Australian National University

# A MC-AIXI-CTW Implementation Group Project

Johannes Kirschner
Kerry Olesen
Jesse Wu

October 20, 2013

## 1 Introduction

The AIXI model **?** 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. AIXI does not require any previous knowledge of an environment, actions are chosen based on past perceptions, which are used to build a model of the environment. More specifically, AIXI chooses in cycle $k$ an action

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

TODO: explain what $\xi$ is, and why it is good (because it is so large)
Unfortunately the AIXI model is incomputable. For all practical applications, the agent must be approximated. One approach in approximating AIXI is the MC-AIXI-CTW **?** model. Here the expectimax search is solved by an Monte-Carlo approach. The UTC **?** algorithm is used to balance exploration and exploitation. The class of environment models used in the implementation is a mixture of $d$-th order Markov Decision Process. Notably, Context Tree Weighting allows efficient linear time computation of this rather general class of models **?**. 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 4 describes the results of the model on several experimental environments.

# 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.

# 3 Code Documentation

Do we need this??

# 4 Experimental Results

## 4.1 Experimental Setup

- List/Make table with configurations used for each environment.
- Present results of 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 the exploration constant affect learning? (tiger env)
How does aixi handle a change of the environment?
– Compare 0.3 coinflip to 0.7 coinflip
– biased rps vs tiger

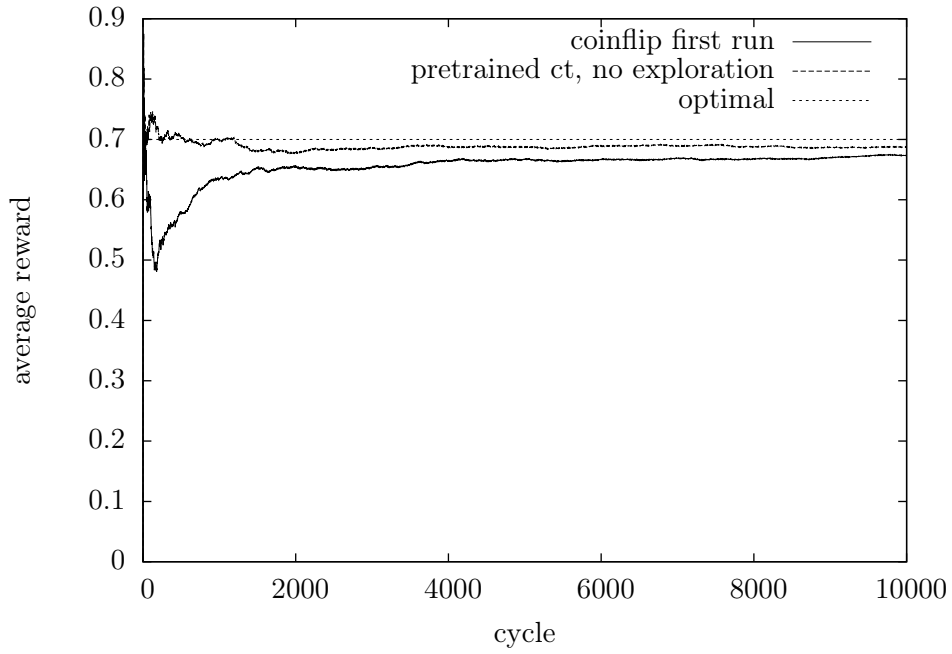


Figure 4.1: coinflip environment