Australian National University

A implementation of MC-AIXI-CTW Group Project

Johannes Kirschner
Kerry Olesen
Jerry Wu

October 19, 2013

1 Introduction

The AIXI model [Hut05] is an attempt to solve the general AI problem. The AIXI agent interacts with an environment in cycles. Denote by $\mathcal{A}, \mathcal{O}$ and $\mathcal{R}$ an action, observation and reward space respectively. In each cycle, AIXI takes an action $a \in \mathcal{A}$ and receives an observation $o \in \mathcal{O}$ and a reward $r \in \mathcal{R}$. The goal of the agent is to maximize the total future reward. While the environment is not known to the agent, actions are chosen based on past perceptions, which are used to build an model of the environment. More specificly AIXI chooses in cycle k an action

$$a_k = \arg\max_{a_k} \sum_{o_k r_k} \dots \max_{a_m} \sum_{o_m r_m} (r_k + \dots + r_m)\xi(o_1 r_1 \dots o_m r_m | a_1 \dots a_m)$$

The AIXI model is incomputable. In order to make it feasible we have to approximate it. One way to approximate AIXI is the MC-AIXI-CTW [ref] model. Here the expectimax search is solved by an Monte-Carlo approach. The UTC [ref] algorithm is used to balance exploration and exploitation. The class of environment models used in the implemntation is a mixture of d-th order Marcov Decision Process. To effectivly compute this model the Context Tree Weighning method is used [ref]. In the following we present our implementation of the MC-AIXI-CTW model. In section 2 we explain how to use the program and specify different options. Section 3 consists of different experiments we conducted.

2 User Manual

Our approximation of aixi is written in C++ and requires g++ for compilation.

2.1 Setup

Compile:
cd aixi
make

Run:
./aixi file.conf [--option1=value1 --option2=value2 . . .]
Include trained ctw data?? I think this is a good idea Johannes

2.2 Configuration Options

Options can be either specified in the configuration file or passed directly as –option=value to the program. Several configuration files are available, each specifies a particular environment and a set of default options.

Available Options

--environment=env Specifies the environment. Available environments are

- biased_rock_paper_scissor
- coinflip
- kuhn_poker
- pacman
- tiger

--mc-timelimit=N The number N of MC simulations per cycle.

--write-ct=file Write CTW to file before agent termination.

--load-ct=file : Specifies a (trained) CTW for the agent to load at initialisation.

--log=file

--terminate-age=N The number N of agent/environment interaction cycles.

--exploration=P Probability $0 \leq P \leq 1$ that a action is chosen randomly.

--explore-decay=D Decay $0 \leq D \leq 1$ of exploration constant. P is multiplied by D in each cycle.

Domain	CTW depth	m	ϵ	γ	ρUCT Simulations
Biased Rock-Paper-Scissor					
Coinflip					
Kuhn-Poker					
Pacman					
Tiger					

Figure 4.1: Agent configurations

3 Code Documentation

Do we need this??

4 Experimental Results

4.1 Experimental Setup

- List/Make table with configurations used for each environment.
- List hardware (cpu/clock speed/cache/ram)

4.2 Results

Present Results/Graphs of each environment. Maybe mention optimal result and/or scale results accordingly

4.3 Further Experiments

How does aixi handle a change of the envirnment?
Compare 0.3 coinflip to 0.7 coinflip
biased rps vs tiger

4.4 Discussion

How well did it do.
Include results to do with forgetting past model - changing environments
Include statistics about cycles required for optimal performance, time per cycle as in the VNHS paper [1].
TODO: Does anyone have a bibtex library? Or shall we do references manually? We probably won't have many.

References

[1] J. Veness et al. Reinforcement learning via AIXI approximation Technical report, Australian National University, 2009.