Journal, Vol. XXI, No. 1, 1-5, 2013
Additional note

DeepRTS: Machine Learning platform for Real Time Strategy Games

Per-Arne Andersen[1], Dr. Morten Goodwin[1], Prof. Ole-Christoffer Granmo[1]

Abstract
We propose a Deep learning platform, ***DeepRTS*** for edge research in the field of artificial intelligence. Games can be used to compare and develop novel AI techniques which can later be used in real world scenarios. The proposed platform consist of a high performance, real time strategy game with several API gateways for reinforcement learning, deep learning and machine learning development. RTS Games are known to have a immensely high branching factor. This requires algorithms to have a high discover rate, or a technique to easily eliminate the state space pool. This work also present results of existing machine learning techniques applied to the learning platform. An survey of applicable algorithms to the RTS problem is also outlined to create a future work plan for the learning platform.

Keywords
Deep Learning — Reinforment Learning — Neural Network — Deep-Q-Network — Tree Search — Monte Carlo Methods — POMDP — MDP — Real Time Strategy

[1] *Department of Information and Communication Technology, University of Agder, Grimstad, Norway*

Contents

Introduction

Controlling an agent in an environment where the sensory data is sparse, abstract and hard to interpret directly, is one of the most challenging tasks yet to be solved in the realm of reinforcement learning (RL). RL is an area of machine learning, where the agent directly interacts with the environment to learn a *policy*. This policy determines how the agent reacts to its environment, and does a action based on this.

Challenges arises when the problem becomes complex, such as large state and action spaces. To give an idea of what a large state space is we can estimate chess to be 10^{47} [1] , the game of Go to be 10^{170} [1]. The game of Go, was until recently a impossible problem because of hardware limitation, but with todays proccessing power, this became possible. But Go is still not possible to beat using a regular computer, as it required google to construct a distributed machine with 1920 CPUs and 280GPUs having 64 concurrent threads searching for the next best move. [2]

Real-Time-Strategy-Games (RTS) are a game genre which features real time events from multiple players. This renders current methods of state-space searching almost impossible with the technology today. Starcraft 2, worlds most popular RTS game are expected to be one of the biggest challenge of reinforcement learning because of the huge search-space. Its hard to measure its maximum state-space, but one are certain that it expands beyond 10^{1024}. To solve this problem, the algorithm must be able to understand and construct abstractions to the state-space in order to drastically reduce its search space.

This work attempts to take *microRTS*, a RTS simulator closer to the ultimate goal of solving a RTS game. DeepRTS is a RTS engine which attempts to simplify the development of machine learning techniques to solve highly complex games like Starcraft 2. Following sections introduces the platform and some introductory results using Monte-Carlo-Tree-Search (MCTS) and Deep-Q-Learning (DQN).

1. Real-Time Strategy Games

- What is an RTS?

- Why is it hard?
- What exists? (Other research, microRTS.. etc)
- Which algorithms are worth trying out? any algorithms that looks promising? DeepNNs?, MCTS?

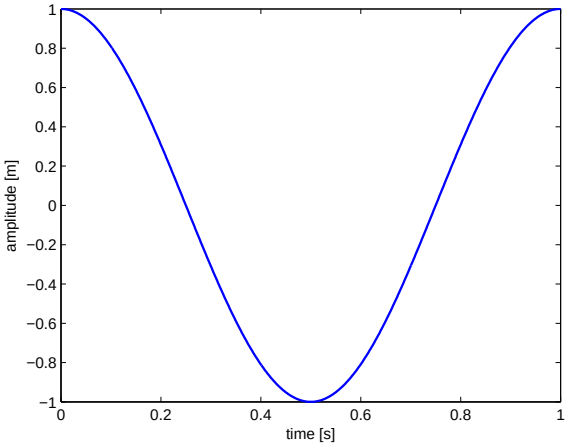


Figure 1. In-text Picture

2. DeepRTS

RTS is as stated, one of the most difficult problems to solve using machine learning [NEED SOURCE]. Continuous state-space with huge action spaces requires computational power beyond the capabilities of current hardware [NEED SOURCE]. In order to solve Star Craft 2, **DeepRTS** is a suited problem to start at.

Figure 2. DeepRTS Simulator

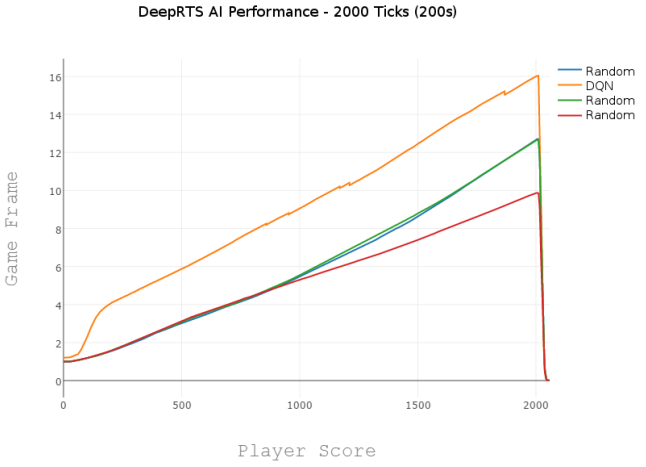
The game objective of DeepRTS is to build a base consisting of an Town-Hall and then expand the base in order to gain military power to defeat the opponents. Each of the players start with a worker. Workers can build and gather resources in the game to improve the players base. [NEED REVISING, skriver så en tulling]. The game consist of two main termonologies, *Micro* and *Macro* management. In order to win, the player with the best ability to both micro and macro mange their resources are most likely to win. [OMMMG REVISE]

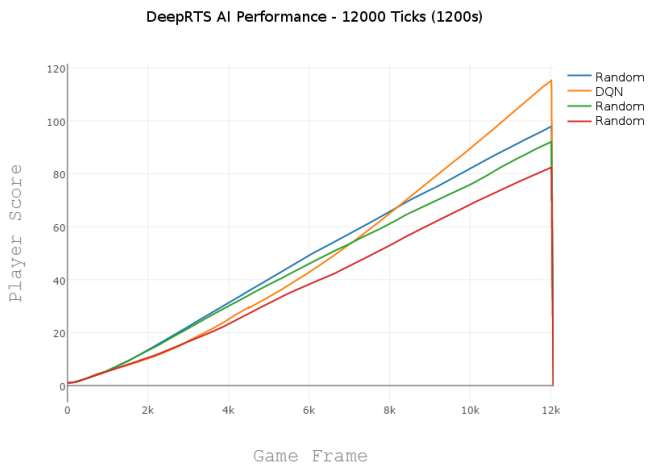
DeepRTS was specifically developed for high-performance simulation for the RTS genre. It is developed in C++ with API for Python, websockets, LUA and ZeroMQ. It focuses on being flexible in configuration to allow for development of different AI approaches, i.e reinforcement learning and unsupervised learning. The simulator also features a python version which can be used with Gym, however it does not perform as good as the C++ implementation, but it is well suited for GPU based machine learning.

The game interface attempts to display all of the important statistics meanwhile a game is ongoing. This being *action execution distribution*, player resources, player scores and a live performance graph. Additionally the game have multiple hot-keys for moderating game-speed and graphics. A Detailed list of these hot-keys can be found by pressing the *G*-hotkey.

One of its major features is that it allow up to 16 players. At current level of AI, a reasonable goal is 3 players. This is because it eliminates possibility for regular min-max strategies [NEED SOURCE]. Some algorithms have been developed for basic game-play, specifically algorithms based on *Monte-Carlo Methods*, these will be thoroughly discussed in the upcoming chapters. The goal of DeepRTS is to improve productivity and the understanding of artificial intelligence, and making it possible to further expand towards a super intelligent AI.

3. Deep Q-Learning





4. Monte-Carlo Methods

Monte-Carlo methods (MCM) works by using random-sampling to attempt to find an optimal solution. Eventually an optimal state will be found given an indefinite time budget. Monte-Carlo Tree search is the algorithm of research covered here, with some alterations which branch into a new algorithm *Monte-Carlo Graph Search*. Tree-Search algorithms are widely used in chess and other turn-based games, and perform well in scenarios where the time budget is high. The more computational time, the bigger branching scope are being discovered and thus yielding a high expertise level. In DeepRTS MCM is challenge becuase it has yet to perform well in games with more than two players . DeepRTS is also an complex game where state-space is beyond the known scope of success for this algorithm genre. Specifically, *Monte-Carlo Direct Approach* shown good results which may be an indication that high-level abstractions can improve performance of tree-seach algorithms [NEED SOURCE]

EED
URCE]

In DeepRTS, all MCM methods have a finite time-budget set to **18** milliseconds. Additionally all of the algorithms are configured with an depth-limit of **10**. Each algorithm uses a unique heuristic for selecting the optimal action per cycle, and these will be discussed thoroughly in the upcoming subsections.

4.1 Upper Confidence Bounds

blablablac

4.2 UCB Action Search

blablala

4.3 Direct Greedy Approach

blablabla

4.4 Graph Search

5. Theoretical Aproaches

- Algorithms that may work

6. Results

Acknowledgments

So long and thanks for all the fish [1].

References

[1] A. J. Figueredo and P. S. A. Wolf. Assortative pairing and life history strategy - a cross-cultural study. *Human Nature*, 20:317–330, 2009.