Generalised Discount Functions applied to a Monte-Carlo AI$_\mu$ Implementation[*]

(Extended Abstract)

Sean Lamont
Research School of Computer Science,
Australian National University
sean.a.lamont@outlook.com

John Aslanides
Research School of Computer Science,
Australian National University
john.stewart.aslanides@gmail.com

Jan Leike
Future of Humanity Institute,
Oxford University;
Google Deepmind,
London
leike@google.com

Marcus Hutter
Research School of Computer Science,
Australian National University
marcus.hutter@anu.edu.au

ABSTRACT

In recent years, work has been done to develop the theory of General Reinforcement Learning (GRL). However, there are no examples demonstrating the known results regarding generalised discounting. We have added to the GRL simulation platform (AIXIjs) the functionality to assign an agent arbitrary discount functions, and an environment which can be used to determine the effect of discounting on an agent's policy. Using this, we investigate how geometric, hyperbolic and power discounting affect an informed agent in a simple MDP. We experimentally reproduce a number of theoretical results, and discuss some related subtleties. It was found that the agent's behaviour followed what is expected theoretically, assuming appropriate parameters were chosen for the Monte-Carlo Tree Search (MCTS) planning algorithm.

1. INTRODUCTION

Reinforcement learning (RL) is a branch of artificial intelligence focused on agents that learn how to achieve a task through rewards. Classically, RL methods focus on one specialised area and often assume a fully observable *Markovian* environment. Many problems of interest lack the necessary assumptions to apply such methods. Scaling RL to non-Markovian and *partially observable* real world domains provides the motivation for General Reinforcement Learning, which focuses on designing agents effective in a wide range of environments. (G)RL agents use a *discount function* when choosing their future actions, controlling how they weight future rewards. Several theoretical results have been proven for arbitrary discount functions relating to GRL agents [8].

We present some contributions to the platform AIXIjs [1][2][1], which enables the simulation of GRL agents for gridworld problems. Being web-based allows this platform to be used as an educational tool, as it provides a simple visual demonstration of theoretical results. It also allows the testing of agents in different environments and scenarios, which can be used to analyze and compare models. This makes it a useful tool for demonstrating results within the GRL field. Our main work here is to extend AIXIjs to arbitrary discount functions. Using this, we show it is possible to induce theoretically predicted agent behaviours in a simple concrete setting.

2. BACKGROUND

2.1 Generalised Discounting

Samuelson [11] first introduced the model of discounted utility, which is still used in both RL and other disciplines. Hutter and Lattimore [8] address several issues with this model, using the GRL framework to include the agent's history and the possibility of change in discounting over time. This facilitated a classification of *time consistent* discounting. A policy is time consistent if it agrees with previous plans. As an example, if I plan to complete a task in 2 hours but then after 1 hour plan to do it after another 2 hours, my policy will be *time inconsistent.* They also present a list of common discount functions and which of these are time consistent. These form the basis for our experiments, and we introduce the most notable below. We have omitted geometric discounting in the interest of space, as the results for this function provide little insight. Given a current time k, future time $t > k$, and a discount vector γ, we have:

Hyperbolic Discounting: $\gamma_t^k = \frac{1}{(1+\kappa(t-k))^\beta}, \kappa \in \mathbb{R}^+, \beta \geq 1$, (Time Inconsistent). Hyperbolic discounting is of interest as it is thought to model some irrational (time inconsistent) human behaviour [15].

[*] The full paper can be accessed with the arXiv ID, arXiv:1703.01358 [cs.AI]

Appears in: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

[1] For a thorough introduction to the AIXIjs platform: aslanides.io/docs/masters_thesis.pdf

Power Discounting: $\gamma_t^k = t^{-\beta}, \beta > 1$, (Time Consistent). Power discounting causes the agent to become more far sighted over time, with future rewards becoming relatively more desirable as time progresses *(a growing effective horizon*). This is flexible as there is no need to fix an effective horizon, it will instead grow over time.

2.2 AIXIjs

We implement our experiments using AIXIjs, a JavaScript platform designed to demonstrate GRL results. There are several GRL agents currently implemented to work in (toy) gridworld and MDP environments. Using these, demos each designed to showcase some theoretical result in GRL are presented on the web page. The user can alter parameters and run the demo, while the simulation specific data is used to visualise the interaction. The API allows for anyone to design their own demos based on current agents and environments, and for new agents and environments to be added into the system. There is the option to run simulations as experiments, collecting desired data from the simulation and storing it in a JSON file for analysis.

The source code can be accessed on: `https://github.com/aslanides/aixijs`

While the demos can be found at: `http://aslanides.io/aixijs/` or `http://www.hutter1.net/aixijs/`

There has been some related work in adapting GRL results to a practical setting. In particular, the Monte-Carlo AIXI approximation [16] implemented a AIXI model using the ρUCT algorithm and successfully applied this to various toy settings.

Related to AIXIjs is the REINFORCEjs web demo by Karpathy [6]. This example is restricted by the Q-Learning and SARSA algorithms being defined only for Markovian environments.

3. EXPERIMENTS AND RESULTS

The environment we use is a deterministic MDP, structured to provide a simple means to differentiate myopic and far-sighted agent policies. The idea behind the environment is to give the agent the option of receiving an instant reward r_I at any point, which it will take if it is sufficiently myopic. The other option gives a very large reward r_L only after following a different action for N steps. If the agent is far-sighted enough, it will ignore the low instant reward and plan ahead to reach the very large reward in N time steps. We instantiate to $N = 6$, $r_I = 4$ and $r_L = 1000$ in our experiments. In order to determine time inconsistency, we find the agents plan by traversing the MCTS.

The GRL agent AIμ [4] is designed to find the optimal reward in a known environment. To isolate the effect of discounting, this is the agent used for our experiments to remove uncertainty in the agent's model. We use Monte-Carlo Tree Search (MCTS) as the planning algorithm to approximate the agents search tree (specifically ρUCT [16]). Although UCT [7] would suffice for our deterministic environment, ρUCT is already incorporated into AIXIjs, and as such was used for the planning.

Hyperbolic Discounting: These experiments were performed on commit 3911d of the provided github link. We varied κ between 1.0 and 3.0 in increments of 0.2, and kept β constant at 1. The MCTS horizon was 10, Samples were 10 000 and UCB was 0.01. We found that for $\kappa \geq 1.8$ the agent behaved myopically (red/lowest reward plot in Figure 1), and for $\kappa < 1.8$ the agent behaved far-sightedly (black/largest reward plot Figure 1). For $\kappa = 1.8$, the agent planned to stay at the instant reward for the next cycle and then move off to collect a delayed reward. This plan was the same for all cycles, with the agent constantly pursuing the instant reward and planning to do the better long term action later (essentially procrastinating). The agent was therefore time inconsistent at every cycle. The fact that this behaviour can be induced with this function supports the claim that hyperbolic discounting can model certain irrational human behaviours.

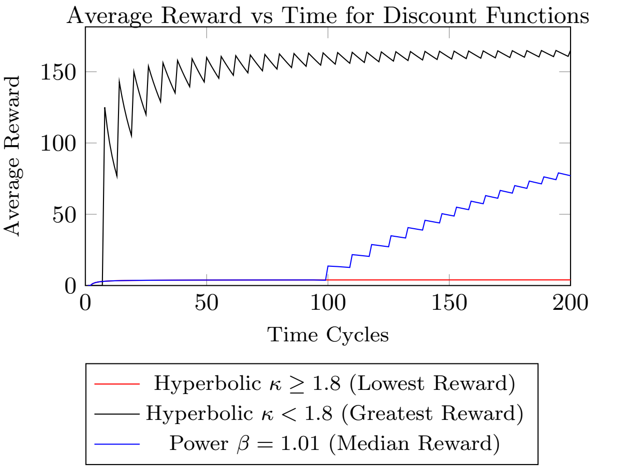

Figure 1: Reward Plot for Discounting Experiments

Power Discounting: We used β= 1.01 for this case. We note that any change in β would result in similar behaviour, with only the length and time between these stages changing. The MCTS horizon was 7, Samples were 100 000 and UCB was 0.001. No time inconsistency was detected for this function. These results can be replicated with the latest version of AIXIjs. This result (See the blue/middle line in Figure 1) shows how a growing effective horizon can effect an agent's policy. Initially the agent is too short sighted to collect the delayed reward, but over time this reward becomes more heavily weighted compared to the instant reward. After some time the agent starts to collect the delayed reward and is fixed to a far-sighted policy. This shows it is possible to recreate this theoretically predicted behaviour in a practical setting.

4. SUMMARY

We have adapted AIXIjs to include arbitrary discount functions. Using this, we were able to isolate time inconsistent behaviour and empirically validate known results on generalised discounting using a simple MDP. We show hyperbolic discounting can induce procrastinating agent behaviour, and that it is possible to observe the impact of a growing effective horizon with power discounting. The AIXIjs platform now permits a larger class of experiments and demos with general discounting, which will be useful for future research on the topic. There will continue to be new results proven for GRL, so an avenue for future work would be to demonstrate those results in a similar fashion to the work presented here. Our contributions would allow for this to be done easily for new results on agent discounting.

REFERENCES

[1] J Aslanides. AIXIjs: A software demo for general reinforcement learning, Australian National University, 2016.

[2] J Aslanides and M. Hutter and J. Leike., General Reinforcement Learning Algorithms: Survey and Experiments, 2016. http://www.hutter1.net/publ/grlsurexp.pdf

[3] R Bellman. Dynamic programming. *Princeton, NJ: Princeton University Press*, 1957.

[4] M. Hutter. A theory of universal artificial intelligence based on algorithmic complexity. *ISDIA-14-00, ISDIA, arXiv:cs.AI/0004001*, 2000.

[5] M. Hutter. Universal artificial intelligence: Sequential decisions based on algorithmic probability. *Springer*, 2005.

[6] A. Karpathy. Reinforcejs, 2015. https://cs.stanford.edu/people/karpathy/reinforcejs /index.html.

[7] L. Kocsis and C. Szepesvari. Bandit based Monte-Carlo planning. *Euro. Conf. Mach. Learn. Berlin, Germany : Springer, pp. 282-293.*, 2006.

[8] T. Lattimore and M. Hutter. General time consistent discounting. *Theoretical Computer Science, 519:140-154*, 2014.

[9] J. Leike. What is AIXI? - An Introduction to General Reinforcement Learning, 2015. https://jan.leike.name/AIXI.html.

[10] G. A. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. *Technical Report CUED/F-INFENG/TR 166, Cambridge University Engineering Department*, 1994.

[11] P. Samuelson. A note on measurement of utility. *The Review of Economic Studies, 4(2) : 155-161*, 1937.

[12] D. Sliver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, N. Kalchbrenner, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature 529, 484-489*, 2016.

[13] R. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning, 3, 9-44.*, 1988.

[14] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, MA, 1998.

[15] R. Thaler. Some empirical evidence on dynamic inconsistency. *Economics Letters, 8(3) : 201 - 207*, 1981.

[16] J. Veness, M. Hutter, W. Uther, D. Silver, and K. S. Ng. A Monte-Carlo AIXI Approximation. *Journal of Artificial Intelligence Research 40: 95-142*, 2011.

[17] C. Watkins and P. Dayan. Q-learning. *Machine Learning, 8, 279-292*, 1992.