Using Markov Chains to Manage Resource Gathering, Base Building & Unit Building in StarCraft

Giovanni Campo, Patrick O'Halloran & Jeremiah Dunn

This paper will discus an implementation the implementation of certain aspects of an AI controller which could be used in the Student StarCraft AI Tournament (SSCAI).[1] The implementation will focus on the use of Markov chains to manage recourse gathering.

1 Introduction

StarCraft is a *real time strategy* game developed by *Blizzard* and released in 1998. In the game, the player gathers and manages resources, constructs a base, builds an army and attempts to destroy all enemy players. Players choose one of three *races* which each have different play-styles.

In the game's multi-player and single-match skirmish modes, the player can be controlled by a computer agent using the *Brood War API (BWAPI)*. A yearly AI competition for the game, the SSCAI, has been running since 2011 where students submit their own bots to compete in a tournament.

Many academic papers have been written exploring the use of different AI techniques for different components of the game. Our AI will use Markov chains to manage the resource gathering, base construction and unit building for the *Terran* race.

2 Markov Chains

Markov chains are similar to state machines, the difference being that their transitions are governed by some random process.

Agents can be controlled using these systems similar to how they would be controlled by a state machine. To make an agents actions meaningful, some sort of performance metric needs to be created for the agent. The values which dictate the random process which governs state transitions is then tweaked to maximize the performance value.

By applying Markov chains to unit/building construction and managing resource gathering in StarCraft we hope to be able to optimise this component of the game.

3 Mathematical Model

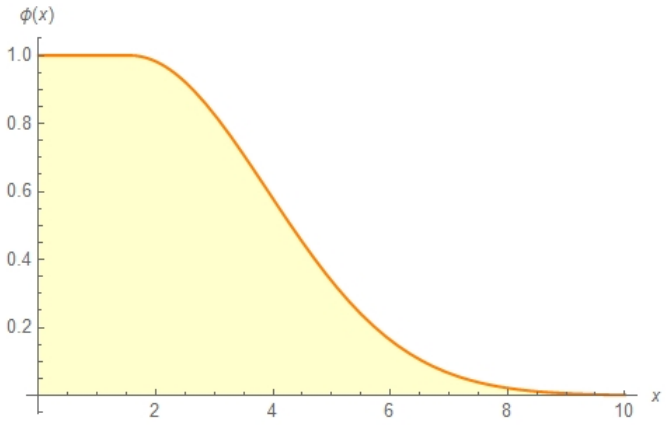
In order to calculate the probability for the Markov chain we decided to use a Gaussian Distribution for our probability density function. In probability theory, the normal (or Gaussian) distribution is a continuous probability distribution, which is a function that tells the probability that any real observation will fall between any two real limits or real numbers, as the curve approaches zero on either side. In our case we want to consider the distribution for positive values of x.

$$\phi(x, \sigma, \mu) = \begin{cases} 1 & x < \mu \\ e^{-\frac{(x-\mu)^2}{2\sigma^2}} & x \geq \mu \end{cases} \tag{1}$$

Equation 1 shows our normal distribution where σ is the standard deviation and μ is the mean. With these two parameters, it is possible to tune our function for different scenarios. For example to decide when an SCV should go building a new Supply Depot or going back harvesting as well as to decide when to build a new unit.

Figure 1 shows the distribution plotted with the aid of *Mathematica*. We picked to random values for mean and standard deviation, but those two will be tuned accordingly to a specific event that the Markov chain will represent.

Figure 1: Gaussian Distribution



4 Implementation

4.1 SCV Units

The *Terran* race uses SCV units to harvest resources and build structures. At the start of a game there are four SCVs available to the player.

Figure 4.1 shows the Markov Chain which dictates the behaviour of the SCV units. The transition from Building to Idle has no probability assigned to it. Building is taken to be a distinct state but exiting the state is controlled by the game, not the agent. The probabilities P_{IB} and P_{IH} refer to the probability that an idle agent will chose to start harvesting or to start building. P_{HB} refers to the probability of a harvesting agent stopping and beginning building.

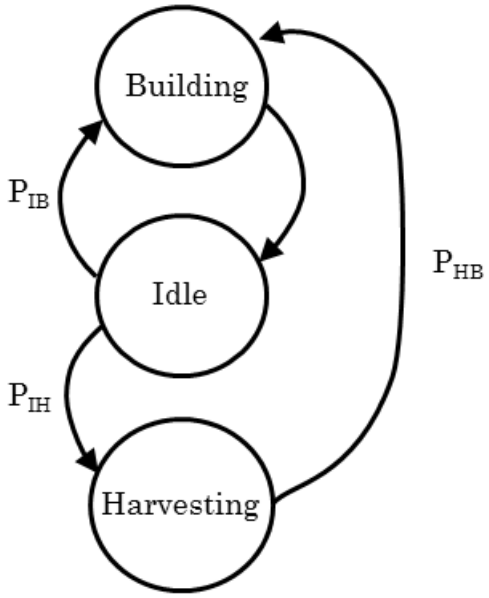


Figure 2: A Markov chain for SCV agent.

4.2 Structures

4.3 Unit Construction Structures

Figure 4.3 shows the Markov Chain which dictates the behaviour of unit constructing structures. The unlabelled actions represent the change state change from building a unit to being idle which the player has no control over. The probabilities P_I and P_B refer to the probability that an idle structure remains idle or builds a unit. P_{UX} refers to the probability of the structure building unit X.

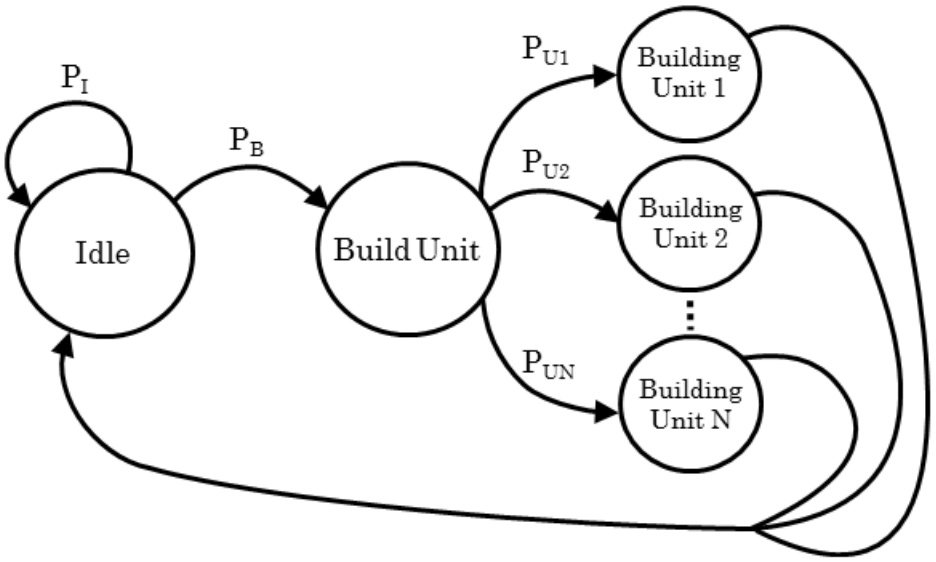


Figure 3: A generalised Markov chain for unit building structures.

4.4 Research Orientated Structures

Figure 4.4 shows a the Markov chain for research oriented structures. The states are in a chain with one direction of flow showing the next available research stage. There are probabilities for continuing research, $P_{R1}, P_{R2}, \ldots, P_{RN}$, and probabilities to remain idle $P_{I1}, \ldots, P_{I(N-1)}$.

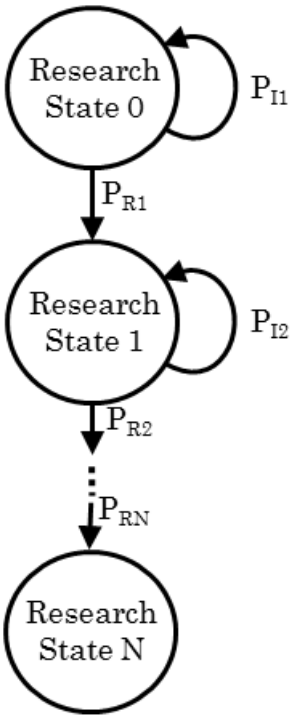


Figure 4: A generalised Markov chain for unit building structures.

5 Results

-what did we achieve?

-have we any way of evaluating our performance?

6 Conclusion

-Difficulty in testing performance *evaluating building and resource gathering not straight forward *what is an optimal result of this process (speed in building attack units?) *how to test against other systems

Future work - better evaluation techniques

-what do we think about the results?

-Patrick might write about future work using genetic algorithms?

-other forms of learning algorithms?

References

[1] Sscai student starcraft ai tournament 2014. http://www.sscaitournament.com/, December 2014.