Dynamic Interruption Policies for Reinforcement Learning Game Playing Using Multi-Sampling Multi-Armed Bandits

Fa Wu [1] **Rendong Chen** [2] **Shouchao Wang** [1] **Ningzi Zhang** [1] **Sunyun Qi** [1] **Dexing Kong** [3]

Abstract

In many reinforcement learning (RL) game tasks, episodes should be interrupted after a certain time, as the agent could sometimes fall into a deadlock state. This paper provides a dynamic interruption policy in RL game training via a newly defined multi-sampling multi-armed bandit (MSMAB) model, and offers an efficient algorithm named Exp3.P.MS for the new bandit setting. The experimental results show that the dynamic interruptions can adapt to the weak-to-strong performance of the RL agent and spur a fast learning in game playing.

1. Introduction

Reinforcement learning (RL) is a semi-supervised learning model in machine learning, which allows an agent to take actions and interact with an environment so as to maximize the total rewards (Sutton, 1984). This technique has been widely used in game playing (Gelly et al., 2006; Kocsis & Szepesvári, 2006; Lample & Chaplot, 2017). However, in many game tasks, there is a troublesome state called *deadlock* (Junghanns & Schaeffer, 1998). When the RL agent is deadlocked in an episode, the episode will be unsolved, and the learning would come to a standstill or progress slowly. The conventional way to avoid such situations is to interrupt the episode after a fixed number of steps (or time), no matter whether the situation is deadlocked or not. After that, the game level resets to a new one, and the learning process is revitalized in a new environment.

Using a fixed interruption length to avoid unsolvable states in deadlocks is feasible but definitely not a perfect solution. One serious drawback is that the learning process is sensitive to the varying values of interruption lengths, and thus it is hard to get a relatively optimal choice, especially when no prior knowledge exists. If the interruption length is defined too short, many difficult game levels would never be solved. If the interruption length is defined long enough (In some cases, we may even have no idea of how long is long enough), it means that the deadlocked agent would take many useless trails until the episode being interrupted, which directly affects the overall training efficiency.

The above mentioned shortcomings motivate the theme of this paper, where we propose a dynamic interruption policy that can spur a fast learning even when no prior knowledge of optimal parameter settings of the interruption length exists. The dynamic interruption policy models the interruption rules of each episode as a sequential decision problem, and optimize it with the multi-armed bandit (MAB) where each arm is labeled a number representing the steps (or time) after which the episode should be interrupted. In this MAB model, a reward is obtained if an episode is finished within the interruption length, and a cost component is added to measure the training steps (or time). they are both under the nonoblivious adversarial setup. Specially, since the performance of the RL agent changes slowly, in one round, the reward and cost are considered to be from a stochastic joint distribution. And we allow a multi-sampling process in one round. As far as we know, no prior studies have included a similar multi-sampling process in the adversarial MAB problem. So in this work, we call the new MAB model *multi-sampling multi-armed bandit* (MSMAB).

We present an efficient algorithm named Exp3.P.MS for the MSMAB model, achieving an expected regret bound of $\mathcal{O}(\sqrt{nK\ln(K)})$. To the best of the authors' knowledge, Exp3.P.MS is the first approach for solving the nonoblivious adversarial MAB problem with reward and cost pattern. For the Sokoban game, the experimental results show that the proposed dynamic interruption policy is in a gradually increasing level of trialed steps, adapting to the weak-to-strong performance of the RL agent. Therefore, it can spur a fast learning in RL game playing tasks, even if no prior knowledge of optimal parameter settings of the interruption length exists.

[1]Zhejiang Demetics Medical Technology Co.,Ltd. [2]National Engineering Laboratory for Educational Big Data, Central China Normal University [3]School of Mathematical Sciences, Zhejiang University. Correspondence to: Rendong Chen <chenrendonger@163.com>.

Real-world Sequential Decision Making workshop at ICML 2019.

2. Problem Setup and Main Result

2.1. Problem Formulation

According to the statement in Section 1, the proposed MSMAB is an adversarial MAB model enriched by a cost component and a multi-sampling process in each round. Specifically, suppose there are some number of arms. Each arm is labeled with a number representing the steps (or time) after which the game level should be interrupted. At the beginning of each round, first an arm is pulled, i.e., one label is selected, to give an interruption length. For a game level, a reward is obtained if the level is finished within the interruption length. As the steps of trial in each game level are inherently costly, a cost component is added to measure the truly consumed steps. Since the performance of the RL agent changes slowly, in one round, the reward and cost are considered to be from a stochastic joint distribution. And we allow a multi-sampling process in one round. It is not until the current interruption policy is abandoned that we will pull a new arm and a new round will start.

From a global point of view, MSMAB is a nonoblivious adversarial bandit model, which can be clarified by the following two aspects. First, the performance of the RL agent is positively related to the learning time. This means that the reward-cost pattern is not a stochastic setting. Second, in most cases, the generated game levels would be recycled for training. When a game level reappears in training, its reward will be assigned a different value. This equals to say that the reward setting depends on the bandit agent's previous behaviors. Therefore, the adversary is nonoblivious.

First, we take the most general case of the MSMAB problem considering that the sequence of reward and cost vectors are set by the adversary that can depend on the bandit agent's previous behaviors. A concrete example of the reward and cost setting is described in Section 3 for the Sokoban game. Consider a machine with K ($K \geq 2$) distinct arms. At each round t, an adversary assigns arm i a joint distribution $D_{i,t}$ of the reward and cost. The bandit agent selects exactly one of the arms and is dedicated to pulling this arm with a sampling mechanism. The distribution $D_{i,t}$ at one round is assumed to be unchanged. For a stochastic sample $(r_{i,t},\ c_{i,t})$ drawn from $D_{i,t}$, the cost is bounded, say $c_{i,t} \in (0, \tau_i]$ ($\tau_i \leq 1$), and the reward is bounded by the cost, say $r_{i,t} \in [0, a \cdot c_{i,t}]$, where a is a positive constant (WLOG, we adopt $r_{i,t} \in [0, c_{i,t}]$). The regret is to measure the difference between the cumulative rewards when always playing the optimal arm, defined as the expected reward under the resource the bandit agent has consumed, and the realized rewards of the bandit agent,

$$\mathrm{Reg}(n) = \max_{i=1,\dots,K} \sum_{t=1}^{n} \frac{\mathbb{E}[r_{i,t}]}{\mathbb{E}[c_{i,t}]} C_{I_t,t} - \sum_{t=1}^{n} R_{I_t,t}, \qquad (1)$$

where $R_{I_t,t}$ and $C_{I_t,t}$ denote the total rewards and costs at round t.

Algorithm 1 Exp3.P.MS

Parameters: $\eta \in \mathbb{R}^+$ and $\gamma, \beta \in [0, 1]$.
Let p_1 be the uniform distribution over $1, \dots, K$.
For each round $t = 1, \dots, n$
(1) Draw an arm I_t from the probability distribution p_t, and let $R_{I_t,t} = 0, C_{I_t,t} = 0$.
while $C_{I_t,t} < 2 \ln(n+1)$ **do**
  Pull arm I_t and record the reward $r_{I_t,t}$ and the cost $c_{I_t,t}$,

$$R_{I_t,t} \leftarrow R_{I_t,t} + r_{I_t,t}, \quad C_{I_t,t} \leftarrow C_{I_t,t} + c_{I_t,t};$$

  Break the loop with probability $c_{I_t,t}$.
end while
(2) Compute the estimated gain for each arm:

$$\tilde{R}_{i,t} = \frac{R_{I_t,t} \mathbb{1}_{I_t=i} + 3\beta}{p_{i,t}},$$

and update the estimated cumulative gain for each arm:

$$q_{i,t} = \sum_{s=1}^{t} \tilde{R}_{i,s}.$$

(3) Compute the new probability distribution over the arms $p_{t+1} = (p_{1,t+1}, \dots, p_{K,t+1})$ where:

$$p_{i,t+1} = (1-\gamma) \frac{\exp(\eta q_{i,t})}{\sum_{i=1}^{K} \exp(\eta q_{i,t})} + \frac{\gamma}{K}.$$

The definition of (1) is similar to the classic regret form described in the seminal paper (Bubeck et al., 2012). But since at round t, we can only observe the sampling process in arm I_t and the realized cost $C_{I_t,t}$ and reward $R_{I_t,t}$, the cumulative rewards of the best arm are defined as the expected reward under the resource the bandit agent has consumed. A desirable property of this setting is that it allows the comparison of reward under the same time horizon.

2.2. Expected Regret Bound

In this part, we propose an algorithm called Exp3.P.MS and analyze its expected regret bound. It can be considered as a general solver of the MSMAB model (the pseudocode is given in Algorithm 1).

Similar to the Exp3.P algorithm developed in (Auer et al., 2002), algorithm Exp3.P.MS maintains the exponential reweighting of the cumulative estimated gains to define the probability distribution from which the bandit agent will select the arm I_t. The key task is to build an unbiased estimator of $\frac{\mathbb{E}[r_{i,t}]}{\mathbb{E}[c_{i,t}]}$ of any other arm. To achieve this goal, we

take a carefully designed rule for the multi-sampling process. Specifically, for round t after pulling an arm with a cost of $c_{I_t,t}$, the bandit agent will continue to play the current arm with a probability of $1 - c_{I_t,t}$, i.e., to pull a new arm with a probability of $c_{I_t,t}$. This rule is understandable in practical scenario. Suppose a player facing with several bandit arms. When he has little knowledge about these choices, he is much likely to set a psychological budget for each choice. If the cost is small, he is more inclined to make another trial of the current choice, and vice versa. More specifically, at each round, we denote $\lambda_j = (r_j, c_j, \epsilon_j)$, where the index j is used to count the number of sampling and we omit the notations of I_t and t without ambiguity. Here, ϵ_j is a random variable where $\epsilon_j = 1$ means the sampling process would continue, and $\epsilon_j = 0$ represents the sampling process would end. And it satisfies $\mathbb{P}(\epsilon_j = 1) = 1 - c_j$, and $\mathbb{P}(\epsilon_j = 0) = c_j$. Then we can get,

$$\mathop{\mathbb{E}}_{\lambda_{[1:\infty]}}[C_{I_t,t}] = 1, \tag{2}$$

$$\mathop{\mathbb{E}}_{\lambda,I_t}\left[\frac{R_{I_t,t}\mathbb{1}_{I_t=i}}{p_{i,t}}\right] = \frac{\mathbb{E}[r_{i,t}]}{\mathbb{E}[c_{i,t}]}, \tag{3}$$

where the notation J denotes the last time of sampling at round t. Proofs of (2) and (3) are presented in the supplemental materials.

To prevent the sampling process from accumulating too much cost (compared with the fact $\mathbb{E}[C_{I_t,t}]$ equals to 1), we give a comfortable constraint $B = 2\ln(n+1)$ to bound the value of $C_{I_t,t}$. Once $C_{I_t,t} > B$, the procedure of sampling will be interrupted, and we can always get the inequality of $C_{I_t,t} < B + 1$.

Theorem 1 (Expected regret of Exp3.P.MS) *If the algorithm Exp3.P.MS is run with*

$$\beta = \sqrt{\frac{\ln(K)}{nK}}, \eta = 0.25\sqrt{\frac{\ln(K)}{nK}},$$
$$\gamma = \sqrt{\frac{K\ln(K)}{n}}, n \geq 100,$$

then we can obtain that,

$$\mathbb{E}[\mathrm{Reg}(n)] \leq 17.33\sqrt{nK\ln(K)} \tag{4}$$

The proof of Theorem 1 is given in the supplemental materials.

2.3. Modification by Observed Side-Information

In the classic MAB model, the bandit agent can only observe the rewards of the arm pulled, but not the rewards of other arms. However, in some practical situations, the knowledge of the observed actions can sometimes infer some information of other actions. In both Alon et al. (2017) and Alon et al. (2015), they study these partial-information models with the feedback specified by a graph, with which more optimal regret bounds have been reported. Similarly, in MSMAB the label of each arm is visible to the bandit agent, So pulling one arm can partially monitor some information of other arms. The useful side-information can improve the stability of the Exp3.P.MS algorithm.

Reconsider part of the setting in the MSMAB model. There is a machine with K ($K \geq 2$) distinct arms. To link the interruption policy, each arm is labeled a number τ_i (WLOG, for $i_1 < i_2$, we assume $\tau_{i_1} < \tau_{i_2}$) representing the steps after which a game level should be interrupted. To discover the side-information, consider a motivating scenario: at round t, arm I_t is pulled and we receive the cost $c_{I_t,t}$ ($c_{I_t,t} \leq \tau_{I_t}$) and the reward $r_{I_t,t}$. Based on the observation of arm I_t, we assert that for any arm i ($i < I_t$), its reward $r_{i,t}$ and cost $c_{i,t}$ at round t can be confirmed. Specifically, if the RL agent successfully completes the game level within τ_{I_t}, then we can get that the RL agent at time t needs exactly $c_{I_t,t}$ steps to complete the current game level. Thus, for $i < I_t$, if $\tau_i < c_{I_t,t}$, the cost and reward of arm i are τ_i and 0, otherwise, they are $c_{I_t,t}$ and $r_{I_t,t}$. If the RL agent failed to complete the level, this means that completing the current game level will need more than τ_{I_t} steps. Thus, for $i < I_t$, the cost and reward of arm i are τ_i and 0, respectively. In summary, pulling arm I_t can observe the reward-cost information of arms $1, \ldots, I_t - 1$.

Meanwhile, for $i > I_t$, it is easy to verify that,

$$\mathbb{E}[r_{I_t,t}] \leq \mathbb{E}[r_{i,t}],\ \mathbb{E}[c_{I_t,t}] \leq \mathbb{E}[c_{i,t}].$$

Since,

$$\begin{aligned}\mathbb{E}[c_{i,t}] &= \mathbb{E}[\min(\hat{c}_t, \tau_i)] \leq \mathbb{E}[\min(\frac{\tau_i}{\tau_{I_t}}\hat{c}_t, \tau_i)] \\ &= \frac{\tau_i}{\tau_{I_t}}\mathbb{E}[\min(\hat{c}_t, \tau_{I_t})] \\ &= \frac{\tau_i}{\tau_{I_t}}\mathbb{E}[c_{I_t,t}],\end{aligned}$$

where notation $\hat{c}_t$ denotes the exact steps for completing the level at round t. We get a lower bound of $\frac{\mathbb{E}[r_{i,t}]}{\mathbb{E}[c_{i,t}]}$ based on arm I_t,

$$\frac{\mathbb{E}[r_{i,t}]}{\mathbb{E}[c_{i,t}]} \geq \frac{\tau_{I_t}}{\tau_i}\frac{\mathbb{E}[r_{I_t,t}]}{\mathbb{E}[c_{I_t,t}]} \tag{5}$$

Based on the above mentioned facts, we make some modifications of Exp3.P.MS, the details of which is given in the supplemental materials.

3. Experiments

We demonstrate the performance of MSMAB guided by Exp3.P.MS in the Sokoban game. We solve this planning

problem with the deep RL method, where the MSMAB model is used to sequentially provide the game levels with a dynamic interruption policy. The proposed method is compared with the RL approach with constant interruption length, which, to the best of the authors' knowledge, is almost the only way used in RL game learning. We also analyze the impact of MSMAB on driving the RL agent to play through an increasingly difficult level of game tasks.

3.1. Sokoban Problem

Sokoban is a challenging one-player puzzle game whose goal is to push a number of boxes onto target tiles that are scattered over a maze (Ge, 2018; Leme et al., 2015). The maze is defined by a grid occupied by walls and free tiles. The RL agent can walk through the free tiles and push a box to any adjacent free tile of the maze. Since it does not have pull action, many moves are irreversible, the game becomes unsolvable if a box is pushed into a corner or wall in many cases. Such an unsolvable state in Sokoban is called deadlock, which, in the training process, should be interrupted after some steps of trial. Despite its simple rule set, Sokoban has been proved to be NP-hard (Gupta & Nau, 1992) and PSPACE-complete (Culberson, 1999), and there is still no general solver available at present.

3.2. Experimental Setup

The generation algorithm of Sokoban game levels is adapted from Weber et al. (2017), which creates levels with a wild variance in difficulty. A state-of-art deep RL algorithm known as advantage-actor-critic (A2C) (Schulman et al., 2015) is presented for training the generated game levels. For the A2C model with constant interruption length, the game level would be interrupted if it is not completed by a fixed number of steps. For the proposed method (the A2C model combined with the dynamic interruption policy), however, the interruption length is not fixed but dynamic. In the rest of the paper, we call the A2C model with the dynamic interruption policy A2CD for short.

In MSMAB setting, we add 186 arms labeled from 15 to 200, where the number is used to represent the candidate value of the interruption length. The cost c is defined as the consumed steps, and the reward r is generated by a nonoblivious setting. In detail, at the beginning of training, a weighting factor w for every game level is set to 1. If the RL agent completes a level, the bandit agent will earn a reward of $w \cdot c$, and at the same time, the weighting factor for this game level is set to be halved. If the RL agent fails to complete this level, the bandit agent will get 0 reward, and we increase the weighting factor of this game level. Formally, consider it is the m-th training time for a game level. If the game level is successfully completed within the interruption length, then the reward is defined as,

$$r_m = w_m \cdot c_m. \tag{6}$$

The weighting factor of this game level is updated by,

$$w_{m+1} = 0.5 \cdot w_m. \tag{7}$$

If not completed, the reward is,

$$r_m = 0. \tag{8}$$

The weighting factor is updated by,

$$w_{m+1} = w_m + 0.5 \cdot (1 - w_m). \tag{9}$$

3.3. Comparative Results

One configuration of Sokoban is in a 8×8 grid world with 4 boxes, about 3 million game levels are generated, and we stop the training course after 500 million steps. In addition to the A2CD method, we run several A2C models with a constant interruption length of 20, 50, 100, 200 steps, respectively. The fraction of solved levels is calculated in the validation dataset (1% in total), where the interruption length is defined as 200 steps for all methods. Figure 1 shows the learning curves of A2CD and several A2C models with constant interruption lengths. The horizontal axis shows the sum of steps of the RL agent, which represents the cost of training. The vertical axis is the fraction of the solved game levels. We observe that in the first 100 million steps, the A2C model with the interruption length of 20 steps performs the best. But globally, it only has a performance of less than 80% of level solved vs. nearly 85% for the A2CD

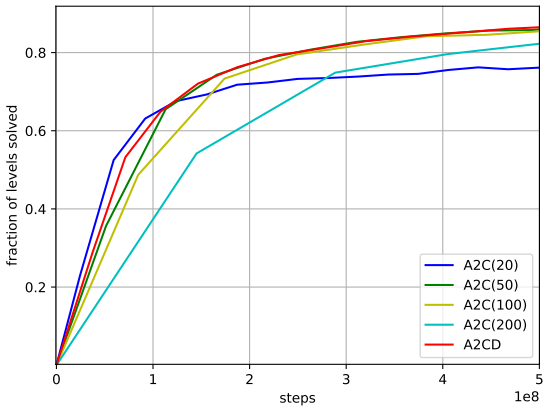

Figure 1. Learning curves of A2CD and A2C models with the interruption length of 20, 50, 100, 200 steps in a 8×8 grid world with 4 boxes.

method. This shows that a too small interruption length may interrupt many game levels which are not yet falling into deadlocks. The A2C model with the interruption length of 200 steps is long enough to be capable of completing the tasks if not deadlocked. But we can see that it wastes lots of time especially in the early stage of training. The truth is that it needs about 420 million steps to get a performance of 80% of levels solved vs. 250 million steps of the same performance in A2CD. The A2C model with the interruption length of 50 steps performs the best in A2C models, with a similar performance as A2CD. So under the current game configuration, the A2C model with the interruption length of 50 steps can be considered as a good choice. But given that to find such an optimal parameter is costly, in this sense, the A2CD model is more advisable.

To validate that the A2CD model is applicable for different configurations, we implement another experiment of Sokoban in a 12×12 grid world with 5 boxes. In Figure 2, it can be seen that the A2CD model shows obvious superiority compared to the A2C models because the fraction of solved game levels is quite higher than that of the A2C models. It is clear that this is achieved with the effect of the dynamic interruption policy for saving a lot of learning time. Remarkably, under this environment, the A2C model with the interruption length of 50 steps (worked well in 8×8 grid world) is not a good solution anymore, with a performance of less than 40% of levels solved. From these, we conclude that A2CD can be used in different game configurations, and can improve the learning efficiency.

3.4. Learning with a Gradually Increasing Interruption Length

In Figure 3, we show the dynamic values of interruption length of A2CD in the 12×12 grid world setting, where the values are averaged for every 50000 steps. At the beginning, the interruption length is uniformly distributed over the interval, and its average is around 108. Then it descends quickly to a low level and finally increases gradually to a stable value. This result shows that the dynamic interruption policy can automatically adapts to the performance of the RL agent. More concretely, for the RL approach, the RL agent is initially unaware of its environment and must learn everything. Therefore, at the early stage of training, the behavior of the RL agent approximates a random walk. This means that the agent could only solve very simple tasks, and for relatively difficult levels, it is very likely to be deadlocked. So a small interruption length at the early training stage is reasonable. With the agent's performance growing over time, a short interruption length may hinder the learning process of the episode with some difficulties (cf. A2C model with the interruption length of 50 steps in Figure 2). This drives the interruption length to increase gradually. With time goes by, the performance of the RL agent and the interruption length will achieve a balance and remain steady as shown in the late part of the curve in Figure 3. The results above suggest that even when no prior information of optimal interruption parameters exits, the proposed dynamic interruption policy can still progress a healthy learning and guarantee a good performance.

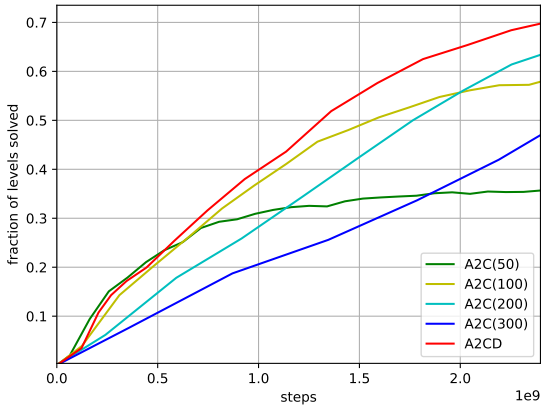

Figure 2. Learning curves of A2CD and A2C models with the interruption length of 50, 100, 200, 300 steps in a 12×12 grid world with 5 boxes.

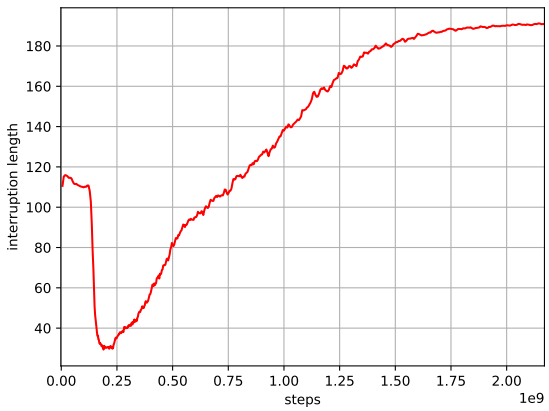

Figure 3. The dynamic interruption length in Sokoban game playing. The displayed results are averaged for every 50000 steps.

4. Conclusion

In this work, we propose a dynamic interruption policy for RL game playing. To achieve it, we have presented a newly defined MSMAB model, which is a nonoblivious adversarial MAB problem enriched by a cost component and a multi-sampling process at each round. We present, analyze and evaluate an algorithm named Exp3.P.MS for this new bandit setting. The experimental results demonstrate that the proposed A2CD model can achieve better performance than the standard A2C method even in different game configurations. Moreover, we show that the dynamic interruption length is in a gradually increasing level of trialed steps, which automatically adapts to the weak-to-strong performance of the RL agent. So it can guarantee an efficient learning in game playing tasks, even if no prior knowledge of optimal parameter settings of the interruption length exists.

References

Alon, N., Cesa-Bianchi, N., Dekel, O., and Koren, T. Online learning with feedback graphs: Beyond bandits. In *JMLR WORKSHOP AND CONFERENCE PROCEEDINGS*, volume 40. Microtome Publishing, 2015.

Alon, N., Cesa-Bianchi, N., Gentile, C., Mannor, S., Mansour, Y., and Shamir, O. Nonstochastic multi-armed bandits with graph-structured feedback. *SIAM Journal on Computing*, 46(6):1785–1826, 2017.

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.

Bubeck, S., Cesa-Bianchi, N., et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1): 1–122, 2012.

Culberson, J. Sokoban is pspace-complete. In *Proceedings in Informatics*, volume 4, pp. 65–76. Citeseer, 1999.

Ge, V. *Solving planning problems with deep reinforcement learning and tree search*. PhD thesis, 2018.

Gelly, S., Wang, Y., Teytaud, O., Patterns, M. U., and Tao, P. Modification of uct with patterns in monte-carlo go. 2006.

Gupta, N. and Nau, D. S. On the complexity of blocks-world planning. *Artificial Intelligence*, 56(2-3):223–254, 1992.

Junghanns, A. and Schaeffer, J. Sokoban: Evaluating standard single-agent search techniques in the presence of deadlock. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pp. 1–15. Springer, 1998.

Kocsis, L. and Szepesvári, C. Bandit based monte-carlo planning. In *European conference on machine learning*, pp. 282–293. Springer, 2006.

Lample, G. and Chaplot, D. S. Playing fps games with deep reinforcement learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Leme, R. R., Pereira, A. G., Ritt, M., and Buriol, L. S. Solving sokoban optimally with domain-dependent move pruning. In *2015 Brazilian Conference on Intelligent Systems (BRACIS)*, pp. 264–269. IEEE, 2015.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

Sutton, R. S. Temporal credit assignment in reinforcement learning. 1984.

Weber, T., Racanière, S., Reichert, D. P., Buesing, L., Guez, A., Rezende, D. J., Badia, A. P., Vinyals, O., Heess, N., Li, Y., et al. Imagination-augmented agents for deep reinforcement learning. *arXiv preprint arXiv:1707.06203*, 2017.