Shield Decentralization for Safe Multi-Agent Reinforcement Learning

Daniel Melcer, **Christopher Amato**, **Stavros Tripakis**
Northeastern University, Boston, MA, USA
{melcer.d, c.amato, stavros}@northeastern.edu

1 Introduction

It is often difficult to verify the properties of a reinforcement learning system; this limits the application of reinforcement learning in safety-critical systems.

One recent approach to this problem is the idea of a shield—a reactive system which monitors the agents and environment of a reinforcement learning system [1]. As the agent proposes each action, the shield evaluates whether the action is safe, and replaces unsafe proposed actions with a safe action. This method has been extended to multi-agent environments, but with communication assumptions [3].

Our main contribution in this work is an algorithm which, given a multi-agent shield, produces a *decentralized* shield requiring no communication between the agents. We find that performance in various grid-based reinforcement learning tasks is similar both in agents which use a communication-based centralized shield, and agents equipped with a decentralized shield. Analysis of the shield structure itself reveals that the algorithm preserves the vast majority of behaviors allowed by communication-based shields for these tasks.

2 Preliminaries

Our characterization of a centralized shield is based off of the definitions of a centralized shield in [1; 3].

Briefly, agents operate in an environment such as a Markov Decision Process (MDP). We denote the state space of the environment by $\mathbb{S}$ and the agent's action space by $\mathbb{A}$. It is often useful to define an *abstraction function* $f : \mathbb{S} \to L$, such that the *label space* L captures a subset of environment information sufficient to ensure safety.

We denote by Σ the transition alphabet of a Deterministic Finite Automaton (DFA), by δ its transition function, by s_0 its initial state, by Q its state space, and by F its set of accepting states. A *safety specification* for some environment is a DFA where $\Sigma = L \times \mathbb{A}$, $s_0 \in F$ and there are no transitions from $Q \setminus F$ to F. A safety specification can *enforce* another specification if accepts a subset of traces.

A *centralized shield* is a safety specification in which there is always some *safe action* available, when the shield is composed with the environment as follows: At runtime, a centralized shield maintains the current shield state g, initialized to s_0. At every time step, it observes the state s from the environment, and calculates $l = f(s)$. The shield receives a proposed action a from the agent. If $\delta(g, (l, a)) \in F$, the shield sends this action to the environment. Otherwise, the shield chooses some safe action a' such that $\delta(g, (l, a')) \in F$. In this case, the agent observes, and is trained on, the environment transition (s, a', r, s'). The agent is also trained on a synthetic transition $(s, a, r + r_p, s')$ where r_p is some negative reward modifier; while not strictly necessary for safety, this additional transition improves performance.

Prior multi-agent shielding methods [3] treat all agents as one collective agent. For a n-agent environment, this agent's action space $\mathbb{A} = \mathbb{A}_1 \times \ldots \times \mathbb{A}_n$ represents the Cartesian product of each individual agents' action spaces.

3 Decentralized Shielding

The use of a centralized shield in a multi-agent environment requires communication. First, the set of safe individual actions for a given agent may depend on the actions which other agents choose. In contrast, a decentralized shield must allow each agent to independently choose its own action.

Furthermore, a centralized shield may transition to different states in response to different joint actions. Without direct knowledge of the global joint action, a decentralized shield must guarantee that the next shield state is unambiguous.

We structure a *decentralized shield* as a collection of *individual shields*. An individual shield for agent i is a DFA whose transition alphabet is $L \times \mathbb{A}_i$.

Centralized shield synthesis is strictly an easier problem than decentralized shielding; therefore, given a safety specification ϕ^s, we first attempt to use existing methods to synthesize a centralized shield ϕ^g which enforces ϕ^s. If it fails, we do not attempt decentralized shield synthesis. Otherwise, we instead focus our efforts on decomposing the centralized shield to obtain a decentralized shield which enforces ϕ^g. By transitivity, the decentralized shield will also enforce ϕ^s.

There exists a trivial method to decompose shields: for every shield state and label, prescribe a single safe joint action in advance. While this technically satisfies the requirements, it doesn't allow the agents any freedom to explore actions and optimize for expected future rewards.

Instead, we would like to synthesize a *maximally permissive* shield; i.e. a shield for which there does not exist an alternate shield that admits a strict superset of label traces. This is a fundamentally difficult problem, and we therefore

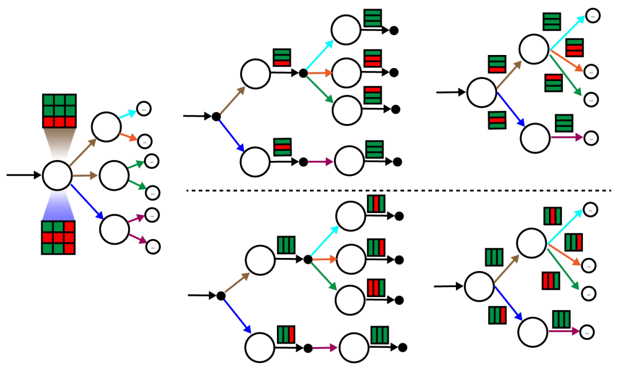

Figure 1: An overview of our algorithm. Colored arrows represent labels, green squares represent allowed actions. Step 1 finds a set of safe individual actions for each state and label. This is shown for the leftmost state: upon observing brown, agent 1 may only choose actions 1 or 2, while agent 2 may choose any action. Step 2 projects these individual actions onto transient-state individual shields. For example, after observing the label brown from the initial state, agent 1 transitions to a state which allows actions 1 or 2. Step 3 combines label and action transitions to conform to the shield interface.

present a "best-effort" procedure to produce a decentralized shield which is still safe, but significantly more permissive than the trivial solution. Our sequence of algorithms is illustrated in Figure 1, and described as follows.

First, for every centralized shield state, and for every label which can be observed from that state, find sets of individual actions per agent such that the Cartesian product is safe and unambiguous (Figure 2). Second, project these joint action sets to *transient-state individual shields*; this intermediate structure treats the agent and environment as players in a two-player game automaton. Lastly, eliminate transient states to obtain individual shields. The entire sequence takes linear time with respect to the size of the input shield.

Given an input shield which is already unambiguous and allows independent action selection, the output decentralized shield allows the same behaviors. At worst, the decentralized shield is equivalent to one generated by the trivial solution; in practice (c.f. Section 4), it is closer to the former case.

4 Experiments

We aim to show with our experiments that decentralized shields correctly prevent the agents from taking unsafe actions, while an equivalent unshielded agent would take unsafe actions in the same situation. Furthermore, we hypothesize that agents which use our decentralized shielding method perform comparably to agents which use a centralized shield.

4.1 The Gridworld Collision Domain

We adapt the 2-agent gridworld maps from Melo and Veloso [4] to test our method, ranging in size from 8x10 to 10x23. Each can move in the four cardinal directions, or stay in place. Agents receive a -10 reward when hitting a wall, a +100 reward if all agents reach the goal, or -1 otherwise. The environment gives a -30 reward for a collision or crossing.

We used a reactive synthesis tool [2] to create a centralized shield which avoids collisions in any gridworld environment.

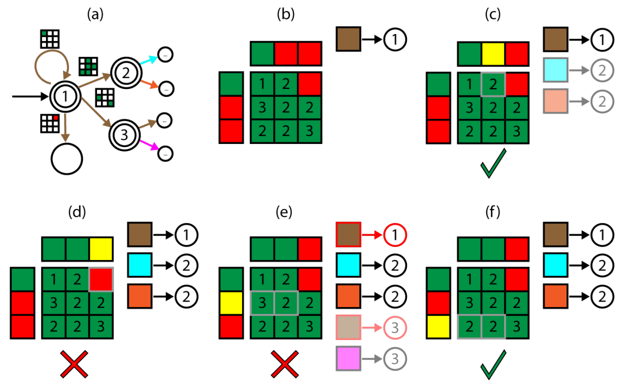

Figure 2: An illustration of step 1. (a) An example input centralized shield. This figure focuses on the process for the state-label pair of (1, brown). (b) The grid represents the next state which will be seen after taking each joint action. A single safe joint action—both agents choosing action 1—is chosen arbitrarily. The algorithm notes that if the next label is brown, the next state must be state 1. (c) The algorithm considers adding agent 1, action 2. This is safe and unambiguous, so the action is added. (d) The algorithm considers agent 1, action 3. This would add an unsafe joint action and is rejected. (e) The algorithm considers agent 2, action 2 (implying two different joint actions). Adding state 3 as a possible next state would cause ambiguity, so the action is rejected. (f) The algorithm considers agent 2, action 3. Neither joint action causes any issues, so this individual action is added. The step has completed: Agent 1 may take actions 1 or 2, and agent 2 may take actions 1 or 3.

We then combined duplicate DFA states and utilized our algorithm to obtain a decentralized shield. Analysis of the shield's structure reveals that the decentralized shield allows an average of 22.07 joint actions for a given shield state, compared to the centralized shield's 23.30 and trivial shield's 1.00.

We trained independent tabular Q-learning agents, with a linear ϵ annealing schedule from 1 to 0.05. We set $\gamma = 0.9$ and $r_p = -10$. Agents were trained for 2.5 million steps each, using 10 different random seeds for each configuration.

Our results (Table 1) show that both centralized and decentralized shielding methods effectively prohibit unsafe actions, while unshielded agents occasionally take such actions. Additionally, no shielding method consistently obtains the best results; the agents tend to be competitive with each other. Therefore, decentralized shielding is a reasonable choice for enforcing safety in no-communication environments.

4.2 The Relative Particle + Momentum Domain

We also evaluate our shield decomposition method in an environment where the shield must take action in advance to prevent an unsafe situation several steps later. In the relative particle-momentum domain, agents observe only their relative positions (capped at 10 units in each direction) and velocities (up to 2 units in each direction). An agent can move in any cardinal direction or do nothing; in either case, its movement is added to its velocity from the previous time step. The agents obtain a reward of 100 when they reach a specific arrangement relative to each other, and a -1 reward otherwise. Our safety specification is that agents can never collide, or stray more than 10 units away from each other. If an agent

Start Type	Map Name	Centralized	Decentralized	No Shield
Fixed	ISR	90.4 ± 1.8 (0)	89.9 ± 1.6 (0)	90.0 ± 2.0 (12.3)
	MIT	72.0 ± 2.6 (0)	71.6 ± 1.0 (0)	72.8 ± 1.6 (0)
	Pentagon	89.0 ± 1.2 (0)	82.0 ± 18.3 (0)	89.7 ± 1.0 (4.8)
	SUNY	83.3 ± 4.4 (0)	86.6 ± 1.8 (0)	86.2 ± 1.3 (0)
Random	ISR	78.6 ± 7.0 (0)	76.1 ± 6.2 (0)	83.6 ± 4.5 (2.0)
	MIT	82.6 ± 0.7 (0)	83.2 ± 0.6 (0)	81.4 ± 2.8 (0.4)
	Pentagon	88.4 ± 1.5 (0)	80.3 ± 11.1 (0)	89.1 ± 1.0 (0.9)
	SUNY	78.1 ± 1.2 (0)	73.1 ± 5.2 (0)	77.3 ± 1.3 (0.4)

Table 1: Agent performance in the gridworld collision domain. Results show average reward and total safety violations over 100 testing episodes, averaged over 10 random seeds. Note that in a no-communication environment, centralized shielding would not be feasible.

Observability	Start Type	Centralized	Decentralized	No Shield
Partial	Fixed	68.6 ± 40.4 (0)	85.5 ± 5.1 (0)	62.8 ± 26.9 (35.3)
	Random	69.1 ± 16.0 (0)	79.9 ± 8.0 (0)	33.0 ± 18.4 (123.9)
Full	Fixed	91.6 ± 0.1 (0)	91.6 ± 0.1 (0)	90.7 ± 0.6 (3.0)
	Random	94.6 ± 0.4 (0)	94.6 ± 0.4 (0)	93.7 ± 0.6 (2.8)

Table 2: DQN Agent Performance in the particle momentum domain. Partial observability uses 50 random seeds, full observability uses 10.

does either of these, it also receives a -30 penalty from the environment. Because of the momentum, it is possible for both agents to take the no-op action and still violate the specification. We also created a partially observable variation; these agents do not observe the relative velocities.

We synthesized shields which use only the relative positions of the agents as the label set. The synthesized shields are automatically able to track the relative velocities of the agents by observing the changes in relative positions, this behavior arises without needing to explicitly describe the calculation. Using the same structure analysis as earlier, the centralized shield allows an average of 21.67 actions, while the decentralized shield allows an average of 20.81 actions.

To show that decentralized shielding is independent of the agent itself, we trained DQN agents to solve this task [5]. The resulting policies were evaluated, as shown in Table 2. We reach similar conclusions as in the gridworld-collision domain: the agents with centralized and decentralized shielding achieve comparable performance, while neither takes any unsafe actions as the unshielded agents do. This is magnified in the partially observable case, where reactive agents do not have sufficient information to learn a policy by themselves. Because shielding is agnostic to the choice of the underlying agent, and these shields only require position information, the shields are still able to enforce the safety specification.

5 Conclusion

Our experiments show that the shield decomposition method described in this paper results in shields that are safe, do not require communication, and allow agents to learn a solution to the reinforcement learning task. We are currently working to extend our results in several ways, by implementing additional agents and training schemes, improving our shield decomposition algorithm to produce more permissive shields, and exploring direct synthesis of decentralized shields.

Acknowledgements

This material is based on work supported by NSF SaTC Award CNS-1801546, NSF CAREER Award IIS-2044993, and Army Research Office Award W911NF2010265. The experiments were run using the Discovery cluster, supported by Northeastern University's Research Computing team.

References

[1] ALSHIEKH, M., BLOEM, R., EHLERS, R., KÖNIGHOFER, B., NIEKUM, S., AND TOPCU, U. Safe reinforcement learning via shielding, 2017.

[2] EHLERS, R., AND RAMAN, V. Slugs: Extensible gr(1) synthesis. In *Computer Aided Verification* (2016), S. Chaudhuri and A. Farzan, Eds., Lecture Notes in Computer Science, Springer International Publishing, p. 333–339.

[3] ELSAYED-ALY, I., BHARADWAJ, S., AMATO, C., EHLERS, R., TOPCU, U., AND FENG, L. Safe multi-agent reinforcement learning via shielding. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems* (Richland, SC, 2021), AAMAS '21, International Foundation for Autonomous Agents and Multiagent Systems, p. 483–491.

[4] MELO, F. S., AND VELOSO, M. Learning of coordination: Exploiting sparse interactions in multiagent systems. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 2* (Richland, SC, 2009), AAMAS '09, International Foundation for Autonomous Agents and Multiagent Systems, p. 773–780.

[5] MNIH, V., KAVUKCUOGLU, K., SILVER, D., GRAVES, A., ANTONOGLOU, I., WIERSTRA, D., AND RIEDMILLER, M. Playing atari with deep reinforcement learning, 2013.