European Workshop on Reinforcement Learning 14 (2018) October 2018, Lille, France.

Towards learning to best respond when losing control

Richard Klima RICHARD.KLIMA@LIVERPOOL.AC.UK
University of Liverpool, UK
Daan Bloembergen D.BLOEMBERGEN@CWI.NL
Centrum Wiskunde & Informatica, NL
Michael Kaisers MICHAEL.KAISERS@CWI.NL
Centrum Wiskunde & Informatica, NL
Karl Tuyls K.TUYLS@LIVERPOOL.AC.UK
University of Liverpool, UK

Abstract

Many applications require control policies that are robust against rare but significant deviations, for example caused by hardware failures or adversarial attacks. In this article we propose a reinforcement learning method that uses a prior belief over potential compromisation to learn such a robust policy without requiring observations of the event during training. Our method can be easily combined with different standard reinforcement learning algorithms such as Q-learning and Expected SARSA. Experiments in a multi-agent coordination domain show that our method improves performance of standard algorithms, both with and without communication between the agents that occasionally lose control.

Keywords: Reinforcement Learning; Multi-Agent Systems; Safety; Robust Learning

1. Motivation and related work

Our research is motivated by the need for safety in critical systems, which can be put at risk by rare but significant deviations caused by one or more system components failing or being compromised in an attack. Innovations in critical systems may introduce vulnerabilities to such attacks: for example, in smart grids communication channels are needed for distributed intelligent energy management strategies, while simultaneously forming a potential target that could compromise safety (Yan et al., 2013).

In this article we present a new approach for learning safe and robust multi-agent policies in such systems that uses a priori information about the potential nature of the attack or failure. As such, the algorithm is model free with respect to the environment, and model based with respect to internal faults of the individual agents. We build on our recent preliminary work in this direction (Klima et al., 2018) by providing improved empirical results in a more complex cooperative domain, and by investigating the effect of moving from centralised learning and execution to a decentralised multi-agent approach based on fictitious play. The proposed algorithms are safe with respect to temporary loss of control, and empirically robust with respect to estimation errors of control loss probabilities.

Our work touches upon multiple fields including robust control (Zhou and Doyle, 1998) and robust learning, security games, safe reinforcement learning and multi-agent reinforcement learning. In relation to the taxonomy of **safe reinforcement learning** of García and Fernández (2015) our method falls in between *Worst-Case Criterion under Parame-*

ter Uncertainty and *Risk-Sensitive Reinforcement Learning Based on the Weighted Sum of Return and Risk*, depending on the chosen alternate controller objectives. The domain of **security games** has expanded in recent years with many real-world applications (Pita et al., 2008; Shieh et al., 2012), where the main approach has been computing exact solutions and deriving strong theoretical guarantees, mostly using equilibria concepts such as Nash and Stackelberg equilibria (Korzhyk et al., 2011; Lou et al., 2017). This has been very important to theoretically underpin the field, however it often seems difficult to deploy exact theoretical solution methods in real world settings due to strict model assumptions or severe simplifications. To overcome some of the weaknesses of these theoretical approaches we base our approach on *reinforcement learning from interactions* with the environment, thus we do not need to know the system model. Until now there has been substantially less work done on using **reinforcement learning** for security games compared to the game-theoretic approaches, exceptions being for example Ruan et al. (2005) and Klima et al. (2016) who use reinforcement learning in the context of patrolling and illegal rhino poaching problems, respectively. Our work adopts the information asymmetry assumption often used in Stackelberg Security games (Korzhyk et al., 2011), providing the control transition model for the *leader*, and allowing leader-strategy-informed best response strategies by attackers. However, we arrive at a more general safe learning approach where the attacks might be very rare and not necessarily adversarial (e.g. technical failures). Similar to our approach is the work of Ciosek and Whiteson (2017) who use a form of importance sampling to better deal with *significant rare events*, however their work focuses on a single-agent environment and requires that the agent actually observes the rare events, while in our case the agents can reason about these events in advance. Adversarial attack models may draw on the **multi-agent reinforcement learning** algorithm Minimax-Q (Littman, 1994) for zero-sum games, which assumes minimisation over the opponent action space. However, in contrast, we define an attack to minimise over our own action space, and thus learn (but not enact) simultaneously our optimal policy and the (rare) attacks it is susceptible to.

2. General model

We use the *stochastic game* formulation defined by a tuple $(n, S, A_1 \dots A_n, R_1 \dots R_n, T)$, where n is the number of agents, S is the state space, and A_i is the action space of agent i. The joint action space is $A = A_1 \cup \ldots \cup A_n$, and a joint action is $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)$.[1] $R_i(s, a_i, \boldsymbol{a_{-i}}) \rightarrow r_i$ is the reward function of agent i for given state s and joint action $\boldsymbol{a}$, and $T(s, \boldsymbol{a}) \rightarrow s'$ is the state transition function. The n agents are assumed to have similar goals and thus cooperate with each other, with a possibility of any of them being compromised or to fail causing deviations from the optimal policy. We consider several *control domains*, defining whether the agents are in control of their actions or not. During the interaction with the system there are assumed to be transitions of such control. The model of the control transition is assumed to be (approximately) known and is defined by the control transition function $\mathcal{C}$, which can for example define if and when an agent (and which one) is compromised and thus not in control of her actions. The control function is defined in a general manner, expressing various scenarios of malfunction or of a malicious attack, as

1. We use the common shorthand $\boldsymbol{a_{-i}}$ to denote the joint action of all agents except agent i, i.e. $\boldsymbol{a_{-i}} = (a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n)$.

$\mathcal{C}(s, \boldsymbol{a}, c) \rightarrow c'$, where $c, c' \in C$ are the current and next control domain, respectively, with C being the set of possible control functions. In the proposed reinforcement learning method the value function then becomes a function of both the state and the control function as $V(s, \mathcal{C})$. We introduce our algorithm first for a basic case in which all possible control policies are conditioned on the same reward function (and thus on the same value function). This is the version used in the remainder of this paper, with a concrete example given in Section 3. Afterwards we provide a generalisation that relaxes this assumption.

Basic model We assume one of two entities having control: either the (safe-learning) agent[2] with policy π, or an external (adversarial) entity with policy μ. The control transition function $\mathcal{C}$ determines whether we move between control domains c_π and c_μ at any given moment; thus, in our example $C = \{c_\pi, c_\mu\}$. The objective of the external policy μ is defined in a general way and can for example be to minimise value (malicious attack) or to randomise (random errors). Based on a prior assumption about the nature of μ we want to learn simultaneously a model of the environment and a model of μ without necessarily observing actual attacks or failures. This means learning a safe policy π right from the start. In the basic case discussed here we define μ in terms of our own Q-value function, for example an attacker that is minimising our expected return. Thus we need to learn only one Q-value function Q^π. This is similar to the standard assumption in Stackelberg games that the attacker is able to fully observe our past actions and thus knows our Q-value function. We define the Q-value function update, based on standard Q-learning, given the control transition function $\mathcal{C}$ as

$$Q^\pi(s,a) \leftarrow Q^\pi(s,a) + \alpha \left[r(s,a) + \gamma V^{\tilde{\pi}}(s', \mathcal{C}) - Q^\pi(s,a) \right]. \tag{1}$$

Note that where we can normally write $V^\pi(s) = \sum_a \pi(s,a) Q^\pi(s,a)$, in our case we have

$$V^{\tilde{\pi}}(s, \mathcal{C}) = \sum_{c_\sigma \in C} p(c_\sigma) \sum_a \sigma(s,a) Q^\pi(s,a), \tag{2}$$

computed as a weighted sum over all possible control policies σ under $\mathcal{C}$. The composite policy $\tilde{\pi}$ represents the actual (stochastic) policy that is executed by the system, including any potential malfunctions and/or attacks. Note that we can learn Q^π without actually experiencing any attack or malfunction, based only on prior assumptions about the nature of $\mathcal{C}$.

General model Here we relax the assumption that all possible control policies σ are conditioned on the same value function Q^π. Instead, each σ can now be based on a different Q-value function Q^σ. We tackle this generalisation by learning separate Q-functions for both our target policy π and any possible not-in-control (e.g. attacker) policy, and then using a mix of all Q-functions based on the control transition model (e.g. probability of attack) to evaluate the next state. The value function of Equation 2 now becomes $V^{\tilde{\pi}}(s, \mathcal{C}) = \sum_{c_\sigma \in C} p(c_\sigma) V^\sigma(s)$, a weighted sum[3] of the different value functions learned for all possible

2. For clarity we present a single agent formulation here; an example of our model in a two-agent setting is given in Section 3.
3. In general this can be any operator, not just the *sum* (i.e. linear combination), which would allow for more complex scenarios.

policies σ. Thus, we want to simultaneously learn Q-functions for all policies σ we consider (including our own policy π). We can learn these from the same experience stream $\langle s, a, r, s' \rangle$ generated by our behaviour policy π by using *importance sampling*[4] as $Q^{\sigma}(s,a) \leftarrow Q^{\sigma}(s,a) + \alpha \frac{\sigma(a|s)}{\pi(a|s)} \left[r(s,a) + \gamma V^{\tilde{\pi}}(s', \mathcal{C}) - Q^{\sigma}(s,a) \right]$. This gives us our own estimation of the Q-functions for all policies σ.

3. Two-agents under severe rare attack

We demonstrate the effectiveness of our approach in a two-agent coordination domain. We consider two versions of the joint action learning (JAL) approach (Claus and Boutilier, 1998), differing by the level of communication between the agents: (i) full communication (FC), where the agents can communicate/observe each other's actions, rewards and policies and (ii) no communication (NC), where the agents do not communicate but can still model each other by observing past actions. We assume an intentional and intelligent attack, which minimises the possible return (minimising the Q-function) as explained above. An important aspect is the sparsity of such attacks with the aim not to be too cautious when not needing to be. The risk of attack is assumed to be known to the agents and is expressed by the probability of attack per state, which is defined by the control transition function $\mathcal{C}$. In our algorithm we use a parameter κ which expresses the risk of attack and allows the algorithm to learn an accordingly safe strategy π. Thus, we consider several control domains $c_\sigma \in C$, where $C = \{\max_{A_1} \max_{A_2} Q^{\pi}, \max_{A_1} \min_{A_2} Q^{\pi}, \min_{A_1} \max_{A_2} Q^{\pi}, \min_{A_1} \min_{A_2} Q^{\pi}\}$ with $p(c_\sigma) = \{1-\kappa, \kappa/2, \kappa/2, 0\}$, representing the situation where only one agent can be attacked at a time (with probability $\kappa/2$). We use Equation 2 and present an off-policy algorithm based on standard Q-learning (Watkins and Dayan, 1992), and an on-policy algorithm based on Expected SARSA (van Seijen et al., 2009).

Joint action learners with full communication (JAL-FC) We assume two agents with different action spaces but an identical reward function and thus a shared joint action Q-value function. Moreover, the agents can coordinate a priori on a joint action to take in the current state, and thus simply find the joint action with the maximum Q-value. We propose off-policy $\mathbf{Q(\kappa)}$, where we define the optimal joint action as $\langle a_1^\star, a_2^\star \rangle = \arg\max_{a_1 \in A_1, a_2 \in A_2} Q^{\pi}(s, \langle a_1, a_2 \rangle)$ and the value function as

$$V^{\tilde{\pi}}(s, \mathcal{C}) = (1-\kappa) Q^{\pi}(s, \langle a_1^\star, a_2^\star \rangle) + \frac{\kappa}{2} \min_{a_1 \in A_1} Q^{\pi}(s, \langle a_1, a_2^\star \rangle) + \frac{\kappa}{2} \min_{a_2 \in A_2} Q^{\pi}(s, \langle a_1^\star, a_2 \rangle).$$

Note that for $\kappa = 0$ the algorithm is equal to standard Q-learning. We also present an on-policy version called **Expected SARSA(κ)**, where the value function $V^{\tilde{\pi}}(s, \mathcal{C})$ with the attacker reacting to the current stochastic policy π is defined as

$$\begin{aligned} V^{\tilde{\pi}}(s, \mathcal{C}) = {} & (1-\kappa) \mathbb{E}_{a_1^\star, a_2^\star \sim \pi} \left[Q^{\pi}(s, \langle a_1^\star, a_2^\star \rangle) \right] \\ & + \frac{\kappa}{2} \min_{a_1 \in A_1} \mathbb{E}_{a_2^\star \sim \pi} \left[Q^{\pi}(s, \langle a_1, a_2^\star \rangle) \right] + \frac{\kappa}{2} \min_{a_2 \in A_2} \mathbb{E}_{a_1^\star \sim \pi} \left[Q^{\pi}(s, \langle a_1^\star, a_2 \rangle) \right]. \end{aligned}$$

Joint action learners with no communication (JAL-NC) Here we assume two agents without communication, each learning an individual Q-value function. Moreover,

4. Or other similar methods such as Retrace(λ) (Munos et al., 2016).

the agents can no longer coordinate on a joint action, but can only best-respond to an estimated policy. We define the algorithm from perspective of agent 1, the algorithm for agent 2 is analogous. We assume that agent 1 can observe agent 2's past moves and base her estimate of agent 2's policy $\hat{\pi}_2$ on fictitious play (FP) (Fudenberg and Levine, 1998). We choose the most frequently taken action (pure strategy) as the simplest estimate of the other agent's policy $\hat{\pi}$.[5] Since in this scenario the agents do not necessarily have the same Q-function, they cannot reason about the other agent being attacked, and thus each agent considers only an attack on herself in order to compute $V^{\tilde{\pi}}(s, \mathcal{C})$. We again present off-policy $\mathbf{Q(\boldsymbol{\kappa})}$ with the value function

$$V^{\tilde{\pi}}(s, \mathcal{C}) = (1 - \frac{\kappa}{2}) \max_{a_1 \in A_1} \mathbb{E}_{a_2^\star \sim \hat{\pi}_2} \left[Q^\pi(s, a_1, a_2^\star)\right] + \frac{\kappa}{2} \min_{a_1 \in A_1} \mathbb{E}_{a_2^\star \sim \hat{\pi}_2} \big[Q^\pi(s, \langle a_1, a_2^\star \rangle)\big].$$

The value function of on-policy **Expected SARSA($\boldsymbol{\kappa}$)** is now defined as

$$V^{\tilde{\pi}}(s, \mathcal{C}) = (1 - \frac{\kappa}{2}) \mathbb{E}_{a_1^\star \sim \pi_1, a_2^\star \sim \hat{\pi}_2} \big[Q^\pi(s, \langle a_1^\star, a_2^\star \rangle)\big] + \frac{\kappa}{2} \min_{a_1 \in A_1} \mathbb{E}_{a_2^\star \sim \hat{\pi}_2} \big[Q^\pi(s, \langle a_1, a_2^\star \rangle)\big].$$

4. Experiments

We use a variation of the cliff walking task described by Sutton and Barto (1998) for experimental evaluation of our proposed algorithms Q(κ) and Expected SARSA(κ). Our environment is a grid world with puddles which need to be avoided by the joint-learning agents. The two agents jointly control the movement of a single robot in this puddle world, with each controlling a single direction $\langle \text{up}, \text{down} \rangle$ or $\langle \text{left}, \text{right} \rangle$. Agent 1 can take the actions {*stay, move down, move up*} and agent 2 can choose {*stay, move left, move right, move right by 2*}. The joint action is the combination of the two selected actions. We assume a reward of -1 for every move and -1000 for stepping into a puddle (ending the episode and returning to the start node). The agents have to move together from the start node at the top left corner to the goal at the bottom right corner. Figure 1 shows the policy learned by our proposed algorithm Q(κ) for the scenario JAL-FC. Note how a safer path (longer, avoiding the puddles) is learned with increasing parameter κ. For $\kappa = 0$ our algorithm degenerates to Q-learning (left panel).

Performance Figure 2 shows the performance of the proposed κ-algorithms against the standard TD algorithms for an increasing probability of attack. All algorithms use a fixed exploration rate of $\epsilon = 0.1$ (in training) and are trained for 200000 episodes without attacks and evaluated over 50000 episodes with attacks. In Figure 2 we can observe the superior performance of Q(κ) over unmodified Q-learning for both the full communication (JAL-FC) and the no communication (JAL-NC) scenarios. Similarly, Expected SARSA(κ) performs at least as good as Expected SARSA, and much better than regular SARSA. The latter is very unstable in the NC case and often does not reach the goal state (see Appendix A). Therefore we also include the filtered performance of only the best runs (approximately best 25%), however this still does not get SARSA's performance close to the κ algorithms. We can see that Q(κ) learns a safer path of the same distance as Q-learning, yielding much

5. One could also assume a mixed strategy estimate proportional to the frequency of past moves, or any other policy estimator.

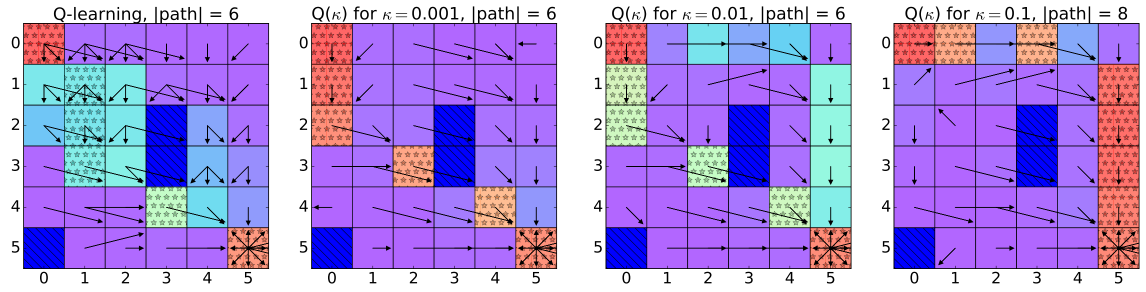
Q-learning, |path| = 6
Q(κ) for κ = 0.001, |path| = 6
Q(κ) for κ = 0.01, |path| = 6
Q(κ) for κ = 0.1, |path| = 8
0
1
2
3
4
5

Figure 1: The puddle world: Q(κ) learns safer path with increasing κ. Puddles are dark blue, the arrows show the optimal actions in each state and the heatmap shows the number of visits to each state (▬, blue is none). A learned path is depicted by stars.

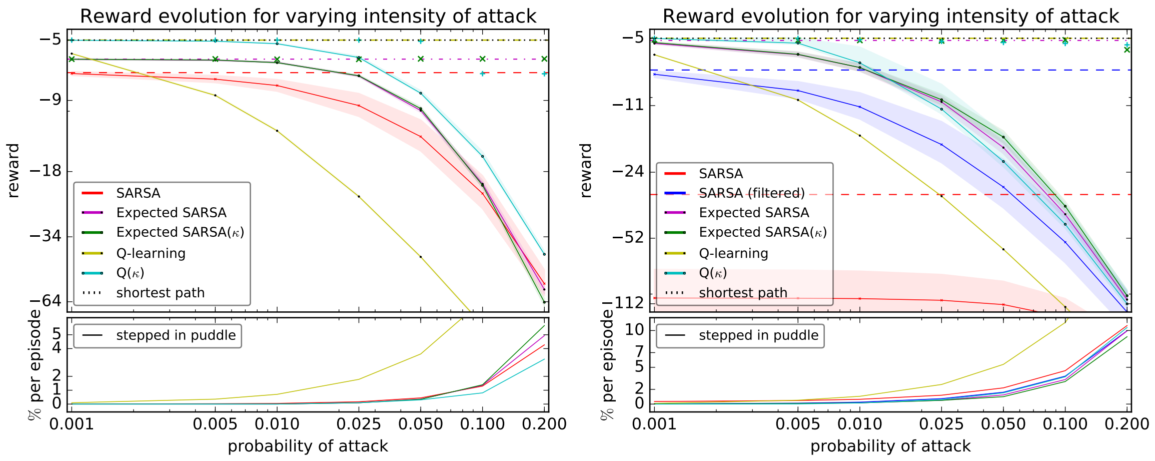

Figure 2: Total reward per episode for increasing probability of attack. JAL-FC (**left**) and JAL-NC (**right**). The dotted/dashed lines and '+' and 'x' markers in respective colours are the rewards of the learned path (without attacks). The black dotted line is the reward of the shortest path. The solid lines (with 95% confidence intervals) are the rewards obtained after training when attacks do happen.

better rewards when attacked (as also shown in Figure 1). Note that we train Q(κ) and Expected SARSA(κ) with κ equal to the true probability of attack, however both algorithms are robust against small errors in the expected probability κ as discussed in the following.

Robustness Here we test the robustness of the proposed algorithms. We no longer assume the probability of attack to be known and thus it is not possible to correctly set the κ parameter for Q(κ) and Expected SARSA(κ). Note that in our previous experiments we set the parameter κ to be equal to the actual probability of attack. In Figure 3 we show the performance of our algorithms for a range of actual attack probabilities (y-axis) while learning using a fixed parameter $\kappa = 0.01$ (left) and fixed $\kappa = 0.1$ (right), for the JAC-FC scenario; Figure 4 shows the JAL-NC case. One can see that even for the cases where κ is not equal to the actual probability of attack the proposed κ-algorithms still outperform the baselines in most cases. This robustness evaluation confirms strong properties of our proposed algorithms. We conclude that the proposed algorithms are robust to inaccurate estimates of the probability of attack (within some bounds). In Figure 3 we can see how Q(κ) for fixed $\kappa = 0.01$ outperforms the baselines up to a probability of attack of 0.05, and for fixed $\kappa = 0.1$ it outperforms the baselines from a probability of attack of 0.1 onward. In Figure 4 for $\kappa = 0.01$ we can see that Q(κ) and Expected SARSA(κ) have very similar performance to Expected SARSA while still clearly beating Q-learning and SARSA. For $\kappa = 0.1$ we can see that Expected SARSA(κ) outperforms all the baselines.

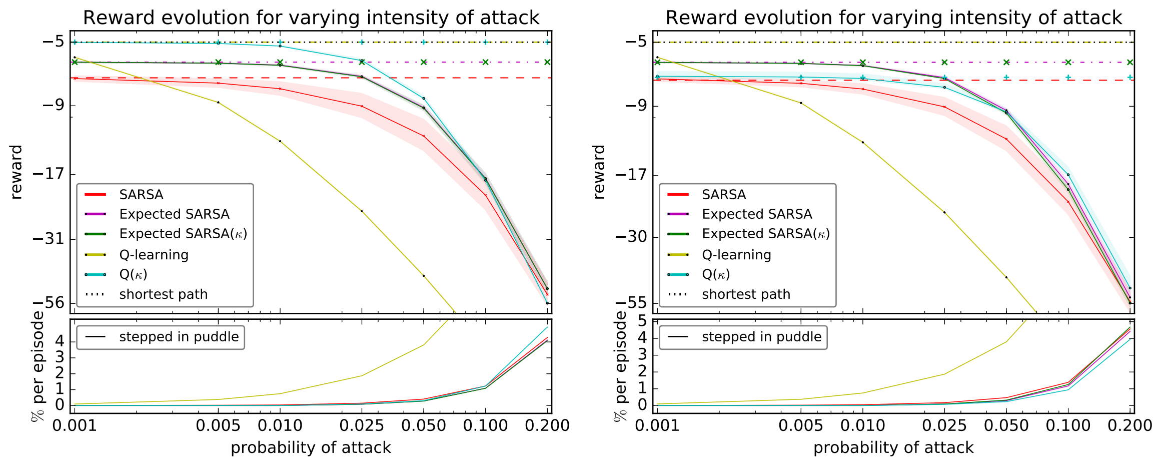

Figure 3: Robustness: Fixed $\kappa = 0.01$ (left) and $\kappa = 0.1$ (right) for Q(κ) and Expected SARSA(κ). Joint learners with full communication (JAL-FC).

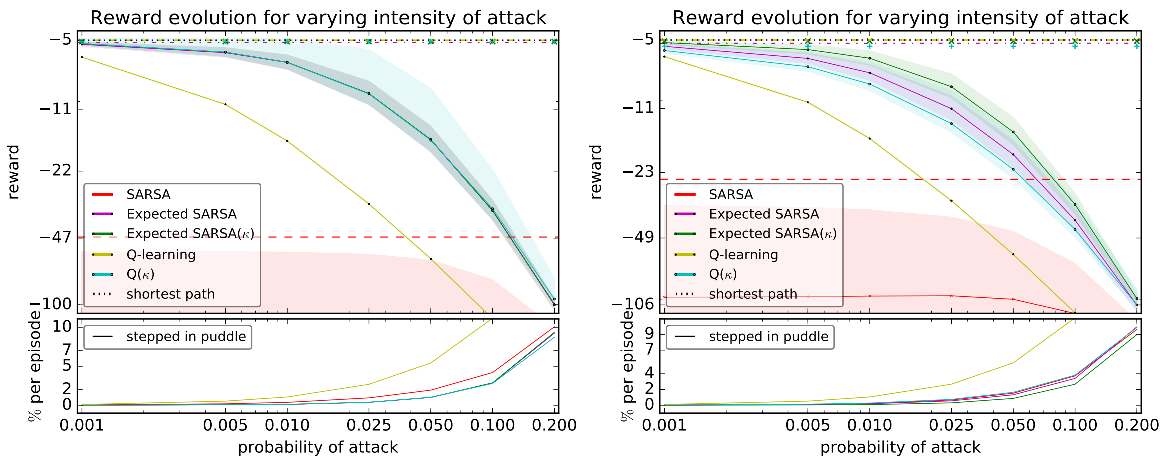

Figure 4: Robustness: Fixed $\kappa = 0.01$ (left) and $\kappa = 0.1$ (right) for Q(κ) and Expected SARSA(κ). Joint learners with no communication (JAL-NC).

5. Discussion and conclusion

We presented a new method based on standard reinforcement learning algorithms which can learn robust policies in the presence of rare malfunctions or attacks in cooperative multi-agent systems. While the extended setting of losing control could be seen just as a new Markov game, we argue that a) this class of games is relevant and useful in applications, and b) our formulation based on a control loss model provides a solution that exploits the identical action space of the attacker, which model free approaches applied to the extended game would not. The proposed framework provides robustness against a chosen stochastic control transition model, which describes the probability of attack or malfunction of every part of the studied system. This control transition function is assumed to be a chosen (and thus known) robustness target set by the system designer, which provides a useful degree of freedom for many real world domains where expertise is available on the nature of expected attacks or potential malfunctions that may arise. The preliminary experimental evaluation in a two-agent grid world domain shows promising results, outperforming standard Q-learning, SARSA and Expected SARSA algorithms in both full-communication and more daring no-communication scenarios.

There are several interesting directions for future work. We could assume several agents being attacked or malfunctioning with different intensity. This would be expressed by a more complex control transition function with potentially more parameters depending for example on state or time. A state-dependent control transition function is motivated by the fact that in practice some states might be critical and more prone to malicious attacks or malfunction. Such extensions would narrow the reality gap and would allow for learning more complex policies, where we believe our approach could prove even more competitive. Our proposed method can be closely linked or even combined with some recent state-of-the-art reinforcement learning methods. For example, our method could be combined with Retrace(λ) (Munos et al., 2016) in a multi-step Q-update, which would potentially speed up convergence. Another promising extension of our model would be to combine it with Q(σ) (Asis et al., 2018) to allow for mixed multi-step updates. Note that the parameter σ in this algorithm can also be time- or state-dependent similarly to the potential extensions of the control transition function in our model. This would allow to learn robust policies against more complex control transition functions such as multi-step attacks. Another interesting extension along this line would be to model the control transition function similar to the options framework (Sutton et al., 1999; Bacon et al., 2017), in which case the alternate control policies could be seen as "malicious" options over which the agent has no control, with potentially complex initiation sets and termination conditions. Such extensions would increase the flexibility of our proposed method, making it applicable to a wide range of real-world scenarios.

Acknowledgments

This project has received funding in the framework of the joint programming initiative ERA-Net Smart Energy Systems' focus initiative Smart Grids Plus, with support from the European Union's Horizon 2020 research and innovation programme under grant agreement No 646039.

References

Kristopher De Asis, J. Hernandez-Garcia, G. Holland, and Richard Sutton. Multi-step reinforcement learning: A unifying algorithm. In *AAAI Conference on Artificial Intelligence*, 2018.

Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *Proceedings of Association for the Advancement of Artificial Intelligence Conference (AAAI)*, pages 1726–1734, 2017.

Kamil Andrzej Ciosek and Shimon Whiteson. OFFER: Off-environment reinforcement learning. In *Proceedings of Association for the Advancement of Artificial Intelligence Conference (AAAI)*, 2017.

Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. *AAAI/IAAI*, 1998:746–752, 1998.

Drew Fudenberg and David K Levine. *The theory of learning in games*, volume 2. MIT press, 1998.

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

Richard Klima, Karl Tuyls, and Frans Oliehoek. Markov Security Games: Learning in Spatial Security Problems. *NIPS Workshop on Learning, Inference and Control of Multi-Agent Systems*, pages 1–8, 2016.

Richard Klima, Daan Bloembergen, Michael Kaisers, and Karl Tuyls. Learning robust policies when losing control. *Adaptive and Learning Agents workshop at AAMAS*, 2018.

Dmytro Korzhyk, Zhengyu Yin, Christopher Kiekintveld, Vincent Conitzer, and Milind Tambe. Stackelberg vs. Nash in Security Games: An Extended Investigation of Interchangeability, Equivalence, and Uniqueness. *Journal of Artificial Intelligence Research*, 41:297–327, 2011.

Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings 1994*, pages 157–163. Elsevier, 1994.

Jian Lou, Andrew M Smith, and Yevgeniy Vorobeychik. Multidefender security games. *IEEE Intelligent Systems*, 32(1):50–60, 2017.

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1054–1062, 2016.

James Pita, Manish Jain, Janusz Marecki, Fernando Ordonez, Christopher Portway, Milind Tambe, Craig Western, Praveen Paruchuri, and Sarit Kraus. Deployed ARMOR Protection: The Application of a Game Theoretic Model for Security at the Los Angeles International Airport. In *International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, volume 3, pages 1805–1812, 2008.

Sui Ruan, Candra Meirina, Feili Yu, Krishna R Pattipati, and Robert L Popp. Patrolling in a stochastic environment. Technical report, Electrical and Computer Engineering Department, University of Connecticut, Storrs, 2005.

Eric Shieh, Bo An, Rong Yang, Milind Tambe, Craig Baldwin, Joseph DiRenzo, Ben Maule, and Garrett Meyer. PROTECT: A Deployed Game Theoretic System to Protect the Ports of the United States. *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 1:13–20, 2012.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

Richard S Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112 (1-2):181–211, 1999.

Harm van Seijen, Hado van Hasselt, Shimon Whiteson, and Marco Wiering. A theoretical and empirical analysis of Expected Sarsa. In *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, ADPRL 2009*, pages 177–184, 2009.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Ye Yan, Yi Qian, Hamid Sharif, and David Tipper. A survey on smart grid communication infrastructures: Motivations, requirements and challenges. *IEEE communications surveys & tutorials*, 15(1):5–20, 2013.

Kemin Zhou and John Comstock Doyle. *Essentials of robust control*, volume 104. Prentice hall, Upper Saddle River, NJ, 1998.

Appendix A. Instability of SARSA

In the experiments one can notice the unstable performance of SARSA, demonstrated by very wide confidence intervals (see Figure 2). Therefore, we analyse the convergence properties of the algorithms (baselines and the proposed κ-versions) by showing the temporal difference (TD) error over the number of episodes in Figure 5. The figure shows the sum of the absolute values of the two-agents' TD errors for both cases, JAL-FC and JAL-NC. Notice the different y-axis ranges in Figure 5 for JAL-FC and JAL-NC, meaning that SARSA is much less stable in JAL-NC than in JAL-FC; however, importantly, it fails to converge in either case. The reason for SARSA being very unstable is mainly the fixed exploration rate $\epsilon = 0.1$ and the fact it is an on-policy learning algorithm, which is known to have high variance. In our model the latter has an even more profound effect due to the extreme difference in possible rewards caused by the rare but significant events. Furthermore, in our case additional stochasticity is brought into the model by the occasionally loss of control.

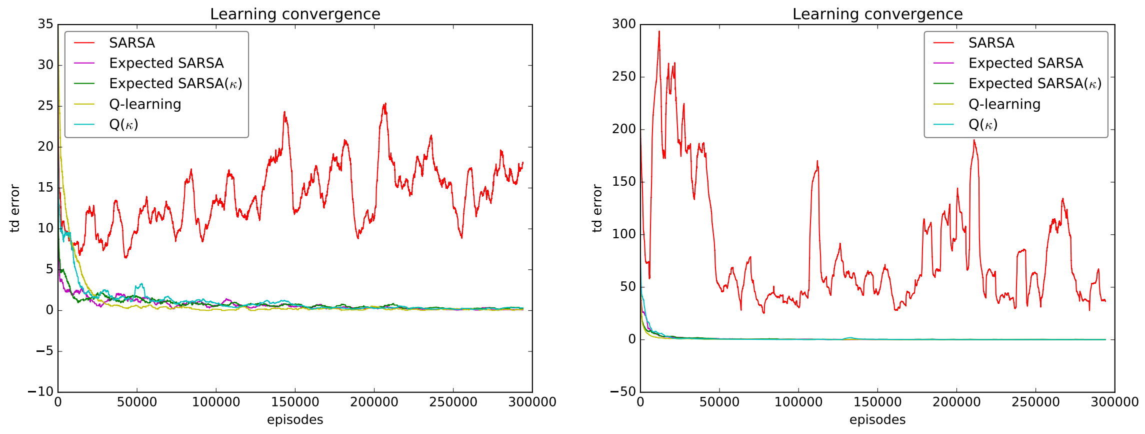

Figure 5: Temporal difference (TD) error evolution. JAL-FC (**left**) and JAL-NC (**right**).