Learning Density-Based Correlated Equilibria for Markov Games

Libo Zhang*
University of Electronic Science and
Technology of China, China
University of Auckland, New Zealand
lzha797@aucklanduni.ac.nz

Yang Chen*,†
University of Auckland, New Zealand
yang.chen@auckland.ac.nz

Toru Takisaka
University of Electronic Science and
Technology of China, China
takisaka@uestc.edu.cn

Bakh Khoussainov
University of Electronic Science and
Technology of China, China
bmk@uestc.edu.cn

Michael Witbrock
University of Auckland, New Zealand
m.witbrock@auckland.ac.nz

Jiamou Liu†
University of Auckland, New Zealand
jiamou.liu@auckland.ac.nz

ABSTRACT

Correlated Equilibrium (CE) is a well-established solution concept that captures coordination among agents and enjoys good algorithmic properties. In real-world multi-agent systems, in addition to being in equilibrium, agents' policies are often expected to meet requirements with respect to safety, and fairness. Such additional requirements can often be expressed in terms of the *state density* which measures the state-visitation frequencies during the course of a game. However, existing CE notions or CE-finding approaches cannot explicitly specify a CE with particular properties concerning state density; they do so implicitly by either modifying reward functions or using value functions as the selection criteria. The resulting CE may thus not fully fulfil the state-density requirements. In this paper, we propose *Density-Based Correlated Equilibria* (DBCE), a new notion of CE that explicitly takes state density as a selection criterion. Concretely, we instantiate DBCE by specifying different state-density requirements motivated by real-world applications. To compute DBCE, we put forward the *Density Based Correlated Policy Iteration* algorithm for the underlying control problem. We perform experiments on various games where results demonstrate the advantage of our CE-finding approach over existing methods in scenarios with state-density concerns.

KEYWORDS

Correlated Equilibrium; State Density; Markov Games

1 INTRODUCTION

A central question in the study of multi-agent systems is finding policies for rational game players to reach a particular form of

---

* Equal contributions
† Corresponding author

---

equilibrium. A more recent trend in the investigation of this question is to incorporate policies' side effects [23]. Indeed, in many real-world scenarios, it is difficult to define a reward function that captures all aspects of desired outputs of the agents. For example, a robotic system may gain a high reward by performing a specific risky manoeuvre that is less-than-desirable or engaging in actions that are seen as unethical [31]. When finding policies for agents, it is therefore not optimal to simply enable agents to achieve the highest possible rewards. Still, more importantly, the procedures must also satisfy other desirable properties, such as safety and fairness, that are not reflected by rewards.

For simplicity, we formulate this type of problem as taking an N-player Markov game as input while asking for policies of agents that satisfy two types of requirements:

(1) **Reward Requirement:** First, we expect that the agents, being rational, will not unilaterally deviate from their policies due to utility concerns, and
(2) **Non-reward Requirements:** Then, the policies must satisfy certain non-utility-based requirements that confine the runs of the multi-agent system.

In this paper, we study an instance of the general problem above. (1) For the reward requirement, we specify a solution concept that factors into the possible coordination among agents. More specifically, we adopt *correlated equilibrium* (CE) [17] as the solution concept. Compared to Nash equilibrium (NE), widely adopted in this field [19], CE does not require independence among agents and is suitable for a wider range of practical scenarios. Moreover, the set of CEs constitutes a convex polytope. Therefore, it is easy to compute via linear programming. Many adaptive procedures are shown to converge to CE rather than the more restricted NE [13, 16]. (2) For the non-reward requirements, we consider essential properties which can be loosely translated to, e.g., *"certain situation should not take place"*, *"certain situation should happen with a prescribed frequency"* and *"two situations should happen with the same frequency"*. These properties can be captured by examining a *run*, i.e., the sequence of states the agents are in during the game. More precisely, they are *state-distribution requirements* that are defined in terms of the visitations to states in the game:

- *Safety requirements.* These conditions demand that *certain bad states should not be visited.* Many industrial applications involve

dangerous states that should never happen. Take, e.g., the low-power status of a robot system [29]. Similar concerns can happen from an ethical perspective as well [31].

- *Frequency requirements.* These conditions demand that *certain states should be visited with a fixed frequency.* To generalise safety, the system may be expected to visit certain states with a certain proportion in the long term. For example, we may want a robotic system to run in high-efficiency mode 30% of the time, and the rest 70% time in normal mode.
- *Fairness requirements.* These conditions demand that *two states should be visited with the same frequency.* One may also wish to balance the visitation frequency of two different states of the system for the sake of, e.g., system stability. For instance, if there are two charging stations for a team of unmanned aerial vehicles, one may wish to balance their use rate to avoid unnecessary queuing. Or at a crossroads, traffic lights in two directions should be green with equal frequency.

The *state density function* may be employed to measure the state visitation frequency when navigating the environment using a policy [29]. The function can express the aforementioned state-distribution requirements. However, so far, no work on CE or CE-finding algorithms has *explicitly* incorporated requirements defined by the state density function. On the other hand, methods have been introduced to *implicitly* express these non-reward requirements by imposing additional constraints on rewards. Yet, these methods may not be sufficient to meet these desired requirements. In detail, the existing techniques fall into two categories:

(1) *Risk-sensitive reward modification.* This method adds additional terms that tweak the reward structure, such as negative rewards for undesired states or imposing variance as risk-terms [5, 12, 25, 32]. This method has been preferred when the target is simple, and a tweaking strategy can be efficiently designed. However, it requires parameter fine-tuning as the optimal policy is sensitive to reward settings. When computing CE, this method changes the shape of the CE set of the original game. As a result, the optimal policy found in a modified game may not be a CE to the original game. Moreover, designing a reward modification for complex targets such as *value-targeting requirement* requires expert domain knowledge, which is challenging in real-world applications.

(2) *Constrained methods.* This method directly takes the subset of policies by adding explicit constraints such as *constrained Markov games* [3]. However, it requires parameter fine-tuning because the threshold for constraints can directly impact the game's performance and feasibility. Before solving the game, the optimal solution is invisible to the designers, so setting a correct threshold is challenging. Additionally, when computing CE, the introduced constraints may reduce the size of the CE set of the original game.

Both methods above may change the shape or size of the feasible CE set of the original game. Such changes are illustrated in Fig. 1.

In this paper, we introduce *Density-Based Correlated Equilibria* (DBCE) in the context of Markov games. By using density functions as a selection criterion, DBCE explicitly integrates state-distribution requirements (non-reward requirements) and reward requirements
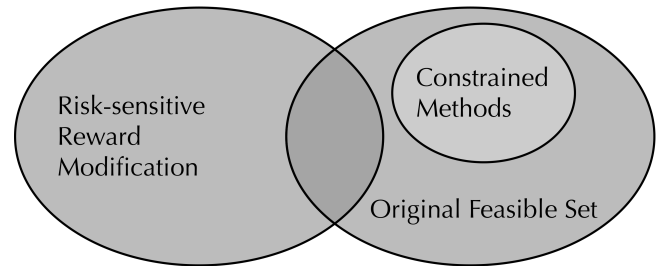

Figure 1: This diagram shows the changes to the feasible CE sets by the two existing methods. A constrained method leads to a size-reduced feasible set, which can be empty if additional constraints are infeasible. The risk-sensitive reward modification generates a new game; thereby, the feasible set may be shifted. Moreover, an improper reward modification may cause an empty intersection between the feasible sets for the modified game and the original game.

to a novel CE concept without suffering the issues above in the existing CE notions or CE-finding approaches.

However, having an equilibrium concept does not necessarily imply an effective way to find it. Directly computing a DBCE is intractable due to the inconsistency between the measurements of a policy's state density and cumulative rewards, preventing us from optimising the two in the same space. To settle this challenge, we employ the notion of *occupancy measure*, i.e., the cumulative state-action visitation frequency, in terms of which both the state density and cumulative rewards can be represented; it thus allows us to optimise the two in a unified fashion. This machinery gives rise to our proposed algorithm for computing DBCE named *Density Based Correlated Policy Iteration* (DBCPI). More specifically, DBCPI runs in such an iterative manner that alternates between the update of agents' policies and the occupancy measure: the policies are updated by finding a CE that is induced by the current occupancy measure and satisfies the non-reward requirements; the current occupancy is subsequently updated in accordance with the updated policy. Moreover, we provide a theoretical justification for DBCPI where the convergence conditions are given.

Our primary contributions are summarised as follows:

(1) We propose a new CE concept for Markov games– Density-Based Correlated Equilibria (DBCE) – which exploits the state density function to explicitly capture non-reward requirements without changing the set of all feasible CEs.
(2) To compute DBCE, we come up with Density-Based Correlated Policy Iteration (DBCPI). We show that under certain assumptions, this mechanism converges to a valid DBCE.
(3) We test DBCPI against existing approaches on different simulated scenarios motivated by real-world applications. Experimental results demonstrate our machinery's advantage in finding CE with those mentioned above, additional non-reward requirements, i.e., safety, frequency and fairness.

2 RELATED WORKS

Equilibrium Concepts. Solving a game in multi-player games, especially non-cooperative ones, amounts to finding an equilibrium.

Nash Q-learning [20] extended the canonical Q-learning to general-sum Markov games to find Nash equilibrium. As for finding CEs, some work [7, 26] attempted to calculate the whole set of CEs by determining or approximating the boundary of the resulting expected-reward space, which has been shown to be a convex polytope. Alternatively, Greenwald et al. [14] proposed a Q-learning-like algorithm to find an instance of CE in a Markov game rather than the whole set. Another line of work focuses on exploiting the application value of CEs in real-world scenarios, such as Yu et al. [37] and Han et al. [15] used CEs to coordinate equipment in industrial scenarios; Jin et al. [22] used CEs as the solution to a outsource task pricing problem. A series of work captures some particular properties by selecting a special subclass of CEs from the entire set. For instance, Ortiz et al. [28] and Ziebart et al. [38] choose the CE with the maximum policy entropy to ensure the uniqueness of the solution to a game.

Non-reward Requirements. In real-world applications, the non-reward requirements are inevitable. Some work adopts an implicit way to satisfy these requirements by modifying the reward functions [9]. One popular method is to augment the reward function with risk-sensitive terms such as variance [24] and exponential utility function [6]. Rather than implicit reward tweaking, logic instruction [18] that explicitly describes the goal is also considered as one method to modify the reward function. Recently, reinforcement learning algorithms with different risk-sensitive factors have been studied in various aspects [5, 12, 25, 32]. Some work along this line adds constraints to the learned policy in order to capture safety concerns: Altman [2] studied constrained MDP, and subsequently, Q-learning was extended to constrained MDP by [10]. Constrained method was later further extended to Markov Games in the multi-agent setting [1, 3, 11, 21]. Among these constraints, the state density stands out as a particular one. Typical work includes [12] that directly specified the unwanted states to avoid getting in, and Qin et al. [29] proposed to use density functions as constraints to guide the finding of an optimal policy in reinforcement learning.

3 PRELIMINARIES

The set of all natural numbers, reals, and non-negative reals are denoted by $\mathbb{N}$, $\mathbb{R}$, and $\mathbb{R}_{\geq 0}$, respectively. For a natural number N, the set $\{1, \ldots, N\}$ is denoted by $[N]$.

Markov Games. Markov games, also known as stochastic games, are extensions of Markov decision processes to the multi-agent setting, where agents act in a stochastic environment, each aiming to maximise its cumulative rewards.

DEFINITION 1. *An N-agent* Markov game *is a tuple*

$$(S, \{\mathcal{A}_i\}_{i=1}^N, P, \{r_i\}_{i=1}^N, \eta, \gamma), \text{ where}$$

- S *is the set of* states,
- $\mathcal{A}_i$ *is the set of* actions *for the ith agent,*
- $P : S \times \mathcal{A} \to \Delta(S)$ *is the* transition function *that specifies the transition probability between two states given a* joint action $\boldsymbol{a} = (a_1, \ldots, a_n)$, *where $\mathcal{A} = \times_{i=1}^N \mathcal{A}_i$ is the space of joint actions and $\Delta(S)$ denotes the set of probability distributions over S,*
- $r_i : S \times \mathcal{A} \to \mathbb{R}$ *is a* reward function *that determines agent i's immediate reward of a joint action in a state,*

- $\eta \in \Delta(S)$ *is the* initial distribution *of states,*
- $\gamma \in (0, 1)$ *is a* discount factor.

Throughout, we use bold variables without subscripts to represent the concatenation of the corresponding variables for all agents and use the subscript $-i$ to denote all agents other than i, e.g., $\boldsymbol{a} = (a_1, \ldots, a_n) = (a_i, \boldsymbol{a}_{-i})$ denotes a *joint action* of all agents.

DEFINITION 2. *The agents' (stationary)* joint policy *is a function*

$$\boldsymbol{\pi} : S \to \Delta(\mathcal{A})$$

which specifies agents' probabilistic choice of actions according to the current state. The set of all joint policies is denoted by Π.

Each agent aims to find a policy to maximise its own *cumulative rewards* during the whole course of a game: $\sum_{t=0}^\infty \gamma^t r_i(s^t, \boldsymbol{a}^t)$. For each agent i, the *expected return* of a state-joint action pair under a joint policy $\boldsymbol{\pi}$ is defined as:

$$Q_i^{\boldsymbol{\pi}}(s, \boldsymbol{a}) \triangleq \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t r_i(s^t, \boldsymbol{a}^t) \middle| s^0 = s, \boldsymbol{a}^0 = \boldsymbol{a}, P, \boldsymbol{\pi}\right].$$

Correlated Equilibria. A solution to a Markov is called an equilibrium that amounts to a joint policy where no agent has the incentive to deviate in order to gain rewards unilaterally. Two canonical equilibrium concepts stand out concerning assumptions on different degrees of independence among agents' policies. The well-known *Nash equilibrium* (NE) [8] requires independence among the agents, i.e., $\boldsymbol{\pi} = \times_{i=1}^N \pi_i$ where $\pi_i : S \to \Delta(\mathcal{A}_i)$ denotes the policy of an individual agent. In comparison, *correlated equilibrium* (CE) [4] generalises NE by capturing the coordination among agents, which is more suitable for multi-agent systems where agents coordinate their actions. Conceptually, agents are coordinated by a *correlation device* that recommends an action $a_i \in \mathcal{A}_i$ to each agent i, who is aware of all other agents' conditional distribution $\boldsymbol{\pi}_{-i}(\boldsymbol{a}_{-i}|s, a_i)$. To be in a CE, each agent has no incentive to disobey the recommendation, i.e., selecting an alternate action $a_i' \in \mathcal{A}_i$, called the *deviation action*.

DEFINITION 3. *A* correlated equilibrium *(CE) for a Markov game is a joint policy $\boldsymbol{\pi}$ that satisfies:*

$$\forall i \in [N], s \in S, a_i, a_i' \in \mathcal{A}_i, \quad \text{reg}_{\boldsymbol{\pi}}(s, i, a_i, a_i') \leq 0. \quad (1)$$

Here, the regret $\text{reg}_{\boldsymbol{\pi}}(s, i, a_i, a_i')$ *embodies the expected reward gain of shifting to a deviation action:*

$$\text{reg}_{\boldsymbol{\pi}}(s, i, a_i, a_i') \triangleq \mathbb{E}_{\boldsymbol{a}_{-i} \sim \boldsymbol{\pi}_{-i}(\cdot|s, a_i)} \left[Q_i^{\boldsymbol{\pi}}(s, a_i', \boldsymbol{a}_{-i}) - Q_i^{\boldsymbol{\pi}}(s, a_i, \boldsymbol{a}_{-i})\right].$$

The general existence of NE [8] implies the existence of CE. CE has nicer mathematical properties than NE in the sense that the constraints in Eq. (1) define an N-dimension polytope in agent's expected returns while the set of NE consists of isolated points [27] in the polytope. Consequently, the set of CEs for normal-form games (equivalent to one-shot Markov games) can be derived using linear programming as Eq. (1) is a system of linear inequalities. Still, exactly computing CE for Markov games is generally intractable due to two reasons: (i) the constraints of CE turn to non-linear inequalities because both Q and $\boldsymbol{\pi}$ are unknown in Eq. (1); (ii) the number of corners of the CE polytope grows exponentially with the horizon increases [38].

Density Functions. A *density function* [30] $\rho : \mathcal{S} \to \mathbb{R}_{\geq 0}$ measures the visitation frequency of states when navigating the environment with a policy. Formally, for an infinite-horizon Markov game with its initial distribution η, discounted factor γ, the density function under a joint policy $\boldsymbol{\pi}$ is defined as

$$\rho^{\boldsymbol{\pi}}(s) \triangleq \sum_{t=0}^{\infty} \gamma^t \Pr(s^t = s | \boldsymbol{\pi}, s^0 \sim \eta).$$

Notice that the density function can also be written in a recursive form as

$$\rho^{\boldsymbol{\pi}}(s) = \eta(s) + \boldsymbol{\pi}(s,a)\gamma \sum_{s' \in \mathcal{S}} \sum_{\boldsymbol{a} \in \mathcal{A}} P(s|s', \boldsymbol{a})\rho^{\boldsymbol{\pi}}(s').$$

Occupancy Measure. Similar to the density function, the *occupancy measure* $\rho(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_{\geq 0}$ measures the visitation frequency of state-action pairs given a stationary policy. Formally, the occupancy measure $\rho^{\boldsymbol{\pi}}$ under $\boldsymbol{\pi}$ is defined as

$$\rho^{\boldsymbol{\pi}}(s, \boldsymbol{a}) \triangleq \sum_{t=0}^{\infty} \gamma^t \Pr(s^t = s, \boldsymbol{a}^t = \boldsymbol{a} | \boldsymbol{\pi}, s^0 \sim \eta).$$

We can calculate the density of a state by an equation $\rho^{\boldsymbol{\pi}}(s) = \sum_{\boldsymbol{a} \in \mathcal{A}} \rho^{\boldsymbol{\pi}}(s, \boldsymbol{a})$. Occupancy measure also has several properties useful in policy synthesis via optimisation [33]. First, a function $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the occupancy measure under some stationary policy if and only if it satisfies the following *Bellman flow (BF) constraints*:

$$\text{BFError}_f(s) = 0, \forall s \in \mathcal{S} \quad \text{AND} \quad f(s, \boldsymbol{a}) \geq 0, \forall s \in \mathcal{S}, \boldsymbol{a} \in \mathcal{A}, \quad (2)$$

where $\text{BFError}_f(s)$ denotes the *Bellman residual* with respect to the state-action visitation frequency:

$$\text{BFError}_f(s) = \sum_{\boldsymbol{a} \in \mathcal{A}} f(s, \boldsymbol{a}) - \eta(s) - \gamma \sum_{s' \in \mathcal{S}} \sum_{\boldsymbol{a} \in \mathcal{A}} P(s|s', \boldsymbol{a})f(s', \boldsymbol{a}).$$

On the other direction, for an f satisfying BF constraints, there is a unique stationary policy $\boldsymbol{\pi} \in \Pi$ associated with f such that f is the occupancy measure under $\boldsymbol{\pi}$ (i.e., $\rho^{\boldsymbol{\pi}} = f$); furthermore, such a policy can be constructed by

$$\boldsymbol{\pi}(s, \boldsymbol{a}) = f(s, \boldsymbol{a}) \Big/ \sum_{\boldsymbol{a}' \in \mathcal{A}} f(s, \boldsymbol{a}'). \quad (3)$$

Non-reward Requirements. In addition to reward requirements captured by equilibrium concepts, non-reward requirements have also drawn attention. Here, we consider the three typical non-reward requirements: safety, frequency and fairness requirements. These requirements can be measured as the counts of occurrences of certain states in a *game trajectory*, i.e., a sequence of states generated by a policy in a Markov game. A finite trajectory with length $n+1$ is written as $\tau \triangleq s_0, s_1, \ldots, s_n$. We can formalise the above-mentioned three types of non-reward requirements in a trajectory-centric way:

- **Safety:** For a set of undesired states S^*, we expect the count of undesired states in the trajectory equals zero, $\sum_{i \in [0,n]} \mathbb{I}(s_i \in S^*) = 0$ where $\mathbb{I}$ is the indicator function;
- **Frequency:** For a set of specific states S^*, we expect the count of such states occur in trajectory with a certain proportion c, $\sum_{i \in [0,n]} \mathbb{I}(s_i \in S^*)/(n+1) = c$
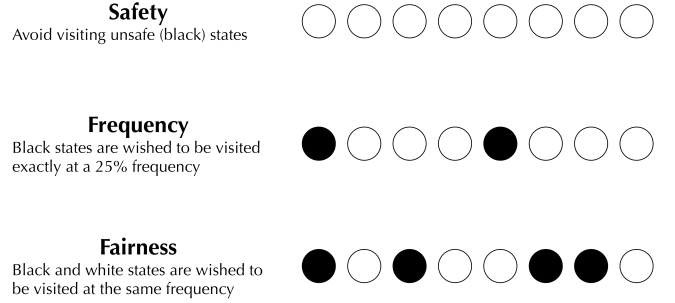

Figure 2: Demonstrations of three types of non-reward requirements. Sequences of circles represent the trajectory, and black and white colors represent two different states.

- **Fairness:** For 2 sets of states S_1^*, S_2^*, we expect the counts of such states from 2 sets to be equal in trajectory, $\sum_{i \in [0,n]} \mathbb{I}(s_i \in S_1^*) = \sum_{i \in [0,n]} \mathbb{I}(s_i \in S_2^*)$.

Intuitively, we demonstrate the three types of non-reward requirements in Fig. 2 on a Markov game with two different states.

4 DENSITY-BASED CORRELATED EQUILIBRIA

This section first proposes the general definition of *Density-Based Correlated Equilibria* (DBCE). We then instantiate it by specifying the selection criterion as above mentioned three types of non-reward requirements.

Recall that the motivation of DBCE is to find an equilibrium that can capture both agents' coordination and policies' side effects that cannot be simply represented in terms of rewards but can instead be interpreted using the density function. To this end, we formalise DBCE by taking one or a set of density functions as the selection criterion to identify the subset of CEs that satisfy the desired non-reward requirements. In such a way, DBCE is defined as a solution to a constrained optimisation problem, where the density functions serve as the objective and the constraints enforce the conditions of being a CE.

DEFINITION 4. *Let the following be given:*
- *A Markov Game $(\mathcal{S}, \{\mathcal{A}_i\}_{i=1}^N, P, \{r_i\}_{i=1}^N, \eta, \gamma)$;*
- *A subset of states $S^* = \{s_1, \ldots, s_m\} \subseteq \mathcal{S}$;*
- *A real-valued function $F : \mathbb{R}^m \to \mathbb{R}$;*
- *A function $\varphi(\boldsymbol{\pi}) = F(\rho^{\boldsymbol{\pi}}(s_1), \ldots, \rho^{\boldsymbol{\pi}}(s_m))$, which we call the density error of $\boldsymbol{\pi}$.*

A joint policy $\boldsymbol{\pi}$ is called an (F-specified) density-based correlated equilibria (DBCE) if it is a solution to the following constrained optimisation problem:

$$\min_{\boldsymbol{\pi} \in \Pi} \varphi(\boldsymbol{\pi}) \quad \text{subject to}$$
$$\text{reg}_{\boldsymbol{\pi}}(s, i, a_i, a_i') \leq 0, \forall i \in [N], s \in \mathcal{S}, a_i, a_i' \in \mathcal{A}_i. \quad (4)$$

We can use the value of the density error $\varphi(\boldsymbol{\pi})$ to indicate the quality of $\boldsymbol{\pi}$ in terms of the state density. By choosing a suitable function F, we can instantiate the DBCE that captures specific non-reward requirements. Here, we introduce the following specific DBCEs with respect to safety, frequency and fairness requirements: a DBCE is called a

- *Minimum Density CE (MDCE)* when we have
$\varphi(\boldsymbol{\pi}) = \sum_{s \in \mathcal{S}^*} \rho^{\boldsymbol{\pi}}(s)$;
- *Frequency Matching CE (FMCE)* when we have
$\varphi(\boldsymbol{\pi}) = |\sum_{s \in \mathcal{S}^*} \rho^{\boldsymbol{\pi}}(s) - c|$ for some $c \in \mathbb{R}_{\geq 0}$;
- *Minimum Density Gap CE (MDGCE)* when we have
$\varphi(\boldsymbol{\pi}) = |\sum_{s \in \mathcal{S}_1} \rho^{\boldsymbol{\pi}}(s) - \sum_{s \in \mathcal{S}_2} \rho^{\boldsymbol{\pi}}(s)|$ for $S_1, S_2 \subseteq \mathcal{S}^*$.

Intuitively, we use MDCE, FMCE and MDGCE to represent CEs with requirements concerning safety, frequency, and fairness requirements, respectively. These instantiations indicate the general ability of DBCE to characterise the equilibria with some density-related properties, which are not yet able to be represented by other equilibrium notions.

5 DENSITY-BASED CORRELATED EQUILIBRIA FINDING

This section is devoted to the introduction of a policy iteration algorithm to compute DBCE and the proof of its convergence under certain assumptions. For simplicity, our analysis centres around the Minimum Density CE (MDCE); but it applies to any other instance of DBCE.

5.1 Density-Based Correlated Policy Iteration

Recall that computing an MDCE requires solving the constrained optimisation problem defined in Eq. (4) where $\varphi(\boldsymbol{\pi}) = \sum_{s \in \mathcal{S}^*} \rho^{\boldsymbol{\pi}}(s)$. However, directly solving it is intractable because the density functions in the objective and the expected return in the constraints are defined in two different spaces; this prevents us from optimising the density-related objective whilst satisfying rewards-related constraints in a unified fashion. We thus ask for a way to unify the representations of the state density and expected return. Fortunately, we observe that both the state density and expected return can be rewritten in terms of the occupancy measure introduced in Sec. 3. We can thereby simultaneously control the two by maintaining a single variable, rather than in two separate spaces.

We next show how to derive an equivalent yet tractable form of the original constrained optimisation problem. We first rewrite the constraints of Eq. 4 by occupancy measure, $\text{reg}'_f(s, i, a_i, a'_i)$, which is defined as follows:

$$\sum_{\boldsymbol{a}_{-i}} f(s, a_i, \boldsymbol{a}_{-i}) \left[Q_i^{\boldsymbol{\pi}}(s, a'_i, \boldsymbol{a}_{-i}) - Q_i^{\boldsymbol{\pi}}(s, a_i, \boldsymbol{a}_{-i}) \right].$$

This is equivalent to $\text{reg}'_{\rho^{\boldsymbol{\pi}}}(s, i, a_i, a'_i)$, shown via Eq. (3).

With the relationship between the occupancy measure and the density function, the objective function $\varphi(\boldsymbol{\pi})$ can also be rewritten as $\varphi'(f)$, where

$$\varphi'(f) \triangleq F\left(\sum_{\boldsymbol{a}} f(s_1, \boldsymbol{a}), \ldots, \sum_{\boldsymbol{a}} f(s_m, \boldsymbol{a})\right). \tag{5}$$

Recall that the density function can be rewritten as the sum of occupancy measures within one state: $\rho^{\boldsymbol{\pi}}(s) = \sum_{\boldsymbol{a} \in \mathcal{A}} \rho^{\boldsymbol{\pi}}(s, \boldsymbol{a})$, the new objective function becomes equivalent to the original objective function. So far, we achieve the consistency between non-reward requirements and reward requirements by occupancy measure. By the properties of occupancy measure discussed in the previous section, we recast the problem as follows:

PROBLEM 1.
$$\min_{f: \mathcal{S} \times \mathcal{A} \to \mathbb{R}} \sum_{s \in \mathcal{S}^*} \sum_{\boldsymbol{a} \in \mathcal{A}} f(s, \boldsymbol{a}) \quad \text{subject to}$$

$$\text{reg}'_f(s, i, a_i, a'_i) \leq 0, \qquad \forall i \in [N], s \in \mathcal{S}, a_i, a'_i \in \mathcal{A}_i; \tag{6}$$

$$\text{BFError}_f(s) = 0, \qquad \forall s \in \mathcal{S}; \tag{7}$$

$$f(s, \boldsymbol{a}) \geq 0, \qquad \forall s \in \mathcal{S}, \boldsymbol{a} \in \mathcal{A}. \tag{8}$$

Here, the Bellman flow constraints (7) and (8) enforce f to be the occupancy measure under some $\boldsymbol{\pi} \in \Pi$. For such an f, the new objective function $\sum_{\boldsymbol{a} \in \mathcal{A}} f(s^*, \boldsymbol{a})$ is equal to $\rho^{\boldsymbol{\pi}}(s^*)$, and (6) enforces $\boldsymbol{\pi}$ to be a CE. Due to the one-to-one correspondence between occupancy measures and stationary policies, a solution of Prob. 1 is the occupancy measure under a solution to the original problem.

However, Prob. 1 is still difficult to solve directly because both f and Q (involved in reg'_f) are unknown. We introduce an iterative approach to handle the problem that we call *Density-Based Correlated Policy Iteration* (DBCPI). It alternates between (i) *policy evaluation*: estimating Q values according to the current policy; and (ii) *policy improvement*: computing a DBCE under the current Q function. More formally, let t denote the index of iterations. At each iteration, $\mathbf{Q}^t = \{Q_i^t\}_{i \in [N]}$ defines a *stage game* with constant Q values. Define $\text{reg}_f^t(s, i, a_i, a'_i)$ as follows:

$$\sum_{\boldsymbol{a}_{-i}} f(s, a_i, \boldsymbol{a}_{-i}) \left[Q_i^t(s, a'_i, \boldsymbol{a}_{-i}) - Q_i^t(s, a_i, \boldsymbol{a}_{-i}) \right].$$

By substituting reg'_f in Prob. 1 with reg_f^t, the stage game $\mathbf{Q}^t$ is now tractable to solve using linear programming. After deriving the DBCE $\boldsymbol{\pi}^t$ of the current stage game $\mathbf{Q}^t$, we head back to update Q functions and derive $\mathbf{Q}^{t+1}$. The pseudocode is presented in Alg. 1.

Algorithm 1 Density-Based Correlated Policy Iteration

1: **Input:** A Markov game $(\mathcal{S}, \mathcal{A}, P, \{r_i\}_{i=1}^N, \eta, \gamma)$.
2: **Initialisation:** Q_i for each $i \in [N]$, learning rate α
3: $\boldsymbol{\pi}(s, \boldsymbol{a}) \leftarrow f(s, \boldsymbol{a}) / \sum_{\boldsymbol{a}' \in \mathcal{A}} f(s, \boldsymbol{a}')$
4: **for** each iteration **do**
5:     $f \leftarrow$ (solution to Prob. 1 with $\{Q_i\}_{i \in [N]}$)
6:     $\boldsymbol{\pi}(s, \boldsymbol{a}) \leftarrow f(s, \boldsymbol{a}) / \sum_{\boldsymbol{a}' \in \mathcal{A}} f(s, \boldsymbol{a}')$
7:     **while** Not converge **do**
8:         Initialise state $s \in \mathcal{S}$
9:         Observe transition $(s, \boldsymbol{a}, \boldsymbol{r}, s')$
10:         **for** each $i \in [N]$ **do**
11:             $V_i(s') \leftarrow \sum_{\boldsymbol{a}' \in \mathcal{A}} \boldsymbol{\pi}(s', \boldsymbol{a}') Q_i(s', \boldsymbol{a}')$
12:             $Q_i(s, \boldsymbol{a}) \leftarrow (1 - \alpha) Q_i(s, \boldsymbol{a}) + \alpha(r_i + \gamma V_i(s'))$
13:         **end for**
14:         Decay α
15:     **end while**
16: **end for**
17: **Output:** A joint policy $\boldsymbol{\pi}$, and $\varphi'(f)$ as the error of $\boldsymbol{\pi}$.

5.2 Convergence Analysis

We next prove that $\mathbf{Q}^t$ converges to the Q values under a DBCE as Alg. 1 is applied. We begin by introducing the following useful technical assumptions.

ASSUMPTION 1. *Each state $s \in \mathcal{S}$ and action $a_i \in \mathcal{A}_i$ for all $i \in [N]$ are visited infinitely often.*

ASSUMPTION 2. *The reward is bounded by some constant.*

ASSUMPTION 3. *The learning rate α_t satisfies the following conditions:* $0 \leq \alpha_t < 1 \, \forall t$, $\sum_t \alpha_t = \infty$ *and* $\sum_t \alpha_t^2 < \infty$.

We also need the following lemma that guarantees the policy estimation procedure after solving each stage game can converge to Q values under π^t.

LEMMA 1 ([34]). *Let $\mathbb{Q}$ be the space of all Q functions. Under Assumption 1-3, the iteration defined by the following converges to Q^{π} with probability 1:*

$$Q^{t+1}(s, \boldsymbol{a}) = (1 - \alpha)Q^t(s, \boldsymbol{a})$$
$$+ \alpha_t \Big(r(s', \boldsymbol{a}) + \gamma \sum_{\boldsymbol{a}} \pi(\boldsymbol{a}|s')Q^t(s', \boldsymbol{a}) \Big).$$

Now, we are ready to present our main theorem which shows that DBCPI converges to a DBCE Q function under assumptions.

THEOREM 1. *Under Assumption 1-3, the Q function iteratively updated in Alg. 1 will converge to the one under a DBCE if for all t, $s \in \mathcal{S}$, and $i \in [N]$, the policy π_t is recognised as the global optimum expressed as:*

$$\forall \pi' \in \Pi, \quad \mathbb{E}_{\boldsymbol{a} \sim \pi^t}[Q_i^t(s, \boldsymbol{a})] \geq \mathbb{E}_{\boldsymbol{a} \sim \pi'}[Q_i^t(s, \boldsymbol{a})].$$

PROOF SKETCH. The basic idea is to show that the policy guided by DBCPI monotonically improves in terms of rewards. By Lemma 1, for all t, after sufficient rounds of updates, we derive Q functions $\mathbf{Q}^{t+1} = \{Q_i^{t+1}\}_{i \in [N]}$ under π^t. By assumption, there always exists a globally optimal policy for each encountered stage game $\mathbf{Q}^t$. As a result, in every iteration the policy monotonically improves as the iteration progresses:

$$\mathbb{E}_{\boldsymbol{a} \sim \pi^{t+1}}[Q_i^{t+1}(s, \boldsymbol{a})] \geq \mathbb{E}_{\boldsymbol{a} \sim \pi^t}[Q_i^{t+1}(s, \boldsymbol{a})],$$

for all t, $s \in \mathcal{S}$, and $i \in [N]$. This implies that after a sufficient number of iterations, the policy converges to a globally optimal one, and so does the Q function. By solving each stage game $\mathbf{Q}^t$ using linear programming, all constraints in Prob. 1 can be satisfied. At convergence, the policy is thus a feasible DBCE. □

Although Thm. 1 tells us that the convergence holds true under strong constraints on every stage game, in experiments we find the constraint is not necessary for DBCPI to converge. This fact is in accordance with the empirical analysis in [20, 36].

6 EXPERIMENTS

We seek to answer the following questions via experiments:

Q1. Does our algorithm find a CE better than other approaches?
We evaluate this by checking the following values upon the termination of Alg. 1 after K iterations:

$$\text{MaxReg} \triangleq \max_{s, i, a_i, a_i'} \text{reg}_f^{K+1}(s, i, a_i, a_i'),$$

$$\text{MaxBF} \triangleq \max_s |\text{BFError}_f(s)|.$$

MaxReg can be seen as a "distance" between π and the CE-set: the larger the value is, the larger incentive there exists for some agent to deviate from π. In particular, π is a CE when this value is non-positive. MaxBF evaluates the soundness of the computation in line 5: the larger the value is, the further f deviates from ρ^{π}, i.e.,

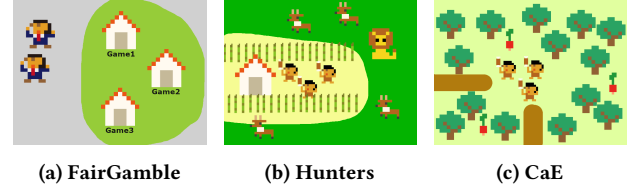(a) FairGamble   (b) Hunters   (c) CaE

Figure 3: Screenshots of games.

the occupancy measure of π. Such a deviation of f implies that the value $\varphi'(f)$ is unreliable as the error of π.

To evaluate the algorithms in this question, we focus on the MaxReg and MaxBF values in all cases. The smaller the values are, the better the algorithms are. 9 extra MDCE tasks are carried out to compare the MaxReg in modified games and original games for risk-sensitive reward modification.

Q2. Does our policy generate desired trajectories?
We evaluate this by examining the patterns of individual trajectories under the policy computed by Alg. 1 on all three requirements.

Q3. What is the accuracy of our DBCPI compared to existing ones?
To make the comparison in accuracy, we compare the errors (i.e., the value $\varphi'(f)$ in the output of Alg. 1), and select a few instances from our data and illustrate them with plots.

Q4. What is the convergence performance of our DBCPI on learning?
We evaluate this by performing the iteration error plots of our DBCPI in the experiments.

6.1 Experiment Setup

6.1.1 Game environments and tasks. We consider three game models (Fair Gamble, Hunters, Collect and Explore); for each of them, we impose three different state-distribution requirements, which make nine instances of the input to Alg. 1 in total. We have anonymously published animated demonstrations of our algorithm on these games, which are available at https://github.com/nanaralala/Density-based-Correlated-Equilibrium/. The screenshots are presented in Fig. 3. The description of each game is shown below.

(1) *Fair Gamble.* In this game, two gamblers play games with each other, and they choose from 3 different games. 3 games are extremely fair, so no matter what they do, the reward will be given randomly. Game 1 gives 0 rewards fairly; Game 2 gives 0.5 to a gambler and -0.5 to another; Game 3 gives 1 to a gambler and -1 to another.

In each round, the gamblers choose a number from 0,1,2 and we compare the number to select which game they play. See Fig. 3a for the explanation.

We consider three state-related requirements, namely
  (a) a safety requirement for gamblers to avoid game 3.
  (b) a frequency matching requirement for gamblers to choose game 3 in 10% of the time.
  (c) a fairness requirement that demands game 1 and game 2 to have an equal frequency.
(2) *Hunters.* In this non-cooperative game, 3 hunters live in one village. In each round, they are inside the village or outside the village, and they can choose between going hunting or guarding

the village against the animals, see Fig. 3b. If one hunter goes hunting from the village, the hunter will get a high reward (1) and the rest of the hunters will get a low reward (0.1). If one hunter guards the village, all hunters will get the same mid-level reward (0.5). If one hunter is outside of the village and still stays hunting outside, we consider the behaviour is not safe enough, so the reward he gains becomes smaller (0.5), and the others will get a punishment reward (-0.5). Additionally, if there is less or equal than 1 hunter guarding the village, they will receive high punishment reward (-3).

We consider three state-related requirements, namely

(a) a safety requirement that demands at least 2 hunters to stay in the village.
(b) a frequency matching requirement that requires less or equal to one agent guarding the village 10% of the time.
(c) a fairness requirement that demands that the frequency of hunter 1 goes hunting and the sum of hunter 2 and 3 go hunting to be equal.

(3) *Collect and Explore* (CaE). In this cooperative game, 3 agents are trapped in a forest, see Fig. 3c. They can choose to explore the environment or collect some food nearby their accommodation. If more than one agent chooses to go out, we randomly choose one of them to go out, and the others will get no rewards. In each round, if existing agents go exploration, we add 1 to the reward; if any agent collects foods nearby, we add 0.3 to the reward. Since it's a cooperative environment, we set the same reward for all agents.

We consider three state-related requirements, namely

(a) a safety requirement for agent 1 to not go out.
(b) a frequency matching requirement that confines agent 1 to go out 10% of the time.
(c) a fairness requirement that demands that the frequency of agent 1 go explore be equal to the sum of the frequencies that agent 2 and agent 3 go explore.

6.1.2 Baselines. We implement *utilitarian CE-Q* [14] with risk-sensitive reward modification and constrained methods. The detail is shown as follows:

(1) *Risk-sensitive Reward modification (RM).* A negative constant $p < 0$ is added to the reward at $s \in S^*$. We write RM-p to denote the algorithm with a specific p.
(2) *Constrained method (CM).* An additional constraint $\varphi'(f) \leq b$ is added (cf. eq. (5)). We write CM-b to denote the algorithm with a specific b.

RM is used as a baseline for the safety requirement only; we are not aware if there exists a canonical way to do that for frequency matching and fairness requirements. CM is used for all state requirements we consider in the experiment.

6.1.3 Implementation details. The iteration number of Alg. 1 is set to 250. Parameters in Alg. 1 are set to $\gamma = 0.99$ and α decays from 0.3 to 0.001. We run the algorithm 3 times for each experiment environment, and the results are taken as the mean of 3 runs. In the program, the optimisation problem in line 5 of Alg. 1 was solved using an optimizer in [35].

Table 1: Comparisons on capabilities of found CE.

Game	Metric	Method	Requirement		
			Safety	Fairness	Freq-10
FairGamble	Error	DBCE	**1.225**	32.967	8.683
		CM-0.05	8.331	**0.466**	**6.053**
		CM-5	9.65	8.814	5.668
		RM-1.5	17.725	—	—
	MaxBF	DBCE	0.464	**0.13**	**0.521**
		CM-0.05	16.496	8.16	11.612
		CM-5	18.054	7.781	6.623
		RM-1.5	**0.032**	—	—
	MaxReg	DBCE	0.164	**0.034**	**0.08**
		CM-0.05	0.174	0.172	0.22
		CM-5	0.391	0.107	0.231
		RM-1.5	**0.11**	—	—
Hunters	Error	DBCE	14.129	4.442	3.643
		CM-0.05	2.313	2.283	**0.05**
		CM-5	2.124	**2.174**	5
		RM-1.5	**0.828**	—	—
	MaxBF	DBCE	**0**	**0**	0.035
		CM-0.05	0.005	0.001	**0**
		CM-5	0.004	0.032	**0**
		RM-1.5	0.001	—	—
	MaxReg	DBCE	**0.044**	**0.061**	**0.037**
		CM-0.05	9.171	0.52	0.225
		CM-5	0.479	1.18	0.18
		RM-1.5	0.842	—	—
CaE	Error	DBCE	0.419	**0.002**	7.608
		CM-0.05	**0.242**	0.05	**1.174**
		CM-25	16.802	9.688	11.854
		RM-0.5	23.838	—	—
	MaxBF	DBCE	**0**	**0**	**0**
		CM-0.05	0.023	**0**	**0**
		CM-25	0.004	1.151	**0**
		RM-0.5	0.023	—	—
	MaxReg	DBCE	**0.002**	**0.001**	0.003
		CM-0.05	1.266	0.002	0.174
		CM-25	0.419	6.695	**0.001**
		RM-0.5	1.434	—	—

6.2 Experiment Results

The following results and discussions answer questions asked at the beginning of this section.

Q1. The results for this question are found in the MaxBF and MaxReg rows in Tab. 1. DBCE has the smallest MaxBF value in 7 of 9 cases, which means DBCE performs better in finding occupancy measures and mapping them to policies. DBCE also has the smallest MaxReg value in 7 of 9 cases, which means DBCE performs better in finding policies in the CE-set. In conclusion of this observation, our selection criteria perform better in finding CE policies in these experiments. In 9 extra MDCE runs, 7 of 9 runs show that the distance to CE-set in the original game is longer than

(a) FairGamble-MDCE   (b) Hunters-Freq-10   (c) CaE-Fairness
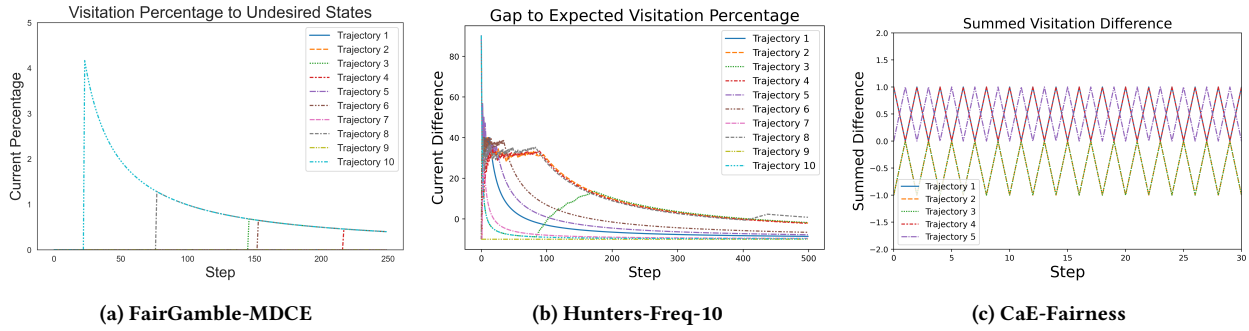
Figure 4: Results for trajectory performance are shown in 4a 4b 4c Close to 0 means desired trajectory.

the distance to CE-set in the modified game, which indicates the disadvantage of the RM method.

Q2. The result for this question is found in Fig. 4a for MDCE, Fig. 4b for FMCE and Fig. 4c for MGDCE. Each plot includes a few trajectories generated by our algorithm, we estimate the gap between the trajectories and the expected visitation counts, so the closer to 0 the value is, the better the trajectory is. In Fig. 4a, we observe the trajectories have the expected property, which is the count of visitations to undesired states is low. In Fig. 4b, we observe the trajectories generated by our algorithm gradually converge to the expected proportion of visitation. Few trajectories have a larger deviation from the desired value, which indicates a larger deviation, but others perform well. In Fig. 4c, all trajectories perform well in balancing visitation in 30 steps (a similar pattern appears in the rest 220 steps).
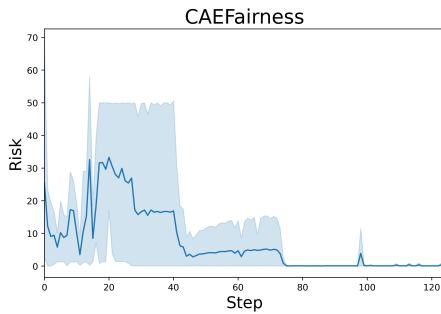

Figure 5: The error plot of our DBCE finding algorithm during learning. Take CAE-Fairness as an example. The error plot illustrates the process of the algorithm getting stabilised.

Q3. The selected result for this question is shown in Fig. 2. There, we picked up the cases with comparable MaxReg and MaxBF values, which means we drop the cases with large MaxReg or MaxBF values; notice that these numbers should be (close to) zero to claim that the algorithm found a CE with the computed error. The exhaustive result is found in Tab. 1.

In detail, DBCE has the smallest MaxReg and MaxBF values in most cases. In *CaE-MinGap* case, DBCE and Cons005 have almost the same performance in MaxReg and MaxBF, but DBCE has a smaller error.

Q4. The results for this question are found in Fig. 5. We can observe the error-step line gradually gets stable. Fluctuations in the early stages vanish along with learning. We believe this early-stage fluctuation can be caused by the updating of Q-functions, and risk gets stable after the Q-functions get stable, which is reasonable since it's a policy-iteration process.

Table 2: The explanation on comparable instances.

Task	Method	MaxReg	MaxBF	Error
	DBCE	**0.061**	**0**	4.442
Hunters-MinGap	Cons005	0.52	0.001	2.283
	Cons5	1.18	0.032	2.174
FairGamble-MDCE	DBCE	0.164	0.464	1.225
	RewMod	**0.11**	**0.032**	17.725
	DBCE	**0.001**	**0**	0.002
CaE-MinGap	Cons005	0.002	**0**	0.05
	Cons5	6.695	1.151	9.688
	DBCE	**0.002**	**0**	0.419
CaE-MDCE	Cons005	1.266	0.023	0.242
	Cons25	0.419	0.004	16.802
	RewMod	1.434	0.023	23.838

7 CONCLUSION AND FUTURE WORK

In this paper, we propose a new concept of the correlated equilibrium, the Density-Based correlate equilibrium (DBCE). It enables us to find joint policies that satisfy both reward requirements, i.e., equilibrium, and non-reward requirements characterised by state density functions. Different from existing methods, DBCE neither modifies the shape or size of feasible CE-set to a game nor suffers the parameter tuning problem. We connect density and reward by occupancy measure, and design Density-Based Correlated Policy Iteration (DBCPI) to compute DBCE. Experiments on various games prove the advantage of our method in finding desired CEs. In future works, one may be interested in implementing parameterised version of DBCPI to solve more complex games with continuous state-action space games. Additionally, one density-based objective may lead to multiple points in the CE space, so further selection among those candidates can also be the next step.

REFERENCES

[1] E. Altaian, K. Avrachenkov, Nicolas Bonneau, mérouane Debbah, Rachid El-Azouzi, and Daniel Menasché. 2007. Constrained Stochastic Games in Wireless Networks. *GLOBECOM - IEEE Global Telecommunications Conference*, 315 – 320. https://doi.org/10.1109/GLOCOM.2007.66

[2] Eitan Altman. 1993. Asymptotic properties of constrained Markov decision processes. *Zeitschrift für Operations Research* 37, 2 (1993), 151–170.

[3] Eitan Altman and Adam Shwartz. 2000. Constrained markov games: Nash equilibria. In *Advances in dynamic games and applications*. Springer, 213–221.

[4] Robert J Aumann. 1987. Correlated equilibrium as an expression of Bayesian rationality. *Econometrica: Journal of the Econometric Society* (1987), 1–18.

[5] Vivek S Borkar. 2002. Q-learning for risk-sensitive control. *Mathematics of operations research* 27, 2 (2002), 294–311.

[6] Kun-Jen Chung and Matthew J Sobel. 1987. Discounted MDP's: Distribution functions and exponential utility maximization. *SIAM journal on control and optimization* 25, 1 (1987), 49–62.

[7] Liam Dermed and Charles Isbell. 2009. Solving Stochastic Games. *Advances in Neural Information Processing Systems 22 - Proceedings of the 2009 Conference*, 1186–1194.

[8] Arlington M Fink. 1964. Equilibrium in a stochastic n-person game. *Journal of science of the hiroshima university, series ai (mathematics)* 28, 1 (1964), 89–93.

[9] Javier Garcıa and Fernando Fernández. 2015. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research* 16, 1 (2015), 1437–1480.

[10] Ather Gattami, Qinbo Bai, and Vaneet Agarwal. 2019. Reinforcement learning for multi-objective and constrained Markov decision processes. *arXiv preprint arXiv:1901.08978* (2019).

[11] Yangyang Ge, Fei Zhu, Wei Huang, Peiyao Zhao, and Quan Liu. 2020. Multi-agent cooperation Q-learning algorithm based on constrained Markov Game. *Computer Science and Information Systems* 17, 2 (2020), 647–664.

[12] Peter Geibel and Fritz Wysotzki. 2005. Risk-sensitive reinforcement learning applied to control under constraints. *Journal of Artificial Intelligence Research* 24 (2005), 81–108.

[13] Geoffrey J Gordon, Amy Greenwald, and Casey Marks. 2008. No-regret learning in convex games. In *Proceedings of the 25th international conference on Machine learning*. 360–367.

[14] Amy Greenwald, Keith Hall, Roberto Serrano, et al. 2003. Correlated Q-learning. In *ICML*, Vol. 3. 242–249.

[15] Zhu Han, Charles Pandana, and KJ Ray Liu. 2007. Distributive opportunistic spectrum access for cognitive radio using correlated equilibrium and no-regret learning. In *2007 IEEE wireless communications and networking conference*. IEEE, 11–15.

[16] Sergiu Hart and Andreu Mas-Colell. 2000. A simple adaptive procedure leading to correlated equilibrium. *Econometrica* 68, 5 (2000), 1127–1150.

[17] Sergiu Hart and Andreu Mas-Colell. 2001. A reinforcement procedure leading to correlated equilibrium. In *Economics essays*. Springer, 181–200.

[18] Mohammadhosein Hasanbeig, Alessandro Abate, and Daniel Kroening. 2018. Logically-constrained reinforcement learning. *arXiv preprint arXiv:1801.08099* (2018).

[19] Charles A Holt and Alvin E Roth. 2004. The Nash equilibrium: A perspective. *Proceedings of the National Academy of Sciences* 101, 12 (2004), 3999–4002.

[20] Junling Hu and Michael P Wellman. 2003. Nash Q-learning for general-sum stochastic games. *Journal of machine learning research* 4, Nov (2003), 1039–1069.

[21] Xiaofeng Jiang, Shuangwu Chen, Jian Yang, Han Hu, and Zhenliang Zhang. 2020. Finding the Equilibrium for Continuous Constrained Markov Games Under the

[22] Haiming Jin, Hongpeng Guo, Lu Su, Klara Nahrstedt, and Xinbing Wang. 2019. Dynamic task pricing in multi-requester mobile crowd sensing with markov correlated equilibrium. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 1063–1071.

[23] Toryn Q Klassen, Sheila A McIlraith, and Christian Muise. 2022. An AI Safety Threat from Learned Planning Models. (2022).

[24] Harry Markowitz. 1952. PORTFOLIO SELECTION*. *The Journal of Finance* 7, 1 (1952), 77–91. https://doi.org/10.1111/j.1540-6261.1952.tb01525.x arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1540-6261.1952.tb01525.x

[25] Oliver Mihatsch and Ralph Neuneier. 2002. Risk-sensitive reinforcement learning. *Machine learning* 49, 2 (2002), 267–290.

[26] Chris Murray and Geoff Gordon. 2007. *Finding correlated equilibria in general sum stochastic games*. Carnegie Mellon University, School of Computer Science, Machine Learning …

[27] Abraham Neyman. 1997. Correlated equilibrium and potential games. *International Journal of Game Theory* 26, 2 (1997), 223–227.

[28] Luis E Ortiz, Robert E Schapire, and Sham M Kakade. 2007. Maximum entropy correlated equilibria. In *Artificial Intelligence and Statistics*. PMLR, 347–354.

[29] Zengyi Qin, Yuxiao Chen, and Chuchu Fan. 2021. Density constrained reinforcement learning. In *International Conference on Machine Learning*. PMLR, 8682–8692.

[30] Anders Rantzer. 2001. A dual to Lyapunov's stability theorem. *Systems & Control Letters* 42, 3 (2001), 161–168.

[31] Daniel Schiff, Justin Biddle, Jason Borenstein, and Kelly Laas. 2020. What's next for ai ethics, policy, and governance? a global overview. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 153–158.

[32] Yun Shen, Michael J Tobia, Tobias Sommer, and Klaus Obermayer. 2014. Risk-sensitive reinforcement learning. *Neural computation* 26, 7 (2014), 1298–1328.

[33] Umar Syed, Michael Bowling, and Robert E Schapire. 2008. Apprenticeship learning using linear programming. In *Proceedings of the 25th international conference on Machine learning*. 1032–1039.

[34] Csaba Szepesvári and Michael L Littman. 1999. A unified analysis of value-function-based reinforcement-learning algorithms. *Neural computation* 11, 8 (1999), 2017–2060.

[35] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272. https://doi.org/10.1038/s41592-019-0686-2

[36] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. 2018. Mean field multi-agent reinforcement learning. In *International conference on machine learning*. PMLR, 5571–5580.

[37] Tao Yu, HZ Wang, Bin Zhou, Ka Wing Chan, and J Tang. 2014. Multi-agent correlated equilibrium Q (λ) learning for coordinated smart generation control of interconnected power grids. *IEEE transactions on power systems* 30, 4 (2014), 1669–1679.

[38] Brian D Ziebart, Drew Bagnell, and Anind K Dey. 2010. Maximum causal entropy correlated equilibria for Markov games. In *Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Average Criteria. *IEEE Trans. Automat. Control* 65, 12 (2020), 5399–5406.