Exploring the synergies between transfer in reinforcement learning and procedural content generation

Müller-Brockhausen, M.F.T.

Summary

In this dissertation (titled: *Exploring the Synergies between Transfer in Reinforcement Learning and Procedural Content Generation*) we explore how the two research fields named in the title, namely *Transfer in Reinforcement Learning (TRL)* and *Procedural Content Generation (PCG)* can synergize together.

Our journey began with a comparison of different AI algorithms on the board game *Tetris Link*. We compared a heuristic, Monte Carlo Tree Search (MCTS), and Reinforcement Learning (RL). For both MCTS & RL, the number of possible actions per turn indicates how difficult a game is. In Tetris Link, the number of possible actions per turn is between chess and go. This combined with difficult game mechanics where most actions result in negative points that are hard to recover from make it an interesting game for research. We expected MCTS to perform best here, especially due to our implementation of the game in Rust with a focus on speed, and the fact that our MCTS implementation was heavily parallelized and got a reasonable thinking time. Especially given the thinking time we expected different results, because in principle MCTS will always find the optimal solution, provided that it has simulated every possible outcome. Nevertheless, the heuristic outperformed both RL and MCTS, and even RL outperformed MCTS. Still, we were disappointed by the performance of RL and we were determined to find out how to make it work better.

One promising avenue is using transfer in RL to increase the difficulty of tasks slowly. Instead of learning a problem from scratch, one starts with an easier task. After understanding the simple version, one continues training on a more difficult version. Every change in difficulty is then a transfer in RL. To be sure about the field and what to do we set out to create an overview of the field. To do so we scraped all available papers that contained the three keywords *transfer reinforcement, and learning* in the Microsoft Academic Graph. After combing out duplicates and irrelevant papers we still ended up with almost 300 contributions that we analyzed according to Taylor & Stones TRL categories. The main takeaway from our overview is that TRL mainly focuses on a static amount of tasks that only slightly differ and are all within the same domain. A true cross-domain transfer is an unsolved challenge and the few papers that actually did it had domains that were fairly similar. However, the small incremental task changes gave us the idea that this could be improved by leveraging PCG so that the learned policy generalizes better to unseen task varieties that have not been included in the static task set. This can also be extended to small difficulty increases which then is akin to automated curriculum learning.

Having an idea of how to sensefully apply TRL we attempted our first experiments but quickly hit the problem of reproducibility. Training RL policies based on neural networks is hardly reproducible, a problem that plagues the whole machine-learning field. However, reproducibility is a strongly desirable trait because in TRL every task switch creates a chain of non-reproducibility. While we did not come up with a solution to the problem we did find an intermediary patch: Replay traces. Most RL environments are deterministic and hence reproducible if the same actions are played out with the same initial seed. If we save both we can re-simulate the rest of the data such as observations and rewards while saving hard-disk storage as we do not have to save these. Moreover, sharing replay traces enables reviewers of scientific publications to verify the validity of portrayed figures and tables.

With a solid foundation to do verifiable experiments, we set out to actually apply TRL and PCG together. For that, we transformed the creative *Linerider* game from 2D into 3D space. The game itself is purely creative and does not have a set goal, which has sparked players to come up with wonderful creations such as tracks that play out synchronously to music. This is challenging as after a track is drawn one presses a play button and then a rider on a sled starts to drive down the track without any interaction from the user. Physics and gravity are the only guiding powers here. For our RL-environment, we had to of course define a goal, which in our case is to ensure that the rider arrives at a target position. This goal can be below, above, or at the same height as the rider's starting position. The interesting thing about this environment is that the task to solve itself is an application of PCG. Initial experiments showed that the distance between the starting position and the goal has to be small for RL to learn the task. However, through small incremental increases in distance, hence applying TRL to the problem, we were able to train policies that were able to build tracks for larger distances as well.

Lastly, in the vein of PCG, we also investigated Large Language Models (LLMs) to generate content for games. Related research focuses on having dynamic dialogues where the user can input anything they want to. However, user input to LLMs can easily be used to jailbreak its safety systems and derail them to spew toxic content. Our idea is *Chatter*. Only generate small utterances of NPCs based on pre-defined prompts of developers. We empirically show that this works quite well. Moreover, in this work, we show that the majority of consumer gaming hardware ($\approx 70\%$) would be powerful enough to run a local LLM next to a AAA-Game such as Cyberpunk.

In conclusion, we have shown that PCG and TRL synergize well. PCG can be applied to create many different tasks that help train more general RL policies. And

TRL can help to achieve better results when applying RL to a PCG problem.