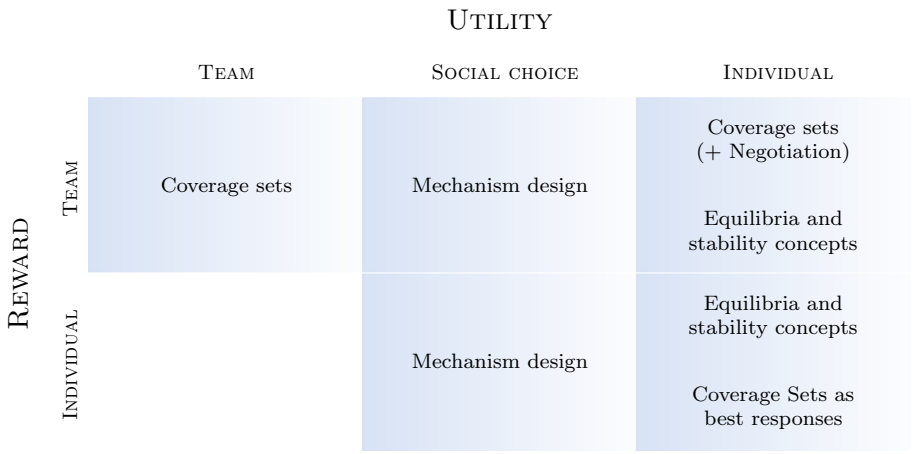

Fig. 10 This mapping outlines which solution concepts (Sect. 5) are relevant to each of the different reward and utility settings identified in our taxonomy (Sect. 4) of decision making in multi-objective multi-agent systems. Algorithmic approaches and applications for each of the above combinations of solution concepts and reward and utility settings are discussed in Sect. 6

As noted in Sect. 4.1.1, the team reward team utility setting is similar enough to single-agent multi-objective settings such that methods developed for one may be easily applied to the other; mixture policies are one such method which could feasibly be used in the team reward team utility setting.

5.2 Coverage sets

The optimal solution in single-agent multi-objective decision making is called a *coverage set (CS)* [132,133]. A coverage set contains at least one optimal policy for each possible utility function, $u(\mathbf{V}^{\pi})$, i.e., if a set $\mathcal{C}$ is a coverage set then, under SER,

$$\forall u \in \mathcal{U} : \max_{\pi \in \Pi} u(\mathbf{V}^{\pi}) = \max_{\pi \in \mathcal{C}} u(\mathbf{V}^{\pi}),$$

and under ESR,

$$\forall u \in \mathcal{U} : \max_{\pi \in \Pi} \mathbb{E}[u(\boldsymbol{\rho})|\pi, \mu_0] = \max_{\pi \in \mathcal{C}} \mathbb{E}[u(\boldsymbol{\rho})|\pi, \mu_0],$$

where Π is the space of all possible (and allowed) policies, $\boldsymbol{\rho}$ are the vector-valued returns, i.e., $\boldsymbol{\rho} = \sum_{t=0}^{\infty} \gamma^t \mathbf{r_t}$ and $\mathcal{U}$ is the set of all possible utility functions. Furthermore, coverage sets do not contain dominated policies,

$$\pi \in \mathcal{C} \rightarrow \exists u \in \mathcal{U} : u(\mathbf{V}^{\pi}) = \max_{\pi' \in \mathcal{C}} u(\mathbf{V}^{\pi'}),$$

under SER, and,

$$\pi \in \mathcal{C} \rightarrow \exists u \in \mathcal{U} : \mathbb{E}[u(\boldsymbol{\rho})|\pi, \mu_0] = \max_{\pi' \in \mathcal{C}} \mathbb{E}[u(\boldsymbol{\rho})|\pi', \mu_0],$$

under ESR, i.e., a coverage set should only contain policies that are optimal for some utility function u. Finally, algorithms should aim to construct coverage sets that are as small as possible, but as coverage sets are not unique, constructing a minimally sized one is far from trivial.

5.2.1 Motivations for coverage sets in multi-agent settings

In single-agent settings, coverage sets need to be constructed with respect to any possible utility function allowed by the problem specification. However, due to the single-agent nature, it can be assumed that ultimately, in the execution phase, there will be one true utility function that governs user utility. Multi-agent settings are more complex; the different agents can represent different interests, and may be optimising for different utility functions. Nonetheless, there are many multi-agent settings for which coverage sets are the appropriate solution concept.

The first and most straightforward motivation is the **team reward and team utility** setting described in Sect. 4.1.1. This is a fully cooperative setting; all rewards and the utility derived from that is shared between all agents. Therefore, there is only one true utility function in the execution phase, and the motivation for coverage sets being the right solution concept is the same as for single-agent multi-objective decision making. For example, this is the case when there are multiple agents belonging to the same team or organisation are tackling a problem together, e.g., a soccer team or different agents belonging to the same company.

However, team utility is not strictly necessary for coverage sets to be useful. In a team reward but individual utility setting, coverage sets could be used if all agents will agree (preferably contractually) that they will always execute a policy that is potentially optimal. In this case, a coverage set can be computed as the *input to a negotiation* [73,74,115] between the agents of which policy to execute.

Note that this strategy of computing a coverage set and then negotiating does not trivially apply to individual reward settings. In the case of individual rewards, a joint policy can be optimal for one agent, while it can be strictly dominated for another. Generalising the concept of a coverage set to individual reward settings is an open question that would merit investigation.

Furthermore, in an individual utility setting, a coverage set can also be a *set of possible best responses to the behaviours of the other agents*. Of course, one needs a different coverage set per combination of possible behaviours for all the individual other agents. This may quickly become infeasible if the set of possible policies of the other agents becomes large. However, if one can model the opponents using a small set of possible behaviours this may be a viable approach.

Finally, there is a uniquely individual rewards setting coverage set, for the special case that each objective corresponds to one agent, and the objectives of other agents are seen as secondary objectives. In other words, this is the case where agents are at least partially altruistic. This concept is called a Lorenz optimal set, which we discuss in Sect. 5.2.4.

5.2.2 Convex coverage sets

A convex coverage set is the optimal solution set when it can be assumed that the utility functions of all agents are linear. This is a salient case in the multi-objective decision making literature, and for example applies in the case where each objective corresponds to a resource that can be bought or sold on an open market. Specifically, the utility functions are assumed to be the inner product between a vector of weights $\mathbf{w}$ and the value vector of the joint policy $\mathbf{V}^{\boldsymbol{\pi}}$, i.e.,

$$u_i(\mathbf{V}^{\boldsymbol{\pi}}) = \mathbf{w} \cdot \mathbf{V}^{\boldsymbol{\pi}}. \tag{9}$$

Please note that for this type of utility function, there is no difference between SER and ESR, as $E[\mathbf{w} \cdot \boldsymbol{\rho}|\boldsymbol{\pi}, \mu_0] = \mathbf{w} \cdot E[\boldsymbol{\rho}|\boldsymbol{\pi}, \mu_0] = \mathbf{w} \cdot \mathbf{V}^{\boldsymbol{\pi}}$.

In the case of linear utility functions, the undominated set—the convex hull—is defined as follows:

Definition 2 The *convex hull (CH)* is the subset of the set of all admissible joint policies Π for which there exists a $\mathbf{w}$ for which the linearly scalarised value is maximal:

$$CH(\Pi) = \{\boldsymbol{\pi} : \boldsymbol{\pi} \in \Pi \wedge \exists \mathbf{w} \forall (\boldsymbol{\pi}' \in \Pi)\ \mathbf{w} \cdot \mathbf{V}^{\boldsymbol{\pi}} \geq \mathbf{w} \cdot \mathbf{V}^{\boldsymbol{\pi}'}\}. \tag{10}$$

Please note that in this definition, we assume a team reward setting, such that $\mathbf{V}^{\boldsymbol{\pi}}$ is a single vector.

One problem with the CH is that it can be undesirably large; and in the case of stochastic policies, often infinitely large. However, in such cases a *convex coverage set (CCS)* can often be defined that is much more compact:[8]

Definition 3 A set $CCS(\Pi)$ is a *convex coverage set (CCS)* if it is a subset of $CH(\Pi)$ and if, for every $\mathbf{w}$, it contains a policy whose linearly scalarised value is maximal:

$$CCS(\Pi) \subseteq CH(\Pi) \wedge (\forall \mathbf{w})(\exists \boldsymbol{\pi}) \left(\boldsymbol{\pi} \in CCS(\Pi) \wedge \forall (\boldsymbol{\pi}' \in \Pi)\ \mathbf{w} \cdot \mathbf{V}^{\boldsymbol{\pi}} \geq \mathbf{w} \cdot \mathbf{V}^{\boldsymbol{\pi}'}\right). \tag{11}$$

While in the case of individual utility, the actual $\mathbf{w}_i$ can differ per agent, the CCS contains at least one optimal policy for every $\mathbf{w}_i$, and therefore forms a suitable starting point for finding possible compromises based on the assumption that all agents have a linear utility function. For example, a strategy could be to try to estimate each $\mathbf{w}_i$ and take the average weights vector across all agents to select the default compromise. Of course, agents may also want to negotiate [73,74] in order to get a better deal than such a default compromise.

5.2.3 Pareto coverage sets

For monotonically increasing but non-linear utility functions, the undominated and coverage sets become significantly larger than for linear utility functions. To be able to define a coverage set for this setting under SER we must first define the concept of Pareto dominance:

Definition 4 A joint policy $\boldsymbol{\pi}$ *Pareto-dominates* ($\succ_P$) another joint policy $\boldsymbol{\pi}'$ when its value is at least as high in all objectives and strictly higher in at least one objective:

$$\mathbf{V}^{\boldsymbol{\pi}} \succ_P \mathbf{V}^{\boldsymbol{\pi}'} \Leftrightarrow \forall o, V_o^{\boldsymbol{\pi}} \geq V_o^{\boldsymbol{\pi}'} \wedge \exists o', V_{o'}^{\boldsymbol{\pi}} > V_{o'}^{\boldsymbol{\pi}'}. \tag{12}$$

Looking at this definition, it is clear that no Pareto-dominated policy can ever have a higher utility under a monotonically increasing utility function:

$$\mathbf{V}^{\boldsymbol{\pi}} \succ_P \mathbf{V}^{\boldsymbol{\pi}'} \rightarrow u_i(\mathbf{V}^{\boldsymbol{\pi}}) \geq u_i(\mathbf{V}^{\boldsymbol{\pi}'}).$$

As long as being monotonically increasing is the only assumption we can make about the utility function, this is in fact the only thing that can be said of the relative preferences across all possible utility functions. Therefore, we use the concept of Pareto dominance to define the undominated set for monotonically increasing utility functions, the Pareto front:

Definition 5 The *Pareto front* is the set of all joint policies that are not Pareto dominated by any other joint policy in the set of all admissible joint policies Π:

$$PF(\Pi) = \{\boldsymbol{\pi} : \boldsymbol{\pi} \in \Pi \wedge \neg \exists (\boldsymbol{\pi}' \in \Pi), \mathbf{V}^{\boldsymbol{\pi}'} \succ_P \mathbf{V}^{\boldsymbol{\pi}}\}. \tag{13}$$

[8] For details on why this is so, please refer to [137].

A *Pareto coverage set* (PCS) of minimal size can be constructed by selecting only one policy of the policies with identical value vectors from the $PF(\Pi)$:

Definition 6 A set $PCS(\Pi)$ is a *Pareto coverage set* if it is a subset of $PF(\Pi)$ and if, for every policy $\boldsymbol{\pi'} \in \Pi$, it contains at least one policy that either dominates $\boldsymbol{\pi'}$ or has equal value to $\boldsymbol{\pi'}$:

$$PCS(\Pi) \subseteq PF(\Pi) \wedge \forall(\boldsymbol{\pi'} \in \Pi)(\exists \boldsymbol{\pi}) \left(\boldsymbol{\pi} \in PCS(\Pi) \wedge (\mathbf{V}^{\boldsymbol{\pi}} \succ_P \mathbf{V}^{\boldsymbol{\pi'}} \vee \mathbf{V}^{\boldsymbol{\pi}} = \mathbf{V}^{\boldsymbol{\pi'}})\right). \tag{14}$$

Negotiating a good compromise from a set of alternatives with different values for all objectives is a typical setting for negotiation [74]. Note that in multi-objective settings, agents and/or users are often incapable of specifying their utilities numerically [194]. However, recently there has been research in automated negotiation focusing on preference uncertainty [83,166], i.e., uncertainty about the individual utility functions, and eliciting preferences [10,83], making realistic negotiation with the PCS of a multi-objective decision problem as input, possible.

Under ESR the situation becomes significantly more complex, i.e., in general, the undominated set is defined as:

Definition 7 The *undominated set of policies (U)* under possibly non-linear monotonically increasing u, under ESR, is the subset of the set of all admissible joint policies Π for which there exists a u for which the scalarised value is maximal:

$$U(\Pi) = \left\{\boldsymbol{\pi} : \boldsymbol{\pi} \in \Pi \wedge \exists(u \in \mathcal{U})\ \forall(\boldsymbol{\pi'} \in \Pi)\ \mathbb{E}[u(\boldsymbol{\rho})|\boldsymbol{\pi}, \mu_0] \geq \mathbb{E}[u(\boldsymbol{\rho})|\boldsymbol{\pi'}, \mu_0]\right\}. \tag{15}$$

However, this is very hard to determine without further information about u. To our knowledge no research has yet been done into constructing (approximate) undominated or coverage sets under ESR. In fact, the available research in single-agent MORL typically assumes that u is known [131].

5.2.4 Lorenz optimal sets

A uniquely multi-agent coverage set is the Lorenz optimal set [117]. Underlying this solution concept is the assumption that each objective corresponds to the interest of each individual agent. Furthermore, it is assumed that the interests of the other agents are an objective for every agent. In other words, the agents are at least in part altruistic. Finally, it is assumed that "more equal" solutions - we will define this exactly below—are better if the sum of utilities does not increase. This final assumption corresponds to a (rather minimal) concept of fairness.

The use-case for Lorenz optimal sets is: all agents agree that fair solutions are better, hence a set of possibly fair solutions will be computed, after which the agents will negotiate which solution from this set to select. It is thus vital that the agents can rely on the selected solution being followed and that no individual agent will enrich itself to the detriment of the group. This can either be enforced contractually, or simply by the notion that the group of agents will have to rely on each other in the future, and that agents that do not follow the convention will no longer be allowed to participate in other decision problems in which the same agents will need to cooperate.

The underlying idea of the Lorenz notion of fairness is the so-called Robin-Hood transfer: if in a vector objective i has a higher value, v_i than objective j, v_j, then transferring part of

the difference, i.e., setting the value of v_i to $v_i - \delta$ and v_j to $v_j + \delta$, for $0 < \delta \leq v_i - v_j$, yields a fairer, and therefore better value vector. More formally, this can be captured in the concept of Lorenz domination. To test whether a vector $\mathbf{V}^{\boldsymbol{\pi}}$ Lorenz dominates a vector $\mathbf{V}^{\boldsymbol{\pi}'}$, both vectors are first projected to their corresponding Lorenz vectors:

Definition 8 The Lorenz vector $\mathbf{L}(\mathbf{V}^{\boldsymbol{\pi}})$ of a vector $\mathbf{V}^{\boldsymbol{\pi}}$ is defined as:

$$\left(v_{(1)},\ v_{(1)} + v_{(2)}, \ldots,\ \sum_{i=0}^{N} v_{(i)} \right),$$

where, $v_{(1)} \leq v_{(2)} \leq \cdots \leq v_{(N)}$, correspond to the values in the vector $\mathbf{V}^{\boldsymbol{\pi}}$ sorted in increasing order.

NB: this definition is under SER. To our knowledge no research has been done with regards to Lorenz optimality under ESR.

Definition 9 A vector $\mathbf{V}^{\boldsymbol{\pi}}$ Lorenz dominates ($\succ_L$) a vector $\mathbf{V}^{\boldsymbol{\pi}'}$ when:

$$\mathbf{V}^{\boldsymbol{\pi}} \succ_L \mathbf{V}^{\boldsymbol{\pi}'} \Leftrightarrow \mathbf{L}(\mathbf{V}^{\boldsymbol{\pi}}) \succ_P \mathbf{L}(\mathbf{V}^{\boldsymbol{\pi}'}),$$

i.e., when the Lorenz vector of $\mathbf{V}^{\boldsymbol{\pi}}$ Pareto dominates the Lorenz vector of $\mathbf{V}^{\boldsymbol{\pi}'}$.

Definition 10 The *Lorenz Optimal Set* is the set of all joint policies that are not Lorenz dominated by any other joint policy in the set of all admissible joint policies Π:

$$LOS(\Pi) = \left\{ \boldsymbol{\pi} : \boldsymbol{\pi} \in \Pi \wedge \neg\exists(\boldsymbol{\pi}' \in \Pi), \mathbf{V}^{\boldsymbol{\pi}'} \succ_L \mathbf{V}^{\boldsymbol{\pi}} \right\}. \tag{16}$$

A *Lorenz coverage set* (LCS) of minimal size can be constructed by selecting only one policy of the policies with identical value vectors from the $LOS(\Pi)$, similar to constructing a PCS from a PF.

5.3 Equilibria concepts

We now shift our attention to the individual utility setting and discuss game theoretic equilibria as solution concepts for multi-objective multi-agent systems. We present an in-depth discussion with respect to Nash and correlated equilibria, together with their extension under the two multi-objective optimisation criteria ESR and SER. Section 5.7 offers an overview of some other solution concepts proposed in the literature of multi-objective games.

5.3.1 Nash equilibria

When multiple self-interested agents learn and act together in the same environment, it is generally not possible for all agents to receive the maximum possible reward. Therefore, MAS are often designed to converge to a Nash equilibrium [152] (NE). This notion of equilibrium was first introduced by Nash [106], and is one of the most important concepts used to analyse MAS [183].

Consider a multi-agent system with n agents, where $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_i, \ldots, \pi_n)$ is their joint policy, with π_i representing the stochastic policy of agent i. We also define $\boldsymbol{\pi}_{-i} = (\pi_1, \ldots, \pi_{i-1}, \pi_{i+1}, \ldots, \pi_n)$ to be a joint policy without the policy of agent i. We can thus write $\boldsymbol{\pi} = (\pi_i, \boldsymbol{\pi}_{-i})$.

Definition 11 A joint policy $\boldsymbol{\pi}^{NE}$ leads to a Nash equilibrium if for each agent $i \in \{1, \ldots, n\}$ and for any alternative policy π_i:

$$V_i^{(\pi_i^{NE}, \boldsymbol{\pi}_{-i}^{NE})} \geq V_i^{(\pi_i, \boldsymbol{\pi}_{-i}^{NE})} \tag{17}$$

Whenever the above inequality holds true for all possible policies and for all agents in a MAS, a Nash equilibrium exists. In other words, a Nash equilibrium occurs whenever any individual agent cannot improve its own return by changing its behaviour, assuming that all other agents in the MAS continue to behave in the same way.

In cooperative MAS (i.e., the **team reward** scenario), coordinating agents' actions to achieve the highest possible system welfare is already a difficult problem. While it is possible for multiple individual learners in a cooperative MAS to converge to a point of equilibrium, whether they will converge to an optimal joint policy (one which maximises the system welfare) depends on the specific learning algorithm and reward scheme used [42,89,122].

5.3.2 Nash equilibria in multi-objective multi-agent settings

In multi-objective decision making, each agent is trying to optimise her return along a set of objectives. Each agent needs to also make compromises between competing objectives on the basis of her utility function. As a motivation for why NE is an appropriate solution concept in MOMAS, we look at the **team reward individual utility** (Sect. 4.1.2) and **individual reward individual utility** (Sect. 4.2.1) scenarios. In both these cases, the utility derived by each agent from the received reward is different, regardless if this reward is the same or not for all the agents. These constitute the most difficult scenarios in our taxonomy. Furthermore, one should also consider which optimisation criteria are best to use, based on what each agent is looking to optimise. Depending on whether an agent cares about average performance over a number of policy executions, or just the performance of a single policy execution [131], we can define the concept of a Nash equilibrium from the perspective of the two multi-objective optimisation criteria defined in Sect. 2.3: ESR and SER.

To simplify our notation let us denote the discounted sum of rewards received by agent i by: $\boldsymbol{\rho}_i = \sum_{t=0}^{\infty} \gamma^t \mathbf{r}_{i,t}$. We can then re-write the expected value for agent i under a joint policy $\boldsymbol{\pi}$, given the distribution μ_0 over initial states as: $\boldsymbol{V}_i^{\boldsymbol{\pi}} = \mathbb{E}\left[\boldsymbol{\rho}_i \mid \boldsymbol{\pi}, \mu_0\right]$.

Definition 12 (*Nash equilibrium for Expected Scalarised Returns*) A joint policy $\boldsymbol{\pi}^{NE}$ leads to a Nash equilibrium under the Expected Scalarised Returns criterion if for each agent $i \in \{1, \ldots, n\}$ and for any alternative policy π_i:

$$\mathbb{E}\left[u_i(\boldsymbol{\rho}_i) \mid \left(\pi_i^{NE}, \boldsymbol{\pi}_{-i}^{NE}\right), \mu_0\right] \geq \mathbb{E}\left[u_i(\boldsymbol{\rho}_i) \mid \left(\pi_i, \boldsymbol{\pi}_{-i}^{NE}\right), \mu_0\right] \tag{18}$$

i.e., $\boldsymbol{\pi}^{NE}$ is a Nash equilibrium under ESR if no agent can increase the *expected utility of her returns* by deviating unilaterally from $\boldsymbol{\pi}^{NE}$.

Definition 13 (*Nash equilibrium for Scalarised Expected Returns*) A joint policy $\boldsymbol{\pi}^{NE}$ leads to a Nash equilibrium under SER if for each agent $i \in \{1, \ldots, n\}$ and for any alternative policy π_i:

$$u_i\left(\mathbb{E}\left[\boldsymbol{\rho}_i \mid \left(\pi_i^{NE}, \boldsymbol{\pi}_{-i}^{NE}\right), \mu_0\right]\right) \geq u_i\left(\mathbb{E}\left[\boldsymbol{\rho}_i \mid \left(\pi_i, \boldsymbol{\pi}_{-i}^{NE}\right), \mu_0\right]\right) \tag{19}$$

i.e. $\boldsymbol{\pi}^{NE}$ is a Nash equilibrium under SER if no agent can increase the *utility of her expected returns* by deviating unilaterally from $\boldsymbol{\pi}^{NE}$.

Under non-linear utility functions, it has been shown that the choice of optimisation criterion can alter the set of Nash equilibria [145]. Furthermore, in multi-objective normal form games with non-linear utility functions under SER, NE need not exist [145].

5.4 ϵ-approximate Nash equilibria

An ϵ-approximate Nash equilibrium [110] occurs when an individual agent cannot increase its return by more than an additive $\epsilon > 0$ by deviating from its policy, assuming that all other agents continue to behave in the same way. In other words, an agent will not care to switch her policy, if the obtained gain is too small.

Definition 14 A joint policy $\boldsymbol{\pi}^{NE}$ leads to a ϵ-Nash equilibrium if for each agent $i \in \{1, \ldots, n\}$ and for any alternative policy π_i:

$$V_i^{(\pi_i^{NE}, \boldsymbol{\pi}_{-i}^{NE})} \geq V_i^{(\pi_i, \boldsymbol{\pi}_{-i}^{NE})} - \epsilon \tag{20}$$

ϵ-Nash equilibria can be envisioned as regions surrounding any Nash equilibrium. All the definitions for NE under SER and ESR can also be adapted for the case of ϵ-Nash equilibria by subtracting ϵ from the right side of each inequality.

5.4.1 Correlated equilibria

A correlated equilibrium (CE) is a game theoretic solution concept proposed by Aumann [8] in order to capture correlation options available to the agents when some form of communication can be established prior to the action selection phase. Another way to think about this concept is to envision an external device sampling from a given distribution and providing each agent with a private signal (e.g., a recommended action) at each time-step. Given this private signal, each agent can then independently make a decision on how to act next. For this work we will consider that the signals take the form of action recommendations.

While previously discussed policies define state-based action probabilities independently for each agent, a *correlated policy* σ represents a probability distribution over the joint-action space $\mathcal{A}$ of all the agents in the system (i.e., $Pr(\mathbf{a}|s) \in [0, 1], \forall \mathbf{a} \in \mathcal{A}$). Thus, correlated policies introduce explicit dependencies between the agents' behaviours. Let us define $\boldsymbol{\pi}^{\sigma} = (\pi_1^{\sigma}, \ldots, \pi_n^{\sigma})$ as the joint policy of the agents when following the recommendation provided according to a correlated policy σ.

Definition 15 A correlated policy σ^{CE} is a correlated equilibrium if for each agent $i \in \{1, \ldots, n\}$ with its corresponding policy under σ^{CE}, $\pi_i^{\sigma^{CE}}$, and for any alternative policy π_i:

$$V_i^{\left(\pi_i^{\sigma^{CE}}, \boldsymbol{\pi}_{-i}^{\sigma^{CE}}\right)} \geq V_i^{\left(\pi_i, \boldsymbol{\pi}_{-i}^{\sigma^{CE}}\right)} \tag{21}$$

Thus, a correlated equilibrium ensures that no player can gain additional return by deviating from the suggestions, given that the other players follow them as well.

5.4.2 Correlated equilibria in multi-objective multi-agent settings

Similarly to the Nash equilibria case, solution concepts such as correlated equilibria can be used in scenarios in which each agent derives a different utility from the received reward

vector, i.e., the **team reward individual utility** (Sect. 4.1.2) and **individual reward individual utility** (Sect. 4.2.1) settings. Furthermore, we can also define correlated equilibria from the perspective of the two possible optimisation criteria: ESR and SER, when considering multi-objective multi-agent decision making problems. We will again denote the value of a joint policy $\boldsymbol{\pi}$ for agent i as $\boldsymbol{V}_i^{\boldsymbol{\pi}} = \mathbb{E}\left[\boldsymbol{\rho}_i \mid \boldsymbol{\pi}, \mu_0\right]$, where $\boldsymbol{\rho}_i = \sum_{t=0}^{\infty} \gamma^t \mathbf{r}_{i,t}$.

Definition 16 (*Correlated equilibrium for Expected Scalarised Returns*) A correlated policy σ^{CE} is a correlated equilibrium under ESR if for any agent $i \in \{1, \ldots, n\}$ with its corresponding policy under σ^{CE}, $\pi_i^{\sigma^{CE}}$, and for any alternative policy π_i:

$$\mathbb{E}\left[u_i(\boldsymbol{\rho}_i) \mid \left(\pi_i^{\sigma^{CE}}, \boldsymbol{\pi}_{-i}^{\sigma^{CE}}\right), \mu_0\right] \geq \mathbb{E}\left[u_i(\boldsymbol{\rho}_i) \mid \left(\pi_i, \boldsymbol{\pi}_{-i}^{\sigma^{CE}}\right), \mu_0\right] \tag{22}$$

i.e. σ^{CE} is a correlated equilibrium under ESR if no agent can increase the *expected utility of her returns* by deviating unilaterally from the action recommendations in σ^{CE}.

Definition 17 (*Correlated equilibrium for Scalarised Expected Returns*) A correlated policy σ^{CE} is a correlated equilibrium under SER if for any agent $i \in \{1, \ldots, n\}$ with its corresponding policy under σ^{CE}, $\pi_i^{\sigma^{CE}}$, and for any alternative policy π_i:

$$u_i\left(\mathbb{E}\left[\boldsymbol{\rho}_i \mid \left(\pi_i^{\sigma^{CE}}, \boldsymbol{\pi}_{-i}^{\sigma^{CE}}\right), \mu_0\right]\right) \geq u_i\left(\mathbb{E}\left[\boldsymbol{\rho}_i \mid \left(\pi_i, \boldsymbol{\pi}_{-i}^{\sigma^{CE}}\right), \mu_0\right]\right) \tag{23}$$

i.e. σ^{CE} is a correlated equilibrium under SER if no agent can increase the *utility of her expected returns* by deviating unilaterally from the given action recommendations in σ^{CE}.[9]

It has also been shown that the set of correlated equilibria will be altered, depending on the optimisation criteria used [145]. Furthermore, under SER, with non-linear utility functions, in multi-objective normal form games, CE need not exist when taking the expectation over all the possible correlation signals [145].

5.5 Coalition formation and stability concepts

A different perspective on multi-agent decisions is that taken by *cooperative game theory* [35]. Cooperative game theory studies settings where binding agreements among agents are possible. A central problem is therefore that of *coalition formation*, i.e., finding (sub)groups of agents that are willing to make such a binding agreement with each other. In the models in cooperative game theory, the utility for each agent is directly derived from the coalition the agents end up in, however, one can imagine that under the hood, the coalition works together cooperatively (based on their binding agreement) in a sequential decision problem that results in this utility. We further note, that the word cooperative does not imply team utility; typically, the agents will have their own utility functions. Hence, the solution concepts from cooperative game theory apply to the individual utility settings.

To illustrate the solution concepts for multi-objective cooperative game theory, we use the *multi-criteria coalition formation game (MC2FG)* [70,162,163]. Such a game consists of a set of agents, $\mathcal{N}$, each with their own utility function $u_i(\mathbf{q})$, and a quality/reward function

[9] When considering a CE-based approach, an agent is able to calculate her expected return given one correlation signal, but also an expected return given all the possible signals. This allows one to define two variants for CE under SER: the single-signal CE (when agents have multiple interactions under the same given signal) and multi-signal CE (when agents receive a new signal after every interaction) [145]. For this work we define the more general case of multi-signal CE.

$\mathbf{q}(S)$ that maps each possible subset, i.e., coalition, of the agents $S \in \mathcal{N}$ to a value or quality vector, that each agent in that coalition will receive. That is, we are in an individual utility setting.

Definition 18 A multi-criteria coalition formation game (MC2FG) is a triple $(N, q, \mathcal{U})$ where N is a finite set of agents, $\mathbf{q} : 2^N \rightarrow \mathbb{R}^d$ is a vector-valued reward function that represents the quality $\mathbf{q}(S)$ of a subset, i.e. coalition, of agents $S \subseteq N$, and $u_i \in \mathcal{U}$ are the utility functions for each agent $i \in N$.

The MC2FG is a useful model to study for multi-objective multi-agent systems. Specifically, if in a multi-agent system with multiple objectives, the agents need to form coalitions to cooperate to gain a value vector, the most straightforward case is a MC2FG, i.e., given the coalition the value vector can exactly be predicted independently of the other coalitions, but agents can have different preferences between possible value vectors. Therefore, MC2FGs form a minimal model to study the feasibility of contract negotiations between agents in multi-objective multi-agent decision making.

The goal in an MC2FG is to find a partition, ψ, of agents into coalitions that are stable. That is, the coalitions will not break apart. For this notion of stability, there are multiple possible versions, from strong to weak: *core stability*, *Nash stability*, and *individual stability*.

We denote the coalition (subset of agents) which agent i is in according to ψ as $\psi(i)$. A partition ψ is *individually rational* if no agent strictly prefers staying alone to their own coalitions, i.e. $\forall i : u_i(\mathbf{q}(\psi(i)) \geq u_i(\mathbf{q}(\{i\}))$.

Definition 19 A coalition $S \subseteq N$ is said to *block* a partition ψ if every agent strictly prefers S to $\psi(i)$, i.e., $\forall (i \in S) : u_i(\mathbf{q}(\psi(i)) < u_i(\mathbf{q}(S))$.

Definition 20 A partition ψ of N is *core stable (CR)* if no (non-empty) coalition $S \subseteq N$ blocks ψ.

Beside CR, there are two key stability concepts that represent immunity to deviations by individual players. An agent i, wants to deviate from $\psi(i)$ to another coalition in ψ, S, if it prefers $S \cup \{i\}$ to $\psi(i)$, i.e., $u_i(\mathbf{q}(\psi(i)) < u_i(\mathbf{q}(S \cup \{i\}))$. A player $j \in S$ would accept such a deviation if it prefers $S \cup \{i\}$ to S, i.e., $u_j(\mathbf{q}(S) \leq u_j(\mathbf{q}(S \cup \{i\}))$.

Definition 21 A partition ψ is *Nash stable (NS)* if there are no NS-deviations (Definition 22) for any agent i, from its coalition $\psi(i)$ to any other coalition $S \in \psi$ or to $\emptyset$.

Definition 22 A deviation of i from $\psi(i)$ to S is an NS-deviation if i wants to deviate from $\psi(i)$ to S.

Definition 23 A partition ψ is *Individually stable (IS)* if there are no IS-deviations (Definition 24) for any agent i, from its coalition $\psi(i)$ to any other coalition $S \in \psi$ or to $\emptyset$.

Definition 24 A deviation of i from $\psi(i)$ to S is an IS-deviation if it is an NS-deviation and all players in S accept it.

Every single-criterion coalition formation game has at least one partition that is core stable and individually stable [70]. However, this is not necessarily so in the multi-objective case. This is because in a single-objective coalition formation game, the utility of a coalition is the same for each agent, i.e., the scalar quality/reward of the coalition. However, in the multi-objective case, all agents that are in a coalition S receive the same reward vector $\mathbf{q}(S)$, but they may value these vectors differently. In fact, Igarashi and Roijers [70] show that MC2FGs do not necessarily have core, Nash, nor individually stable partitions by counter-example resulting in the following Theorem:

Theorem 1 *For any positive integer n and for any $0 < \varepsilon < 1$ there exists an MC2FG $(N, \mathbf{q}, \{\mathbf{w}_i : i \in N\})$, where $\mathbf{w}_i$ is the weights vector for the linear utility function of agent i, which admits neither a core nor individually stable partition, where the number of players $|N| = n$, the number of criteria $m = 2$, and $|w_{i,k} - w_{j,k}| \leq \varepsilon$ for any $i, j \in N$ and either objective (k).*

So, this theorem implies that even when the number of objectives is smaller than the number of agents, and the difference between the utility functions (even if they are linear) is arbitrarily small, stable partitions do not need to exist. This has important consequences for multi-objective multi-agent systems in general, as MC2FGs are such a minimal model of finding cooperative subsets of agents that could contractually agree on a value vector. Because no stable solutions need to exist, such contract negotiations could go on forever (agents repeatedly switching between coalitions before signing the contract), if all agents just aim to optimise their individual utilities. We believe this means that a thorough investigation of (the compatibility of) negotiation techniques for various multi-objective multi-agent decision problems on the basis of coverage sets, under different optimisation criteria (i.e., ESR versus SER) is required. Furthermore, the fact that the stability of coalitions cannot be guaranteed could have a strong impact on future *interactive approaches*[10] as well. While the prospects of such interactive approaches seem good, as [70] have shown that individually stable coalitions can often be found interactively under linear utility functions in MC2FGs, it is not clear what will happen for non-linear utility functions under SER or ESR, or in learning settings where the estimated value vectors of different joint policies of changing coalitions, may change.

5.6 Social welfare and mechanism design

In this section, we have so far taken the position of the individual agents. However, we can also take a system perspective, i.e., we can look at what the socially desirable outcomes of a multi-agent decision problem would be. In Sects. 4.1.3 and 4.2.2, we have looked at the execution phase of such settings and defined the social welfare function, i.e., a function that should be maximised if we want to find socially desirable outcomes.

In game theory, the field of mechanism design takes the system's perspective for multi-agent decision problems: taking an original decision problem where the agents have individual reward functions that are unknown to the other agents and the "owner" of the game, as well as a social welfare function as input, the aim is to design a system of additional payments that would a) force the agents to be truthful about their individual utilities, and b) leads to solutions that are (approximately) optimal under the social welfare function.

In single-objective multi-agent decision problems, the individual utilities of the agents are simply the individual (expected) (cumulative) rewards that the agents receive. The agents can be assumed to know these rewards, and act accordingly. This is for example the case in public tenders, where different companies know their own costs and profit margins of their possible proposals, but do not broadcast this information to others. In multi-objective settings, the situation is more complex, as the individually received rewards determine the individual utilities via individual private utility functions. These utility functions can have different properties. In general, it might even be very hard, or even impossible to articulate these functions, so being "truthful" about their utilities might be infeasible from the get-go.

Nevertheless, it is possible to design mechanisms for some multi-objective multi-agent problems if the individual utilities can be articulated. First, we observe that if the utility

[10] Interactive approaches intertwine preference elicitation and learning about the decision problem [138,139].

functions are linear, the inner product with weights distributes over all expectations. Hence, it is possible to even design mechanisms that are agnostic about the weights, compute a convex coverage set (see Sect. 5.2.2) of possibly socially desirable outcomes, and choose the weights a posteriori. This enables the designer/owner of the decision problem to make an informed decision about which weights to use. For example, in a public tender for traffic maintenance by Scharpff et al. [136,147], the objectives of costs and traffic hindrance should both be minimised. Because of mechanism design, all agents need to be truthful; whatever weights (and resulting penalties) are put on traffic hindrance, it is in the best interest of the agents to be truthful about their costs, making it possible for the owner of the game to assume that given the mechanism, all agents will be fully cooperative, solve the problem as an MOMMDP, and choose the weights a posteriori.

For specific cases of non-linear utility functions, it is also possible to devise mechanisms. For example, Grandoni et al. [61] assume individual utility functions with a primary objective that should be maximised, and other objectives that need to achieve at least a threshold value. The utility is the value of the first objective in the case that all thresholds are met, but negative infinity if the thresholds are not met. They show that for such cases, effective mechanisms can be designed, and solutions can be found within a reasonable amount of time.

An interesting and different approach to social welfare is taken by Mouaddib et al. [105], who cast a decentralised sequential multi-agent problem with individual (scalar) reward functions as a multi-objective problem. Specifically, besides its main objective an agent will model its positive impact on the group as well as the nuisance it causes to other group members as separate objectives. Even though this work provides no strong guarantees, the authors show empirically that these additional objectives in combination with a social welfare function can lead to good emergent group behaviour in very hard—decentralised partially observable multi-agent—decision problems.

5.7 Other solution concepts

The concepts discussed so far do not form in any way an exhaustive list for what constitutes a solution in a MOMAS. We briefly present below a few other possible solution concepts that have been discussed in the literature.

Multi-criteria or multi-objective games [19] have been extensively discussed in the literature, together with several possible solution concepts that we shortly introduce below. An early discussion on how to extend equilibria concepts from single-objective games to multi-objective settings is presented by Shapley and Rigby [150], where the concepts of weak and strong equilibria are proposed as extensions of NE. These concepts are defined using vector domination and thus are called *Pareto(-Nash) Equilibria* and have been vastly discussed and analysed in many different settings [23,24,87,175,176,184,187,189].

Continuing the game theoretic perspective, Kawamura et al. [76] extends the concept of *evolutionary stability* for multi-objective games. Borm et al. [22] define the idea of *perfect equilibrium points*, based on the perfectness concept of Bielefeld [18]. Voorneveld et al. [174] introduce the notion of *ideal Nash equilibrium*, i.e., players have no incentive to deviate regardless of weights chosen over the objectives. Patrone et al. [116] extend the theory of potential games to multicriteria games and investigate the existence of pure (approximate) Pareto equilibria from this perspective. In the context of non-cooperative multicriteria games, Pusillo and Tijs [123] use improvement sets to propose the *E-equilibrium* as a solution concept and prove its existence in the case of potential games. Finally, considering coalition formation processes, Pieri and Pusillo [118] formalise multicriteria partial cooperative games (or multi-

objective environmental games) and define a corresponding solution concept: the *coalitional Pareto equilibrium*.

Ghose and Prasad [59] have studied multi-criteria games by also taking into account the security level against strategy deviations of the opponent and have proposed the concept of Pareto-optimal security strategies. On a related track, Fernández et al. [50] propose goal games, where a player can set a minimal goal to achieve in each objective and define the solution concept of G-goal security strategies.

Cyclic equilibria [52,101,192] have been proposed as a solution concept for games where no stationary equilibrium exists. A cyclic equilibrium is a non-stationary joint policy where agents have no incentive to deviate unilaterally [192]. Cyclic equilibria cycle repeatedly through a set of stationary policies. Similar to ϵ-NE, an ϵ-correlated cyclic equilibrium is defined as a situation where no agent can improve its value by more than ϵ at any stage by deviating unilaterally [192].

6 Algorithmic approaches and applications

In this section, we survey related work on algorithmic approaches to MOMAS, as well as applications of multi-objective multi-agent decision making. The works we survey are organised into three broad categories: those which aim to derive coverage sets are discussed in Sect. 6.1, those which aim to apply stability and equilibria concepts are discussed in Sect. 6.2 and finally methods which employ mechanism design are discussed in Sect. 6.3. Within these categories, the works surveyed are further classified according to our taxonomy on the basis of the reward and utility functions used. A summary of the papers which fall under each category discussed below is presented in Table 5.

6.1 Coverage sets

6.1.1 Team reward: team utility (TRTU)

Multi-objective coordination graphs (MOCoGs)[11] are one of the most studied models for cooperative multi-objective multi-agent systems, and in particular for team reward team utility. One reason for this is that it is the simplest model to express and exploit *loose couplings*, i.e., the fact that in multi-agent systems the rewards can often be factorised into a sum over small components, i.e., local reward functions, that depend on small (but possibly overlapping) subsets of agents. However, finding suitable joint actions in a MOCoG is also key to finding, e.g., coverage sets for MOMMDPs (as is also the case for single-objective CoGs and MMDPs [79,80]).

In MOCoGs, a lot of research focuses on finding (approximate) Pareto coverage sets (PCSs), using various algorithmic approaches. These approaches often extend single-objective methods by adapting the inner workings of such methods to be able to handle sets of value vectors rather than single scalar values. Examples of such methods are multi-objective bucket elimination (MOBE) [140,141], also known as multi-objective variable elimination (MOVE, which is the more common name in the planning and reinforcement learning com-

[11] Because the MOCoG model is a flexible multi-objective graphical model that can be used for many types of problems, and has been used by many research communities, the MOCoG is known under many different names. These include: multi-objective weighted constraint satisfaction problems (MO-WCSPs) [141] and Multi-objective Constraint Optimisation Problems (MOCOPs) [97,99].

Table 5 Summary of algorithmic approaches and applications to MOMAS, classified according to solution concept used and our taxonomy

Solution concept	Reward and utility	Papers
Coverage sets (Sect. 6.1)	Team reward–team utility	[2,11,21,41,68,71,79,80,90–92,94–99,134–137,140–143,179,185,186]
	Team reward–individual utility	[7,57,102,117,155]
	Individual reward–individual utility	[9,29,46,47,170]
Equilibria and stability concepts (Sect. 6.2)	Team reward–individual utility	[70,84]
	Individual reward–individual utility	[26,88,124,127,164]
Mechanism design (Sect. 6.3)	Team/individual reward–social choice	[4,28,31,49,61,82,105,111,120,125]

munities), which solves a series of local sub-problems to eliminate all agents from a MOCoG in sequence, by finding local coverage sets as best responses to neighbouring agents. Other such methods include multi-objective Russian doll search [142], multi-objective (AND/OR) branch-and-bound tree search [97,98,143] using mini-bucket heuristics [97,141], Pareto local search [71], and multi-objective max-sum [41]. Many of these papers note that PCSs can grow very large very quickly, making finding exact PCSs infeasible. Therefore computing bounded approximations [98] can be necessary.

On the other hand, Roijers et al. [135–137] compare the computational and memory complexity of computing convex and Pareto coverage sets (CCSs and PCSs). They observe that the size of PCSs typically grows much faster with the number of agents in a MOCoG than the size of CCSs, and that often CCSs suffice, e.g., in the case that mixture policies are allowed (Sect. 5.1). It can therefore be highly preferable to focus on finding CCSs, especially in problems with many agents. They propose several methods to do so with different computation-memory trade-offs. Specifically for finding CCSs, they propose and compare inner loop methods to outer loop methods, i.e., methods that construct CCSs by iteratively solving scalarised instances of the multi-objective decision problem. Outer loop methods are more memory-efficient, and significantly faster for smaller numbers of objectives, while inner loop methods are faster for many objectives. Finally, they note that ε-approximate CCS can efficiently be computed using outer loop methods. Anytime approximations to CCSs can also be effectively constructed using an outer loop method that employs variational (inference) methods [134].

Wilson et al. [179] consider methods to compute coverage sets for MOCoGs when more information about the possible utility function(s) is available. They assume that along with the standard notion of the shape of the utility function,[12] they are provided with a set of preferences that users expressed a priori. They integrate these preferences into AND/OR branch-and-bound, and show that this can often lead to much more efficient computation than would be required to compute a PCS. To achieve this, they pose additional constraints on the utility function, that are only fulfilled by linear utility functions. Therefore, doing the same for arbitrarily shaped utility functions is still an open problem.

Multi-objective multi-agent MDPs (MOMMDPs)—in the literature often referred to as cooperative multi-objective stochastic games (cooperative MOSGs)—are another frequently used model for cooperative multi-objective multi-agent decision making problems. Some recent works have sought to derive coverage sets in MOMMDPs using reinforcement learn-

[12] Or, in their original paper, the equivalent concept of domination.

ing or evolutionary algorithms (e.g. [90–92,95,185,186]). As in single-objective MMDPs, learning joint policies which coordinate agents' actions to get the desired outcome(s) in MOMMDPs is a difficult problem. When individual agents learn using the system reward and the same utility function (i.e. TRTU), it is difficult for any one agent to evaluate how its actions affected the system utility, due to the effect of the other agents in the system. This is referred to as the credit assignment problem in the single-objective MAS literature. Reward shaping is one solution which has been proposed to address this problem, where the reward which is usually received from the environment is augmented with an additional shaping term, with the goal of providing a more informative reward signal to the agents in a MAS. Specific forms of reward shaping which have been applied to cooperative MOMAS include difference rewards (D) [182] and potential-based reward shaping (PBRS) [107].[13]

Yliniemi [186] and Yliniemi and Tumer [185] present the first work that considers the use of reward shaping in a cooperative MOMARL setting. Their work compares the effectiveness of the difference reward with that of two typical MARL reward structures: local rewards (L) and global rewards (G). Experimental work in a multi-objective congestion problem, and a multi-objective robot coordination problem confirms that D can improve MOMARL performance when compared to L or G, both in terms of learning speed and the quality of the set of non-dominated solutions found. Yliniemi and Tumer [185] also demonstrate that D can be used effectively with multi-objective evolutionary algorithms, in a series of experiments where it is applied to shape the fitness function of the Elitist Non-dominated Sorting Genetic Algorithm-II (NSGA-II). Mannion et al. [96] evaluate the effect of D in an electricity generator scheduling problem. Mannion et al. [92] provide a theoretical analysis of $PBRS$ in single- and multi-agent MORL settings, demonstrating that the Pareto relation between (joint) policies is preserved when $PBRS$ is used. Mannion et al. [94] also provide a theoretical analysis of D in MOMMDPs, demonstrating that the relative values of actions (and therefore the Pareto relation between actions) is preserved when D is applied in MOMMDPs. A comprehensive analysis of the effects of both D and $PBRS$ when generating coverage sets via learning in cooperative MOMARL settings is presented in [90,91].

On the application side, Ahmad et al. [2] consider the problem of multi-core processing, or, more specifically, energy aware task allocation for optimising energy use versus performance. They employ a cooperative game theory perspective and transform the problem into a max-max-min one to generate solutions for different energy-time trade-offs. Bone and Dragićević [21] are interested in the setting of natural resource management, where each agent represents a forest harvesting company. Agents need to learn how to harvest wood in order to maximise economic profit and minimise ecological impact. They use a step utility function to transform the problem into a single-objective one and learn an optimal policy as independent reinforcement learners. Babbar-Sebens and Mukhopadhyay [11] consider a water resource management system modelled as a multi-objective game, where the players use a simple reinforcement learning or genetic algorithm approach in order to find a set of solutions corresponding to various linear utility functions. Focusing on traffic optimisation, Houli et al. [68] develop the multi-RL algorithm, a multi-agent reinforcement learning approach which selects an optimisation objective depending on the real-time traffic conditions.

[13] Although individualised reward shaping implies that each agent receives a different reward, we have classified these works under the TRTU setting as all agents use the same utility function and the individual shaped rewards are still aligned with the global (system) rewards. Reward shaping might also be useful in combination some of the other settings in our taxonomy and solution concepts discussed in Sect. 5, although only the TRTU setting with coverage sets has been explored to date.

6.1.2 Team reward: individual utility (TRIU)

By far the most papers that study coverage sets motivate this from a team reward and team utility setting. However, as we noted in Sect. 5.2.1 coverage sets in general can be motivated from a team reward but individual utility perspective as well. Specifically, when the objective correspond to quantities that all agents are interested in, dominated solutions (Sect. 5.2) are suboptimal for all agents. Hence, computing the coverage set is to return the set of solutions on which agents may disagree which of these solutions is optimal, making it the ideal starting point for a negotiation. As such all algorithmic approaches that compute a coverage set mentioned in Sect. 6.1.1 also apply to this setting.

As previously mentioned, it is also possible to consider varying levels of altruism by considering the (single-objective) rewards of each agent as an objective. For example, Aoki et al. [7] model a multi-stage flow system as a MAS, where each agent (i.e., service centre) represents a different objective. They use a distributed reinforcement learning framework and propose a bi-directional decision making mechanism to address the multi-objective nature of the problem. Furthermore, it is possible to explicitly impose fairness between these objectives (i.e., interests of different agents). A rather loose condition for fairness is that a joint policy is not Lorenz dominated (Definition 9). Imposing this condition leads to the Lorenz optimal set as coverage set (see Sect. 5.2.4). Such a Lorenz optimal set can then be input for a negotiation of which policy to execute in practice. Lorenz optimal sets have been studied in different problem domains [57,102,117], but the multi-agent motivation in terms of what it means for the utility of the individual agents is often not made explicit. Furthermore, we believe that it would be an opportunity to investigate the full cycle, i.e., to study the negotiation phase resulting from producing a Lorenz optimal set, and comparing it to negotiations with a Pareto or convex coverage set.

Finally, coverage sets can also be used in asymmetric games, where one agent has multiple objectives, while the other does not. For example, Sinha et al. [155] study a Stackelberg game, where a government agent is interested in maximising tax revenue, while minimising environmental impact, and a mining company agent is only interested in profit. The government agent computes a convex coverage set to find different trade-offs between its objectives, while the latter agent just computes best responses (maximising its single objective) to the resulting different behaviours of the government agent.

6.1.3 Individual reward: individual utility (IRIU)

Investigating multi-objective games, Avigad et al. [9] propose an evolutionary search algorithm for finding the Pareto set of strategies for a player, given each possible strategy of an opponent. Also looking at competitive multi-objective games, Eisenstadt et al. [47] proposes a co-evolutionary algorithm for finding solutions simultaneously for both players, under worst-case assumptions, given that the opponent's preferences are unknown.

Brys et al. [29] apply MOMARL to a traffic signal control problem, where each intersection in a 2×2 grid is controlled by an individual agent. Their work demonstrates that rewarding agents with a linear scalarised combination of delay and throughput can improve delay times when compared to agents rewarded using delay alone. However, their approach uses local rewards (i.e. each agent is rewarded based on conditions at its assigned intersection only, and hence this work is classified as IRIU), and does not make any attempt to explicitly encourage coordination between the agents.

Dusparic and Cahill [46] propose the Distributed W-Learning algorithm, an RL-based approach for multi-policy optimisation in collaborative multi-agent systems, such as urban

traffic. Each agent represents a traffic light at an intersection and implements a set of local and remote policies (i.e., involving its neighbours). Even though the agents here have possibly conflicting goals and receive an individual reward, at every time-step they exchange information with their neighbours regarding their current states and rewards, allowing them to develop, if necessary, a cooperative behaviour.

Van Moffaert et al. [170] apply MORL to a multi-objective multi-agent smart camera problem. They develop an adaptive weight algorithm (AWA) which is used to choose the weighting between the two system objectives when linear scalarisation is applied to individual reward vectors for each camera agent in the system. The AWA algorithm is found to improve learning speed, obtaining solutions with a better spread in the objective space, when compared with other weight selection methods that were tested.

6.2 Equilibria and stability methods

6.2.1 Team reward: individual utility (TRIU)

Lee [84] takes a game theory perspective on the reservoir watershed management domain and develops for this purpose a multi-objective game theory model. The goal in this case is to balance between maximising economic gain and minimising environmental impact. Each player represents a different objective and multiple bargaining rounds are used in order to arrive at a Nash equilibrium.

As previously mentioned, Igarashi and Roijers [70], apply the team reward individual reward setting in multi-objective coalition formation games. We believe that, even though they prove that stable coalitions need not exist in these settings, it is worth developing new methods for such settings, as the utility functions of the individuals in real-world teams are seldom exactly aligned.

6.2.2 Individual reward: individual utility (IRIU)

Qu et al. [124] examine a multi-objective non-zero sum game, where each agent has a set of weights for each objective (i.e., linear utility function where the exact weights are not known). They propose an approach for finding a robust weighted Nash equilibrium. In the context of multi-objective games with incomplete information, Yu and Liu [188] propose and study robust multiple objective games, where players try to maximise their worst-case expected returns, and define corresponding robust equilibria for this setting.

Brown et al. [26] propose the multi-objective security game, where each objective represents a threat from a different type of attacker. When a defending agent needs to face multiple attacker types, instead of offering one solution to the end-user, like in the case of Bayesian security games, they propose to return a Pareto front that would explicitly show the trade-offs between the possible strategies.

Taylor et al. [164] propose Parallel Transfer Learning (PTL) as a mechanism to accelerate learning, by sharing experience among agents. PTL is tested on a multi-objective multi-agent smart grid problem, and is found to improve learning speed and final performance when compared to agents learning without PTL.

Madani and Lund [88] consider a Monte Carlo Game Theory approach for stochastic multi-criteria decision making settings, such as water resource management. They propose the use of Monte Carlo simulations to map the problem to deterministic settings transformed then into strategic games, solved using non-cooperative stability methods.

Considering the equilibrium design problem, Rettieva [127] discusses dynamic discrete-time settings, with agents sharing or exploiting a common resource, while having different utilities to optimise (e.g., harvesting and selling fish). An equilibrium is found as a solution of a Nash bargaining scheme [128], thus defining a possible solution concept for the setting of dynamic multi-objective games.

6.3 Social welfare and mechanism design

In the context of mechanism design, where we strive to optimise a social welfare function, the distinction between team versus individual rewards will not carry anymore an important significance. We will thus survey work on mechanism design for multi-objective multi-agent setting without making a distinction between then two possible reward settings.

On the line of reward engineering in multi-objective congestion problems, Ramos et al. [125] consider the route choice traffic problem and develop a reward signal based on the marginal cost tolling mechanism. This allows one to reach a system optimum performance, even when agents have heterogeneous preferences with respect to travelling time and monetary costs. To model the agent preferences they use a linear utility function that attributes different weights to the considered objectives. They show that if all the agents use their proposed learning approach, i.e., Generalised Toll-based Q-learning, they will converge to a system optimum.

Mouaddib et al. [105] develop a multi-objective multi-agent planning framework in the form of a regret-based algorithm to improve the resulting social behaviour for the considered vector-valued Dec-MDP. This framework assumes a true objective—the regular Dec-MDP objective—and then adds two extra objectives, i.e., the positive and negative impact an agent has on the team. This leads to social behaviour, and therefore better functioning teams. Such addition of artificial objectives to attain better policies can be seen as a form of multi-objectivisation [28]. Multi-objectivisation to improve team behaviour through social welfare is an interesting research direction that we believe needs to be further explored.

Grandoni et al. [61] assume individual utility functions with a primary objective that should be maximised, and other objectives that need to achieve at least a threshold value. This type of utility function is similar to constrained MDPs [6] in the single-agent literature, and can be seen as a special case of multi-objective MDPs, i.e., one where the utility has this shape. They show that truthful mechanisms exist that allow solutions to be found in a reasonable amount of time.

Pla et al. [120] study auctions in which agents make bids that lead to a value-vector in different objectives. For such auctions the social welfare must be optimised via a mechanism that determines the payments w.r.t. the bid. These payments, together with the costs of the bids, constitute the individual utilities. They show that the social welfare function must obey three properties for a mechanism to be possible: it must be real-valued and monotonically increasing in all objectives—as with the utility functions for any multi-objective decision problem—and it must be bijective, i.e., given the bid attributes values and the result of an evaluation function, the cost corresponding to a bid can take only one possible value. This is necessary to be able to calculate the payments in a mechanism.

Mechanisms for multi-objective decision problems are used in a variety of applications, for example: Buettner and Landes [31] apply mechanism design in order to match employers looking for temporary workers, to workers looking for contracts. Because these contracts have several aspects that may lead to utility, such as hourly salary, benefits, sick pay or overtime premiums, this is a multi-objective setting; Fard et al. [49] use mechanism design in for cloud work-flow management, where the agents have costs and completion time as objectives

when trying to schedule tasks; and Kruse et al. [82] study multi-objective airline service procurement using mechanisms. Aleksandrov and Walsh [4] consider an online allocation setting, with food items arriving one by one and having to be distributed to agents with different utilities/preferences. They analyse different allocation mechanisms and investigate computation and counting issues regarding their Nash equilibria and properties such as envy-freeness and number of allocated items, when agents have various type of utilities (i.e., binary, 0/1/2).

Nwulu et al. [111] use a multi-objective perspective for a power generation system that minimises fuel costs and minimises emissions, as well as adhering to constraints such as a maximum load. In this system, costumers are agents, which can be incentivised to adjust their consumption patterns. They show how to optimise for different (linear) weightings of the objectives, while calculating the incentives that should be put to the consumers.

7 Conclusions and new horizons

In this paper, we analysed multi-objective multi-agent decision problems from a utility-based perspective. Starting from the execution phase and working backwards, we derived when different solution concepts apply. We surveyed the literature on the applicable solution concepts, methods that compute such solutions, and practical applications. The taxonomy of problem settings and solution methods we propose structures this relatively new line of research from the perspective of user utility, and it is therefore our hope that this survey helps to place existing research papers in the larger multi-objective multi-agent decision problem context, and informs and helps to inspire further research. To this end, in this last section we discuss what we consider to be the key new horizons and open problems in the field of multi-objective multi-agent decision making.

7.1 Optimisation criteria and solution concepts

In future work, it would be worthwhile to further explore the link between multi-objective optimisation criteria (ESR vs. SER) and solution concepts for MOMAS with non-linear utility functions. The body of literature on theory and experimental results is limited up until this point with respect to this topic, apart from an initial analysis conducted by Rădulescu et al. [145] for multi-objective normal-form games under SER which proves by example that Nash equilibria need not exist, and that correlated equilibria can exist under certain conditions. This line of research should be extended to sequential settings (e.g., MOSGs), as well as to consider the other solution concepts discussed in Sect. 5. For example, algorithms to construct (approximate) coverage sets (Sect. 5.2.3) and Lorenz optimal sets (Sect. 5.2.4) under ESR merit further investigation. In multi-objective coalition formation games, open questions remain about how stable outcomes may be reached (Sect. 5.5); in this setting an investigation of negotiation techniques on the basis of coverage sets, under different optimisation criteria (i.e., ESR versus SER) would be worthwhile. The differences between negotiation phases resulting from producing a Lorenz optimal set and negotiations resulting from a Pareto or convex coverage set should also be explored. Further work also needs to be done on generalising the concept of a coverage set to the individual rewards setting (Sect. 5.2.1).

It is also possible that not all agents in a MOMAS would choose the same optimisation criterion; it is currently not known how mixing optimisation criteria would affect the collective

behaviour of MOMAS in practice. Developing stronger theoretical guarantees, as well as a better understanding of these issues using comprehensive empirical studies represents an important research direction one can pursue.

7.2 ESR planning and reinforcement learning and SER game theory

For multi-objective multi-agent decision problems, there is a large discrepancy between the game theory literature and the planning and reinforcement learning literature. The former focuses mostly on ESR settings, while the latter focuses almost exclusively on SER settings. Perhaps this is an artefact of the single-shot nature of most game-theoretic models and the sequential nature of planning and reinforcement learning models. However, as we recently noted [145], both optimality criteria are well-motivated, as they apply to different real-world decision problem settings, and lead to vastly different theoretical results as well as practical solutions in single-shot settings with non-linear utility functions. The same argument can be made for sequential decision making settings. Therefore, we believe that analysing sequential decision problems under ESR, and game-theoretic (single-shot) models under SER, is both highly important and almost entirely unstudied.

7.3 Opponent modelling and modelling opponent utility

In single-objective reinforcement learning, an agent often aims to learn a model of the other agents' behaviours and uses this model when selecting or learning best responses. In multi-objective multi-agent settings, a good and possibly even sufficient predictor for this behaviour would be the utility function of the other agents. Therefore, explicitly estimating the utility functions of the other agents in a MOMAS is likely to be important in future research. In team-utility settings, i.e., when there is only one true utility function, Zintgraf et al. [194] show that this utility function can be estimated effectively by posing preference queries, and using monotonicity information about the utility function. However, this assumes that there is a single user to pose such queries to, who "owns" the utility function. In multi-agent settings, there may be multiple utility functions, and users, that have conflicting interests. Furthermore, if they can benefit from not revealing their true preferences, they might lie. This motivates two important open questions for future research: can we design mechanisms that force agents/users to be truthful about revealing their preferences over value/return vectors? And if not, can we estimate their utility functions solely from the agent's behaviour in a multi-objective decision problem? Albrecht and Stone recently published a comprehensive survey on opponent modelling for single-objective MAS [3]; many of the methods they surveyed could plausibly be adapted or extended to model other agents' intentions and utilities in MOMAS.

7.4 Closing the loop

From our analysis of prior works on MOMAS in Sect. 6, it is apparent that the field to date has been quite fractured; some settings from our taxonomy (e.g. TRTU) have received much more attention than others, and a limited number of authors are currently active in the field. Consequently, there is not yet a standardised approach to identifying and completing all the steps which are necessary for a successful application of multi-objective multi-agent decision making.

We propose the following sequence of steps: selecting an appropriate decision making model from among those listed in Sect. 3, identifying which setting from our taxonomy the decision making problem fits into (Sect. 4), defining the environment including the state and action spaces and reward and utility schemes, selecting an appropriate solution concept (Sect. 5), completing the planning/learning and/or negotiation phase, executing the policies found and finally evaluating the outcome by measuring the utilities achieved. We hope that by following these main steps, it will become easier for other researchers to apply multi-objective multi-agent decision making theory in their work.

7.5 Interactive approaches

In most of the survey we have assumed that there is a separate learning or planning phase first, then a policy selection and/or negotiation phase, and finally an execution phase. However, it is also possible to elicit preferences from users while planning or learning, leading to a combined planning/learning and preference-elicitation/negotiation phase. This has been studied in multi-objective single-agent systems [138,139] and in cooperative game theory [70]. Furthermore, the incorporation of preference information during planning [179] can also be seen in this line. This previous research however focuses either on eliciting preferences with respect to a team utility function [138,139,179] or individual utilities in the context of checking whether deviations from current coalitions are desired [70]. Parallel negotiation and learning or planning is, to our knowledge, still unexplored territory. Such interactive querying approaches could be helpful for mechanism design in settings where individual reward vectors are common knowledge, but agent preferences are (partially) unknown.

7.6 Deep multi-objective multi-agent decision making

Most of research discussed in this survey so far considers domains with discrete states and actions. For challenging real-world applications of MOMAS, it will be necessary to develop methods which consider continuous or high-dimensional state and action spaces. Considerable progress has been made on developing single-objective deep RL methods for single-agent decision making. In the last couple of years, interest in deep MORL has intensified, although primarily in single-agent settings (see e.g. [1,56,75,104,108,129,153,160,161]). Very recently, single-objective deep multi-agent RL has received considerable attention as well [53,55,65,86,126,144,158,191]. An important next step is therefore to extend existing deep RL methods for multi-objective multi-agent decision making settings.

7.7 Applications and broader applicability

Now that we have identified the different settings and solution concepts which are relevant to MOMAS, significant opportunities exist to revisit problems that were initially modelled as single-objective multi-agent decision problems using a multi-objective perspective. This could provide a richer set of potential solutions for cooperative MAS using the concept of coverage sets (Sect. 6.1), or potentially improve performance by considering additional synthetic objectives which represent sub-tasks explicitly (a process known as multi-objectivisation [27,28]). One promising direction for future work is to use multi-objectivisation to improve team behaviour through social welfare. The possibility also exists to use MORL techniques to develop agents which may be tuned to adopt a range of different behaviours during deploy-

ment in MAS (e.g. cooperative vs. competitive), as recently demonstrated by Källström and Heintz [75], or even creating populations of agents that develop effective behaviours against a large range of opponents [13].

Another interesting and related line of work concerns internal reward optimisation using population-based training, with the goal of obtaining a robust team of agents able to deal with a variety of environments and opponents, as presented by Jaderberg et al. [72]. On the same line of work, Liu et al. [85] study how coordination can emerge in a competitive setting, through reward shaping. They propose to linearly combine a series of rewards in order to shape the original sparse signal. They then proceed to use population based training to determine how to best weigh each component for the reward shaping process, i.e, in multi-objective terms, to learn the utility function that outputs the desired cooperative strategy.

References

1. Abels, A., Roijers, D. M., Lenaerts, T., Nowé, A., & Steckelmacher, D. (2019). Dynamic weights in multi-objective deep reinforcement learning. In *ICML 2019: Proceedings of the 36th international conference on machine learning* (pp. 11–20).
2. Ahmad, I., Ranka, S., & Khan, S. U. (2008). Using game theory for scheduling tasks on multi-core processors for simultaneous optimization of performance and energy. In *2008 IEEE international symposium on parallel and distributed processing* (pp. 1–6). IEEE.
3. Albrecht, S. V., & Stone, P. (2018). Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence*, *258*, 66–95.
4. Aleksandrov, M., & Walsh, T. (2017). Pure nash equilibria in online fair division. In *Proceedings of the twenty-sixth international joint conference on artificial intelligence, IJCAI-17* (pp. 42–48). https://doi.org/10.24963/ijcai.2017/7.
5. Alonso, E., D'inverno, M., Kudenko, D., Luck, M., & Noble, J. (2001). Learning in multi-agent systems. *The Knowledge Engineering Review*, *16*(3), 277–284.
6. Altman, E. (1999). *Constrained Markov decision processes* (Vol. 7). Boca Raton: CRC Press.
7. Aoki, K., & Kimura, H., & Kobayashi, S. (2004). Distributed reinforcement learning using bi-directional decision making for multi-criteria control of multi-stage flow systems. In *The 8th conference on intelligent autonomous systems* (pp. 281–290).
8. Aumann, R. J. (1974). Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, *1*(1), 67–96.
9. Avigad, G., Eisenstadt, E., & Cohen, M. W. (2011). Optimal strategies for multi objective games and their search by evolutionary multi objective optimization. In *2011 IEEE conference on computational intelligence and games (CIG'11)* (pp. 166–173). IEEE.
10. Baarslag, T., & Kaisers, M. (2017). The value of information in automated negotiation: A decision model for eliciting user preferences. In *Proceedings of the 16th conference on autonomous agents and multiagent systems* (pp. 391–400). International Foundation for Autonomous Agents and Multiagent Systems.
11. Babbar-Sebens, M., & Mukhopadhyay, S. (2009). Reinforcement learning for human-machine collaborative optimization: Application in ground water monitoring. In *2009 IEEE international conference on systems, man and cybernetics* (pp. 3563–3568). IEEE.
12. Bahmankhah, B., & Coelho, M. C. (2017). Multi-objective optimization for short distance trips in an urban area: Choosing between motor vehicle or cycling mobility for a safe, smooth and less polluted route. *Transportation Research Procedia*, *27*, 428–435.
13. Balduzzi, D., Garnelo, M., Bachrach, Y., Czarnecki, W. M., Perolat, J., Jaderberg, M., & Graepel, T. (2019). Open-ended learning in symmetric zero-sum games. ArXiv preprint arXiv:1901.08106.
14. Bargiacchi, E., Verstraeten, T., Roijers, D. M., Nowé, A., & Hasselt, H. (2018). Learning to coordinate with coordination graphs in repeated single-stage multi-agent decision problems. In *International conference on machine learning* (pp. 491–499).
15. Becker, R., Zilberstein, S., Lesser, V., & Goldman, C. V. (2004). Solving transition independent decentralized Markov decision processes. *Journal of Artificial Intelligence Research*, *22*, 423–455.
16. Bellman, R. (1957). *Dynamic programming*. Princeton, NJ: Princeton University Press.
17. Bernstein, D. S., Givan, R., Immerman, N., & Zilberstein, S. (2002). The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, *27*(4), 819–840.

18. Bielefeld, R. S. (1988). *Reexamination of the perfectness concept for equilibrium points in extensive games* (pp. 1–31). Dordrecht: Springer. https://doi.org/10.1007/978-94-015-7774-8_1.
19. Blackwell, D., et al. (1956). An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, *6*(1), 1–8.
20. Bloembergen, D., Tuyls, K., Hennes, D., & Kaisers, M. (2015). Evolutionary dynamics of multi-agent learning: A survey. *Journal of Artificial Intelligence Research*, *53*, 659–697.
21. Bone, C., & Dragićević, S. (2009). Gis and intelligent agents for multiobjective natural resource allocation: A reinforcement learning approach. *Transactions in GIS*, *13*(3), 253–272.
22. Borm, P., van Megen, F., & Tijs, S. (1999). A perfectness concept for multicriteria games. *Mathematical Methods of Operations Research*, *49*(3), 401–412.
23. Borm, P., Tijs, S., & Van Den Aarssen, J. (1988). Pareto equilibria in multiobjective games. *Methods of Operations Research*, *60*, 302–312.
24. Borm, P., Vermeulen, D., & Voorneveld, M. (2003). The structure of the set of equilibria for two person multicriteria games. *European Journal of Operational Research*, *148*(3), 480–493.
25. Bourdache, N., & Perny, P. (2019). Active preference learning based on generalized gini functions: Application to the multiagent knapsack problem. In *Thirty-third AAAI conference on artificial intelligence (AAAI 2019)*.
26. Brown, M., An, B., Kiekintveld, C., Ordóñez, F., & Tambe, M. (2012). Multi-objective optimization for security games. In *Proceedings of the 11th international conference on autonomous agents and multiagent systems-volume 2* (pp. 863–870). International Foundation for Autonomous Agents and Multiagent Systems.
27. Brys, T., Harutyunyan, A., Vrancx, P., Nowé, A., & Taylor, M. E. (2017). Multi-objectivization and ensembles of shapings in reinforcement learning. *Neurocomputing*, *263*, 48–59. **(Multiobjective Reinforcement Learning: Theory and Applications)**.
28. Brys, T., Harutyunyan, A., Vrancx, P., Taylor, M. E., Kudenko, D., & Nowé, A. (2014). Multi-objectivization of reinforcement learning problems by reward shaping. In *2014 international joint conference on neural networks (IJCNN)* (pp. 2315–2322). IEEE.
29. Brys, T., Pham, T. T., & Taylor, M. E. (2014). Distributed learning and multi-objectivity in traffic light control. *Connection Science*, *26*(1), 65–83. https://doi.org/10.1080/09540091.2014.885282.
30. Buşoniu, L., Babuška, R., De Schutter, B., et al. (2008). A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *38*(2), 156–172.
31. Buettner, R., & Landes, J. (2012) Web service-based applications for electronic labor markets: A multidimensional price VCG auction with individual utilities. In *ICIW 2012 proceedings* (pp. 168–177).
32. Calisi, D., Farinelli, A., Iocchi, L., & Nardi, D. (2007). Multi-objective exploration and search for autonomous rescue robots. *Journal of Field Robotics*, *24*(8–9), 763–777. https://doi.org/10.1002/rob.20216.
33. Calvaresi, D., Marinoni, M., Sturm, A., Schumacher, M., & Buttazzo, G. (2017). The challenge of real-time multi-agent systems for enabling IoT and CPS. In *Proceedings of the international conference on web intelligence, WI '17* (pp. 356–364). New York, NY: ACM.
34. de Castro, M. S., Congeduti, E., Starre, R. A., Czechowski, A., & Oliehoek, F. A. (2019) Influence-based abstraction in deep reinforcement learning. In *Proceedings of the AAMAS workshop on adaptive learning agents (ALA)*.
35. Chalkiadakis, G., Elkind, E., & Wooldridge, M. (2011). Computational aspects of cooperative game theory. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *5*(6), 1–168.
36. Chung, J. J., Rebhuhn, C., Yates, C., Hollinger, G. A., & Tumer, K. (2018). A multiagent framework for learning dynamic traffic management strategies. *Autonomous Robots*, pp. 1–17.
37. Crites, R. H., & Barto, A. G. (1996). Improving elevator performance using reinforcement learning. In *Advances in neural information processing systems*, pp. 1017–1023.
38. Current, J., & Min, H. (1986). Multiobjective design of transportation networks: Taxonomy and annotation. *European Journal of Operational Research*, *26*(2), 187–201.
39. De Hauwere, Y. (2011). *Sparse interactions in multi-agent reinforcement learning*. Ph.D. Thesis, Vrije Universiteit Brussel
40. Deb, K. (2014). Multi-objective optimization. In *Search methodologies* (pp. 403–449). Springer.
41. Delle Fave, F., Stranders, R., Rogers, A., & Jennings, N. (2011). Bounded decentralised coordination over multiple objectives. In *Proceedings of the tenth international joint conference on autonomous agents and multiagent systems* (pp. 371–378).
42. Devlin, S., Yliniemi, L., Kudenko, D., & Tumer, K. (2014). Potential-based difference rewards for multiagent reinforcement learning. In *Proceedings of the 2014 international conference on autonomous*

agents and multi-agent systems (pp. 165–172). International Foundation for Autonomous Agents and Multiagent Systems.
43. Diaz-Balteiro, L., & Romero, C. (2008). Making forestry decisions with multiple criteria: A review and an assessment. *Forest Ecology and Management*, *255*(8–9), 3222–3241.
44. Dubus, J., Gonzales, C., & Perny, P. (2009). Choquet optimization using GAI networks for multiagent/multicriteria decision-making. In F. Rossi & A. Tsoukias (Eds.) *ADT 2009: Proceedings of the first international conference on algorithmic decision theory* (pp. 377–389). Berlin: Springer.
45. Dubus, J., Gonzales, C., & Perny, P. (2009) Multiobjective optimization using GAI models. In *IJCAI 2009: proceedings of the twenty-third international joint conference on artificial intelligence* (pp. 1902–1907).
46. Dusparic, I., & Cahill, V. (2009). Distributed w-learning: Multi-policy optimization in self-organizing systems. In *2009 Third IEEE international conference on self-adaptive and self-organizing systems* (pp. 20–29). IEEE.
47. Eisenstadt, E., Moshaiov, A., & Avigad, G. (2015). Co-evolution of strategies for multi-objective games under postponed objective preferences. In *2015 IEEE conference on computational intelligence and games (CIG)* (pp. 461–468). IEEE.
48. Espinasse, B., Picolet, G., & Chouraqui, E. (1997). Negotiation support systems: A multi-criteria and multi-agent approach. *European Journal of Operational Research*, *103*(2), 389–409.
49. Fard, H. M., Prodan, R., Moser, G., Fahringer, T. (2011). A bi-criteria truthful mechanism for scheduling of workflows in clouds. In *2011 IEEE third international conference on cloud computing technology and science* (pp. 599–605). IEEE.
50. Fernández, F., Monroy, L., & Puerto, J. (1998). Multicriteria goal games. *Journal of Optimization Theory and Applications*, *99*(2), 403–421.
51. Fernandez, F. R., Hinojosa, M. A., & Puerto, J. (2002). Core solutions in vector-valued games. *Journal of Optimization Theory and Applications*, *112*(2), 331–360.
52. Flesch, J., Thuijsman, F., & Vrieze, K. (1997). Cyclic Markov equilibria in stochastic games. *International Journal of Game Theory*, *26*(3), 303–314.
53. Foerster, J., Assael, I. A., de Freitas, N., & Whiteson, S. (2016). Learning to communicate with deep multi-agent reinforcement learning. In *Advances in neural information processing systems* (pp. 2137–2145).
54. Foerster, J., Nardelli, N., Farquhar, G., Afouras, T., Torr, P.H., Kohli, P., & Whiteson, S. (2017). Stabilising experience replay for deep multi-agent reinforcement learning. In *Proceedings of the 34th international conference on machine learning-volume 70* (pp. 1146–1155). JMLR.org.
55. Foerster, J. N., Farquhar, G., Afouras, T., Nardelli, N., & Whiteson, S. (2018). Counterfactual multi-agent policy gradients. In *Thirty-second AAAI conference on artificial intelligence*.
56. Friedman, E., & Fontaine, F. (2018). *Generalizing across multi-objective reward functions in deep reinforcement learning*. ArXiv preprint arXiv:1809.06364.
57. Galand, L., & Lust, T. (2015). Exact methods for computing all lorenz optimal solutions to biobjective problems. In *International conference on algorithmic decision theory* (pp. 305–321). Springer.
58. Gardi, A., Sabatini, R., Marino, M., & Kistan, T. (2016). Multi-objective 4d trajectory optimization for online strategic and tactical air traffic management. In T. H. Karakoc, M. B. Ozerdem, M. Z. Sogut, C. O. Colpan, O. Altuntas, & E. Açıkkalp (Eds.), *Sustainable aviation: Energy and environmental issues* (pp. 185–200). Berlin: Springer.
59. Ghose, D., & Prasad, U. (1989). Solution concepts in two-person multicriteria games. *Journal of Optimization Theory and Applications*, *63*(2), 167–189.
60. Golden, B., & Perny, P. (2010). Infinite order lorenz dominance for fair multiagent optimization. In *Proceedings of the 9th international conference on autonomous agents and multiagent systems: Volume 1* (pp. 383–390). International Foundation for Autonomous Agents and Multiagent Systems.
61. Grandoni, F., Krysta, P., Leonardi, S., & Ventre, C. (2010). Utilitarian mechanism design for multiobjective optimization. In *Proceedings of the twenty-first annual ACM-SIAM symposium on discrete algorithms* (pp. 573–584). Society for Industrial and Applied Mathematics.
62. Guestrin, C., Koller, D., & Parr, R. (2002). Multiagent planning with factored MDPs. In *Advances in neural information processing systems 15 (NIPS'02)*.
63. Hamidi, H., & Kamankesh, A. (2018). An approach to intelligent traffic management system using a multi-agent system. *International Journal of Intelligent Transportation Systems Research*, *16*(2), 112–124.
64. Hansen, E. A., Bernstein, D. S., & Zilberstein, S. (2004). Dynamic programming for partially observable stochastic games. In *Proceedings of the 19th national conference on artifical intelligence, AAAI'04* (pp. 709–715). AAAI Press.

65. He, H., Boyd-Graber, J., Kwok, K., & Daumé III, H. (2016) Opponent modeling in deep reinforcement learning. In *International conference on machine learning* (pp. 1804–1813).
66. Hernandez-Leal, P., Kaisers, M., Baarslag, T., & de Cote, E. M. (2017). *A survey of learning in multiagent environments: Dealing with non-stationarity*. ArXiv preprint arXiv:1707.09183.
67. Hernandez-Leal, P., Kartal, B., & Taylor, M. E. (2019). A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-agent Systems*, *33*(6), 750–797. https://doi.org/10.1007/s10458-019-09421-1.
68. Houli, D., Zhiheng, L., & Yi, Z. (2010). Multiobjective reinforcement learning for traffic signal control using vehicular ad hoc network. *EURASIP Journal on Advances in Signal Processing*, *2010*(1), 724035.
69. Hurtado, C., Ramirez, M. R., Alanis, A., Vazquez, S. O., Ramirez, B., & Manrique, E. (2018). Towards a multi-agent system for an informative healthcare mobile application. In *KES International symposium on agent and multi-agent systems: Technologies and applications* (pp. 215–219). Springer.
70. Igarashi, A., & Roijers, D. M. (2017) Multi-criteria coalition formation games. In *International conference on algorithmic decision theory* (pp. 197–213). Springer.
71. Inja, M., Kooijman, C., de Waard, M., Roijers, D. M., & Whiteson, S. (2014) Queued pareto local search for multi-objective optimization. In *International conference on parallel problem solving from nature* (pp. 589–599). Springer.
72. Jaderberg, M., Czarnecki, W. M., Dunning, I., Marris, L., Lever, G., Castaneda, A. G., et al. (2019). Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, *364*(6443), 859–865.
73. Jennings, N. R., Faratin, P., Lomuscio, A. R., Parsons, S., Wooldridge, M. J., & Sierra, C. (2001). Automated negotiation: Prospects, methods and challenges. *Group Decision and Negotiation*, *10*(2), 199–215.
74. Jonker, C. M., Aydoğan, R., Baarslag, T., Fujita, K., Ito, T., & Hindriks, K. (2017). Automated negotiating agents competition (ANAC). In *Thirty-first AAAI conference on artificial intelligence*.
75. Källström, J., & Heintz, F. (2019) Tunable dynamics in agent-based simulation using multi-objective reinforcement learning. In *Proceedings of the adaptive and learning agents workshop (ALA-19) at AAMAS*.
76. Kawamura, T., Kanazawa, T., & Ushio, T. (2013). Evolutionarily and neutrally stable strategies in multicriteria games. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, *96*(4), 814–820.
77. Khan, M. W., & Wang, J. (2017). The research on multi-agent system for microgrid control and optimization. *Renewable and Sustainable Energy Reviews*, *80*, 1399–1411.
78. Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., & Osawa, E. (1997). Robocup: The robot world cup initiative. In *Proceedings of the first international conference on autonomous agents, AGENTS '97* pp. 340–347. New York, NY: ACM.
79. Kok, J. R., & Vlassis, N. (2004). Sparse cooperative Q-learning. In *Proceedings of the twenty-first international conference on Machine learning, ICML '04*. New York, NY: ACM.
80. Kok, J. R., & Vlassis, N. (2006). Collaborative multiagent reinforcement learning by payoff propagation. *Journal of Machine Learning Research*, *7*(Sep), 1789–1828.
81. Kraemer, L., & Banerjee, B. (2016). Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, *190*, 82–94.
82. Kruse, S., Brintrup, A., McFarlane, D., Lopez, T. S., Owens, K., & Krechel, W. E. (2013). Designing automated allocation mechanisms for service procurement of imperfectly substitutable services. *IEEE Transactions on Computational Intelligence and AI in Games*, *5*(1), 15–32.
83. Leahu, H., Kaisers, M., & Baarslag, T. (2019) Automated negotiation with Gaussian process-based utility models. In *Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI-19* (pp. 421–427). International joint conferences on artificial intelligence organization. https://doi.org/10.24963/ijcai.2019/60.
84. Lee, C. S. (2012). Multi-objective game-theory models for conflict analysis in reservoir watershed management. *Chemosphere*, *87*(6), 608–613.
85. Liu, S., Lever, G., Merel, J., Tunyasuvunakool, S., Heess, N., & Graepel, T. (2019) Emergent coordination through competition. In *Proceedings of the seventh international conference on learning representations (ICLR 2019)*.
86. Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, O. P., & Mordatch, I. (2017) Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in neural information processing systems* (pp. 6379–6390).
87. Lozovanu, D., Solomon, D., & Zelikovsky, A. (2005). Multiobjective games and determining pareto-nash equilibria. *Buletinul Academiei de Ştiinţe a Republicii Moldova. Matematica*, *3*, 115–122.

88. Madani, K., & Lund, J. R. (2011). A Monte-Carlo game theoretic approach for multi-criteria decision making under uncertainty. *Advances in Water Resources*, *34*(5), 607–616.
89. Malialis, K., Devlin, S., & Kudenko, D. (2016). Resource abstraction for reinforcement learning in multiagent congestion problems. In *Proceedings of the 2016 international conference on autonomous agents and multiagent systems* (pp. 503–511). International Foundation for Autonomous Agents and Multiagent Systems.
90. Mannion, P. (2017). *Knowledge-based multi-objective multi-agent reinforcement learning*. Ph.D. Thesis, National University of Ireland Galway.
91. Mannion, P., Devlin, S., Duggan, J., & Howley, E. (2018) Reward shaping for knowledge-based multi-objective multi-agent reinforcement learning. *The Knowledge Engineering Review*, *33*, e23. https://doi.org/10.1017/S0269888918000292.
92. Mannion, P., Devlin, S., Mason, K., Duggan, J., & Howley, E. (2017). Policy invariance under reward transformations for multi-objective reinforcement learning. *Neurocomputing*, *263*, 60–73.
93. Mannion, P., Duggan, J., & Howley, E. (2016). An experimental review of reinforcement learning algorithms for adaptive traffic signal control. In L. T. McCluskey, A. Kotsialos, P. J. Müller, F. Klügl, O. Rana, & R. Schumann (Eds.), *Autonomic road transport support systems* (pp. 47–66). Berlin: Springer.
94. Mannion, P., Duggan, J., & Howley, E. (2017). A theoretical and empirical analysis of reward transformations in multi-objective stochastic games. In *Proceedings of the 16th international conference on autonomous agents and multiagent systems (AAMAS)*.
95. Mannion, P., Mason, K., Devlin, S., Duggan, J., & Howley, E. (2016). Dynamic economic emissions dispatch optimisation using multi-agent reinforcement learning. In *Proceedings of the adaptive and learning agents workshop (at AAMAS 2016)*.
96. Mannion, P., Mason, K., Devlin, S., Duggan, J., & Howley, E. (2016). Multi-objective dynamic dispatch optimisation using multi-agent reinforcement learning. In *Proceedings of the 15th international conference on autonomous agents and multiagent systems (AAMAS)*.
97. Marinescu, R. (2009). Exploiting problem decomposition in multi-objective constraint optimization. In *CP 2009: Principles and practice of constraint programming* (pp. 592–607). Berlin: Springer.
98. Marinescu, R. (2011). Efficient approximation algorithms for multi-objective constraint optimization. In *ADT 2011: Proceedings of the second international conference on algorithmic decision theory* (pp. 150–164).
99. Marinescu, R., Razak, A., & Wilson, N. (2013). Multi-objective constraint optimization with tradeoffs. In *International conference on principles and practice of constraint programming* (pp. 497–512). Springer.
100. Mendoza, G. A., & Martins, H. (2006). Multi-criteria decision analysis in natural resource management: A critical review of methods and new modelling paradigms. *Forest Ecology and Management*, *230*(1–3), 1–22.
101. Mirrokni, V. S., & Vetta, A. (2004). Convergence issues in competitive games. In *Approximation, randomization, and combinatorial optimization. algorithms and techniques* (pp. 183–194). Springer.
102. Moghaddam, A., Yalaoui, F., & Amodeo, L. (2011). Lorenz versus pareto dominance in a single machine scheduling problem with rejection. In *International conference on evolutionary multi-criterion optimization* (pp. 520–534). Springer.
103. Moradi, M. H., Razini, S., & Hosseinian, S. M. (2016). State of art of multiagent systems in power engineering: A review. *Renewable and Sustainable Energy Reviews*, *58*, 814–824.
104. Mossalam, H., Assael, Y. M., Roijers, D. M., & Whiteson, S. (2016). *Multi-objective deep reinforcement learning*. ArXiv preprint arXiv:1610.02707.
105. Mouaddib, A. I., Boussard, M., & Bouzid, M. (2007). Towards a formal framework for multi-objective multiagent planning. In *Proceedings of the 6th international joint conference on autonomous agents and multiagent systems* (p. 123). ACM.
106. Nash, J. (1951). Non-cooperative games. *Annals of Mathematics*, *54*(2), 286–295.
107. Ng, A. Y., Harada, D., & Russell, S. J. (1999) Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the sixteenth international conference on machine learning, ICML '99* (pp. 278–287).
108. Nguyen, T. T. (2018). *A multi-objective deep reinforcement learning framework*. ArXiv preprint arXiv:1803.02965.
109. Nguyen, T. T., Nguyen, N. D., & Nahavandi, S. (2018). *Deep reinforcement learning for multi-agent systems: A review of challenges, solutions and applications*. ArXiv preprint arXiv:1812.11794.
110. Nisan, N., Roughgarden, T., Tardos, E., & Vazirani, V. V. (2007). *Algorithmic game theory*. Cambridge: Cambridge University Press.
111. Nwulu, N. I., & Xia, X. (2015). Multi-objective dynamic economic emission dispatch of electric power generation integrated with game theory based demand response programs. *Energy Conversion and Management*, *89*, 963–974.

112. Oliehoek, F. A., Spaan, M. T., & Vlassis, N. (2008). Optimal and approximate q-value functions for decentralized pomdps. *Journal of Artificial Intelligence Research*, *32*, 289–353.
113. Oliehoek, F. A., Whiteson, S., & Spaan, M. T. (2013). Approximate solutions for factored Dec-POMDPs with many agents. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems* (pp. 563–570). International Foundation for Autonomous Agents and Multiagent Systems.
114. Oliehoek, F. A., Witwicki, S. J., & Kaelbling, L. P. (2012). Influence-based abstraction for multiagent systems. In *Twenty-sixth AAAI conference on artificial intelligence*.
115. de Oliveira, E., Fonseca, J. M., & Steiger-Garção, A. (1999). *Multi-criteria negotiation in multi-agent systems*. ≪KSSESE p. 190.
116. Patrone, F., Pusillo, L., & Tijs, S. (2007). Multicriteria games and potentials. *Top*, *15*(1), 138–145.
117. Perny, P., Weng, P., Goldsmith, J., & Hanna, J. (2013). Approximation of Lorenz-optimal solutions in multiobjective Markov decision processes. In *Proceedings of the 27th AAAI conference on artificial intelligence* (pp. 92–94).
118. Pieri, G., & Pusillo, L. (2015). Multicriteria partial cooperative games. *Applied Mathematics*, *6*(12), 2125.
119. Pirjanian, P., & Mataric, M. (2000). Multi-robot target acquisition using multiple objective behavior coordination. In *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)* vol. 3 (pp. 2696–2702). IEEE. https://doi.org/10.1109/ROBOT.2000.846435.
120. Pla, A., Lopez, B., & Murillo, J. (2012). Multi criteria operators for multi-attribute auctions. In *International conference on modeling decisions for artificial intelligence* (pp. 318–328). Springer.
121. Van der Pol, E., Oliehoek, & F. A. (2016). Coordinated deep reinforcement learners for traffic light control. In *Proceedings of learning, inference and control of multi-agent systems (at NIPS 2016)*.
122. Proper, S., & Tumer, K. (2013). Multiagent learning with a noisy global reward signal. In *Twenty-seventh AAAI conference on artificial intelligence*.
123. Pusillo, L., & Tijs, S. (2013). E-equilibria for multicriteria games. In *Advances in dynamic games* (pp. 217–228). Springer.
124. Qu, S., Ji, Y., & Goh, M. (2015). The robust weighted multi-objective game. *PLoS ONE*, *10*(9), e0138970.
125. Ramos, G. D. O., Rădulescu, R., & Nowé, A. (2019) A budged-balanced tolling scheme for efficient equilibria under heterogeneous preferences. In *Proceedings of the adaptive and learning agents workshop (ALA-19) at AAMAS*.
126. Rashid, T., Samvelyan, M., de Witt, C. S., Farquhar, G., Foerster, J., & Whiteson, S. (2018). Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML 2018: Proceedings of the thirty-fifth international conference on machine learning*.
127. Rettieva, A. (2017). Equilibria in dynamic multicriteria games. *International Game Theory Review*, *19*(01), 1750002.
128. Rettieva, A. N. (2014). A discrete-time bioresource management problem with asymmetric players. *Automation and Remote Control*, *75*(9), 1665–1676.
129. Reymond, M., & Nowé, A. (2019). Pareto-DQN: Approximating the Pareto front in complex multi-objective decision problems. In *Proceedings of the adaptive and learning agents workshop (ALA-19) at AAMAS*.
130. Roijers, D. M. (2016). *Multi-objective decision-theoretic planning*. Ph.D. Thesis, University of Amsterdam.
131. Roijers, D. M., Steckelmacher, D., & Nowé, A. (2018). Multi-objective reinforcement learning for the expected utility of the return. In *Adaptive and learning agents workshop (at AAMAS/IJCAI/ICML 2018)*.
132. Roijers, D. M., Vamplew, P., Whiteson, S., & Dazeley, R. (2013). A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, *48*, 67–113.
133. Roijers, D. M., & Whiteson, S. (2017). Multi-objective decision making. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *11*(1), 1–129.
134. Roijers, D. M., Whiteson, S., Ihler, A. T., & Oliehoek, F. A. (2015). Variational multi-objective coordination. In *MALIC 2015: NIPS workshop on learning, inference and control of multi-agent systems*.
135. Roijers, D. M., Whiteson, S., & Oliehoek, F. A. (2013). Computing convex coverage sets for multi-objective coordination graphs. In *International conference on algorithmic decision theory* (pp. 309–323).
136. Roijers, D. M., Whiteson, S., & Oliehoek, F. A. (2014). Linear support for multi-objective coordination graphs. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems* (pp. 1297–1304). International Foundation for Autonomous Agents and Multiagent Systems.
137. Roijers, D. M., Whiteson, S., & Oliehoek, F. A. (2015). Computing convex coverage sets for faster multi-objective coordination. *Journal of Artificial Intelligence Research*, *52*, 399–443.
138. Roijers, D. M., Zintgraf, L. M., Libin, P., & Nowé, A. (2018). Interactive multi-objective reinforcement learning in multi-armed bandits for any utility function. In *ALA workshop at FAIM*, vol. 8.

139. Roijers, D. M., Zintgraf, L. M., & Nowé, A. (2017). Interactive Thompson sampling for multi-objective multi-armed bandits. In *International conference on algorithmic decision theory* (pp. 18–34). Springer.
140. Rollón, E. (2008). *Multi-objective optimization for graphical models*. Ph.D. Thesis, Universitat Politècnica de Catalunya, Barcelona.
141. Rollón, E., & Larrosa, J. (2006). Bucket elimination for multiobjective optimization problems. *Journal of Heuristics*, *12*, 307–328.
142. Rollon, E., & Larrosa, J. (2007). Multi-objective Russian doll search. In *AAAI* (pp. 249–254).
143. Rollon, E., & Larrosa, J. (2008). Constraint optimization techniques for multiobjective branch and bound search. In *International conference on logic programming, ICLP*.
144. Rădulescu, R., Legrand, M., Efthymiadis, K., Roijers, D. M., & Nowé, A. (2018). Deep multi-agent reinforcement learning in a homogeneous open population. In *Proceedings of the 30th benelux conference on artificial intelligence (BNAIC 2018)* (pp. 177–191).
145. Rădulescu, R., Mannion, P., Roijers, D.M., & Nowé, A. (2019). Equilibria in multi-objective games: A utility-based perspective. In *Proceedings of the adaptive and learning agents workshop (ALA-19) at AAMAS*.
146. Scharpff, J., Roijers, D. M., Oliehoek, F. A., Spaan, M. T., & de Weerdt, M. M. (2016). Solving transition-independent multi-agent MDPs with sparse interactions. In *AAAI 2016: Proceedings of the thirtieth AAAI conference on artificial intelligence*. To Appear.
147. Scharpff, J., Spaan, M. T., Volker, L., & de Weerdt, M. M. (2013). Coordinating stochastic multi-agent planning in a private values setting. In *Distributed and multi-agent planning* (p. 17).
148. Sen, S., Weiss, G. (1999). Learning in multiagent systems. In G. Weiss (Ed.) *Multiagent systems: A modern approach to distributed artificial intelligence* (pp. 259–298). Cambridge, MA: MIT Press. http://dl.acm.org/citation.cfm?id=305606.305612.
149. Shapley, L. S. (1953). Stochastic games. *Proceedings of the National Academy of Sciences*, *39*(10), 1095–1100.
150. Shapley, L. S., & Rigby, F. D. (1959). Equilibrium points in games with vector payoffs. *Naval Research Logistics Quarterly*, *6*(1), 57–61.
151. Shelton, C. R. (2001). *Importance sampling for reinforcement learning with multiple objectives*. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA.
152. Shoham, Y., Powers, R., & Grenager, T. (2007). If multi-agent learning is the answer, what is the question? *Artificial Intelligence*, *171*(7), 365–377.
153. Si, W., Li, J., Ding, P., & Rao, R. (2017) A multi-objective deep reinforcement learning approach for stock index future's intraday trading. In *2017 10th International symposium on computational intelligence and design (ISCID)*, vol. 2 (pp. 431–436).
154. Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*, 484–503.
155. Sinha, A., Malo, P., Frantsev, A., & Deb, K. (2013). Multi-objective stackelberg game between a regulating authority and a mining company: A case study in environmental economics. In *2013 IEEE congress on evolutionary computation* (pp. 478–485). IEEE.
156. Song, J., Ren, H., Sadigh, D., & Ermon, S. (2018). Multi-agent generative adversarial imitation learning. In *Advances in neural information processing systems* (pp. 7461–7472).
157. Srinivasan, S., Lanctot, M., Zambaldi, V., Pérolat, J., Tuyls, K., Munos, R., & Bowling, M. (2018). Actor-critic policy optimization in partially observable multiagent environments. In *Advances in neural information processing systems* (pp. 3422–3435).
158. Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., & Tuyls, K., et al. (2018). Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th international conference on autonomous agents and multiagent systems* (pp. 2085–2087). International Foundation for Autonomous Agents and Multiagent Systems.
159. Sutton, R., & Barto, A. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.
160. Tajmajer, T. (2017). *Multi-objective deep q-learning with subsumption architecture*. ArXiv preprint arXiv:1704.06676.
161. Tajmajer, T. (2018). Modular multi-objective deep reinforcement learning with decision values. In *2018 Federated conference on computer science and information systems (FedCSIS)* (pp. 85–93).
162. Tanino, T. (2009). Multiobjective cooperative games with restrictions on coalitions. In *Multiobjective programming and goal programming* (pp. 167–174). Springer.
163. Tanino, T. (2012) Vector optimization and cooperative games. In *Recent developments in vector optimization* (pp. 517–545). Springer.

164. Taylor, A., Dusparic, I., Galván-López, E., Clarke, S., & Cahill, V. (2014). Accelerating learning in multi-objective systems through transfer learning. In *2014 International joint conference on neural networks (IJCNN)* (pp. 2298–2305). IEEE.
165. Tesauro, G. (1994). Td-gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computing*, *6*(2), 215–219.
166. Tsimpoukis, D., Baarslag, T., Kaisers, M., & Paterakis, N.G. (2018). Automated negotiations under user preference uncertainty: A linear programming approach. In *International conference on agreement technologies* (pp. 115–129). Springer.
167. Utomo, C., Idrus, A., & Napiah, M. (2009). Methodology for multi criteria group decision and negotiation support on value-based decision. In *2009 International conference on advanced computer control* (pp. 365–369). IEEE.
168. Vamplew, P., Dazeley, R., Barker, E., & Kelarev, A. (2009). Constructing stochastic mixture policies for episodic multiobjective reinforcement learning tasks. In *Australasian joint conference on artificial intelligence* (pp. 340–349). Springer.
169. Vamplew, P., Dazeley, R., Berry, A., Issabekov, R., & Dekker, E. (2011). Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine Learning*, *84*(1–2), 51–80.
170. Van Moffaert, K., Brys, T., Chandra, A., Esterle, L., Lewis, P. R., & Nowé, A. (2014). A novel adaptive weight selection algorithm for multi-objective multi-agent reinforcement learning. In *2014 International joint conference on neural networks (IJCNN)* (pp. 2306–2314).
171. Vickery, W., Brown, J., & FitzGerald, G. (2003). Spite: altruism's evil twin. *Oikos*, *102*(2), 413–416.
172. Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A.S., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., & Schrittwieser, J., et al. (2017). *Starcraft ii: A new challenge for reinforcement learning*. ArXiv preprint arXiv:1708.04782.
173. Vlassis, N. (2007). A concise introduction to multiagent systems and distributed artificial intelligence. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *1*(1), 1–71.
174. Voorneveld, M., Grahn, S., & Dufwenberg, M. (2000). Ideal equilibria in noncooperative multicriteria games. *Mathematical Methods of Operations Research*, *52*(1), 65–77.
175. Voorneveld, M., Vermeulen, D., & Borm, P. (1999). Axiomatizations of pareto equilibria in multicriteria games. *Games and Economic Behavior*, *28*(1), 146–154.
176. Wang, S. (1993). Existence of a pareto equilibrium. *Journal of Optimization Theory and Applications*, *79*(2), 373–384.
177. White, D. (1982). Multi-objective infinite-horizon discounted Markov decision processes. *Journal of Mathematical Analysis and Applications*, *89*(2), 639–647.
178. Wiggers, A. J., Oliehoek, F. A., & Roijers, D. M. (2016). Structure in the value function of two-player zero-sum games of incomplete information. In *Proceedings of the twenty-second european conference on artificial intelligence* (pp. 1628–1629). IOS Press.
179. Wilson, N., Razak, A., & Marinescu, R. (2015). Computing possibly optimal solutions for multi-objective constraint optimisation with tradeoffs. In *IJCAI 2015: Proceedings of the twenty-fourth international joint conference on artificial intelligence* (pp. 815–821).
180. Wolpert, D. H., & Tumer, K. (2001). Optimal reward functions in distributed reinforcement learning. In *Intelligent agent technology: Research and development* (pp. 365–374). World Scientific.
181. Wolpert, D. H., & Tumer, K. (2002). Optimal payoff functions for members of collectives. In *Modeling complexity in economic and social systems* (pp. 355–369). World Scientific.
182. Wolpert, D. H., Wheeler, K. R., & Tumer, K. (2000). Collective intelligence for control of distributed dynamical systems. *EPL (Europhysics Letters)*, *49*(6), 708.
183. Wooldridge, M. (2001). *Introduction to multiagent systems*. New York, NY: Wiley.
184. Xieping, D. (1996). Pareto equilibria of multicriteria games without compactness, continuity and concavity. *Applied Mathematics and Mechanics*, *17*(9), 847–854.
185. Yliniemi, L., & Tumer, K. (2016). Multi-objective multiagent credit assignment in reinforcement learning and NSGA-ii. *Soft Computing*, *20*(10), 3869–3887.
186. Yliniemi, L. M. (2015). *Multi-objective optimization in multiagent systems*. Ph.D. Thesis, Oregon State University, Corvallis, OR.
187. Yu, H. (2003). Weak pareto equilibria for multiobjective constrained games. *Applied Mathematics Letters*, *16*(5), 773–776.
188. Yu, H., & Liu, H. (2013). Robust multiple objective game theory. *Journal of Optimization Theory and Applications*, *159*(1), 272–280.
189. Yuan, X. Z., & Tarafdar, E. (1996). Non-compact pareto equilibria for multiobjective games. *Journal of Mathematical Analysis and Applications*, *204*(1), 156–163.
190. Zhang, M., Filippone, A., & Bojdo, N. (2018). Multi-objective optimisation of aircraft departure trajectories. *Aerospace Science and Technology*, *79*, 37–47.

191. Zheng, Y., Meng, Z., Hao, J., & Zhang, Z. (2018) Weighted double deep multiagent reinforcement learning in stochastic cooperative environments. In *Pacific Rim international conference on artificial intelligence* (pp. 421–429). Springer.
192. Zinkevich, M., Greenwald, A., & Littman, M. L. (2006). Cyclic equilibria in Markov games. In *Advances in neural information processing systems* (pp. 1641–1648).
193. Zintgraf, L. M., Kanters, T. V., Roijers, D. M., Oliehoek, F. A., & Beau, P. (2015). Quality assessment of MORL algorithms: A utility-based approach. In *Benelearn 2015: Proceedings of the twenty-fourth Belgian-Dutch conference on machine learning*.
194. Zintgraf, L. M., Roijers, D. M., Linders, S., Jonker, C. M., Nowé, A. (2018) Ordered preference elicitation strategies for supporting multi-objective decision making. In *Proceedings of the 17th international conference on autonomous agents and multi-agent systems* (pp. 1477–1485). International Foundation for Autonomous Agents and Multiagent Systems.