# LEARNING IN CONSTRAINED STOCHASTIC DYNAMIC POTENTIAL GAMES

*Sergio Valcarcel Macua* *Santiago Zazo* *Javier Zazo*

Universidad Politécnica de Madrid

## ABSTRACT

We extend earlier works on continuous potential games to the most general case: stochastic time varying environment, stochastic rewards, non-reduced form and constrained state-action sets. We provide conditions for a Markov Nash equilibrium (MNE) of the game to be equivalent to the solution of a single control problem. Then, we address the problem of learning this MNE when the reward and state transition models are unknown. We follow a reinforcement learning approach and extend previous algorithms for working with constrained state-action subsets of real vector spaces. As an application example, we simulate a network flow optimization model, in which the relays have batteries that deplete with a random factor. The results obtained with the proposed framework are close to optimal.

***Index Terms—*** Approximate dynamic programming, game theory, multi-agent, network flow, reinforcement learning

## 1. INTRODUCTION

In a noncooperative stochastic dynamic game, the players compete in a time-varying environment that we assume can be characterized by a discrete-time Markov decision process. The game starts at some initial state. Then, the players take some action, receive some reward and the system moves to another state. This state-transition process is repeated at every time step over an *infinite* time horizon. We study the case in which the state transitions and rewards are stochastic and follow some stationary distributions, and where the players aim to maximize their expected long-term cumulative reward. Thus, a stochastic game is represented by a set of coupled stochastic-optimal-control-problems (SOCP), which are generally difficult to solve. Nevertheless, there is a class of problems, named *stochastic dynamic potential games* (SDPG), that can be formulated as a *single multivariate stochastic optimal control problem* (MSOCP), which is generally more tractable than the original set of coupled SOCP.

Static potential games and their applications have been extensively studied [1, 2, 3, 4]. However, SDPG have only been analyzed for particular cases. The pioneering work on deterministic dynamic potential games is that of [5]. More recently, [6] extended the analysis to stochastic transition models and stochastic rewards. However, these works [5, 6] only considered games in *reduced form* (i.e., they assumed that the state-transition equation, which express the future state as a function of the action and current state, can be converted to express the action as a function of the current and future states). A more general treatment in *non-reduced form* can be found in [7, 8] for the non-stochastic case. But none of these works [5, 6, 7, 8] considered explicitly constraints in the state-action variables. Up to our knowledge, the first works to analyze dynamic potential games in non-reduced form by taking state-action constraints explicitly are [9, 10]. However, [9, 10] only study deterministic problems. Our first contribution is to extend earlier analysis to the most general case, by considering SDPG in non-reduced form with constrained state-action sets. Similar to [10], we use the Euler approach, but in this case the optimality conditions will be stochastic.

Our second contribution is to show that we can find a solution to the constrained SDPG even when the state transition and reward models are unknown. Indeed, we just need to observe the state transitions, actions, and rewards to estimate the optimal strategy. That is, we consider the game as a real or simulated black-box and learn by interaction. The idea is that, once we have converted the SDPG to an equivalent SMOCP, we can follow a *reinforcement learning* (RL) approach. Note though that most RL algorithms are only valid for discrete state-action sets. Since we deal with subsets of high dimensional real subspaces, learning the solution of the SMOCP becomes very difficult in practice. Therefore, we first survey RL algorithms that are suitable for continuous state-action sets and remark one that is simple but able to learn complex policies. Since all these algorithms only work for unconstrained problems, we also propose a penalty-type extension for dealing with constraints.

Finally, as an application example, we take the dynamic network flow game due to [10]—in which the users aim to maximize their throughput through some relays subject to bounded capacity while also saving the relays' battery life—and include a random depleting factor that models the stochastic variations of the channel gain. By using the proposed penalty-extension with the chosen RL algorithm we are able to achieve close to optimal results in this problem.

## 2. PROBLEM FORMULATION

Let $\mathcal{Q} \triangleq \{1, \dots, Q\}$ denote the set of players. Let $\mathcal{X} \subseteq \Re^S$ denote the set of states of the game. At every time step $t$, the state-vector of the game is $\mathbf{x}_t \triangleq (x_t^k)_{k=1}^S \in \mathcal{X}$. Every player $i \in \mathcal{Q}$ can be influenced by a subset of states $\mathcal{X}^i \subseteq \mathcal{X}$, determined by a subset of state-vector components denoted by $\mathcal{X}(i) \subseteq \{1, \dots, S\}$, such that $\mathbf{x}_t^i \triangleq (x_t^m)_{m \in \mathcal{X}(i)}$ indicates the value of the state-vector for player $i$ at time $t$. We also define $\mathcal{X}^{-i} \subseteq \mathcal{X}$ as the subset of components that do not influence player $i$, such that $\mathbf{x}_t^{-i} \triangleq (x_t^l)_{l \notin \mathcal{X}(i)} \in \mathcal{X}^{-i}$.

Let $\mathcal{U} \subseteq \Re^Q$ denote the set of actions of all players. Let $\mathcal{U}^i \subseteq \Re^{A^i}$ stand for the subset of actions of player $i$, such that $\mathcal{U} \triangleq \prod_{i=1}^Q \mathcal{U}^i$. We write $\mathbf{u}_t^i \triangleq (u^{ia})_{a=1}^{A^i} \in \mathcal{U}^i$ the action vector of length $A^i$ for player $i$ at time $t$, such that $\mathbf{u}_t \triangleq (\mathbf{u}_t^i)_{i=1}^Q \in \mathcal{U}$ contains the actions of all players. We also define $\mathbf{u}_t^{-i} \triangleq (\mathbf{u}_t^j)_{j \in \mathcal{Q}, j \neq i} \in \mathcal{U}^{-i} \triangleq \prod_{j \neq i} \mathcal{U}^j$ as the vector of all players' actions except that of player $i$. Slightly abusing notation, we rewrite $\mathbf{u}_t = (\mathbf{u}_t^i, \mathbf{u}_t^{-i})$.

The state transitions $\mathbf{x}_{t+1} \sim p_{\mathbf{x}}(\cdot|\mathbf{x}_t, \mathbf{u}_t)$ are determined by the state-transition probability distribution over the future state, given the current state and current action. Theses transitions can be equivalently expressed as a function $f : \mathcal{X} \times \mathcal{U} \times \Theta \to \mathcal{X}$ of a random variable $\boldsymbol{\theta}_t \in \Theta \subseteq \Re^S$, such that $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t)$, where $\{\boldsymbol{\theta}_t\}$ is a sequence of i.i.d. random variables.

Work supported by Spanish Ministry of Economy and Competitiveness grant TEC2013-46011-C3-1-R and an FPU doctoral grant to the third author.

Each player $i$ has an instantaneous reward function $r^i : \mathcal{X}^i \times \mathcal{U} \times \Sigma \to \Re$, such that, at every time step, it receives a random reward $r_t^i \triangleq r^i(\mathbf{x}_t^i, \mathbf{u}_t, \sigma_t^i)$, for some sequence of i.i.d. random variables $\{\sigma_t^i \in \Sigma^i \subseteq \mathcal{X}^i\}$. This can be also rewritten as $r_t^i \sim p_{\mathbf{r}}(\cdot|\mathbf{x}_t^i, \mathbf{u}_t)$.

We include a vector $g \triangleq (g^c)_{c=1}^C$ of $C$ constraints over the system state and all players' actions.

Each player $i$ aims to find the sequence of actions $\{\mathbf{u}_t^i\}_{t=1}^\infty$ that maximizes its long term cumulative utility, given the sequence of other players' actions $\{\mathbf{u}_t^{-i}\}_{t=0}^\infty$. Thus, the stochastic game can be expressed by a set of $Q$ coupled SOCP:

$$\begin{aligned} \mathcal{G}_1 : \underset{\{\mathbf{u}_t^i\}\in\prod_{t=0}^\infty \mathcal{U}^i}{\text{maximize}} \quad & \mathbb{E}\left[\sum_{t=0}^\infty \beta^t r^i(\mathbf{x}_t^i, \mathbf{u}_t^i, \mathbf{u}_t^{-i}, \sigma_t^i)\right] \\ \forall i \in \mathcal{Q} \qquad \text{s.t.} \quad & \mathbf{x}_{t+1} \sim p_{\mathbf{x}}(\cdot|\mathbf{x}_t, \mathbf{u}_t) \\ & g(\mathbf{x}_t, \mathbf{u}_t) \leq 0, \quad \mathbf{x}_0 \text{ given} \end{aligned} \tag{1}$$

where the expectation is taken over all possible transitions and over the reward distribution. The discount factor $0 < \beta < 1$ bounds the infinite-horizon cumulative utility.

Since learning an infinite sequence of actions is impractical, we adopt a closed-loop approach and aim to learn a set of *stationary stochastic policies*, $\pi \triangleq \{\pi^i\}_{i=1}^Q \in \Pi$, such that, each player's policy maps states to distributions over its action set: $\mathbf{u}_t^i \sim \pi^i(\cdot|\mathbf{x}_t^i) \in \Pi^i$. Define the set of other players' policies: $\pi^{-i} \triangleq \{\pi^j\}_{j\in\mathcal{Q}, j\neq i}$.

Introduce each player's value function, $v^i : \mathcal{X} \times \Pi \to \Re$, as the expected long term cumulative reward when the system starts at state $\mathbf{x}_0$, the player follows $\pi^i$ and the rest of players follow $\pi^{-i}$:

$$\begin{aligned} v^i(\mathbf{x}, \pi^i, \pi^{-i}) \triangleq \mathbb{E}\Bigg[\sum_{t=0}^\infty \beta^t r_t^i \;\Big|\; \mathbf{x}_0 = \mathbf{x},\ r_t^i \sim p_{\mathbf{r}}(\cdot|\mathbf{x}_t, \mathbf{u}_t^i, \mathbf{u}_t^{i-1}) \\ \mathbf{x}_{t+1} \sim p_{\mathbf{x}}(\cdot|\mathbf{x}_t, \mathbf{u}_t),\ \mathbf{u}_t^i \sim \pi^i(\cdot|\mathbf{x}_t^i),\ \mathbf{u}_t^{-i} \sim \pi^{-i}(\cdot|\mathbf{x}_t)\Bigg] \end{aligned} \tag{2}$$

We consider the Markov Nash Equilibrium (MNE) (see, e.g., [6]) as the solution concept for an SDPG.

**Definition 1.** *A solution of problem* (1)*, known as MNE, is a set of feasible policies $\pi^\star \triangleq \{\pi^{\star i}\}_{i=1}^Q$ that $\forall i \in \mathcal{Q}$ satisfies:*

$$\begin{aligned} v^i(\mathbf{x}, \pi^{\star i}, \pi^{\star -i}) \geq v^i(\mathbf{x}, \pi^i, \pi^{\star -i}), \quad \forall \pi^i \in \Pi^i, \quad \forall \mathbf{x} \in \mathcal{X} \\ g(\mathbf{x}, (\pi^i(\mathbf{x}), \pi^{\star -i}(\mathbf{x}))) \leq 0 \end{aligned} \tag{3}$$

We require similar assumptions to the deterministic case [10]. Note that no convexity assumption in $r^i$, $f$ or $g$ is required.

**Assumption 1.** *Utilities $\{r^i\}_{i\in\mathcal{Q}}$ are twice continuously differentiable in $\mathcal{X} \times \mathcal{U}$.*

**Assumption 2.** *$\mathcal{X}$, $\mathcal{U}$ are open convex subsets of real vector spaces.*

**Assumption 3.** *$f$ and $g$ are continuously differentiable and satisfy some regularity conditions in $\mathcal{X} \times \mathcal{U}$ (see, e.g., [11, Sec. 3.3.5]).*

**Assumption 4.** *The set of solutions of the SMOCP* (4) *(defined below) is nonempty.*

## 3. STOCHASTIC DYNAMIC POTENTIAL GAMES

In general, finding a MNE of problem (1) is difficult. Nevertheless, for constrained deterministic dynamic games [10] and for unconstrained stochastic games [6], it has been shown that these problems can be solved through a constrained MOCP and unconstrained SMOCP, respectively. In this section we extend previous analysis, giving conditions for the constrained stochastic game (1) to be potential, and proving that it can be solved through the following SMOCP:

$$\begin{aligned} \mathcal{P}_1 : \underset{\pi\in\Pi}{\text{maximize}} \quad & \mathbb{E}\left[\sum_{t=0}^\infty \beta^t J(\mathbf{x}_t, \mathbf{u}_t, \sigma_t)\right] \\ \text{s.t.} \quad & \mathbf{x}_{t+1} \sim p_{\mathbf{x}}(\cdot|\mathbf{x}_t, \mathbf{u}_t), \quad \mathbf{x}_0 \text{ given} \\ & \mathbf{u}_t \sim \pi(\cdot|\mathbf{x}_t), \quad g(\mathbf{x}_t, \mathbf{u}_t) \leq 0 \end{aligned} \tag{4}$$

where $J$ denotes the instantaneous random reward, and $\{\sigma_t\}$ is an i.i.d. sequence of random variables. The expectation in (4) is taken over the state transition and reward distributions.

**Theorem 1.** *Let Assumptions 1–4 hold. Let all players' reward functions satisfy the following conditions $\forall i, j \in \mathcal{Q}$, $\forall m \in \mathcal{X}(i)$, $\forall n \in \mathcal{X}(j)$, $a = [1, \ldots, A^i]$ and $b = [1, \ldots, A^j]$:*

$$\mathbb{E}\left[\frac{\partial^2 r^i(\mathbf{x}_t^i, \mathbf{u}_t, \sigma_t^i)}{\partial u_t^{jb} \partial x_t^m}\right] = \mathbb{E}\left[\frac{\partial^2 r^j(\mathbf{x}_t^j, \mathbf{u}_t, \sigma_t^j)}{\partial u_t^{ia} \partial x_t^n}\right] \tag{5}$$

$$\mathbb{E}\left[\frac{\partial^2 r^i(\mathbf{x}_t^i, \mathbf{u}_t, \sigma_t^i)}{\partial x_t^n \partial x_t^m}\right] = \mathbb{E}\left[\frac{\partial^2 r^j(\mathbf{x}_t^j, \mathbf{u}_t, \sigma_t^j)}{\partial x_t^m \partial x_t^n}\right] \tag{6}$$

$$\mathbb{E}\left[\frac{\partial^2 r^i(\mathbf{x}_t^i, \mathbf{u}_t, \sigma_t^i)}{\partial u_t^{jb} \partial u_t^{ia}}\right] = \mathbb{E}\left[\frac{\partial^2 r^j(\mathbf{x}_t^j, \mathbf{u}_t, \sigma_t^j)}{\partial u_t^{ia} \partial u_t^{jb}}\right] \tag{7}$$

*Then, problem* (1) *is a SDPG that has an MNE equal to the solution of the SMOCP* (4) *with reward function given by*

$$\begin{aligned} J(\mathbf{x}_t, \mathbf{u}_t, \sigma_t) = \sum_{i=1}^Q \Bigg( \int_0^1 \sum_{m\in\mathcal{X}(i)} \frac{\partial r^i(\boldsymbol{\eta}(\lambda), \mathbf{u}_t, \sigma_t^i)}{\partial x_t^m} \frac{d\eta^m(\lambda)}{d\lambda} d\lambda \\ + \int_0^1 \sum_{a\in A^i} \frac{\partial r^i(\mathbf{x}_t, \boldsymbol{\xi}(\lambda), \sigma_t^i)}{\partial u_t^{ia}} \frac{d\xi^{ia}(\lambda)}{d\lambda} d\lambda \Bigg) \end{aligned} \tag{8}$$

*where $\boldsymbol{\eta}(\lambda) \triangleq \left(\eta^k(\lambda)\right)_{k=1}^S$, $\boldsymbol{\xi}(\lambda) \triangleq \left((\xi^{ia}(\lambda))_{a=1}^{A^i}\right)_{i=1}^Q$, and $\boldsymbol{\eta}(0)$-$\boldsymbol{\xi}(0)$ and $\boldsymbol{\eta}(1)$-$\boldsymbol{\xi}(1)$ correspond to the initial and final state-action conditions, respectively, and with $\sigma_t \triangleq (\sigma_t^i)_{i=1}^Q$.*

*Proof.* We combine the analysis in [10, Th. 1] and [6, Sec. 3], for the constrained deterministic and unconstrained stochastic cases, respectively. First, we derive two sets of optimality conditions. For problem (1), each player's Lagrangian is given by

$$\begin{aligned} \mathcal{L}^i\left(\mathbf{x}_t, \mathbf{u}_t, \sigma_t^i, \boldsymbol{\theta}_t, \boldsymbol{\lambda}_t^i, \boldsymbol{\mu}_t^i\right) = \mathbb{E}\Bigg[\sum_{t=0}^\infty \beta^t r^i\left(\mathbf{x}_t^i, \mathbf{u}_t, \sigma_t^i\right) \\ + \boldsymbol{\lambda}_t^{i\top}\left(f\left(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t\right) - \mathbf{x}_{t+1}\right) + \boldsymbol{\mu}_t^{i\top} g\left(\mathbf{x}_t, \mathbf{u}_t\right)\Bigg] \end{aligned} \tag{9}$$

with multipliers $\boldsymbol{\lambda}_t^i \triangleq \left(\lambda_t^{ik}\right)_{k=1}^S$ and $\boldsymbol{\mu}_t^i \triangleq \left(\mu_t^{ic}\right)_{c=1}^C$. Similarly, we derive the Lagrangian for problem (4):

$$\begin{aligned} \mathcal{L}^{\text{MSOCP}}\left(\mathbf{x}_t, \mathbf{u}_t, \sigma_t, \boldsymbol{\theta}_t, \boldsymbol{\gamma}_t, \boldsymbol{\delta}_t\right) = \mathbb{E}\Bigg[\sum_{t=0}^\infty \beta^t \Big(J\left(\mathbf{x}_t, \mathbf{u}_t, \sigma_t\right) \\ + \boldsymbol{\gamma}_t^\top\left(f\left(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t\right) - \mathbf{x}_{t+1}\right) + \boldsymbol{\delta}_t^\top g\left(\mathbf{x}_t, \mathbf{u}_t\right)\Big)\Bigg] \end{aligned} \tag{10}$$

where $\boldsymbol{\gamma}_t \triangleq \left(\gamma_t^k\right)_{k=1}^S$ and $\boldsymbol{\delta}_t \triangleq (\delta_t^c)_{c=1}^C$ are the multipliers. Since $\mathbf{x}_t$ is set after the random value $\boldsymbol{\theta}_{t-1} = s_{t-1}$ is known, we derive the discrete-time stochastic Euler equations of (9) and (10) and obtain

the following KKT conditions for both problems. The KKT of (1) for every player $i \in \mathcal{Q}$, state components $m \in \mathcal{X}(i)$ and action indexes $a = 1, \dots, A^i$, is given by:

$$\mathbb{E}\left[\frac{\partial r^i\left(\mathbf{x}_t^i, \mathbf{u}_t, \sigma_t^i\right)}{\partial x_t^m} + \sum_{k=1}^{S} \lambda_t^{ik} \frac{\partial f^k\left(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t\right)}{\partial x_t^m}\right] + \sum_{c=1}^{C} \mu_t^{ic} \frac{\partial g^c\left(\mathbf{x}_t, \mathbf{u}_t\right)}{\partial x_t^m} - \lambda_{t-1}^{im} = 0 \quad (11)$$

$$\mathbb{E}\left[\sum_{a=1}^{A^i} \frac{\partial r^i\left(\mathbf{x}_t^i, \mathbf{u}_t, \sigma_t^i\right)}{\partial u_t^{ia}} + \sum_{k=1}^{S} \lambda_t^{ik} \frac{\partial f^k\left(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t\right)}{\partial u_t^{ia}}\right] + \sum_{c=1}^{C} \mu_t^{ic} \frac{\partial g^c\left(\mathbf{x}_t, \mathbf{u}_t\right)}{\partial u_t^{ia}} = 0 \quad (12)$$

$$\mathbf{x}_{t+1} = f\left(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t\right), \quad g\left(\mathbf{x}_t, \mathbf{u}_t\right) \leq 0 \quad (13)$$

$$\boldsymbol{\mu}_t^i \leq 0, \quad \boldsymbol{\mu}_t^{i^\top} g\left(\mathbf{x}_t, \mathbf{u}_t\right) = 0 \quad (14)$$

Similarly, the KKT system of (4) for all state-components ($m = 1, \dots, S$) and all action indexes ($ia \in \{11, \dots, QA^Q\}$) is given by

$$\mathbb{E}\left[\frac{\partial J\left(\mathbf{x}_t, \mathbf{u}_t, \sigma_t\right)}{\partial x_t^m} + \sum_{k=1}^{S} \gamma_t^k \frac{\partial f^k\left(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t\right)}{\partial x_t^m}\right] + \sum_{c=1}^{C} \delta_t^c \frac{\partial g^c\left(\mathbf{x}_t, \mathbf{u}_t\right)}{\partial x_t^m} - \gamma_{t-1}^m = 0 \quad (15)$$

$$\mathbb{E}\left[\frac{\partial J\left(\mathbf{x}_t, \mathbf{u}_t, \sigma_t\right)}{\partial u_t^{ia}} + \sum_{k=1}^{S} \gamma_t^k \frac{\partial f^k\left(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t\right)}{\partial u_t^{ia}}\right] + \sum_{c=1}^{C} \delta_t^c \frac{\partial g^c\left(\mathbf{x}_t, \mathbf{u}_t\right)}{\partial u_t^{ia}} = 0 \quad (16)$$

$$\mathbf{x}_{t+1} = f\left(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t\right), \quad g\left(\mathbf{x}_t, \mathbf{u}_t\right) \leq 0 \quad (17)$$

$$\boldsymbol{\delta}_t \leq 0, \quad \boldsymbol{\delta}_t^\top g\left(\mathbf{x}_t, \mathbf{u}_t\right) = 0 \quad (18)$$

Now, by comparing (11)–(14) with (15)–(18), we conclude that the solution of problem (4) is also a MNE of problem (1) if the following conditions are satisfied $\forall i \in \mathcal{Q}$:

$$\mathbb{E}\left[\frac{\partial r^i\left(\mathbf{x}_t^i, \mathbf{u}_t, \sigma_t^i\right)}{\partial x_t^m}\right] = \mathbb{E}\left[\frac{\partial J\left(\mathbf{x}_t, \mathbf{u}_t, \sigma_t\right)}{\partial x_t^m}\right], \ \forall m \in \mathcal{X}(i) \quad (19)$$

$$\mathbb{E}\left[\frac{\partial r^i\left(\mathbf{x}_t^i, \mathbf{u}_t, \sigma_t^i\right)}{\partial u_t^{ia}}\right] = \mathbb{E}\left[\frac{\partial J\left(\mathbf{x}_t, \mathbf{u}_t, \sigma_t\right)}{\partial u_t^{ia}}\right], a = 1 \dots A^i \quad (20)$$

$$\boldsymbol{\lambda}_t^i = \boldsymbol{\gamma}_t, \quad \boldsymbol{\mu}_t^i = \boldsymbol{\delta}_t \quad (21)$$

These are the stochastic extension of the same conditions given by [10, Th. 1] for the deterministic case. Similar to the deterministic case, it is straightforward to see that, under Assumptions 3–4, condition (21) represents a feasible point of the SMOCP that also satisfies the KKT system of the game and, hence, it is a MNE of (1). However, since we do not know $J$ beforehand, (19)–(20) are not easily verifiable in practice. Instead, it is more convenient to check conditions that depend only on the info available in (1). Introduce the vector field $\mathbf{F} = \nabla J$, conservative by construction. Under Assumption 2 and equivalences (19)–(20), the fact that $\mathbf{F}$ is conservative is a necessary and sufficient condition for (5)–(7) (e.g., [12, Th. 10.9]).

Finally, since the objective $J$ is the potential of $\mathbf{F}$, we can calculate it through the line integral (8). □

Instead of verifying (5)–(7), some problems have a separable structure that makes easier to deduce whether they are a SDPG.

**Corollary 1.** *Problem* (1) *is a SDPG if each player's reward function can be expressed as the sum of a term, $J$, common to all players plus another term, $\Psi^i$, that is independent on its own variables:*

$$r^i\left(\mathbf{x}_t^i, \mathbf{u}_t, \sigma_t^i\right) = J\left(\mathbf{x}_t, \mathbf{u}_t, \sigma_t\right) + \Psi^i(\mathbf{x}_t^{-i}, \mathbf{u}_t^{-i}), \quad \forall i \in \mathcal{Q} \quad (22)$$

*Proof.* Taking partial derivatives on (22) with respect to $x_t^m$ and $u_t^{ia}$ yields (19)–(20) (see also [10, Lemma 3] and [1, Prop. 1]). □

## 4. REINFORCEMENT LEARNING FOR SDPG

Having established an equivalence between the SDPG (1) and the SMOCP (4) is a significant step for finding an MNE. However, there are cases in which we the problem parameter are unknown, or where the reward and/or the state-transition distributions are not known at all. Rather, we have access only to the real system that we want to control or to a simulator. Thus, we have to learn by interacting with the real or simulated environment. At every time step, we sense the state of the environment to some extent, take an action, and observe the instant reward, obtaining a sequence of tuples $\{(\mathbf{x}_t, \mathbf{u}_t, r_t)\}_{t=0}^{T}$, where $T$ is the maximum number of samples. RL is a field that studies this kind of problems and its solution methods (see, e.g., [13] for an intuitive introduction to RL; [14] and [15, Vol 2, Ch. 6] for a more formal perspective; [16, 17] for reviews of modern RL algorithms; and [18] for a survey on recent developments in the field).

Traditional RL algorithms work well in environments with finite state and action sets. However, for SDPG, the state set and action sets are subsets of $\Re^S$ and $\Re^{\sum_{i=1}^{Q} A^i}$, respectively. Two main approaches for dealing with continuous state-action sets are: *i)* discretize the continuous variables so that standard techniques can be applied (e.g., [19]); and *ii)* map the state-action variables to a vector of features and learn a parametric approximation of the value function and/or the policy from these features (e.g., [20, 17]). The main problem with discretizing continuous domains is that the number of discrete states grows exponentially with the number of dimensions. Learning parametric approximations presents also some drawbacks: appropriate features have to be discovered or hand-crafted and policies are usually constrained to some parametric class (e.g., [21, 22, 23, 24]), which might not be rich enough to represent the complexities of a constrained problem. Other useful approaches include [25, 26, 27, 28, 29] and, more recently, the promising [30, 31].

Nevertheless, none of the mentioned methods have considered constrained state-actions sets. As in standard optimization methods, we propose to extend the available algorithms by including a penalty function that gives negative reward to solutions that fall outside the feasible set [32, Ch. 9]. A suitable penalty function is of the form:

$$H(\mathbf{x}_t, \mathbf{u}_t) \triangleq -\left(\max\left[0, g(\mathbf{x}_t, \mathbf{u}_t)\right]^\rho\right) \quad (23)$$

where $\rho$ is some positive integer. The new unconstrained MSCOP—to which we can apply the surveyed RL algorithms—becomes:

$$\mathcal{P}_2: \ \underset{\pi \in \Pi}{\text{maximize}} \quad \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t J(\mathbf{x}_t, \mathbf{u}_t, \sigma_t) + H(\mathbf{x}_t, \mathbf{u}_t)\right] \quad \text{s.t.} \quad \mathbf{u}_t \sim \pi, \ \ \mathbf{x}_{t+1} \sim p_{\mathbf{x}}(\cdot|\mathbf{x}_t, \mathbf{u}_t), \ \ \mathbf{x}_0 \ \text{ given} \quad (24)$$

Among the surveyed algorithms, we choose Exa [27, 33] for its simplicity and, at the same time, its capability of learning complex non-parametric policies. This algorithm resembles some features

from the discretization approach. It discretizes the state-action sets and estimate the Q-values (i.e., a variation of (2) that takes the state-action pair $(\mathbf{x}_0, \mathbf{u}_0)$ as input and estimates the cumulative reward for $\mathbf{u}_t \sim \pi$ for $t = 1, \ldots, \infty$) for every discrete state-action pair. Since the Q-values are given for every discrete state-action pair, we can maximize over the actions to obtain the highest value at every discrete state. This is basically the well known *Q-learning* algorithm applied over the discretized sets (see, e.g., [13]). Then, Exa computes the distance from $\mathbf{x}_{t+1}$ to each of the discrete points, takes the $k$ nearest points to $\mathbf{x}_{t+1}$, and returns an average of the actions that maximize their Q-values.

## 5. APPLICATION EXAMPLE: NETWORK FLOW

Consider a game in which $Q$ users aim to maximize their throughput through a network with $S$ relay nodes, while optimizing the use of the relays' batteries. Each player can use multiple paths simultaneously; let $\mathbf{u}_t^i$ be the vector of flow rates across these paths (see Fig. 1). The system-state $\mathbf{x}_t$ is the relays' battery levels. These batteries deplete proportionally to the sum of the outgoing flows, weighted by uniform random variables $\nu_t^{ia} \sim [0, \overline{\nu}]$ that model the channel gain of the outgoing links. Each player reward's is some function $\Gamma : \mathcal{U}^i \to \Re$ of the sum of rates across all available paths, like the square root [34], $\Gamma(\cdot) \triangleq \sqrt{(\cdot)}$. Thus, the stochastic dynamic network flow game is given by

$$\begin{aligned}
\underset{\{u_t^i\} \in \prod_{t=0}^{\infty} \mathcal{U}^i}{\text{maximize}} \quad & \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \left(\alpha \Gamma\left(\sum_{a=1}^{A^i} u_t^{ia}\right) + (1-\alpha) \sum_{k=1}^{S} x_t^k\right)\right] \\
\text{s.t.} \quad & x_{t+1}^k = x_t^k - \sum_{i=1}^{Q} \sum_{u^{ia} \in F_k} \nu_t^{ia} u_t^{ia} \qquad (25) \\
& x_0^k = B_{\max}^k, \;\; 0 \le x_t^k \le B_{\max}^k, \;\; \nu_t^{ia} \sim [0, \overline{\nu}] \\
& \mathbf{M}\mathbf{u}_t \le \mathbf{c}_{\max}, \;\; u_t^{ia} \ge 0 \\
& \forall i \in \mathcal{Q}, \;\; k = 1, \ldots, S, \;\; a = 1, \ldots, A^i
\end{aligned}$$

where $0 < \alpha < 1$ is some weight, $B_{\max}^k$ is the maximum battery level, $\mathbf{c}_{\max} \in \Re^L$ is the vector with maximum capacity for each of the $L$ links; and $\mathbf{M} = [m_{lj}]$ is the $L \times A$ connectivity-matrix between links and users' flows, such that the element $m_{la} = 1$ if link $l$ can be used by path $a$, and $m_{la} = 0$ otherwise. This model generalizes the dynamic game proposed by [10] by making the battery depletion factor a random variable. However, the main novelty of this section is that we learn an MNE of (25) with no knowledge of the reward or state-dynamics, just by interaction with a black-box that simulates the system. At every time step, we observe $\mathbf{x}_t$ and take $\mathbf{u}_t$, then the system returns stochastic reward and $\mathbf{x}_{t+1}$ values.

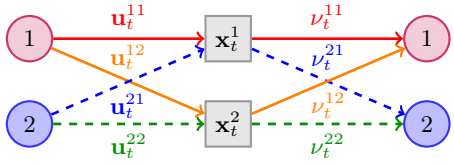


**Fig. 1**. Network flow model. $Q = 2$ users, $S = 2$ relays, and $L = 4$ links (one link at the input of each relay and destination node).

It is straightforward to see that each player's utility can be expressed in separable form. Hence, Corollary 1 establishes that (25) is a SDPG. From Theorem 1, we find its equivalent SMOCP:

$$\begin{aligned}
\underset{\pi \in \Pi}{\text{maximize}} \quad & \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \left(\sum_{i \in \mathcal{Q}} \alpha \Gamma\left(\sum_{a=1}^{A^i} u_t^{ia}\right) + (1-\alpha) \sum_{k=1}^{S} x_t^k\right)\right] \\
\text{s.t.} \quad & x_{t+1}^k = x_t^k - \sum_{i \in \mathcal{Q}} \sum_{u^{ia} \in F_k} \nu_t^{ia} u_t^{ia} \qquad (26) \\
& x_0^k = B_{\max}^k, \;\; 0 \le x_t^k \le B_{\max}^k, \;\; \nu_t^{ia} \sim [0, \overline{\nu}] \\
& \mathbf{M}\mathbf{u}_t \le \mathbf{c}_{\max}, \;\; \mathbf{u}_t \ge 0, \;\; \mathbf{u}_t \sim \pi(\cdot|\mathbf{x}_t) \\
& k = 1, \ldots, S, \;\; a = 1, \ldots, A^i
\end{aligned}$$

Since problem (25) is concave, we can find its optimal solution with convex optimization methods for long enough time horizon (200 steps) and constant gain $\nu_t^{ia} = \bar{\nu}/2$, and use it as benchmark for the RL solution. We simulate problem (26) extended with penalty (23) in the scenario shown in Figure 1. Figure 2 shows results averaged over 500 trials. Although the RL plots (solid) are different from the convex solution (dashed), they follow similar trend. Indeed, both methods provide similar values: $v^{\text{cvx}} = 16.4$ vs. $v^{\text{exa}} = 16.0$, such that the RL solution achieves 97.4% of the optimal value.

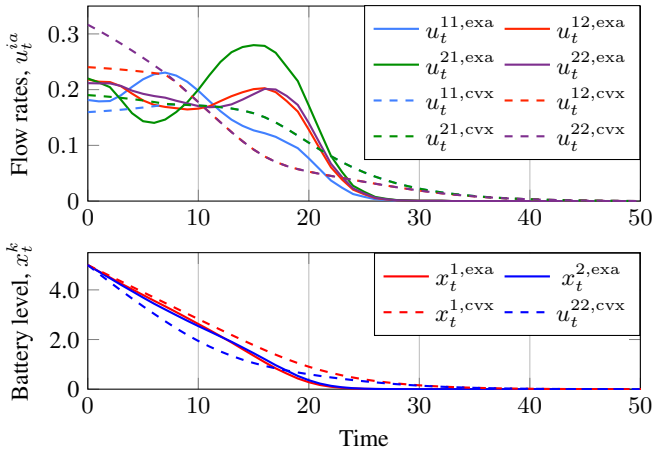


**Fig. 2**. Simulation results Exa [27] (solid) vs. CVX [35] (dashed). Problem parameters: $\beta = 0.9$, $\alpha = 0.9$, $B_{\max}^{1,2} = 5$, $\bar{\nu} = B_{\max}^{1,2}/8$, $\mathbf{c}_{\max} = B_{\max}^{1,2} \cdot [0.08, 0.4, 0.08, 0.4]$. Exa parameters: state grid covers $[0, B_{\max}^{1,2}]^2$ with $10^2$ points, action grid covers $[0, B_{\max}^{1,2} \cdot 0.1]^4$ with $8^4$ points, $k = 2^4$ nearest neighbors, learning rate $5/t^{0.2}$, eligibility trace 0.9, exploration 5%-greedy, run for $3 \cdot 10^4$ episodes.

## 6. CONCLUSIONS

We have extended earlier works on potential games by addressing the most general case: stochastic time varying environment, stochastic rewards, non-reduced form and constrained state-action sets. By applying the stochastic Euler equation to the Lagrangian, we found conditions for the game to be equivalent to a SMOCP. Once we have found this SMOCP, we have shown that it is possible to learn its solution with an RL approach, even when the reward and state transition models are unknown. However, this is a difficult problem for standard RL algorithms because the state and action are continuous vector variables that lie in constrained sets. We have reviewed the RL literature and found algorithms that can work for continuous variables. Then, we extended these algorithms by including a penalty function when the solution does not satisfy the constraints. As an example, we applied this framework to a network flow optimization problem, obtaining near optimal results.

## 7. REFERENCES


[1] M. E. Slade, "What does an oligopoly maximize?," *The Journal of Industrial Economics*, vol. 42, no. 1, pp. 45–61, Mar. 1994.

[2] D. Monderer and L. S. Shapley, "Potential games," *Games and Economic Behavior*, vol. 14, no. 1, pp. 124–143, May 1996.

[3] G. Scutari, S. Barbarossa, and D.P. Palomar, "Potential games: A framework for vector power control problems with coupled constraints," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, May 2006.

[4] O. Candogan, I Menache, A Ozdaglar, and P.A Parrilo, "Flows and decompositions of games: Harmonic and potential games," *Mathematics of Operations Research*, vol. 36, no. 3, pp. 474–503, 2011.

[5] W. D. Dechert, "Optimal control problems from second-order difference equations," *Journal of Economic Theory*, vol. 19, no. 1, pp. 50–63, Oct. 1978.

[6] D. González-Sánchez and O. Hernández-Lerma, "Dynamic potential games: The discrete-time stochastic case," *Dynamic Games and Applications*, pp. 1–20, Mar. 2014.

[7] W. D. Dechert, "Non cooperative dynamic games: a control theoretic approach," Tech. Rep., 1997.

[8] D. Dragone, L. Lambertini, G. Leitmann, and A. Palestini, "Hamiltonian potential functions for differential games," *Proceedings of IFAC CAO*, vol. 9, 2009.

[9] S. Zazo, S. Valcarcel Macua, M. Sánchez-Fernández, and J. Zazo, "A new framework for solving dynamic schedulling games," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2015, Brisbane, Australia.

[10] S. Zazo, M. Sánchez-Fernández, S. Valcarcel Macua, and J. Zazo, "Dynamic potential games in communications: Fundamentals and applications," preprint arXiv:1509.01313, 2015.

[11] D.P. Bertsekas, *Nonlinear programming*, Athena Scientific, 1999.

[12] T.M. Apostol, *Calculus: Multi-variable calculus and linear algebra, with applications to differential equations and probability*, Wiley, 1969.

[13] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 1998.

[14] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, 1996.

[15] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, Athena Scientific, 2012.

[16] C. Szepesvari, *Algorithms for Reinforcement Learning*, Morgan & Claypool Publishers, 2009.

[17] L. Busoniu, R. Babuska, D. Schutter, and D. Ernst, *Reinforcement Learning and Dynamic Programming Using Function Approximators*, CRC Press, 2010.

[18] M. Wiering and M. van Otterlo, *Reinforcement Learning: State-of-the-Art*, Springer, 2012.

[19] A. Geramifard, T. J. Walsh, S. Tellex, G. Chowdhary, N. Roy, and J. P. How, "A tutorial on linear function approximators for dynamic programming and reinforcement learning," *Foundations and Trends in Machine Learning*, vol. 6, no. 4, pp. 375–451, 2013.

[20] G. Konidaris, "Value function approximation in reinforcement learning using the fourier basis," in *Proc. AAAI Conf. on Artificial Intelligence*, Chicago, Illinois, July 2008.

[21] A. Antos, C. Szepesvari, and R. Munos, "Fitted Q-iteration in continuous action-space MDPs," in *Advances in Neural Information Processing Systems 20 (NIPS)*, Vancouver, Canada, Dec. 2007.

[22] H. Van Hasselt and M. et al. Wiering, "Reinforcement learning in continuous action spaces," in *Proc. IEEE Int. Symp. on Approximate Dynamic Programming and Reinforcement Learning (ADPRL)*, Honolulu, Hawaii, Apr. 2007, pp. 272–279.

[23] F. S. Melo and M. Lopes, "Fitted natural actor-critic: A new algorithm for continuous state-action mdps," in *Machine Learning and Knowledge Discovery in Databases*, vol. 5212, pp. 66–81. Springer, 2008.

[24] M. G. Azar, A. Lazaric, and E. Brunskill, "Online stochastic optimization under correlated bandit feedback," in *Proc. Int. Conf. on Machine Learning (ICML)*. Beijing, 2014, JMLR Workshop and Conference Proceedings.

[25] J. Millán, D Posenato, and E. Dedieu, "Continuous-action Q-learning," *Machine Learning*, vol. 49, no. 2-3, pp. 247–265, 2002.

[26] A. Lazaric, M. Restelli, and A. Bonarini, "Reinforcement learning in continuous action spaces through sequential monte carlo methods," in *Advances in neural information processing systems (NIPS)*, Vancouver, Canada, Dec. 2007, pp. 833–840.

[27] J. A. Martin H and J. de Lope, "Ex$\langle a \rangle$: An effective algorithm for continuous actions reinforcement learning problems," in *IEEE Annual Conf. of Industrial Electronics (IECON)*, Nov 2009, pp. 2063–2068.

[28] G. Neumann and J. R. Peters, "Fitted q-iteration by advantage weighted regression," in *Proc. Advances in neural information processing systems (NIPS)*, Vancouver, Canada, Dec. 2009, pp. 1177–1184.

[29] J. Pazis and M.G. Lagoudakis, "Reinforcement learning in multidimensional continuous action spaces," in *IEEE Symp. on Adaptive Dynamic Programming And Reinforcement Learning (ADPRL)*, Paris, France, Apr. 2011.

[30] H. Van Hoof, J. Peters, and G. Neumann, "Learning of non-parametric control policies with high-dimensional state features," in *Proc. Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, San Diego, California, May 2015, pp. 995–1003.

[31] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," preprint arXiv:1509.02971v1, 2015.

[32] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear programming: theory and algorithms*, Wiley-Interscience, Hoboken, N.J, 3rd ed edition, 2006.

[33] J. A. Martin H, J. de Lope, and D. Maravall, "Robust high performance reinforcement learning through weighted k-nearest neighbors," *Neurocomputing*, vol. 74, no. 8, pp. 1251–1259, 2011.

[34] A. Nedic and A. Ozdaglar, "Cooperative distributed multi-agent optimization," in *Convex Optimization in Signal Processing and Communications*, D. P. Palomar and Y. C. Eldar, Eds. Cambridge University Press, 2010.

[35] M. Grant and Stephen Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," Mar. 2014.