ADLR Proposal - Modified MCTS for Elevator Transportation

Tim Pfeifle, Maximilian Rieger

1 Objective

We want to evaluate a variation of the Monte-Carlo-Tree-Search (MCTS) used in AlphaZero [1]. The algorithm proposed in [1] evaluates rewards only at the end of an episode. This is very reasonable for games like Go and Chess, as the only objective in these games is to win the game. For many problems however the performance of the agent depends not only on the final state of an episode, but can already be evaluated during the episode when sub-tasks are solved.
Therefore we propose a change to the state evaluation of the MCTS, which does incorporate observed rewards in every step additionally to the value estimation of a neural network.
We want to apply this modified algorithm to the problem of elevator transportation, which deals with the task of assigning elevators to service passenger's requests. In this environment, rewards such as "passenger arrived at requested floor", can occur in every step. This is common for many other problems as well, to which the modified algorithm could be applied.

In summary our objectives are the following:

1. Modify MCTS for the "reward during episode" problem setting of elevator transportation
2. Compare the performance of the RL approach to heuristic elevator control algorithms

2 Related Work

The problem of elevator transportation was already approached with reinforcement learning in 1996 [2]. Crites et al. conservatively estimated the state space for their example to $\approx 10^{22}$ states and show that reinforcement learning is capable to outperform the best heuristic methods in their problem setting. In a more recent paper Xu Yuan et al. [3] compare Q-value iteration and Q-learning RL algorithms to the task of elevator transportation, as well as describe a mathematical model of an elevator system in detail, making it easy to re-implement. They had to simplify their system so that Q-learning can handle the number of possible states.

In [1] Silver et al. propose a general reinforcement learning algorithm which is hugely successful on several tasks, especially on the game of Go, but only incorporates the reward at the and of an episode. In [4] Laterre et al. show a method of reward modeling, which allows to optimize problems with real-valued outcomes using AlphaZero.

3 Technical Outline

In AlphaZeros version of MCTS, every sample expands one leaf in the action tree. Leaves are then evaluated with a neural network, which gives an estimate of the resulting reward (value v) as well as a prior distribution of promising actions (policy $\boldsymbol{p}$). Each sample descends the current tree by choosing actions a maximizing the sum of the action value $Q(s,a)$ and the upper confidence bound $U(s,a)$ at each state s, while keeping a visit counter $N(s,a)$ for every state-action pair:

$$a* = \arg\max_a Q(s,a) + U(s,a) \tag{1}$$

$$Q(s,a) = \frac{1}{N(s,a)} \sum_{s'} v(s') \tag{2}$$

$$U(s,a) \propto c_{Puct} \frac{\boldsymbol{p}(s,a)}{1+N(s,a)} \tag{3}$$

Our proposed change to the action selection is to use an action value, which incorporates rewards visited in the paths $\pi_{s,s'}$ from the root s to the leaves s':

$$Q_{new}(s,a) = \frac{1}{N(s,a)} \sum_{s'} c_{obs} \cdot f_{norm}\left(\frac{r(\pi_{s,s'})}{|\pi_{s,s'}|}\right) + (1-c_{obs}) \cdot v(s') \tag{4}$$

$$f^k_{norm}(x) = \tanh\left(\frac{x}{k}\right) \tag{5}$$

This method can be combined with ranked reward [4], which maps all episode outcomes, the accumulated rewards, to the value range $[-1, 1]$. This allows better optimization of small improvements and makes training the value network a lot easier, as it can be used with a $\tanh(x)$ activation function. The normalization function in equation (4) maps also the values of the observed rewards to the interval $[-1, 1]$. Now we can choose the weighting between the value net and the observed rewards with c_{obs}.

4 Milestones

The key steps of this project are the following:

- Create an environment for the elevator dispatching problem
- Create a problem generator, which models elevator passengers and their transportation requests
- Implement the AlphaZero algorithm
- Apply the AlphaZero algorithm to the elevateor dispatching problem and evaluate it against heuristics
- Implement our variant of AlphaZero with ranked reward and Q_{new}
- Apply our variant to the problem and evaluate it against the the regular AlphaZero algorithm as well as heuristics

References

[1] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al. "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play". In: *Science* 362.6419 (2018), pp. 1140–1144.

[2] R. H. Crites and A. G. Barto. "Improving elevator performance using reinforcement learning". In: *Advances in neural information processing systems*. 1996, pp. 1017–1023.

[3] X. Yuan, L. Buşoniu, and R. Babuška. "Reinforcement learning for elevator control". In: *IFAC Proceedings Volumes* 41.2 (2008), pp. 2212–2217.

[4] A. Laterre, Y. Fu, M. K. Jabri, A.-S. Cohen, D. Kas, K. Hajjar, T. S. Dahl, A. Kerkeni, and K. Beguir. "Ranked reward: Enabling self-play reinforcement learning for combinatorial optimization". In: *arXiv preprint arXiv:1807.01672* (2018).