Reinforcement Learning with Quantitative Verification for Assured Multi-Agent Policies

Joshua Riley[1][a], Radu Calinescu[1][b], Colin Paterson[1][c], Daniel Kudenko[2][d], and Alec Banks[3][e]
[1]*Department of Computer Science, University of York, York, UK*
[2]*L3S Research Centre, Leibniz University, Hanover, Germany*
[3]*Defence Science and Technology Laboratory, UK*

Keywords:
Reinforcement Learning, Multi-Agent System, Quantitative Verification, Assurance, Multi-Agent Reinforcement Learning

Abstract:
In multi-agent reinforcement learning, several agents converge together towards optimal policies that solve complex decision-making problems. This convergence process is inherently stochastic, meaning that its use in safety-critical domains can be problematic. To address this issue, we introduce a new approach that combines multi-agent reinforcement learning with a formal verification technique termed *quantitative verification*. Our *assured multi-agent reinforcement learning approach* constrains agent behaviours in ways that ensure the satisfaction of requirements associated with the safety, reliability, and other non-functional aspects of the decision-making problem being solved. The approach comprises three stages. First, it models the problem as an *abstract Markov decision process*, allowing quantitative verification to be applied. Next, this abstract model is used to synthesise a policy which satisfies safety, reliability, and performance constraints. Finally, the synthesised policy is used to constrain agent behaviour within the low-level problem with a greatly lowered risk of constraint violations. We demonstrate our approach using a safety-critical multi-agent patrolling problem.

1 INTRODUCTION

Multi-agent systems (MAS) have the potential for use in a range of different industrial, agricultural, and defence domains (Fan et al., 2011). These systems, which allow multiple robots to share responsibilities and work together to achieve goals, can be used in applications where it would not be practical or safe to involve humans. Multiple robotic agents fitted with specialised tools and domain-specific functionality can work together to achieve complex goals which would otherwise require human agents to place themselves at risk. MAS could be particularly beneficial within hazardous work environments, such as search and rescue operations (Gregory et al., 2016), or where tasks need to be completed in irradiated places. Indeed this has been seen previously with the Fukushima nuclear power plant disaster, where multiple robots were used to complete jobs (Schwager et al., 2017).

Many of these complex and hazardous environments require the agents to operate independently of direct human control, and it is these environments which are the focus of our study.

Reinforcement learning (RL) is one promising technique which enables agents to learn how to achieve system objectives efficiently (Patel et al., 2011). MAS with RL has been proposed for work within many scenarios and has become a significant research area, including the use of MAS for nuclear power plant inspections (Bogue, 2011).

However, successful deployment of these systems within safety-critical scenarios must consider hazards within the environment, which if not accounted for, can lead to unwanted outcomes and potentially result in damage to the system, resources, or personnel.

Such safety considerations and guarantees are missing from traditional RL, which aims to learn a policy which maximises a reward function without consideration of safety constraints (Garcia and Fernández, 2012). An RL policy defines which action an agent should take when it finds itself in a particular

[a] https://orcid.org/0000-0002-9403-3705
[b] https://orcid.org/0000-0000-0000-0000
[c] https://orcid.org/0000-0000-0000-0000
[d] https://orcid.org/0000-0000-0000-0000
[e] https://orcid.org/0000-0000-0000-0000

state within the problem space.

Our approach extends previous work on safe single-agent RL (Mason et al., 2017; Mason et al., 2018) by integrating formal verification with multi-agent reinforcement learning (MARL) algorithms to provide policies for use in safety-critical domains.

In this work, we present a 3-stage approach for safe multi-agent reinforcement learning. First, we encode the problem as an abstract Markov decision process (AMDP). Abstracting the problem is a common technique used within safety engineering for reducing complexity (Cizelj et al., 2011). The AMDP must contain all relevant information needed to describe the problem space, including all of the features necessary to capture the mandated safety constraints. Next, we synthesise policies for the abstract model using *quantitative verification* (QV), a mathematically based technique for the verification (Kwiatkowska, 2007; Calinescu et al., 2012) and synthesis (Calinescu et al., 2017; Gerasimou et al., 2018; Calinescu et al., 2018) of probabilistic models whose properties and safety constraints are expressed formally using probabilistic computation tree logic (PCTL) (Ciesinski and Größer, 2004). Using QV for this stage allows for formal guarantees that properties will be met such that the policy generated is safe with respect to the defined constraints. Finally, these policies deemed as safe by the verification stage are used to constrain a multi-agent reinforcement learning problem where the agents learn a policy within a ROS simulator which more closely resembles the real-world environment.

In order to demonstrate our approach, we introduce a MARL safety domain based on a MAS patrolling problem. In this domain, two robots share the responsibility of performing tasks within the rooms of a nuclear power plant. They must work together to ensure these rooms are visited three times in order to complete their tasks successfully. However, one of these rooms has very high amounts of radiation—enough to damage the robots unless the three visits of this room are partitioned between the robots in a sensible way. Another requirement from these robots is to ensure their battery does not drop below a certain level, and ideally to finish the objective with spare battery above the minimum requirement.

Our research contributes to the areas of safe MARL and safe RL, specifically to constrained RL (Garcia and Fernández, 2012). To our knowledge, this is the first piece of work to apply safe RL methods to MAS in this fashion. Our approach allows for the use of MARL while having guarantees on meeting all safety requirements without the need to restrict the environment as strictly as previous approaches (Moldovan, 2012).

The remainder of this paper is structured as follows. Section 2 provides an introduction to the relevant tools and techniques used throughout the paper. Section 3 introduces a domain example which we use to demonstrate our approach. Section 4 provides an overview of each stage in our approach. Section 5 evaluates the effectiveness of our approach. Section 6 reflects on related research, and finally, Section 7 gives a summary of the results and future work.

2 BACKGROUND

2.1 Single-Agent Reinforcement Learning

Reinforcement learning (RL) is a technique that enables an agent to learn the best action to take depending upon the current state of the system. This learning makes use of past experiences to influence an agent's future behaviour. In this way, rewards are associated with each possible action as the agent explores the problem space.

The problem space is typically represented as a Markov Decision Process (MDP) with an agent able to select from a set of actions in each state. As the agent moves through the environment, it may choose between using an action known to be beneficial (exploitation) and those actions about which little is known (exploration).

When the action is taken a reward (or penalty) is obtained and the agent updates the reward associated with the state action pair $Q : (s,a) \rightarrow \mathbb{R}$. Q-learning (Patel et al., 2011) is commonly used to find an optimal value for this mapping.

Once the mapping of state, action pairs to rewards is complete, we can extract a policy by selecting the action which returns the maximum reward for the state we are currently in.

A policy can be seen as a mapping of which actions should be taken in each state. An optimal policy is the most efficient collection of state action pairings possible to reach the desired goal. Standard RL is concerned with finding an optimal policy; however, it does not allow for safety constraints to be defined as part of the learning process, which means that an optimal policy may be unsafe.

2.2 Multi-Agent Reinforcement Learning (MARL)

MARL is an extension of single-agent RL in which multiple agents learn how to navigate and work together towards the desired outcome (Boutilier, 1996). There is a great deal of literature exploring the benefits and challenges of MARL discussed at length in (Buşoniu et al., 2010). Benefits include efficiency, and robustness through the division of labour while challenges include ensuring reliable communications and increased complexity. A number of algorithms have been created explicitly for learning in MAS; these algorithms are commonly classified as independent learners, joint action learners, and gradient-descent algorithms (Buşoniu et al., 2010; Bloembergen et al., 2015).

Independent learners employ techniques in which agents learn within a MARL environment but ignore joint actions for reduced complexity. Independent learners are the primary type of algorithm on which this paper focuses. This is largely due to the lack of assumptions that these algorithms need to make about observations made between different learning agents. This means that they are widely applicable to a range of contexts, including those where environmental variables can diminish the reliability of communication. Independent learners typically learn how to react to the presence of other robots because of how other robots alter the environment. However, we note that while this paper focuses on individual learners, our approach is not limited to solely these algorithms.

The specific individual learner algorithm we look at within the context of this paper is Q-learning. While the Q-learning algorithm was developed for single-agent RL, it has been shown to also provide promising results when used in a MARL setting. This learning approach was therefore selected for use in our work due to its simplicity and popularity (Buşoniu et al., 2010; Zhu et al., 2019).

2.3 Quantitative Verification (QV)

When a system is described as a state transition model, QV allows us to determine if quantitative properties of the model hold. QV relies on efficient algorithms which examine the entire state-space of a given model. Probabilistic model checkers such as PRISM (Parker and Norman, 2014) and Storm (Dehnert et al., 2017) allow for such analysis.

The verification process takes as input the model and a set of properties to be checked against that model. For an MDP, these properties are expressed using probabilistic computation tree logic (PCTL) (Ciesinski and Größer, 2004). PCTL, as the name suggests, is a temporal logic and can be used to express functional and safety specifications which need to be met. PCTL is used to work with temporal properties in the form of sequences, and forms a common basis for describing property specification in model checkers.

We may also associate a reward with states and transitions in the model. In this way, we can check bounds on reachability (of fail states, for example) as well as the cumulative reward associated with actions taken within the problem space (e.g. battery usage).

3 DOMAIN EXAMPLE

In order to demonstrate our approach, we have constructed a domain example that takes the form of a patrolling robot system within a nuclear power plant. There have been many situations in which robots have been used within this setting, and new technologies continue to emerge (Bogue, 2011).

The domain is based on the premise of a two robot system which has the shared responsibility of navigating the rooms and hallways of the plant, shown in Figure 1. The system must fulfil the following requirements:

- C1: Visit each room a minimum of three times
- C2: Complete all tasks without exhausting their batteries

Constraint C1 may be considered a functional requirement whilst C2 is a safety requirement since exhausting the battery would lead to a need for robot extraction putting human life at risk. We may also add a functional requirement with respect to C2 to maximise the amount of remaining battery. For our example, we assume that the battery life for a robot is assumed to decrease with every action undertaken in the problem space.

In addition, we note that Room 4 has a significantly high level of radiation and whilst this room must be visited a minimum of three times the amount of time a single robot spends in the area should be limited. An additional safety constraint is therefore added as:

- C3: The amount of time spent in room 4 should be minimised

Radiation can cause serious damage to robots, as well as humans. Therefore, using radiation and avoiding overexposure is a natural safety constraint for us to use within our example domain.

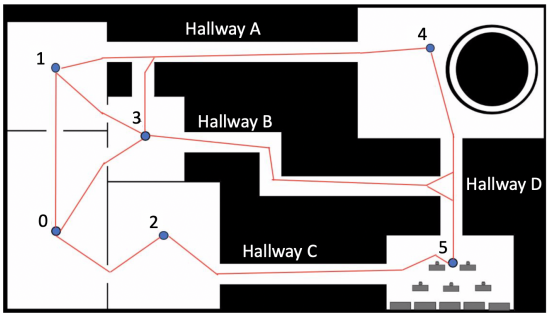

Figure 1: Nuclear reactor map within the simulator, overlaid with states and possible routes.

Table 1: Options for entering and leaving room 4 and the corresponding risk of damage.

Entrance	Exit	Exposure Time	Risk
Hallway A	Hallway A	30 (seconds)	0.03
Hallway A	Hallway D	34 (seconds)	0.04
Hallway D	Hallway D	46 (seconds)	0.07
Hallway D	Hallway A	34 (seconds)	0.04

Figure 1 is a screenshot from the ROS simulator and illustrates the environment within the 3D simulator from a birds-eye view, and the red lines show the movement options between each room for the robots.

Let us consider a robot in room 3. From this state, the robot has 6 possible actions and may move to: Room 0, Room 1 (travelling west), Room 1 via Hallway A, Room 4 via Hallway A, Room 4 via Hallway B or Room 5. Each route will take a different amount of time, and hence a different amount of battery usage will be associated with each transition.

For Room 4, we associate a risk value which is dependent on the route taken through the room and hence the amount of time expected to be spent in the room on average. This is shown in Table 1 and was used as a reward structure for the risk within our model.

4 APPROACH

The approach which we put forward within this paper comprises three main stages, the abstraction of the problem, the synthesising of abstracted safe MARL policies through QV techniques, and finally, the MARL learning within the QV restricted environment. Our approach can be seen visualised in Figure 2, as shown, the domain expert must supply knowledge about the domain, and also supply the desired constraints to allow the problem to be abstracted for easier use within a QV tool. The first two stages of our approach are aimed at obtaining a definition of acceptable safety within the problem domain. The final stage applies MARL techniques in the knowledge that all policies produced will fulfil the safety constraints.

4.1 Stage 1: Constructing an AMDP

In the first stage, it is required that all required information is gathered about the MARL domain. Any information which is not relevant to the constraints that govern the domain problem is abstracted away, with a distinct focus on properties which inform on the safety of the robots. The remaining information should allow for the definition of states, actions or events, rewards, or costs. Our aim in abstracting out all unneeded information is to obtain a model which is small enough for effective and efficient QV whilst retaining sufficient knowledge for meaningful policies to be produced.

For our example, the rooms, as seen in Figure 1, become states in the AMDP. The actions which a robot may undertake in each state are then derived through a consideration of the options available to transition to another room, e.g. 6 possible actions in room 3.

Since constraint C1 requires us to know the number of times a room was visited this leads to each room state being 'expanded' into 4 possible states, i.e. never visited, visited once, twice, three or more times. With this, the policy associated with being in room x is now also a function of how many times the room has been visited.

One way in which the information is abstracted in our domain example concerns the time which it takes for the robots to traverse between locations and the exact routes which an individual robot may follow. However, we take a pessimistic approach to battery usage based on the worst-case distance between the two locations. In the abstracted model, it is assumed that the robots move between locations without complex travel and movement within rooms. Abstracting time from this model dramatically reduces the complexity. Indeed this abstraction is necessary in order for the model to be analysed using QV since large complex models are not able to be analysed using traditional computing resources.

4.2 Stage 2: Synthesising Abstract Policies

In this stage, the AMDP previously generated is analysed using quantitative verification (QV). A QV tool such as Prism allows us to describe the AMDP in a state-based language. Below we show a fragment of the model for our domain example.

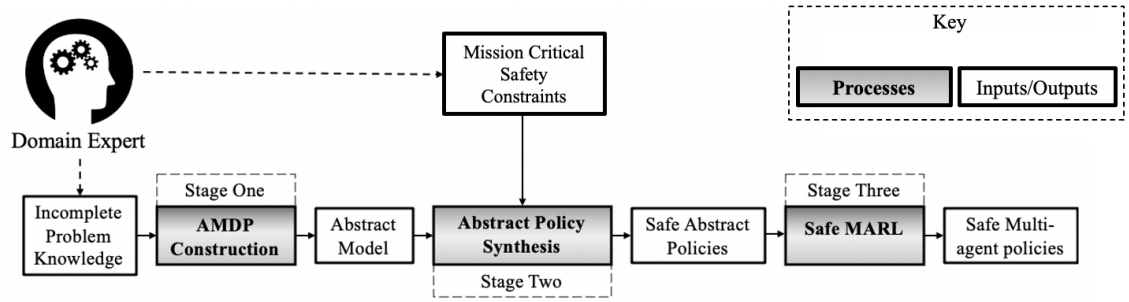

Figure 2: The three stages of our assured MARL approach

```
// In room Zero and making a movement choice
  [visit0_1_1]  !done & r1=0 & visits1<N ->
1:(r1'=1)&(visits1'=visits1+1); // robot 1
visits room 1
  [visit0_1_2]  !done & r2=0 & visits1<N ->
1:(r2'=1)&(visits1'=visits1+1); // robot 2
visits room 1
```

In this fragment, we examine the option of moving from room 0 to room 1 for robots 1 and 2. Here we see a done variable which is used to indicate the task of complete and a counter visits1 which indicates the number of times room 1 has been visited. r1 and r2 indicate the location of robots 1 and 2 respectively. Here we see that if the action is taken to move to room 1 for robot 1, then the robot locations are updated, and the counter associated with room visits is incremented.

```
rewards "energy"
  [visit0_1_1]  true : 3;
  [visit0_2_1]  true : 1.5;
  [visit0_3_1]  true : 2;
```

Within this next fragment, shown above, we show how battery expenditure is represented within the ADMP as a reward structure. The code within the square brackets relates to the option names shown within the first fragment and assigns a numerical reward if the corresponding action is taken. This numerical reward will be larger or smaller, depending on the battery consumption related to the action choice.

Having defined the states, actions and rewards associated with the AMDP, we must now encode the functional properties and safety constraints as PCTL for it to be used in the QV tool in the next stage. First, we need to add bounds to constraints such that they may be analysed. The results PCTL with accompanying bounding values are shown in Table 2.

Finally, the QV tool is presented with the model and the constraints and asked to derive a policy which minimises the battery usage, and the amount of cumulative risk. This is achieved using an RMIN command to direct the QV tool and as shown in the code fragment, framing these properties as reward functions.

Table 2: Constraints of the domain example.

Constraints	**PCTL**
C1: The probability of all rooms being visited three times must be at least 0.8	$P_{\geq 0.80}$[F finished]
C2: The battery of the robots must not drop below 0.35	$R_{\geq 0.35}$[F finished]
C3: The risk of damage must not exceed 0.20	$R_{\leq 0.20}$[F finished]

QV can return multiple policies which are all guaranteed to fulfil the safety requirements. Where multiple policies are generated, the user may select a policy by comparing the rewards associated with each policy. For example, policy 1 may use less battery, but policy 2 may have a lower risk of damage.

Where a policy can not be found, it may be necessary to revisit stage 1 and modify the safety constraints.

The formal guarantees which we reference throughout the paper relate to the quantitative analysis which we perform. We describe the domain within PRISM, as mentioned previously, in the form of an AMDP. This description we create within PRISM includes the six rooms, the relative actions to move between these rooms and two agents which can work through this AMDP. We include reward structures which allow us to monitor the battery usage and also counters to determine how much a room as been visited. In this way, we create a simplified representation of our domain problem, including all the information relevant to the safety constraints. By using QV on this simplified representation, we can determine how likely it is for our safety constraints to be met by any given policy.

We note that the formal guarantees produced relate to the model which is analysed by the QV and that where the AMDP is insufficient to capture the problem domain, these guarantees may not hold. It is

vital, therefore that the problem is abstracted appropriately.

This ability to derive policies for which guarantees are possible is significantly different to other forms of RL, and also most other forms of safe RL, minus (Mason et al., 2017), which this paper is largely influenced by.

The policy synthesised from the tool maps states to actions for each agent, and an example of how the synthesised policy may look like can be seen below. In this example, each line is a tuple (r_{in}, r_{to}, id) such that The first number is representing the room the robot is currently in, the second number is representing the room which the robot will move into, and the final number is acting as an ID to show which robot is taking action.

```
2_0_2
3_1_1
1_3_1
3_1_1
1_3_1
...
```

4.3 Stage 3: Safe Multi-Agent Reinforcement Learning

The third and final part of our approach involves learning policies within the non-abstracted domain but within the constraints learnt in the second stage. This entails the partitioning and constraint of tasks and the domain space based on the synthesised policy. These constraints allow the robots to explore and learn about the problem domain without violating the constraints encoded in the verified policy.

This kind of restriction allows the robots to learn within their partitioned tasks, without being able to unnecessarily enter unsafe situations which will conflict with the mission objectives. While under these restrictions, an action may be taken which holds a quantified level of risk, but with the use of QV, we can guarantee that the cumulative risk, and the probability of risky events happening, is bounded. This approach does not aim for optimality, as the most optimal approach may be disallowed during the QV process; it does, however, guarantee a level of safety and quality while increasing the speed of the learning process.

5 EVALUATION

5.1 Experimental Setup

We demonstrated our approach using an openly available ROS simulator (Portugal et al., 2019). The simulator makes use of simulated lasers for localisation and can be used to control physical robots as well as the simulated robots used in our work. Within the simulation, agents must navigate 'patrol points' which are single geographical coordinates within the domain; these patrol points are connected through action choices.

A model of the nuclear reactor was created in the simulator, as shown in Figure 1, and an AMDP was constructed in the PRISM language to represent the rooms and available transitions and rewards we assigned. The constraints we require to be met were encoded as properties in PCTL, as seen in Table 2.

The AMDP was then analysed using PRISM. PRISM allowed a policy to be produced which met the constraints while minimising risk and battery usage. This policy was then used to constrain the state action pairings of each robot. After this was completed, MARL was allowed to run episodically using the constrained state action pairings in the ROS simulator.

To demonstrate the value of our approach, we conducted two sets of experiments, one which makes use of standard MARL and one which utilises our approach. Within our experiments, we make use of the Q-learning algorithm; within this algorithm, we use the discount factor $\gamma = 0.7$ and a learning rate of $\alpha = 0.3$. Within both experiments, we make use of an exploration of $\varepsilon = 0.5$., this simply being the probability that the agents will choose to explore their environment rather than exploit. These values were found experimentally after multiple interactions of testing based on the non-constrained and constrained learning runs, influenced by the nature of the domain and the number of episodes in a learning run.

For our RL implementation, a reward structure was used, which complements this type of patrolling problem. A numerical reward is given every time a room is reached, based on how long it has been left unattended; this is a common reward structure used within patrolling (Portugal and Rocha, 2016). We tailored this reward structure with additional rewards based on how little battery was expected to be used by making an action. This ensured that eventually, the robots would locate a suitable policy while not frivolously using the battery. We tailored this standard reward function for the example domain's requirements; we also end an episode when a failure event occurs.

5.2 Results

For our experiments, 200 learning episodes were run, with a single episode being completed when all rooms

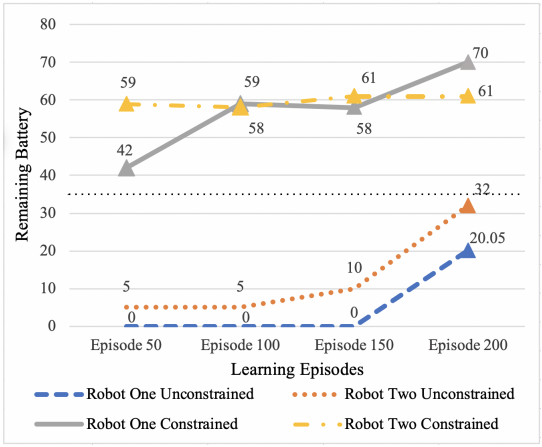

Figure 3: Battery conservation results

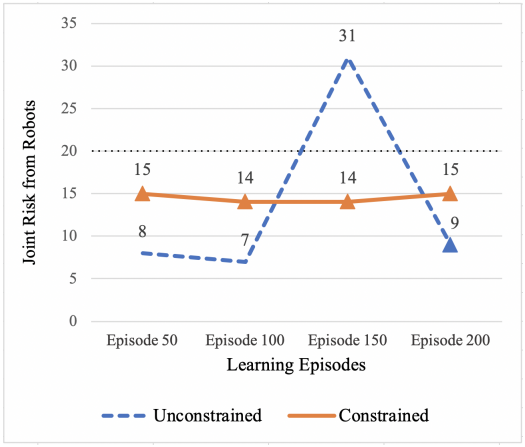

Figure 4: Accumulated risk results

had been patrolled three times. While this is not a large number of learning episodes, it was sufficient for a domain of this simplicity. The results of these experiments are shown in Figure 3 and 4.

The first experiment which was completed was the unsafe baseline experiment with no assured MARL constraints.

It took over 150 learning runs for this to consistently produce an intelligent policy which satisfied the overall mission objective to visit all rooms three times. An intelligent policy is a policy which completes all tasks within the environment without complete loss of battery power. Learning episodes that produce policies that also satisfy constraint C1 from Table 2 are annotated with a triangle.

As can be seen in Figure 3, the battery usage for the unconstrained robots is considerably higher with the battery of robot one often depleted before the end of the episode.

Figure 4 also shows us that the unconstrained robots did violate the amount of permitted time spent in the reactor room. Joint risk, as we refer to it in this paper, is the risk related to time spent exposed to radiation for both of the robots. This drastic increase in joint risk comes from policies which do not complete the mission objective, so continue following their policy until a fail condition is reached, just as with learning episodes 50 and 100, which only visited room 4 twice. However, the final learned policy of the unconstrained robots did produce a policy which completed the mission objective (C1), and to the least amount of possible risk (C3), as seen by unconstrained reaching a risk level of 9, it did this, however, at the cost of failing the battery safety constraint (C2). These experiments show that standard learning, while able to produce intelligent policies, is not guaranteed to meet the safety requirements constraints.

A second experiment was run using a policy constrained by our approach. These constraints saw the responsibilities within the domain partitioned for each agent, resulting in each agent having two rooms which they were solely responsible for and two rooms in which they shared responsibility. This limits the agents' ability to frivolously use their battery, as seen from the constrained values in Figure 3, from very early on, the remaining battery was higher than the safety constraint. This approach, while not removing the most risk of damage, produced a very consistent amount of risk throughout the entire learning run, as seen in Figure 3, never dropping below the safety constraint, this is a function of our approach which constrains the actions to ensure this. It also drastically reduced the search space for both agents and limited the number of learning episodes needed compared to the unconstrained learning run.

The results of the constrained and unconstrained experiments, which are shown in Figure 3 and Figure 4 show that the agents which are constrained using our approach not only learn to complete the patrol quicker but also never exceed the accepted level of risk and quickly learn to conserve its battery to the accepted amount.

6 RELATED WORK

Our approach to safe MARL draws significantly from work within safe RL (Garcia and Fernández, 2015). The majority of these approaches are focused on a single-agent perspective but are directly related to our research. Our approach extends safe MARL past the tailoring of reward functions and the restriction or manipulation in some form on the rewards received; there are many which attempt this in several ways (Serrano-Cuevas et al., 2019; Kroening et al., 2020). There are other techniques for safe RL which

complemented the creation of our approach; these include constraints being placed on which behaviours can be followed and which ones cannot, such as in (Moldovan, 2012) which avoids irreversible actions, amongst others (Moldovan, 2012; Biyik et al., 2019). Our approach to safe MARL uses strict constraints, but unlike other approaches mentioned, does not entirely remove actions which contain risk, and justifies allowing such risk by using guarantees obtained through quantitative analysis. The risk which our approach allows is a calculated risk, which allows missions to be still completed, this differs from these previous approaches,

Assured RL (Mason et al., 2017) made the novel step to incorporate a QV stage into the RL processes, in which it produced very promising results. Our approach applies this directly to our MARL research. Our approach differs from the majority of recent advancements in safe MARL, as it is a multi-step approach that aims to be used in a broad scope of applications, not focusing on one specific problem or domain. As well as this, our approach is not reliant on a specific algorithm, reward structure, or tool, and aims to be flexible to the requirements of the problem. When looking at recent advancements, including research into anti-collision (Zhang et al., 2019; Cheng et al., 2020), learning for automated vehicles (Shalev-Shwartz et al., 2016), limited broad scope approaches to safety have been introduced so far, and others typically follow the trends of single-agent RL. A recent study which works within safe RL makes use of constrained MDPs and proposes a novel policy optimisation algorithm by using convex quadratic functions obtained from policy gradient estimators (Yu et al., 2019). Our approach also makes use of a constrained approach, but through the formal proofs supplied through QV in an abstracted way.

7 CONCLUSION

We introduced a novel approach to Safe MARL, building from a recent advancement in safe RL, utilising QV with a MARL algorithm. Through the use of a domain example, we demonstrated that our three-stage approach allows for MARL policies to be learnt with safety constraints.

Our approach improves upon standard MARL by allowing complex safety, performance, and reliability constraints to be implemented into the learning process of multiple agents. Defining these strict constraints is not possible using reward functions, but we demonstrate how these may be specified using PCTL.

Our approach makes use of an abstracted version of the targeted domain; this means that complete knowledge of the problem does not need to be known to work with the problem. Indeed only limited information is required, including the nature of the performance and safety constraints which are necessary for formal encoding in PCTL. This abstraction also aids the scalability of our approach, which is always a concern when dealing with MAS (Xiao and Tan, 2008).

This approach does not aim for optimality in terms of maximum rewards received from the environment. It aims to produce reliable policies which satisfy all safety constraints.

While our example domain is small in size as it is a first example case. Our aims can be seen being achieved in our example domain, showcasing some of the potentials of this approach.

Future work includes two main points, the first being the examination of how well our approach can be adjusted to work with larger team sizes, and the second how our approach can be used in larger more complex domains and dynamic environment, e.g. (Liu et al., 2020; Gerasimou et al., 2017). Despite the algorithms and tools we used within our example, our approach is largely independent of the learning algorithm chosen, and the tools used. We plan to investigate the generality of our approach by utilising more specialised MARL algorithms within its framework, including MARL algorithms that incorporate deep learning techniques. Lastly, it could be extremely beneficial to apply non-in-dependant learners to our approach, given the plug-in nature of our approach to different tools and algorithms.

Acknowledgements

This paper presents research sponsored by the UK MOD. The information contained in it should not be interpreted as representing the views of the UK MOD, nor should it be assumed it reflects any current or future UK MOD policy.

REFERENCES

Biyik, E., Margoliash, J., Alimo, S. R., and Sadigh, D. (2019). Efficient and safe exploration in deterministic markov decision processes with unknown transition models. In *American Control Conference*, pages 1792–1799.

Bloembergen, D., Tuyls, K., Hennes, D., and Kaisers, M. (2015). Evolutionary dynamics of multi-agent learning: A survey. *Journal of Artificial Intelligence Research*, 53:659–697.

Bogue, R. (2011). Robots in the nuclear industry: a review of technologies and applications. *Industrial Robot: An International Journal*.

Boutilier, C. (1996). Planning, learning and coordination in multiagent decision processes. In *18th Theoretical aspects of rationality and knowledge*, pages 195–210.

Buşoniu, L., Babuška, R., and De Schutter, B. (2010). Multi-agent reinforcement learning: An overview. In *Innovations in multi-agent systems and applications-1*, pages 183–221.

Calinescu, R., Autili, M., Cámara, J., Di Marco, A., Gerasimou, S., Inverardi, P., Perucci, A., Jansen, N., Katoen, J.-P., Kwiatkowska, M., Mengshoel, O. J., Spalazzese, R., and Tivoli, M. (2017). *Synthesis and Verification of Self-aware Computing Systems*, pages 337–373. Springer International Publishing, Cham.

Calinescu, R., Češka, M., Gerasimou, S., Kwiatkowska, M., and Paoletti, N. (2018). Efficient synthesis of robust models for stochastic systems. *Journal of Systems and Software*, 143:140–158.

Calinescu, R., Ghezzi, C., Kwiatkowska, M., and Mirandola, R. (2012). Self-adaptive software needs quantitative verification at runtime. *Communications of the ACM*, 55(9):69–77.

Cheng, R., Khojasteh, M. J., Ames, A. D., and Burdick, J. W. (2020). Safe multi-agent interaction through robust control barrier functions with learned uncertainties. *arXiv preprint arXiv:2004.05273*.

Ciesinski, F. and Größer, M. (2004). On probabilistic computation tree logic. In *Validation of Stochastic Systems*, pages 147–188.

Cizelj, I., Ding, X. C. D., Lahijanian, M., Pinto, A., and Belta, C. (2011). Probabilistically safe vehicle control in a hostile environment. *IFAC Proceedings Volumes*, 44(1):11803–11808.

Dehnert, C., Junges, S., Katoen, J.-P., and Volk, M. (2017). A storm is coming: A modern probabilistic model checker. In *International Conference on Computer Aided Verification*, pages 592–600. Springer.

Fan, Y., Feng, G., Wang, Y., and Qiu, J. (2011). A novel approach to coordination of multiple robots with communication failures via proximity graph. *Automatica*, 47(8):1800–1805.

Garcia, J. and Fernández, F. (2012). Safe exploration of state and action spaces in reinforcement learning. *Journal of Artificial Intelligence Research*, 45:515–564.

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

Gerasimou, S., Calinescu, R., Shevtsov, S., and Weyns, D. (2017). Undersea: an exemplar for engineering self-adaptive unmanned underwater vehicles. In *2017 IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 83–89. IEEE.

Gerasimou, S., Calinescu, R., and Tamburrelli, G. (2018). Synthesis of probabilistic models for quality-of-service software engineering. *Automated Software Engineering*, 25(4):785–831.

Gregory, J., Fink, J., Stump, E., Twigg, J., Rogers, J., Baran, D., Fung, N., and Young, S. (2016). Application of multi-robot systems to disaster-relief scenarios with limited communication. In *Field and Service Robotics*, pages 639–653. Springer.

Kroening, D., Abate, A., and Hasanbeig, M. (2020). Towards verifiable and safe model-free reinforcement learning. CEUR Workshop Proceedings.

Kwiatkowska, M. (2007). Quantitative verification: models techniques and tools. In *6th Joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, pages 449–458.

Liu, Z., Chen, B., Zhou, H., Koushik, G., Hebert, M., and Zhao, D. (2020). Mapper: Multi-agent path planning with evolutionary reinforcement learning in mixed dynamic environments. *arXiv preprint arXiv:2007.15724*.

Mason, G., Calinescu, R., Kudenko, D., and Banks, A. (2018). Assurance in reinforcement learning using quantitative verification. In *Advances in Hybridization of Intelligent Methods*, pages 71–96. Springer.

Mason, G. R., Calinescu, R. C., Kudenko, D., and Banks, A. (2017). Assured reinforcement learning with formally verified abstract policies. In *9th International Conference on Agents and Artificial Intelligence (ICAART)*. York.

Moldovan, T. M. (2012). Safe exploration in markov decision processes. *arXiv preprint arXiv:1205.4810*.

Parker, D. and Norman, G. (2014). Quantitative verification: Formal guarantees for timeliness reliability and performance. *a Knowledge Transfer Report from the London Mathematical Society and Smith Institute for Industrial Mathematics and System Engineering*.

Patel, P. G., Carver, N., and Rahimi, S. (2011). Tuning computer gaming agents using q-learning. In *2011 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 581–588.

Portugal, D., Iocchi, L., and Farinelli, A. (2019). A ros-based framework for simulation and benchmarking of multi-robot patrolling algorithms. In *Robot Operating System (ROS)*, pages 3–28.

Portugal, D. and Rocha, R. P. (2016). Cooperative multi-robot patrol with bayesian learning. *Autonomous Robots*, 40(5):929–953.

Schwager, M., Dames, P., Rus, D., and Kumar, V. (2017). A multi-robot control policy for information gathering in the presence of unknown hazards. In *Robotics research*, pages 455–472. Springer.

Serrano-Cuevas, J., Morales, E. F., and Hernández-Leal, P. (2019). Safe reinforcement learning using risk mapping by similarity.

Shalev-Shwartz, S., Shammah, S., and Shashua, A. (2016). Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*.

Xiao, D. and Tan, A.-H. (2008). Scaling up multi-agent reinforcement learning in complex domains. In *Int. Conf. Web Intelligence and Intelligent Agent Technology*, volume 2, pages 326–329.

Yu, M., Yang, Z., Kolar, M., and Wang, Z. (2019). Convergent policy optimization for safe reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 3127–3139.

Zhang, W., Bastani, O., and Kumar, V. (2019). Mamps: Safe multi-agent reinforcement learning via model predictive shielding. *arXiv preprint arXiv:1910.12639*.

Zhu, C. et al. (2019). A q-values sharing framework for multiple independent q-learners. In *18th Conf. Autonomous Agents and MultiAgent Systems*, volume 1, pages 2324–2326.