2024 American Control Conference (ACC)
July 8-12, 2024. Toronto, Canada

Risk-Aware Distributed Multi-Agent Reinforcement Learning

Abdullah Al Maruf[1], Luyao Niu[2], Bhaskar Ramasubramanian[3], Andrew Clark[4], Radha Poovendran[2]

Abstract— **Autonomous cyber and cyber-physical systems need to perform decision-making, learning, and control in unknown environments. Such decision-making can be sensitive to multiple factors, including modeling errors, changes in costs, and impacts of events in the tails of probability distributions. Although multi-agent reinforcement learning (MARL) provides a framework for learning behaviors through repeated interactions with the environment by minimizing an average cost, it is not adequate to overcome the above challenges. In this paper, we develop a distributed MARL approach to solve decision-making problems in unknown environments by learning risk-aware actions. We use the conditional value-at-risk (CVaR) to define the cost function that is being minimized, and introduce a Bellman operator to characterize the value function associated to a given state-action pair. We prove that this operator satisfies a contraction property, and that it converges to the optimal value function. We then propose a distributed MARL algorithm called the *CVaR QD-Learning* algorithm, and establish that value functions of individual agents reach consensus. We identify several challenges that arise in the implementation of the *CVaR QD-Learning* algorithm, and present solutions to overcome these. We evaluate the *CVaR QD-Learning* algorithm through simulations, and demonstrate the effect of a risk parameter on value functions at consensus.**

I. Introduction

Reasoning about the satisfaction of objectives for complex cyber and cyber-physical systems (e.g., autonomous vehicles, smart manufacturing) typically involves solving a sequential decision-making problem [1]. The operating environment is represented as a Markov decision process (MDP) [2], and transitions between any two states in the system is a probabilistic outcome based on the actions of the decision maker or agent. In dynamic and uncertain environments, the frameworks of reinforcement learning [3] and optimal control [4] have been used to solve sequential decision-making problems by determining actions to minimize an accumulated cost. *Risk-neutral* decision-making solutions determine actions by minimizing an expected or average cost; such techniques have been implemented in applications including robotics, mobile networks, and games [5]–[11].

[1]Department of Electrical and Computer Engineering, California State University, Los Angeles, CA 90032, USA. amaruf@calstatela.edu

[2]Network Security Lab, Department of Electrical and Computer Engineering, University of Washington, Seattle, WA 98195, USA. {luyaoniu, rp3}@uw.edu

[3]Electrical and Computer Engineering, Western Washington University, Bellingham, WA 98225, USA. ramasub@wwu.edu

[4]Electrical and Systems Engineering, Washington University in St. Louis, St. Louis, MO 63130, USA. andrewclark@wustl.edu

This work was supported by the AFOSR grants FA9550-22-1-0054 and FA9550-23-1-0208, and by the US National Science Foundation CNS-2153136.

Although risk-neutral solutions are computationally tractable, they have been shown to have limitations in characterizing sensitivity to changes in costs, modeling errors, and the effect of tails of probability distributions [12]–[14]. In order to solve a sequential decision-making problem by learning *risk-aware* actions, we consider the *conditional value-at-risk (CVaR)* [15]–[18] to characterize the objective function of an MDP. The CVaR corresponds to the average value of the cost conditioned on the event that the cost takes sufficiently large values, and was shown to have strong theoretical justification for its use in [15], [19]. Optimizing a CVaR-based cost will ensure sensitivity of actions to rare high-consequence outcomes [20]. However, different from [19], we assume that the operating environment of the agent is unknown. The agent then learns behaviors by minimizing a cost that is revealed through repeated interactions with the environment. For this setting, we develop a CVaR-based variant of Bellman operator [3], and establish its convergence to an optimal solution.

In multi-agent reinforcement learning (MARL), each agent interacts with both the environment and other agents. Consequently, the evolution of agents' behaviors has been shown to be non-stationary from the perspective of any single agent [21]–[23], making the problem more challenging compared to the single-agent case. The literature examining incorporation of risk-sensitivity in MARL is limited. Recently, the authors of [24], [25] developed a deep-learning based framework to learn risk-sensitive policies in cooperative MARL using CVaR. The algorithms proposed in the above works use the centralized training with decentralized execution (CTDE) paradigm [26] to learn behaviors. An agent using CTDE can use information about other agents' observations and actions to aid its own learning during training, but will have to take decisions independently at test-time [27], [28].

Different from the above works, in this paper, we design a distributed *risk-aware multi-agent reinforcement learning* algorithm. Our solution is inspired by QD-learning [29], wherein at each step, a single update rule incorporates costs revealed by the environment and information from neighboring agents in a graph that describes inter-agent communication. We establish the consensus of agents' value functions when they are optimizing a CVaR-based cost. Our simulations also reveal that as agents become more risk-aware, their value functions at consensus increase in magnitude (corresponding to higher costs incurred); this observation agrees with intuition when the goal is to minimize an accumulated cost. We make the following specific contributions:

979-8-3503-8265-5/$31.00 ©2024 AACC

- We define a Bellman operator to characterize a CVaR-based state-action value function.
- We prove that the Bellman operator is a contraction, and that the fixed point of the operator is the optimal risk-aware value function.
- We develop a risk-aware distributed multi-agent reinforcement learning algorithm called *CVaR QD-Learning* and prove that CVaR-based value functions of individual agents reach consensus.
- We carry out numerical simulations to validate the *CVaR QD-Learning* algorithm, and show that value functions at consensus increase in magnitude as agents become more risk-aware.

The remainder of this paper is organized as follows: Sec. II introduces necessary preliminaries on reinforcement learning and risk-aware decision making. Sec. III presents construction of a Bellman operator, and shows that it is a contraction. Sec. IV presents CVaR-based QD-learning and associated analytical results, and Sec. V presents the *CVaR QD-Learning* algorithm and describes how challenges in the implementation of the algorithm are overcome. Sec. VI shows results of evaluations through simulations and Sec. VII concludes the paper.

II. Setup and Problem Formulation

This section introduces the Markov game setup that we consider, provides necessary preliminaries on reinforcement learning and risk criteria used in decision-making. We then formally state the problem that we will solve in this paper.

A. Setup

We consider a system with N agents. Inter-agent communication is described by an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \cdots, N\}$ is the set of vertices (or nodes) and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is the set of edges between pairs of nodes. Here the nodes and the edges in the graph $\mathcal{G}$ correspond to the agents and the communication link between agents. We assume that $\mathcal{G}$ is simple (no self-loops or multiple edges between any two nodes) and connected (there is a path between every pair of nodes). The set of neighbors of agent n is denoted by $\mathcal{N}(n)$. The graph can be described by an $N \times N$ Laplacian matrix L with entries $L_{ij} = -1$ if $(i,j) \in \mathcal{E}$ or otherwise zero when $i \neq j$, and $L_{ii} = |\mathcal{N}(i)|$ which is equal to the degree of the node i. Since $\mathcal{G}$ is connected, the eigenvalues of L can be ordered as $0 = \lambda_1(L) < \lambda_2(L) \leq \cdots \leq \lambda_N(L)$ [30].

The multi-agent setup we consider here is similar to that of [29]. Specifically, similar to [29] we assume that each agent can fully observe the undertaken action and state of the system. However, the cost/reward received by each agent is local and may not be available to a remotely located controller. As an example, this multi-agent setup can resemble spatially distributed temperature sensors (agents) in a building [29]. A remote controller will have access to all sensor readings but will not be aware of the desired temperatures at different rooms in the building. A possible objective of the controller could be to minimize the average squared deviation between measured temperatures from sensors and their corresponding locations' desired temperatures. Though the assumption of globally observable state and action is restrictive, however this enables establishment of theoretical guarantee [29], [31].

In such a setting, behaviors of agents in the environment can be described by a Markov game $M := (\mathcal{S}, \mathcal{A}, \{c_1, \cdots c_N\}, P, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are assumed to be finite state and action spaces. When the system is in state s and the action taken by the controller is a, the agent n incurs a bounded and deterministic local cost $c_n(s,a) \in [-C_{max}, C_{max}]$. We emphasize that in our setup the state $s \in \mathcal{S}$ and $a \in \mathcal{A}$ are global (i.e. common to all agents) whereas individual costs $c_n(s,a)$ are local to each agent. $P(s'|s,a)$ gives the probability of transitioning to state $s' \in \mathcal{S}$ when taking action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$, and $\gamma \in [0,1)$ is the discounting factor. However, different from [29], we assume that costs are deterministic to aid the development of our theoretical results.

A trajectory of M is an alternating sequence of states and actions $(s_0, a_0, s_1, a_1, \dots)$. A history up to time k, denoted as $h_k \in H_k$, corresponds to a trajectory up to time k i.e. $(s_0, a_0, s_1, a_1, \dots, s_k)$. Formally, with $H_0 = S$, we recursively define the set of possible histories up to time $k \geq 1$ as $H_k = H_{k-1} \times \mathcal{A} \times \mathcal{S}$. A policy at time k is a map $\Pi_{H,k} : H_k \to A$, and we define $\Pi_H := \lim_{k\to\infty} \Pi_{H,k}$ to be the set of all history-dependent policies. A policy $\mu(s_k)$ is called Markov when it only depends on the current state s_k.

Let $c(s_k, a_k) = \frac{1}{N}\sum_{n=1}^{N} c_n(s_k, a_k)$ be the average costs over all agents observed at time k. The discounted average cost up to time k is defined as $C_{0,k} := \sum_{t=0}^{k} \gamma^t c(s_t, a_t) = \frac{1}{N}\sum_{t=0}^{k}\gamma^t \sum_{n=1}^{N} c_n(s_t, a_t)$. Thus discounted average cost over the infinite horizon is given by $C_{0,\infty} = \lim_{k\to\infty} C_{0,k}$.

In reinforcement learning (RL), the transition probability $P(s'|s,a)$ is not known, and costs are revealed to agents through repeated interactions with the environment [3]. The objective is to learn a policy that minimizes the expected accumulated discounted cost. One widely-used approach to learn such a policy is the Q-learning algorithm [32]. Using a state-action value function $Q^\pi(s,a) := \mathbb{E}_\pi[C_{0,\infty}|s_0 = s, a_0 = a, \pi]$, the Q-learning algorithm seeks to find the optimal value $Q^*(s,a)$ corresponding to the optimal policy π^* such that $Q^*(s,a) \leq Q^\pi(s,a)$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ and any policy π.

B. QD-Learning

QD-learning is a multi-agent distributed variant of Q-learning that was first proposed in [29] and has recently been applied to develop distributed algorithms for other settings [31]. In the QD-learning algorithm [29], at time-step k, each agent n maintains a sequence of state-action value functions $\{Q_{n,k}(s,a)\} \in \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$ for all state-action pairs $(s,a) \in \mathcal{S} \times \mathcal{A}$. The sequence $\{Q_{n,k}(s,a)\}$ is updated according to the following rule [29]:

$$Q_{n,k+1}(s_k, a_k) = (1-\alpha_k)Q_{n,k}(s_k, a_k) + \alpha_k\big(c_n(s_k, a_k) + \gamma \min_{a' \in \mathcal{A}} Q_{n,k}(s_{k+1}, a')\big)$$

$$- \beta_k \sum_{l \in \mathcal{N}(n)} \big(Q_{n,k}(s_k, a_k) - Q_{l,k}(s_k, a_k)\big), \quad (1)$$

where the weight sequences $\{\alpha_k\}$ and $\{\beta_k\}$ are given by

$$\alpha_k = \frac{a}{(k+1)^{\tau_1}}, \quad (2)$$

$$\beta_k = \frac{b}{(k+1)^{\tau_2}}, \quad (3)$$

with a and b being positive constants. Eqns. (2) and (3) guarantee that the excitation for the innovation and consensus terms in Eqn. (1) are persistent; i.e., $\sum_k \alpha_k = \infty$ and $\sum_k \beta_k = \infty$. The sequences $\{\alpha_k\}$ and $\{\beta_k\}$ further satisfy $\sum_k \alpha_k^2 < \infty$, $\sum_k \beta_k^2 < \infty$ and $\frac{\beta_k}{\alpha_k} \to \infty$ as $k \to \infty$. Constants a and b in Eqns. (2) and (3) are also chosen so that $(I_N - \beta_k L - \alpha_k I_N)$ is positive semidefinite for all k where I_N denotes the identity matrix in $\mathbb{R}^{N \times N}$. One way to ensure such positive semidefiniteness is to choose a and b so that $a + Nb \leq 1$. We refer to [29] for a detailed discussion on the selection of the weight sequence. When the weight sequences are chosen appropriately, the authors of [29] showed that the sequence $\{Q_{n,k}(s,a)\}$ asymptotically reaches consensus for all agents and the value achieved at consensus was equal to the optimal action-value function $Q^*(s,a)$.

C. Conditional Value-at-Risk (CVaR)

Let Z be a bounded random variable on the probability space $(\Omega, \mathcal{F}, P)$ with cumulative distribution $F(z) = P(Z \leq z)$. The value-at-risk (VaR) at confidence level $y \in (0,1]$ is $VaR_y(Z) := min\{z | F(z) \geq 1 - y\}$ [15]. The conditional value-at-risk (CVaR) at confidence level y is defined as $CVaR_y(Z) := \mathbb{E}[Z | Z \geq VaR_y(Z)]$ [15], and represents the expected value of Z, conditioned on the y quantile of the tail distribution. We note that $CVaR_y(Z) = \mathbb{E}(Z)$ when $y = 1$ and $CVaR_y(Z) \to max\{Z\}$ as $y \to 0$. The CVaR of a random variable Z can be interpreted as the worst-case expectation of Z under a perturbed distribution ξP, as given by the following result from [19], [33].

Proposition 1 (Dual CVaR formulation [19], [33])**.** *Let $\mathbb{E}_\xi[Z]$ denote the ξ-weighted expectation of Z and $\mathcal{U}_{CVaR}(y,P) := \{\xi : \xi(\omega) \in [0, \frac{1}{y}], \int_{\omega \in \Omega} \xi(\omega) P(\omega) d\omega = 1\}$. Then,*

$$CVaR_y(Z) = \max_{\xi \in \mathcal{U}_{CVaR}(y,P)} \mathbb{E}_\xi[Z].$$

The above dual representation, together with the coherent property of the CVaR metric was used to derive a decomposition of $CVaR_y(Z)$ in a recursive manner [19], [34]. In our setting, the random variable Z is a sequence of costs.

Proposition 2 (CVaR decomposition [19])**.** *For $k \geq 0$, let $Z = \big(Z_{k+1}, Z_{k+2}, \cdots\big)$ denote the sequence of costs starting from time $k+1$ to onward. Then, the conditional CVaR, under a policy π satisfies*

$$CVaR_y(Z | h_k, \pi) = \max_{\xi} \mathbb{E}[\xi(s_{k+1}) CVaR_{y\xi(s_{k+1})}(Z | h_{k+1}, \pi) | h_k, \pi],$$

where $\xi \in \mathcal{U}_{CVaR}(y, P(\cdot | s_t, a_t))$, and the expectation is with respect to s_{k+1}.

D. Problem Formulation

Our objective in this paper is to learn risk-aware policies for multi-agent reinforcement learning in a distributed manner. We use CVaR as a measure of risk sensitivity, and aim to learn a policy $\pi \in \Pi_H$ that will minimize a risk-sensitive discounted cost. We use the terms *confidence level* and *risk parameter* interchangeably to refer to the parameter $y \in (0,1]$. The challenge in solving this problem is that the transition probabilities are not known, and costs associated to taking an action in a particular state are revealed to agents only through repeated interactions with the environment. We formally state the problem that we want to address as:

$$\min_{\pi \in \Pi_H} CVaR_y\big(C_{0,\infty} | s_0, \pi\big). \quad (4)$$

In order to solve the problem given in Eqn. (4), we will first propose a CVaR-based Bellman operator for Q-learning and show that the operator has a fixed point that corresponds to the optimal solution. Then we will design a CVaR-based QD-learning algorithm and show that the value functions of individual agents reach a consensus. Through simulation, we also verify the convergence of the algorithm.

III. Bellman Operator Construction

In this section, we define a Bellman operator in order to compute the optimal state-action value function for CVaR-based reinforcement learning. Similar to techniques used for other variants of Q-learning, e.g., [14], [32], [35], [36], we establish the convergence of this operator by showing that it satisfies a contraction property. Of particular relevance is the result for CVaR-based dynamic programming for the convergence of state value functions of MDPs (whose transition probabilities and cost structures are known apriori) presented in [19].

Leveraging Proposition 2, we first augment the state-action pair (s,a) with an additional 'continuous state' $y \in \mathcal{Y} = (0,1]$ which represents the confidence level for CVaR. Then, for a given policy π and CVaR confidence level $y \in \mathcal{Y}$, we define the augmented state-action value function as:

$$Q^\pi(s,a,y) := CVaR_y(C_{0,\infty} | s_0 = s, a_0 = a, \pi). \quad (5)$$

To set up a dynamic programming characterization of $Q^\pi(s,a,y)$, we define a Bellman operator on the space of augmented state-action value functions. Consider the *CVaR Bellman operator* $\mathbf{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{Y} \to \mathcal{S} \times \mathcal{A} \times \mathcal{Y}$ and $\xi \in \mathcal{U}_{CVaR}(y, P(\cdot|s,a))$. Then, we define

$$\mathbf{T}[Q](s,a,y) := c(s,a) + \gamma \min_{a' \in \mathcal{A}} \Big[\max_{\xi} \sum_{s' \in \mathcal{S}} \xi(s') P(s'|s,a) Q(s', a', y\xi(s')) \Big]. \quad (6)$$

Our first result formalizes the fact that the Bellman operator defined in Eqn. (6) satisfies a contraction property.

Lemma 1 (Contraction)**.** *Let $Q^1(s,a,y)$ and $Q^2(s,a,y)$ be two augmented state-action value functions for the same (s,a,y) as defined in Eqn. (5). Then, the Bellman operator $\mathbf{T}$ defined in Eqn.* (6) *is a contraction under the sup-norm. That*

is, $||\mathbf{T}[Q^1](s,a,y) - \mathbf{T}[Q^2](s,a,y)||_\infty \leq \gamma||Q^1(s,a,y) - Q^2(s,a,y)||_\infty$.

Proof. Our proof uses an argument similar to [19]. However, different from [19], the environment in our setting is unknown; this will require reasoning about state-action value functions [3], rather than state value functions used in [19].

Since $\xi(s')P(s'|s,a') \geq 0$ for all $\xi \in \mathcal{U}_{CVaR}(y, P(\cdot|s,a))$, we have $\mathbf{T}[Q^1](s,a,y) \leq \mathbf{T}[Q^2](s,a,y)$ whenever $Q^1(s,a,y) \leq Q^2(s,a,y)$ for all $s \in \mathcal{S}, a \in \mathcal{A}, y \in \mathcal{Y}$. Moreover, since $\sum_{s' \in \mathcal{S}} \xi(s')P(s'|s,a') = 1$ for all $\xi \in \mathcal{U}_{CVaR}(y, P(\cdot|s,a))$, we have $\mathbf{T}[Q + c](s,a,y) = c\gamma + \mathbf{T}[Q](s,a,y)$ where c is a constant. Thus, we have established that $\mathbf{T}[Q](s,a,y)$ exhibits the monotonicity and constant shift properties. Now, from the definition of sup-norm, for all $s \in \mathcal{S}, a \in \mathcal{A}, y \in \mathcal{Y}$ we have $|Q^1(s,a,y) - Q^2(s,a,y)| \leq ||Q^1(s,a,y) - Q^2(s,a,y)||_\infty$ which is equivalent to

$$Q^2(s,a,y) - ||Q^1(s,a,y) - Q^2(s,a,y)||_\infty \leq Q^1(s,a,y) \leq Q^2(s,a,y) + ||Q^1(s,a,y) - Q^2(s,a,y)||_\infty. \quad (7)$$

Applying the Bellman operator to the terms in Eqn. (7) and leveraging the monotonicity and constant shift properties with $c = \pm||Q^1(s,a,y) - Q^2(s,a,y)||_\infty$, for all $s \in \mathcal{S}, a \in \mathcal{A}, y \in \mathcal{Y}$, we obtain

$$\mathbf{T}[Q^2](s,a,y) - \gamma||Q^1(s,a,y) - Q^2(s,a,y)||_\infty \leq \mathbf{T}[Q]^1(s,a,y) \leq \mathbf{T}[Q]^2(s,a,y) + \gamma||Q^1(s,a,y) - Q^2(s,a,y)||_\infty. \quad (8)$$

This is equivalent to $|\mathbf{T}[Q^1](s,a,y) - \mathbf{T}[Q^2](s,a,y)| \leq \gamma||Q^1(s,a,y) - Q^2(s,a,y)||_\infty \; \forall s \in \mathcal{S}, a \in \mathcal{A}, y \in \mathcal{Y}$, which completes the proof. □

Lemma 1 allows us to establish the convergence of the CVar Bellman operator $\mathbf{T}$ to a fixed point $Q^f(s,a,y)$ when applied repeatedly. Let $Q^*(s,a,y)$ be the optimal state-action value function with respect to CVaR confidence level y i.e.

$$Q^*(s,a,y) = \min_{\pi \in \Pi_H} CVaR_y(\mathcal{C}_{0,\infty}|x_0 = s, a_0 = a, \pi). \quad (9)$$

We now derive an intermediate result that will allow us to show the optimality of the fixed point of CVaR Bellman operator i.e. $Q^f(s,a,y) = Q^*(s,a,y)$. To do so, by $\mathbf{T}^k$ we denote the application of the Bellman operator $\mathbf{T}$ for k times for some $k \in \mathbb{N}$.

Lemma 2. *Let* $Q_k(s,a,y) := \mathbf{T}^k[Q_0](s,a,y)$ *for all* $s \in \mathcal{S}, a \in \mathcal{A}, y \in \mathcal{Y}$ *where* $k \in \mathbb{N}$ *and* Q_0 *is an arbitrary initial value of* $Q(s,a,y)$ *for all* $s \in \mathcal{S}, a \in \mathcal{A}, y \in \mathcal{Y}$. *Then* $Q_k(s,a,y) = \min_{\pi \in \Pi_k} CVaR_y(\mathcal{C}_{0,k-1} + \gamma^k Q_0|s_0 = s, a_0 = a, \pi)$.

Proof. Similar to [19], we prove this result using induction. For the base case i.e. $k = 1$, we have

$$\begin{aligned}
Q_1(s,a,y) &= \mathbf{T}[Q_0](s,a,y) = c(s,a) + \gamma \min_{a' \in \mathcal{A}} \Big[\max_{\xi \in \mathcal{U}_{CVaR}(y,P(\cdot|s,a))} \sum_{s' \in \mathcal{S}} \xi(s')P(s'|s,a)Q_0(s',a',y\xi(s'))\Big] \\
&= c(s,a) + \min_{a' \in \mathcal{A}} \Big[\gamma \max_{\xi \in \mathcal{U}_{CVaR}(y,P(\cdot|s,a))} \mathbb{E}_{\xi(s')}[Q_0 \\
&\qquad |s_0 = s', a_0 = a', \pi]\Big] \\
&= \min_{a' \in \mathcal{A}} \Big[c(s,a) + \gamma \max_{\xi \in \mathcal{U}_{CVaR}(y,P(\cdot|s,a))} \mathbb{E}_{\xi(s')}[CVaR_{y\xi(s')} \\
&\qquad (Q_0|s',a')|s_0 = s, a_0 = a, \pi)]\Big] \\
&= \min_{a' \in \mathcal{A}} \Big[c(s,a) + \gamma CVaR_y(Q_0|s_0 = s, a_0 = a, \pi)\Big] \\
&= \min_{\pi \in \pi_1} CVaR_y(\mathcal{C}_{0,0} + \gamma Q_0|s_0 = s, a_0 = a, \pi).
\end{aligned}$$

Thus the base case is proved. Now we assume that the result holds for $k = i$. Now for $k = i + 1$ we have

$$\begin{aligned}
Q_{i+1}(s,a,y) &= \mathbf{T}^{i+1}[Q_0](s,a,y) = \mathbf{T}[Q_i](s,a,y) \\
&= c(s,a) + \gamma \min_{a' \in \mathcal{A}} \Big[\max_{\xi \in \mathcal{U}_{CVaR}(y,P(\cdot|s,a))} \sum_{s' \in \mathcal{S}} \xi(s')P(s'|s,a) \\
&\qquad Q_i(s',a',y\xi(s'))\Big] \\
&= c(s,a) + \gamma \min_{a' \in \mathcal{A}} \Big[\max_{\xi \in \mathcal{U}_{CVaR}(y,P(\cdot|s,a))} \sum_{s' \in \mathcal{S}} \xi(s')P(s'|s,a) \\
&\qquad \min_{\pi \in \Pi_i} CVaR_{y\xi(s')}(\mathcal{C}_{0,i-1} + \gamma^i Q_0|s_0 = s', a_0 = a', \pi)\Big] \\
&= \min_{a' \in \mathcal{A}} \Big[c(s,a) + \max_{\xi \in \mathcal{U}_{CVaR}(y,P(\cdot|s,a))} \mathbb{E}_{\xi(s')}[\min_{\pi \in \Pi_i} \\
&\qquad CVaR_{y\xi(s')}(\mathcal{C}_{1,i} + \gamma^{i+1} Q_0|s',a')|s_0 = s, a_0 = a, \pi)]\Big] \\
&= \min_{\pi \in \pi_{i+1}} CVaR_y(\mathcal{C}_{0,i} + \gamma^{i+1} Q_0|s_0 = s, a_0 = a, \pi). \quad (10)
\end{aligned}$$

Hence the proof is complete by induction. □

The main result of this section proves that the fixed point of the Bellman operator $\mathbf{T}[Q](s,a,y)$ is the optimal value function $Q^*(s,a,y)$.

Theorem 1. *As* $k \to \infty$, $Q_k(s,a,y)$ *converges to a unique fixed point* $\mathbf{T}[Q^*](s,a,y) = Q^*(s,a,y)$ *for all* $s \in \mathcal{S}$, $a \in \mathcal{A}$ *and* $y \in \mathcal{Y}$.

Proof. The proof follows from Lemma 1 and Lemma 2. Since the CVaR Bellman operator $\mathbf{T}[Q](s,a,y)$ has the contraction property, we have that $\lim_{k \to \infty} Q_k(s,a,y)$ converges to a unique fixed point $\mathbf{T}[Q^f](s,a,y) = Q^f(s,a,y)$. From Lemma 2 and using the fact that $\gamma < 1$, we can write

$$\begin{aligned}
&Q^f(s,a,y) \\
&= \lim_{k \to \infty} Q_k(s,a,y) \\
&= \min_{\pi \in \Pi_H} CVaR_y(\lim_{k \to \infty} \mathcal{C}_{0,k-1} + \gamma^k Q_0|x_0 = s, a_0 = a, \pi) \\
&= \min_{\pi \in \Pi_H} CVaR_y(\mathcal{C}_{0,\infty}|x_0 = s, a_0 = a, \pi) \\
&= Q^*(s,a,y).
\end{aligned}$$

Hence the proof is complete. □

We note that the optimal value function $Q^*(s,a,y)$ can be achieved by a stationary (deterministic) Markov policy $\mu^*(s,y)$ where μ^* is a mapping from the current state s and CvaR confidence level y to an action a, i.e., $\mu^* : (\mathcal{S},\mathcal{Y}) \to \mathcal{A}$. We obtain the fixed point $Q^*(s,a,y)$ through repeated application of the CVaR Bellman operator, and then determine a greedy policy μ^* on $Q^*(s,a,y)$ such that $\mu^*(s,y) = \text{argmin}_a\, Q^*(s,a,y)$ [29]. Then, from the definition of $Q^*(s,a,y)$ in Eqn. (9), it follows that the greedy policy μ^* indeed achieves optimality. This makes the problem of finding an optimal policy tractable, even though the original problem in (4) is defined over the set of history-dependent policies.

IV. CVaR-based Distributed Multi-Agent Reinforcement Learning

We use the insight from Theorem 1 to design of a distributed multi-agent reinforcement learning algorithm where agents seek to optimize a risk-aware objective. Such an objective is expressed in terms of a CVaR-based discounted accumulated cost as described in Eqn. (5). The update rule for our *CVaR QD-Learning* algorithm is informed by the QD-learning scheme proposed in [29]. However, optimizing a CVaR-based value function (instead of an expectation-based function in [29]) will require us to develop additional mathematical structure, which will be described below.

At each iteration of the *CVaR QD-Learning* update, the augmented state-action value function of an agent evolves as a weighted sum of (i) the augmented state-action value function at the previous iteration, (ii) an *innovation* term arising from the cost observed by taking a particular action in a given state, and (iii) a *consensus* term corresponding to the difference between augmented state-action values of the agent with its neighbors $\mathcal{N}(\cdot)$ in the inter-agent communication graph $\mathcal{G}$. Specifically, the sequence $\{Q_{n,k}(s,a,y)\}$ evolves for each agent n according to the following equation:

$$\begin{aligned} Q_{n,k+1}(s_k,a_k,y_k) &= (1-\alpha_k)Q_{n,k}(s_k,a_k,y_k) \\ &+ \alpha_k\Big(c_n(s_k,a_k) + \gamma \min_{a'\in\mathcal{A}}\big[\max_{\xi\in\mathcal{U}_{CVaR}(y,P(\cdot|s,a))} \\ &\qquad \xi(s_{k+1})Q_{n,k}(s_{k+1},a',y_k\xi(s_{k+1}))\big]\Big) \\ &- \beta_k \sum_{l\in\mathcal{N}(n)} \big(Q_{n,k}(s_k,a_k,y_k) - Q_{l,k}(s_k,a_k,y_k)\big), \end{aligned} \tag{11}$$

where the weight sequences $\{\alpha_k\}$ and $\{\beta_k\}$ are given by (2) and (3) where $\tau_1 \in (\frac{1}{2},1]$ and $\tau_2 \in (0,\tau_1-\frac{1}{2})$. Like [29], here the weight sequences satisfy $\sum_k \alpha_k = \infty$, $\sum_k \beta_k = \infty$, $\sum_k \alpha_k^2 < \infty$, $\sum_k \beta_k^2 < \infty$ and $\frac{\beta_k}{\alpha_k} \to \infty$ as $k \to \infty$. Constants a and b in Eqns. (2) and (3) also satisfy that $(I_N - \beta_k L - \alpha_k I_N)$ is positive semidefinite for all k.

Since the costs $\{c_n(s,a)\}$ and the parameter $\{\xi(s)\}$ are bounded, and the initial augmented state-action value functions $\{Q_{n,0}(s,a,y)\}$ are chosen to be bounded for all agents n, and for all $s \in \mathcal{S}, a \in \mathcal{A}$ and $y \in \mathcal{Y}$, we can show that $\{Q_{n,k}(s_k,a_k,y_k)\}$ is pathwise bounded, i.e., $P(\sup_k ||Q_{n,k}||_\infty < \infty) = 1$. Our next result establishes that the augmented state-action value functions of the agents asymptotically reach a consensus.

Theorem 2 (Consensus of CVaR QD-Learning)**.** *For the CVaR QD-learning update in Eqn.* (11)*, each agent reaches consensus asymptotically for all $s \in \mathcal{S}, a \in \mathcal{A}$ and $y \in \mathcal{Y}$. That is,*

$$P(lim_{k\to\infty}||Q_{n,k}(s,a,y) - \bar{Q}_k(s,a,y)|| = 0) = 1, \tag{12}$$

where

$$\bar{Q}_k(s,a,y) = \frac{1}{N}\sum_{n=1}^{N} Q_{n,k}(s,a,y). \tag{13}$$

Proof. For agent n we can write the update in Eqn. (11) as:

$$z_{k+1} = (I_N - \beta_k L - \alpha_k I_N)z_k + \alpha_k(U_k + J_k) \tag{14}$$

where L is the Laplacian matrix of the graph $\mathcal{G}$ as defined in Section II and

$$\begin{aligned} z_k &:= [Q_{1,k}(s_k = s, a_k = a, y_k = a)\cdots \\ &\qquad Q_{n,k}(s_k = s, a_k = a, y_k = y)]^T, \\ U_k &:= \gamma \min_{a'\in\mathcal{A}}\big[\max_{\xi\in\mathcal{U}_{CVaR}(y,P(\cdot|s,a))} \xi(s_{k+1} = s') \\ &\qquad Q_{n,k}(s_{k+1} = s', a', y_k\xi(s_{k+1}))\big], \\ J_k &:= c_n(s_k = s, a_k = a). \end{aligned}$$

Let $\hat{z}_k = z_k - \frac{1}{N}1_N^T z_k 1_N$ where 1_N is a vector in $\mathbb{R}^N$ with all entry as 1. Then we can write Eqn. (14) as

$$\hat{z}_{k+1} = (I_N - \beta_k L - \alpha_k I_N)\hat{z}_k + \alpha_k(\hat{U}_k + \hat{J}_k) \tag{15}$$

where $\hat{U}_k = U_k - \frac{1}{N}1_N^T U_k 1_N$ and $\hat{J}_k = U_k - \frac{1}{N}1_N^T J_k 1_N$. Then following the argument of [29] and applying Lemma 4.2 in [29], we can write for $k \geq k_0$

$$||(I_N - \beta_k L - \alpha_k I_N)\hat{z}_k|| \leq (1 - c_2\ r_k)||\hat{z}_k|| \tag{16}$$

where $c_2 \in (0,1)$, $0 \leq r_k \leq 1$ and $k_0 \in \mathbb{N}$. Combining Eqns. (15) and (16), we can write for $k \geq k_0$

$$||\hat{z}_{k+1}|| \leq (1 - c_2\ r_k)||\hat{z}_k|| + \alpha_k(||\hat{U}_k|| + ||\hat{J}_k||). \tag{17}$$

Since $\{Q_{n,k}(s,a,y)\}$, $\{\xi(s)\}$ and $\{c_n(s,a)\}$ are bounded for all agents n and for all $s \in \mathcal{S}, a \in \mathcal{A}$ and $y \in \mathcal{Y}$, we have that $\{||\hat{U}_k||\}$ and $\{||\hat{J}_k||\}$ are bounded. Using Lemma 4.1 of [29] we can conclude that $P((k+1)^\tau \hat{z}_k \to 0) = 1$ as $k \to \infty$ for all $\tau \in (0, \tau_1 - \tau_2 - \frac{1}{2})$. Therefore, we have $P(\hat{z}_k \to 0) = 1 \Rightarrow P(z_k \to \frac{1}{N}1_N^T z_k 1_N) = 1 \Rightarrow P(Q_{n,k}(s,a,y) \to \bar{Q}_n(s,a,y)) = 1$ for all n and all $s \in \mathcal{S}, a \in \mathcal{A}$ and $y \in \mathcal{Y}$ as $k \to \infty$. Hence we recover Eqn. (12). □

In order to solve the maximization over ξ in Eqn. (11) effectively, we establish that the CVaR QD-learning update preserves the concavity of $\{y\ Q_{n,k}(s,a,y)\}$. We observe that this concern is unique to the CVaR-based update, and is not seen in the expectation-based QD-learning update in [29]. Our next result formalizes this insight.

Theorem 3. *Suppose $\{y\ Q_{n,k}(s,a,y)\}$ is concave in y for all agents n and for all $s \in \mathcal{S}, a \in \mathcal{A}$ and $y \in \mathcal{Y}$. Then,*

$\{y\ Q_{n+1,k}(s,a,y)\}$ *is also concave in* y *for all agents* n *and for all* $s \in \mathcal{S}, a \in \mathcal{A}$ *and* $y \in \mathcal{Y}$.

Proof. Let, $y_1, y_2 \in \mathcal{Y}$, $\lambda \in [0,1]$ and $y_\lambda = (1-\lambda)y_1 + \lambda y_2$. Then, we can write

$$
\begin{aligned}
&(1-\lambda)y_1 Q_{n,k+1}(s,a,y_1) + \lambda y_2 Q_{n,k+1}(s,a,y_2) \\
&= (1-\alpha_k - |\mathcal{N}(n)|\beta_k)\Big((1-\lambda)y_1 Q_{n,k}(s,a,y_1) \\
&+ \lambda y_2 Q_{n,k}(s,a,y_2)\Big) + \beta_k \sum_{l\in\mathcal{N}(n)} \big((1-\lambda)y_1 Q_{l,k}(s,a,y_1) \\
&+ \lambda y_2 Q_{l,k}(s,a,y_2)\big) + \alpha_k\Big(((1-\lambda)y_1 + \lambda y_2)c_n(s,a) + \\
&\gamma\Big(\min_{a_1'\in\mathcal{A}}\big[\max_{\xi_1\in\mathcal{U}_{CVaR}(y,P(\cdot|s,a))} \xi_1(s')(1-\lambda)y_1 \\
&Q_{n,k}(s',a',y_1\xi_2(s))\big] + \min_{a_2'\in\mathcal{A}}\big[\max_{\xi_2\in\mathcal{U}_{CVaR}(y,P(\cdot|s,a))} \xi_2(s')y_2 \\
&Q_{n,k}(s',a_2',y_2\xi_2(s))\big]\Big) \\
&\le (1-\alpha_k - |\mathcal{N}(n)|\beta_k)y_\lambda Q_{n,k}(s,a,y_\lambda) + \beta_k \sum_{l\in\mathcal{N}(n)} (y_\lambda \\
&Q_{l,k}(s,a,y_\lambda)) + \alpha_k\big(y_\lambda c_n(s,a) + \gamma\big(\min_{a'\in\mathcal{A}}\big[\\
&\max_{\substack{\xi_1\in\mathcal{U}_{CVaR}(y_1,P(\cdot|s,a))\\ \xi_2\in\mathcal{U}_{CVaR}(y_2,P(\cdot|s,a))}} \xi_1(s')(1-\lambda)y_1 Q_{n,k}(s',a',y_1\xi_2(s)) \\
&+ \xi_2(s')y_2 Q_{n,k}(s',a_2',y_2\xi_2(s))\big]\big) \\
&\le (1-\alpha_k - |\mathcal{N}(n)|\beta_k)y_\lambda Q_{n,k}(s,a,y_\lambda) + \beta_k \sum_{l\in\mathcal{N}(n)} (y_\lambda \\
&Q_{l,k}(s,a,y_\lambda)) + \alpha_k\big(y_\lambda c_n(s,a) + \gamma\big(\min_{a'\in\mathcal{A}} \\
&\big[\max_{\substack{\xi_1\in\mathcal{U}_{CVaR}(y_1,P(\cdot|s,a))\\ \xi_2\in\mathcal{U}_{CVaR}(y_2,P(\cdot|s,a))}} ((1-\lambda)y_1\xi_1(s') + \lambda y_2\xi_2(s')) \\
&Q_{n,k}(s',a',(1-\lambda)y_1\xi_1(s') + \lambda y_2\xi_2(s'))\big]\big)
\end{aligned}
$$

Note that $(1-\alpha_k - |\mathcal{N}(n)|\beta_k) \ge 0$ and $\beta_k \ge 0$. In the above, we have used the concavity of min operation and the concavity of linear combination of concave functions with positive coefficients and our assumption that $\{y\ Q_{n,k}(s,a,y)\}$ is concave in y for all agent n and for all $s \in \mathcal{S}, a \in \mathcal{A}$ and $y \in \mathcal{Y}$.

We define $\xi := \frac{(1-\lambda)y_1\xi_1(s') + \lambda y_2\xi_2(s')}{y_\lambda}$. Note that when $\xi_1 \in \mathcal{U}_{CVaR}(y_1, P(\cdot|s,a))$ and $\xi_2 \in \mathcal{U}_{CVaR}(y_2, P(\cdot|s,a))$ we can write $\xi \in [0, \frac{1}{y_\lambda}]$, where $\sum_{s'\in\mathcal{S}} \xi(s')P(s'|s,a) = 1$. Thus, we can write

$$
\begin{aligned}
&(1-\lambda)y_1 Q_{n,k+1}(s,a,y_1) + \lambda y_2 Q_{n,k+1}(s,a,y_2) \\
&\le (1-\alpha_k - |\mathcal{N}(n)|\beta_k)y_\lambda Q_{n,k}(s,a,y_\lambda) + \beta_k \sum_{l\in\mathcal{N}(n)} (y_\lambda \\
&Q_{l,k}(s,a,y_\lambda)) + \alpha_k\big(y_\lambda c_n(s,a) + \gamma\big(\min_{a'\in\mathcal{A}} \\
&\big[\max_{\xi\in\mathcal{U}_{CVaR}(y,P(\cdot|s,a))} y_\lambda\xi(s')Q_{n,k}(s',a',y_\lambda\xi(s'))\big]\big) \\
&= y_\lambda Q_{n,k+1}(s,a,y_\lambda),
\end{aligned}
$$

which completes the proof. □

V. The CVaR QD-Learning Algorithm

Theorems 2 and 3 are the critical components in the design of our *CVaR QD-Learning* algorithm. This section presents our algorithm, and describes how some challenges in the implementation of the algorithm are overcome.

First, the parameter y in the Q-value $Q_{n,k}(s,a,y)$ takes values in the contiguous interval $\mathcal{Y} = (0,1]$. We overcome this challenge in the manner proposed in [19] by discretizing $\mathcal{Y}$ into sufficiently large number of intervals m, and considering only the extremities of each interval. Then, we can rewrite $\mathcal{Y} = [y_1, \cdots, y_m]$, where $0 < y_1 < \cdots < y_m \le 1$.

The second challenge arises from the maximization of $\xi(s_{k+1})Q_{n,k}(s_{k+1}, a', y_k\xi(s_{k+1}))$ over ξ in the update in Eqn. (11). Our algorithm overcomes this challenge by solving a modified maximization problem,

$$\frac{1}{y_k}\max_{\xi(s_{k+1})}\big[(y_k\xi(s_{k+1}))Q_{n,k}(s_{k+1},a',y_k\xi(s_{k+1}))\big].$$

The concavity property proved in Theorem 3 then allows us to conclude that any local maximum points of $\big[(y_k\xi(s_{k+1}))Q_{n,k}(s_{k+1},a',y_k\xi(s_{k+1}))\big]$ is indeed a global maximum.

The final challenge is identifying an admissible value for $\xi(s_{k+1})$ during the maximization step since the transition probabilities $P(s_{k+1}|s_k,a_k)$ in reinforcement learning are unknown and revealed to agents only during interactions with the environment. Our implementation addresses this challenge by initially choosing $\xi(s') = 1$ for all $s' \in \mathcal{S}$. Then, at every iteration of the *CVaR QD-Learning* algorithm, we update an upper bound $\bar{\xi}(s'|s,a)$ in a manner such that $\bar{\xi}(s'|s,a)\hat{P}(s'|s,a) \le 1$, where $\hat{P}(s'|s,a)$ is an estimate of $P(s'|s,a)$. We use a standard assumption in Q-learning that a (stochastic) base policy π_0 is chosen such that every state-action pair is visited infinitely often [3], [32] to compute an estimate $\hat{P}(s'|s,a)$ and thus obtain the bound $\bar{\xi}(s'|s,a)$. We use $\bar{\xi}(s'|s,a) = \frac{1}{\hat{P}(s'|s,a)}$ if $\hat{P}(s'|s,a) \ne 0$ (otherwise we keep the initial guess $\bar{\xi}(s'|s,a) = 1$). For the maximization over ξ, we will use any $\xi(s_{k+1}) \le \bar{\xi}(s_{k+1}|s_k,a_k)$ such that $\xi(s_{k+1})\ y_k \in \mathcal{Y}$.

The steps of our *CVaR QD-Learning* algorithm is detailed in Algorithm 1.

Algorithm 1 CVaR QD-Learning

1: **Input**: y, $\{\alpha_k\}$, $\{\beta_k\}$, γ, $\mathcal{S}$, $\mathcal{A}$, $\mathcal{Y} = [y_1, \cdots, y_m]$, N.
2: **Initialization**: $\{Q_{n,0}(s,a,y)\}$, $\bar{\xi}(s'|s,a) = 1$, $k = 0$.
3: **Output**: $\{Q_n(s,a,y)\}$.
4: **loop** for each episode
5: Initialize $s_0 \in \mathcal{S}$
6: **loop** take action a_k in s_k and observe next state s_{k+1}.
7: Update $\bar{\xi}(s_{k+1}|s_k,a_k)$.
8: Each agent n locally update $\{Q_{n,k+1}(s_k,a_k,y)\}$ according to (11) for all $y \in \mathcal{Y}$ using any $\xi(s_{k+1}) \le \bar{\xi}(s_{k+1}|s_k,a_k)$ such that $\xi(s_{k+1})\ y_k \in \mathcal{Y}$.
9: $k+1 \leftarrow k$.
10: **end loop** until s is a terminal state.
11: **end loop**

Since all agents asymptotically reach a consensus (Theorem 2), a greedy policy over any agent's evaluation of

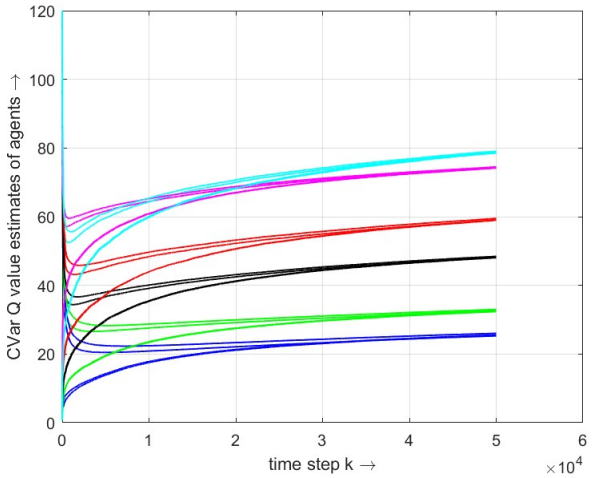

Fig. 1: This figure presents the evolution of $CVaR\ Q-$value estimates of six state-action pairs for four different non-neighbor agents with CVaR confidence level $y = 0.7$. Each state-action pair (s, a) is represented by a different color. We observe that when following Algorithm 1, $CVaR\ Q-$values of the agents reaches consensus for all (s, a) pairs.

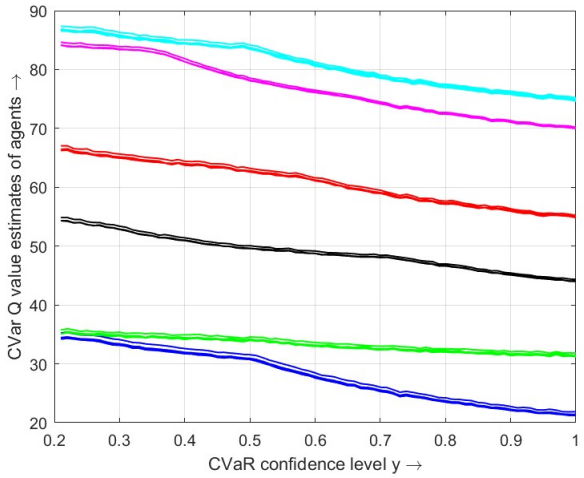

Fig. 2: This figure presents the change in $CVaR\ Q-$value estimates of six state-action pairs at time-step $t = 50000$ for four non-neighbor agents when the CVaR confidence level $y \in [0.2, 1]$. Each color denotes the $CVaR\ Q-$value for one (s, a) pair. We observe that agents' risk sensitivity varies from risk-aware ($y < 1$) to risk-neutral ($y = 1$), the $CVaR\ Q-$value decreases. Intuitively, this shows that risk-aware behaviors will result in higher ($CVaR$) $Q-$values when the objective is to minimize an accumulated cost.

the augmented action-value functions can be used to select the desired policy. Hence, the desired policy for the confidence level y of CVaR can be selected as $\mu(s, y) = lim_{k\to\infty} \operatorname{argmin}_a Q_{n,k}(s, a, y)$ for any $n = 1, \cdots, N$.

VI. Simulation Studies

In this section, we carry out simulation to evaluate the performance of the *CVaR QD-Learning* algorithm (Algorithm 1). We first describe the experiment environment, and then report our results.

A. Environment

Our simulation setup is similar to [29] which consists of a network of $N = 40$ agents. Each agent communicates with two of its nearest neighbors. Different from [29], which considered binary-valued state and action spaces, we assume that both state and action spaces have 3 possible values giving $|\mathcal{S} \times \mathcal{U}| = 9$, and 27 different transition probabilities. Eighteen of these transition probabilities are selected randomly via a uniform sampling; this fixes the remaining nine transition probabilities. For each agent n, the cost $c_n(s, a)$ is chosen from a uniform distribution that has a different mean for each state-action pair (s, a). We set the discount factor $\gamma = 0.7$, and parameters in Eqns. (2) and (3) are chosen to be $\tau_1 = 0.2$, $\tau_2 = 0.3$, $a = 0.2$ and $b = 0.1$. The interval $(0, 1]$ is discretized into 100 equally spaced intervals to quantify confidence levels associated with CVaR. Thus, we have $\mathcal{Y} = \{0.01, 0.02, \cdots, 0.99, 1\}$. We evaluate Algorithm 1 by instantiating a single trajectory $\{s_t, a_t\}$. At each step t in state s_t, actions a_t is chosen randomly via uniform sampling; the next state s_{t+1} is determined by the transition probabilities. The initial state s_0 is chosen randomly. We also set the initial estimates of $Q-$values for the agents to different values.

B. Simulation Results

We show the evolution of the $CVaR\ Q-$value estimates of six state-action pairs when following our *CVaR QD-Learning* algorithm (Algorithm 1) for four different non-neighbor agents in Fig. 1, where each state-action pair (s, a) is represented by a different color. The CVaR confidence level is set to $y = 0.7$ for all state-action pairs. We observe that agents asymptotically reach consensus on estimates of their $CVaR\ Q-$values for shown (s, a) pairs. The rate of convergence is proportional to the number of times the state-action pair is visited in the trajectory (since a larger number of samples will be available in order to update the corresponding $CVaR\ Q-$value).

Fig. 2 demonstrates the variation of $CVaR\ Q-$value estimates of six state-action pairs at time-step $t = 50000$ for different values of the confidence level $y \in [0.2, 1]$, where each state-action pair (s, a) is represented by a different color. As agents reach consensus, we observe that for the state-action pairs, the $CVaR\ Q-$value decreases with increase in y. The $CVaR\ Q-$value is lowest when $y = 1$, i.e., the situation identical to the more conventional expectation-based $Q-$value [3], [29]. Intuitively, this result indicates that as agents' risk sensitivity varies from risk-neutral ($y = 1$) to risk-aware ($y < 1$), their respective ($CVaR$) $Q-$values will be higher (since their objective is to minimize an accumulated cost). We believe that our proposed algorithm will remain effective for other simulation setups,

though the computational complexity will increase with the number of states, actions, and agents.

VII. Conclusion

In this paper, we proposed a distributed multi-agent reinforcement learning (MARL) framework for decision-making by learning risk-aware policies. We used the conditional value-at-risk (CVaR) to characterize a risk-sensitive cost function, and introduced a Bellman operator to describe a CVaR-based state-action value function. Theoretically, we proved that this operator was a contraction, and that it converged to the optimal value. We used this insight to develop a distributed MARL algorithm called the *CVaR QD-Learning* algorithm, and proved that risk-aware value functions associated to each agent reached consensus. We presented solutions to multiple challenges that arose during the implementation of the *CVaR QD-Learning* algorithm, and evaluated its performance through simulations. We also demonstrated the effect of a risk parameter on value functions of agents when they reach consensus.

One possible extension of our approach is to investigate the adversarial setting when some agents may be malicious or corrupt. Some preliminary work in this direction has been studied considering the presence of Byzantine agents, albeit while minimizing an average cost criterion [31]. Another interesting problem is to examine the continuous state-action spaces in distributed setup, where policies are parameterized by (deep) neural networks. Initial research in the design of risk-sensitive policies for MARL has focused on the centralized training regime [24]. Another possible future work direction is extensions to non-cooperative cases and scenarios where agents are heterogeneous, with distinct objectives and state-action spaces.

References

[1] C.-J. Hoel, K. Driggs-Campbell, K. Wolff, L. Laine, and M. J. Kochenderfer, "Combining planning and deep reinforcement learning in tactical decision making for autonomous driving," *IEEE transactions on intelligent vehicles*, vol. 5, no. 2, pp. 294–305, 2019.

[2] M. L. Puterman, *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[3] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[4] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Vol. 1, 4th Ed.* Athena Scientific, 2017.

[5] R. Hafner and M. Riedmiller, "Reinforcement learning in feedback control," *Machine Learning*, vol. 84, pp. 137–169, 2011.

[6] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, 2015.

[7] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, 2016.

[8] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.

[9] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions." in *Robotics: Science and Systems*, 2016.

[10] Z. Yan and Y. Xu, "Data-driven load frequency control for stochastic power systems: A deep reinforcement learning method with continuous action search," *IEEE Transactions on Power Systems*, vol. 34, no. 2, 2018.

[11] C. You, J. Lu, D. Filev, and P. Tsiotras, "Advanced planning for autonomous vehicles using reinforcement learning and deep inverse RL," *Robotics and Autonomous Systems*, vol. 114, pp. 1–18, 2019.

[12] R. A. Howard and J. E. Matheson, "Risk-sensitive Markov decision processes," *Management Science*, vol. 18, no. 7, pp. 356–369, 1972.

[13] A. Nilim and L. El Ghaoui, "Robust control of Markov decision processes with uncertain transition matrices," *Operations Research*, vol. 53, no. 5, pp. 780–798, 2005.

[14] V. S. Borkar, "Q-learning for risk-sensitive control," *Mathematics of Operations Research*, vol. 27, no. 2, pp. 294–311, 2002.

[15] R. T. Rockafellar and S. Uryasev, "Conditional value-at-risk for general loss distributions," *Journal of banking & finance*, vol. 26, no. 7, pp. 1443–1471, 2002.

[16] M. Ahmadi, U. Rosolia, M. D. Ingham, R. M. Murray, and A. D. Ames, "Constrained risk-averse Markov decision processes," in *AAAI Conference on Artificial Intelligence*, 2021.

[17] M. P. Chapman, R. Bonalli, K. M. Smith, I. Yang, M. Pavone, and C. J. Tomlin, "Risk-sensitive safety analysis using conditional value-at-risk," *IEEE Transactions on Automatic Control*, 2021.

[18] L. Lindemann, G. J. Pappas, and D. V. Dimarogonas, "Control barrier functions for nonholonomic systems under risk signal temporal logic specifications," in *IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 1422–1428.

[19] Y. Chow, A. Tamar, S. Mannor, and M. Pavone, "Risk-sensitive and robust decision-making: A CVaR optimization approach," *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[20] G. Serraino and S. Uryasev, "Conditional value-at-risk (CVaR)," *Encyclopedia of Operations Research and Management Science*, pp. 258–266, 2013.

[21] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proceedings of the International Conference on Machine Learning*, 1993, pp. 330–337.

[22] L. Matignon, G. J. Laurent, and N. Le Fort-Piat, "Independent reinforcement learners in cooperative Markov games: A survey regarding coordination problems," *The Knowledge Engineering Review*, vol. 27, no. 1, pp. 1–31, 2012.

[23] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," *Handbook of Reinforcement Learning and Control*, pp. 321–384, 2021.

[24] W. Qiu, X. Wang, R. Yu, R. Wang, X. He, B. An, S. Obraztsova, and Z. Rabinovich, "RMIX: Learning risk-sensitive policies for cooperative reinforcement learning agents," *Advances in Neural Information Processing Systems*, vol. 34, pp. 23 049–23 062, 2021.

[25] J. Zhao, M. Yang, X. Hu, W. Zhou, and H. Li, "DQMIX: A distributional perspective on multi-agent reinforcement learning," *arXiv preprint arXiv:2202.10134*, 2022.

[26] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[27] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proceedings of the AAAI conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[28] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7234–7284, 2020.

[29] S. Kar, J. M. Moura, and H. V. Poor, "QD-Learning: A collaborative distributed strategy for multi-agent reinforcement learning through consensus + innovations," *IEEE Transactions on Signal Processing*, vol. 61, no. 7, pp. 1848–1862, 2013.

[30] F. R. Chung, *Spectral graph theory*. American Mathematical Soc., 1997, vol. 92.

[31] Y. Xie, S. Mou, and S. Sundaram, "Towards resilience for multi-agent QD-learning," in *IEEE Conference on Decision and Control*. IEEE, 2021, pp. 1250–1255.

[32] C. J. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, pp. 279–292, 1992.

[33] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath, "Coherent measures of risk," *Mathematical finance*, vol. 9, no. 3, pp. 203–228, 1999.

[34] G. C. Pflug and A. Pichler, "Time-consistent decisions and temporal decomposition of coherent risk functionals," *Mathematics of Operations Research*, vol. 41, no. 2, pp. 682–699, 2016.

[35] V. S. Borkar and S. Chandak, "Prospect-theoretic Q-learning," *Systems & Control Letters*, vol. 156, p. 105009, 2021.

[36] B. Ramasubramanian, L. Niu, A. Clark, and R. Poovendran, "Reinforcement learning beyond expectation," in *IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 1528–1535.