Safety Limiter Architectures and Frameworks for RL-based Functions

Arvind Somasundaram, Gabrielle W. Poerwawinata
Supervisor: Sina Shafaei

27.07.2018, SS2018
Contents

1 Introduction to Safety Limiters
2 Challenges
3 Safety Monitoring Framework (SMOF)
4 Safe Reinforcement Learning via Shielding
5 Leave No Trace
6 Proposal and Example
7 Conclusion
Introduction to Safety Limiters

What is safety in Reinforcement Learning?

- Exploration process is called safe if no undesirable states are ever visited
- One way to do it is to create limiter/teacher of action for the agent to prevent catastrophic situations
Introduction to Safety Limiters

What is safety monitor?

- The monitor is the ultimate protection against interaction faults or arbitrary behavior of the control channel that adversely affect safety
- It is equipped for observation (i.e., sensors) and able to trigger safety interventions.
A Good Analogy

https://www.youtube.com/watch?v=_b86NK39POQ
## Contents

<table>
<thead>
<tr>
<th></th>
<th>Introduction to Safety Limiters</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td><strong>Challenges</strong></td>
</tr>
<tr>
<td>3</td>
<td>Safety Monitoring Framework (SMOF)</td>
</tr>
<tr>
<td>4</td>
<td>Safe Reinforcement Learning via Shielding</td>
</tr>
<tr>
<td>5</td>
<td>Leave No Trace</td>
</tr>
<tr>
<td>6</td>
<td>Proposal and Example</td>
</tr>
<tr>
<td>7</td>
<td>Conclusion</td>
</tr>
</tbody>
</table>
Challenges
Challenges
## Contents

| 1   | Introduction to Safety Limiters         |
| 2   | Challenges                              |
| 3   | Safety Monitoring Framework (SMOF)       |
| 4   | Safe Reinforcement Learning via Shielding |
| 5   | Leave No Trace                          |
| 6   | Proposal and Example                    |
| 7   | Conclusion                              |
Safety Monitoring Framework (SMOF)

- SMOF is a systematic and formal approach for expressing safety rules.
- It’s a complete framework, starting from hazard analysis and ending in safety rule synthesis.
- The authors make use of a formal language (CTL) NuSMV.
Safety Monitoring Framework (SMOF)

Partition of system states into catastrophic, warning and safe states

Path aborted by action

Path aborted by inhibition

margin

SI

Safe states

Warning states

Catastrophic states

Partition of system states into catastrophic, warning and safe states
Safety Monitoring Framework (SMOF)

Safety Invariant (SI)

- It is a *sufficient condition* to avoid a hazardous situation.
- If a safety invariant is violated, there is no possible recovery action and any state violating the safety invariant is a catastrophic state.

Example: “in crowded street car velocity should be below 80km/h”
Safety Monitoring Framework (SMOF)

Safety Intervention

*Ability of the monitor to constrain* the system behavior in order to prevent the system from violating a safety invariant.

- **Safety Inhibition**: Prevents a change in system state.
- **Safety Action**: Triggers a change in system state.
Safety Monitoring Framework (SMOF)

Safety Rule

• It defines *a way of behaving in some warning states.*
• The condition identifies a subset of warning states. The intervention is intended to abort catastrophic paths via these warning states.

  **Example:** “if the distance of car with front object less than 10 m then apply brake”
Safety Monitoring Framework (SMOF)

Safety Strategy

Collection of safety rules intended to ensure a safety invariant.
Safety Monitoring Framework (SMOF)

Permissiveness

• The strategy is said *permissive* if the monitored system is able to freely move inside a specified state space.

• A maximum permissiveness would authorize all possible states including catastrophic ones.
Safety Monitoring Framework (SMOF)

Safety Monitoring Framework (SMOF)

Hazard Analysis

• The process starts with a HAZOP-UML hazard analysis.
• In this method, the use of the system is modeled with UML use case and sequence diagrams
  • This analysis outputs safety invariants expressed in natural language.
Safety Monitoring Framework (SMOF)

Hazop UML

Example: A mobile robot with a manipulator arm and the safety invariant
Safety Monitoring Framework (SMOF)

Example of Hazard Analysis

| SI1  | The velocity of robot arm must not be greater than $V_0$. |
| SI2  | The velocity of robot platform must not be greater than $V_1$. |
| SI3  | The robot must not enter the restricted area. |
| SI4  | The robot platform must not collide with a human. |
| SI5  | The robot arm must not be extended beyond the platform footprint when the platform moves. |
| SI6  | A gripped box must not be tilted more than $\alpha_0$. |
| SI7  | A collision between a human and the robot arm must not hurt the human. |
| SI8  | The velocity of any point of the robot must not be greater than $V_2$. |
| SI9  | The robot arm must not drop a box. |
| SI10 | The robot arm must not clamp human parts. |
| SI11 | The robot gripper must not clamp human parts. |
| SI12 | The robot must not override boxes laid on tables, shelves and robot storage. |
| SI13 | The robot must follow the hand-guiding. |
Safety Monitoring Framework (SMOF)

Safety Monitoring Framework (SMOF)

Guarantee safety invariant from the monitor point of view

i. **Behavior**: automaton of the system in absence of the monitor, containing all paths to the catastrophic states.

ii. **Interventions**: abilities of the monitor to constrain the system behavior.

iii. **Safety and permissiveness**: desired properties of the monitor action.

Safety Monitoring Framework (SMOF)
Safety Monitoring Framework (SMOF)
Safety Monitoring Framework (SMOF)

Synthesis of strategy

- **n warning states** and **m interventions**, the number of candidate strategies is $2^{mn}$
- Tree of strategies will enumerate all the strategies, encode them one by one in the model and call NuSMV
Safety Monitoring Framework (SMOF)

Tree of Strategies:
Safety Monitoring Framework (SMOF)

Safety Monitoring Framework (SMOF)

Analysis of Consistency

(a) Principle: form a global partition based on the real values

\[
\begin{align*}
\text{INVAR } & pf\_vel=0 \iff si4.pf\_vel=0 \land si2.pf\_vel=0; \\
\text{INVAR } & pf\_vel=1 \iff si4.pf\_vel=1 \land si2.pf\_vel=0; \\
\text{INVAR } & pf\_vel=2 \iff si4.pf\_vel=1 \land si2.pf\_vel=1; \\
\text{INVAR } & pf\_vel=3 \iff si4.pf\_vel=1 \land si2.pf\_vel=2;
\end{align*}
\]

(b) Formal encoding: declare a global variable and glue constraints
## Results

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
<th>Test objective</th>
<th>Executions</th>
<th>Assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Arm movement (above the inner workspace), then platform movement.</td>
<td>Permissiveness</td>
<td>No</td>
<td>PASS</td>
</tr>
<tr>
<td>1</td>
<td>The controller tries to reach an arm velocity that is above the safety threshold, with a normal acceleration.</td>
<td>S11</td>
<td>No</td>
<td>S11</td>
</tr>
<tr>
<td>2</td>
<td>The controller tries to reach an arm velocity that is above the safety threshold, with a high acceleration.</td>
<td>S11</td>
<td>No, Yes √</td>
<td>S11</td>
</tr>
<tr>
<td>3</td>
<td>The controller tries to reach a platform velocity that is above the safety threshold, with a normal acceleration.</td>
<td>S12</td>
<td>No</td>
<td>S12</td>
</tr>
<tr>
<td>4</td>
<td>The controller tries to reach a platform velocity that is above the safety threshold, with a high acceleration.</td>
<td>S12</td>
<td>No, Yes √</td>
<td>S12</td>
</tr>
<tr>
<td>5</td>
<td>The controller tries to reach the restricted area.</td>
<td>S13</td>
<td>No</td>
<td>S13</td>
</tr>
<tr>
<td>6</td>
<td>During the platform movement, the arm moves and goes outside the platform footprint.</td>
<td>S15</td>
<td>No</td>
<td>S15</td>
</tr>
<tr>
<td>7</td>
<td>During the arm movement, the arm goes outside the footprint and stops there. Then the platform moves.</td>
<td>S15</td>
<td>No, Yes √</td>
<td>S15</td>
</tr>
<tr>
<td>8</td>
<td>During the platform movement, the arm moves above the inner workspace. Permissiveness (S15)</td>
<td></td>
<td>No</td>
<td>PASS</td>
</tr>
<tr>
<td>9</td>
<td>The arm moves beyond the platform footprint, and comes back to stop above the inner workspace. Then, the platform moves. Permissiveness (S15)</td>
<td></td>
<td>No, Yes √</td>
<td>PASS</td>
</tr>
</tbody>
</table>
Tesla Autopilots Predicts Crash Compilation 2
## Contents

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction to Safety Limiters</td>
</tr>
<tr>
<td>2</td>
<td>Challenges</td>
</tr>
<tr>
<td>3</td>
<td>Safety Monitoring Framework (SMOF)</td>
</tr>
<tr>
<td>4</td>
<td><strong>Safe Reinforcement Learning via Shielding</strong></td>
</tr>
<tr>
<td>5</td>
<td>Leave No Trace</td>
</tr>
<tr>
<td>6</td>
<td>Proposal and Example</td>
</tr>
<tr>
<td>7</td>
<td>Conclusion</td>
</tr>
</tbody>
</table>
Safe Reinforcement Learning via Shielding

- Designed to work with a learning algorithm
- A framework that learns optimal policies while enforcing properties expressed in temporal logic.
- Given a Linear Temporal Logic (LTL) specification that is to be obeyed by the learning system, the authors propose to synthesize a reactive system called a shield.

- Two Types:
  - Preemptive shielding
  - Post-posed shielding
Shielding

Water tank problem example:

Safety Specification in LTL:

\[ G(\text{level} > 0) \]
\[ \land G(\text{level} < 100) \]
\[ \land G((\text{open} \land \neg \text{close}) \rightarrow XX\text{close} \land XXX\text{close}) \]
\[ \land G((\text{close} \land \neg \text{open}) \rightarrow XX\text{open} \land XXX\text{open}) \]
Shielding

Preemptive Shielding:

- The preemptive shielding approach transforms the original MDP $M$ into a new MDP $M' = (S', s_I, A', P', R')$ with the unsafe actions at each state removed, and where $S'$ is the product of the original MDP and the state space of the shield.

- For each $s \in S'$, we create a new subset of available actions $A'_S \subseteq A_S$ by applying the shield to $A_S$. From each state $s \in S'$ the transition function $P'$ contains only transition distributions from $P$ for actions contained in $A'_S$. 
Shielding

Post-posed Shielding:

- The shield monitors the actions $a_t^1$ selected by the learning agent, and overwrites them with $a_t \neq a_t^1$ if and only if the chosen action is unsafe.

- For the executed action $a_t$, the agent assigns a reward $r_{t+1}$ which could be:
  1. Punishment $r_{t+1}' < 0$ for $a_t^1$
  2. Reward $r_{t+1}$ for $a_t^1$

- Both don’t guarantee safety!
Shielding

Results - Water tank Problem:
Shielding

Results - Grid World:

9x9 Grid with Bombs

15x9 Grid with Opponent
Shielding

Results - Grid World:

Specifications:

1. The robot must not crash into walls or the moving opponent agent. This specification applies to both experiments.
2. The robot must not stay on a bomb for more than two consecutive steps. This specification applies only to the 9x9 experiment.
Shielding

Results - Grid World:

- 9x9 Grid with Bombs
- 15x9 Grid with Opponent
Shielding

Results - Self Driving Car:

- The agent uses a Deep Q-Network (DQN) with a Boltzmann exploration policy
Shielding

Results - Atari 2600 Seaquest:

- Used the OpenAI Gym and Python implementation of DQN
## Contents

<table>
<thead>
<tr>
<th></th>
<th>Introduction to Safety Limiters</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Challenges</td>
</tr>
<tr>
<td>3</td>
<td>Safety Monitoring Framework (SMOF)</td>
</tr>
<tr>
<td>4</td>
<td>Safe Reinforcement Learning via Shielding</td>
</tr>
<tr>
<td>5</td>
<td>Leave No Trace</td>
</tr>
<tr>
<td>6</td>
<td>Proposal and Example</td>
</tr>
<tr>
<td>7</td>
<td>Conclusion</td>
</tr>
</tbody>
</table>
Leave No Trace

• Why not use an autonomous agent to prevent accidents?
• This paper looks at training in the real world!
• Manual resets are necessary when robot or environment breaks
• Leave No Trace: An autonomous method for safe and efficient reinforcement learning that simultaneously learns a forward and reset policy
• Two agents - An agent that learns how to do the task at hand but and another to undo it.
Leave No Trace

Algorithm 1 Joint Training

1: repeat
2:   for max_steps_per_episode do
3:     a ← FORWARD_AGENT.ACT(s)
4:     if RESET_AGENT.Q(s, a) < Q_{min} then
5:       Switch to reset policy.
6:     (s, r) ← ENVIRONMENT.STEP(a)
7:     Update the forward policy.
8:   for max_steps_per_episode do
9:     a ← RESET_AGENT.ACT(s)
10:    (s, r) ← ENVIRONMENT.STEP(a)
11:    Update the reset policy.
12:   Let $S_{reset}^i$ be the final states from the last $N$ reset episodes.
13:   if $S_{reset}^i \cap S_{reset} = \emptyset$ then
14:      s ← ENVIRONMENT.RESET()
Leave No Trace

Strategies for Early Abort:

• Optimistic Aborts: Perform an early abort only if all the Q value are less than $Q_{\text{min}}$
• Realist Aborts: Perform an early abort if the mean Q value is less than $Q_{\text{min}}$
• Pessimistic Aborts: Perform an early abort if any of the Q values are less than $Q_{\text{min}}$
Leave No Trace

Hard Reset:

• Set of safe states $S_{reset} \subseteq S$
• Irreversible state if the set of states visited by the reset policy over the past N episodes is disjoint from $S_{reset}$
• Increasing N decreases the number of hard resets
• In an irreversible state, increasing N means that we remain in that state (learning nothing)
Leave No Trace

Results - Grid World:

Number of Resets
Leave No Trace

Results - Grid World:

Relationship between Q value threshold and number of resets and number of steps to solve
Leave No Trace

Results - Continuous Environments:

- Ball in Cup
- Pusher
- Cliff Cheetah
- Cliff Walker
- Peg Insertion
Leave No Trace

Results - Continuous Environments:

1. Status quo outperforms in terms of Rewards
2. Status Quo needs hard reset after each episode
3. Forward only and Ours don’t use hard resets
# Contents

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction to Safety Limiters</td>
</tr>
<tr>
<td>2</td>
<td>Challenges</td>
</tr>
<tr>
<td>3</td>
<td>Safety Monitoring Framework (SMOF)</td>
</tr>
<tr>
<td>4</td>
<td>Safe Reinforcement Learning via Shielding</td>
</tr>
<tr>
<td>5</td>
<td>Leave No Trace</td>
</tr>
<tr>
<td>6</td>
<td>Proposal and Example</td>
</tr>
<tr>
<td>7</td>
<td>Conclusion</td>
</tr>
</tbody>
</table>
Our Proposal

1. Combine Shielding and SMOF to generate safe, warning and catastrophic states automatically.
2. Create Individual rules for handling collisions at different locations of the car.
3. Use Filtering techniques to predict collisions.
4. Use deep learning and state rules to classify “Safe” states, warning states and catastrophic states.
5. Create agents for the sole purpose of avoiding collisions.
Our Proposal

Meta-model of SMOF modelling template

Exemplary behavior model

\[ \text{fo=true } \land \text{ fd>sd) } \lor \text{ fo=false} \]

\[ \text{ro=true } \land \text{ rd>sd) } \lor \text{ ro=false} \]

\[ \text{lo=true } \land \text{ ld>sd) } \lor \text{ lo=false} \]

\[ \text{fo=true } \land \text{ cd<fd<sd} \land \text{ v>v_{safe}} \]

\[ \text{ro=true } \land \text{ cd<rd<sd} \]

\[ \text{lo=true } \land \text{ cd<ld<sd} \]

\[ \text{ro=true } \land \text{ rd<cd} \]

\[ \text{lo=true } \land \text{ ld<cd} \]

\[ \text{fo=true } \land \text{ fd<cd} \]

\[ \text{ro=true } \land \text{ rd<cd} \]
## Contents

| 1 | Introduction to Safety Limiters |
| 2 | Challenges                      |
| 3 | Safety Monitoring Framework (SMOF) |
| 4 | Safe Reinforcement Learning via Shielding |
| 5 | Leave No Trace                  |
| 6 | Proposal and Example            |
| 7 | Conclusion                      |
Conclusion

• No state of the art results!
• No module that can plug and play with any autonomous system
• Formal methods and rules is a start but a versatile system is much needed.
• There is a long way to go!
Questions?
Reference List


M. Binfet-Kull, P. Heitmann, C. Ameling. “System Safety for an Autonomous Driving Vehicle”. in IEEE Intelligent Vehicles Symposium (IV), Stuttgart, Germany. 1998