Vulture Combat

Potential Fields, Reinforcement Learning and Bayesian Networks Applied to Starcraft Broodwar

Analysis of the Game

Elements for Winning a Match

Build Orders

Information Gathering

Macro

Micro

Terran Tactics

Timing Arrack

Pushing

Harassment

Unit Analysis

Marines

Vultures

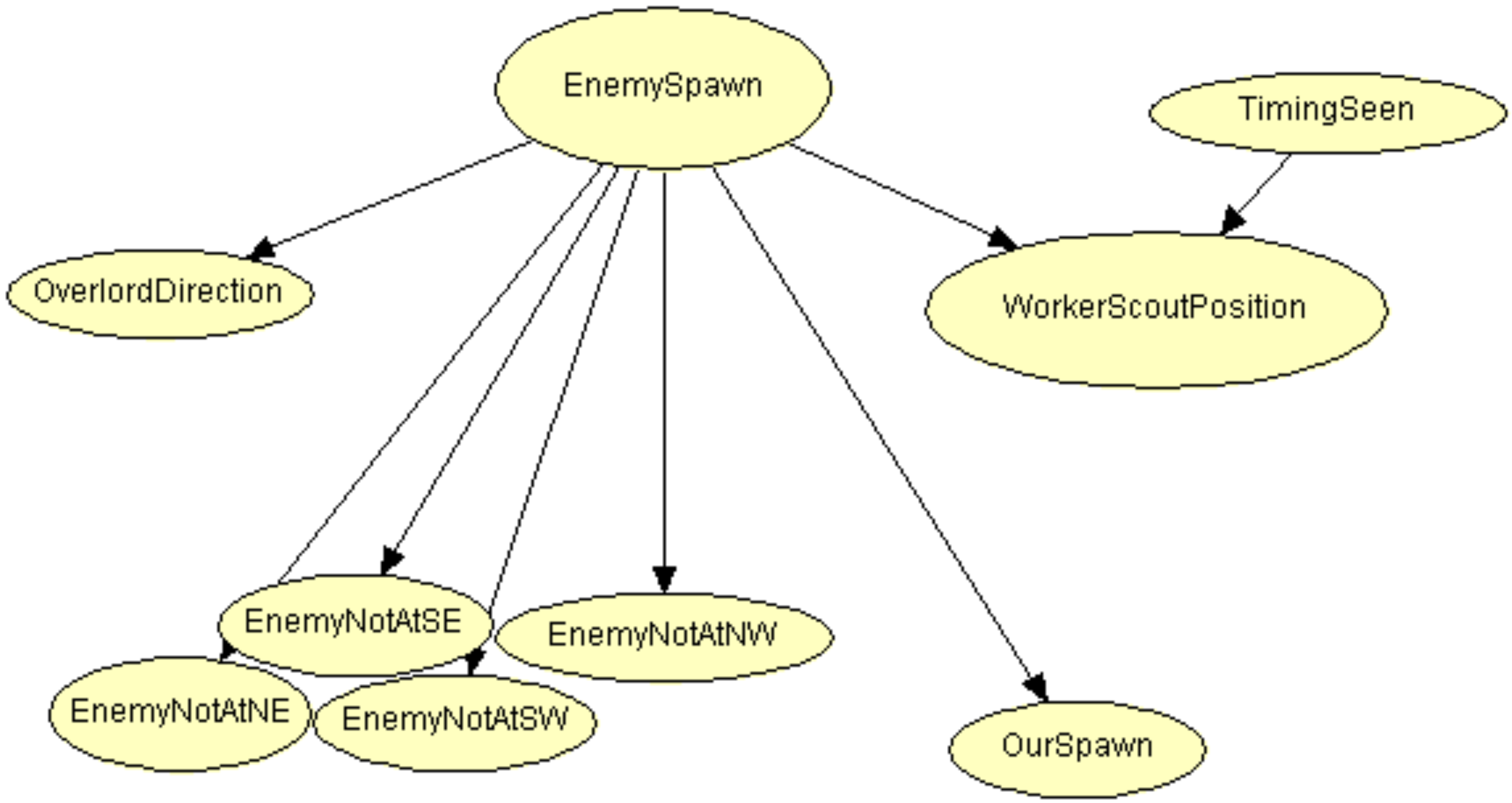
Wraiths

Bayesian Networks

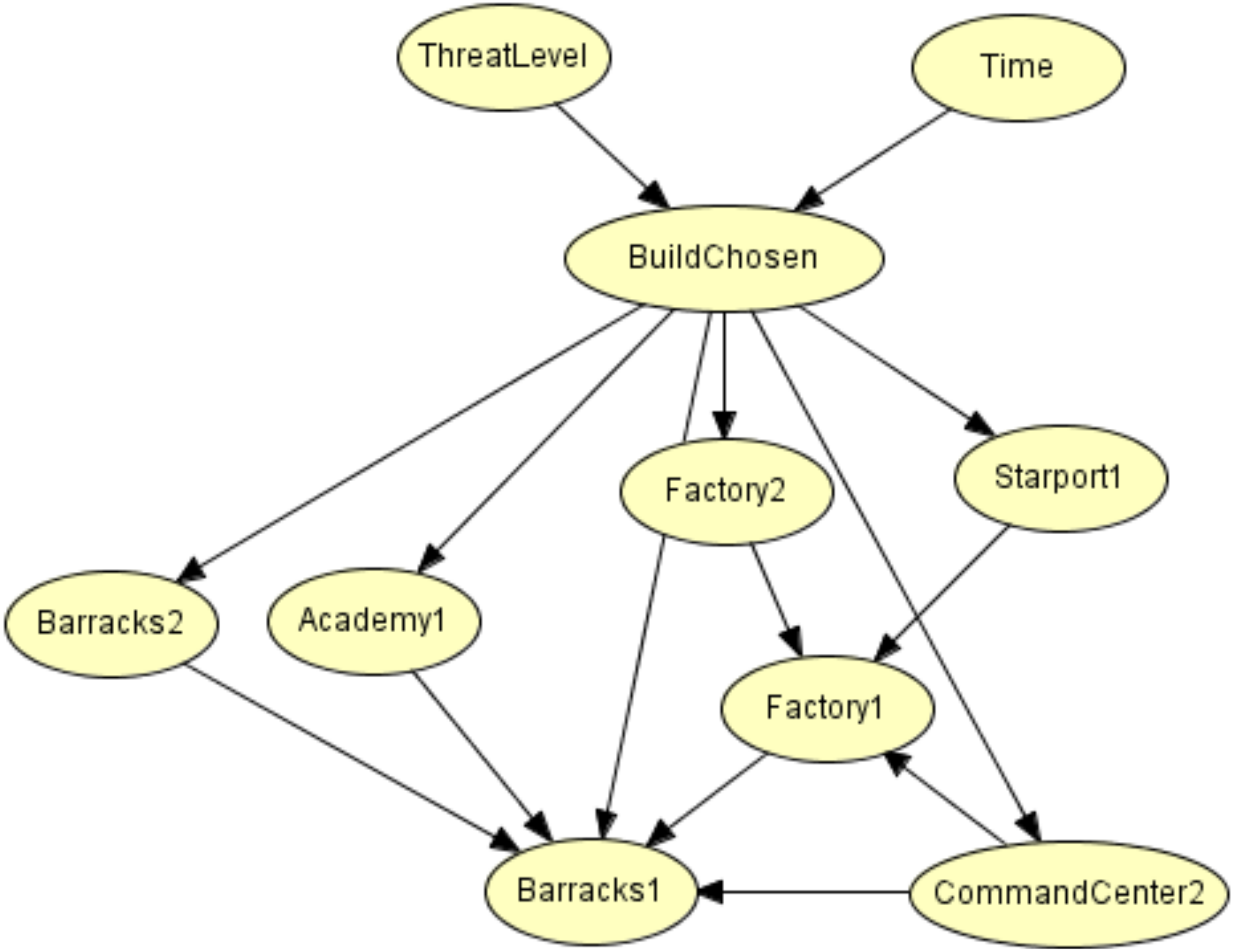
Choice of Decision Model

Bayesian Networks

Decision Trees



Spawn Prediction



Threat Level Prediction

Potential Fields

General Behavior

$$Attractive = \begin{cases} f * c & \text{if } d > s \\ 0 & \text{else} \end{cases}$$

$$Repulsive = \begin{cases} -f * c & \text{if } d > s \\ 0 & \text{else} \end{cases}$$

Example of a Potential Field Function

Behavior determined by sign (+ or -)

Difference between *due* and *de*

Using *(2de - due)*

$$MDP = \begin{cases} f_{MDP} \times (2de - due) & \text{if } de < sr \\ 0 & \text{if } de > sr \end{cases}$$

List of Potential Fields

Maximum Distance Positioning

Edges and Cliffs

Squad Center

Ally Units

Weapon Cooldown

Changes Made During Implementation

Using the numerical representation of a vector.

All weights or forces are positive.

The magnitude of the forces is learned.

Reinforcement Learning

Generalization of Q-Learning

Environment Variables

A State is defined as the combination of all the following characteristics:

Distance to Ally

Distance from Current Tile to Ally

Distance to Center of Squad

Distance from Current Tile to Center of Squad

Distance to Enemy

Distance from Current Tile to Enemy

Distance to Cliff or Edge

Distance from Current Tile to Cliff or Edge

Number of units

Health Lost

Damage Dealt

Number of Units

Time

Weapon’s Cool Down

Shooting Range

Formulas

Q-Approximation

$$\hat{Q}_f = f_{MDP}(2de-due)+f_{AU}(2da-dua)+f_{EAC}(2dc-duc)$$
$$+f_S(2ds-dsv)+f_{CD}(2de-due)$$

Reward

$$R(s) = C_1 numberOfUnits-C_2 healthLost+C_3 damageDealt$$
$$+C_4 numberOfKills-C_5 time$$

Updating Rules

General Rule

$$f_i \leftarrow f_i + \alpha[R(s) + \gamma(max\hat{Q}_f(a', s')) - \hat{Q}_f(a, s)]\frac{\partial \hat{Q}_f(a, s)}{\partial f_i}$$

Application to the Ally Unit Coefficient

$$f_{AU} \leftarrow f_{AU} + \alpha[R(s) + \gamma(max(\hat{Q}_f(a', s'))) - \hat{Q}_f(a, s)](2da - dua)$$

Learning Algorithm

In Every Frame

1. Form the current unit position select the highest $Q(a,s)$ value (where a represents the action of moving to the 8 surrounding tiles).
2. Select the highest value for $Q'(a',s')$ form this state $s' = \delta(a,s)$ (where a' represents the action of moving to the 8 surrounding tiles).
3. Calculate the reward $R(s,a)$.
4. Save all the values.
5. Move to the new position.

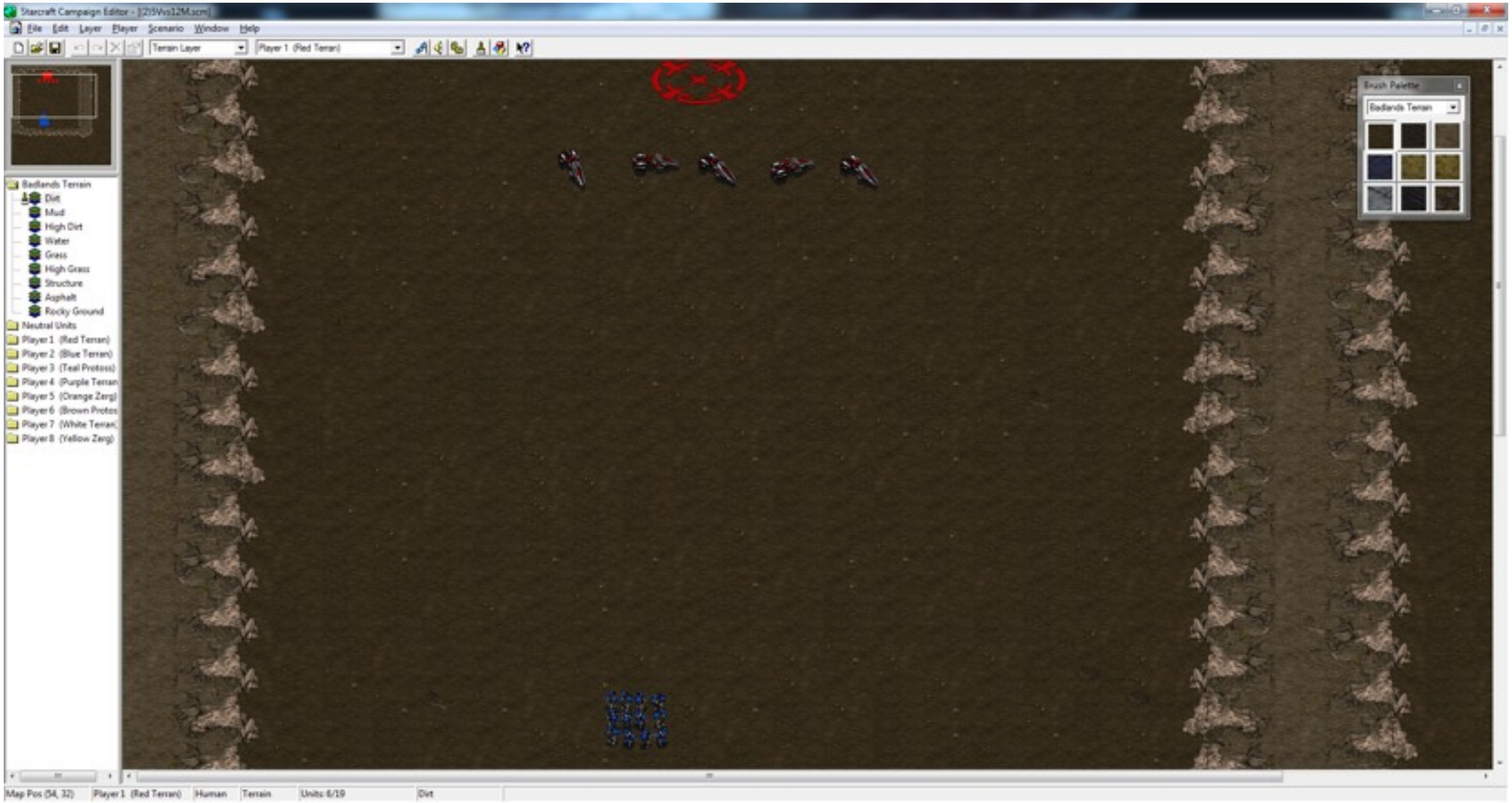
Every 25 frames

1. For every set of values saved in the buffer:

    Calculate the new weights with their corresponding update rule.

2. Save the new accumulated weights to the weights/coefficients used to calculate the next move (*Q* and *Q’* values).

Tests

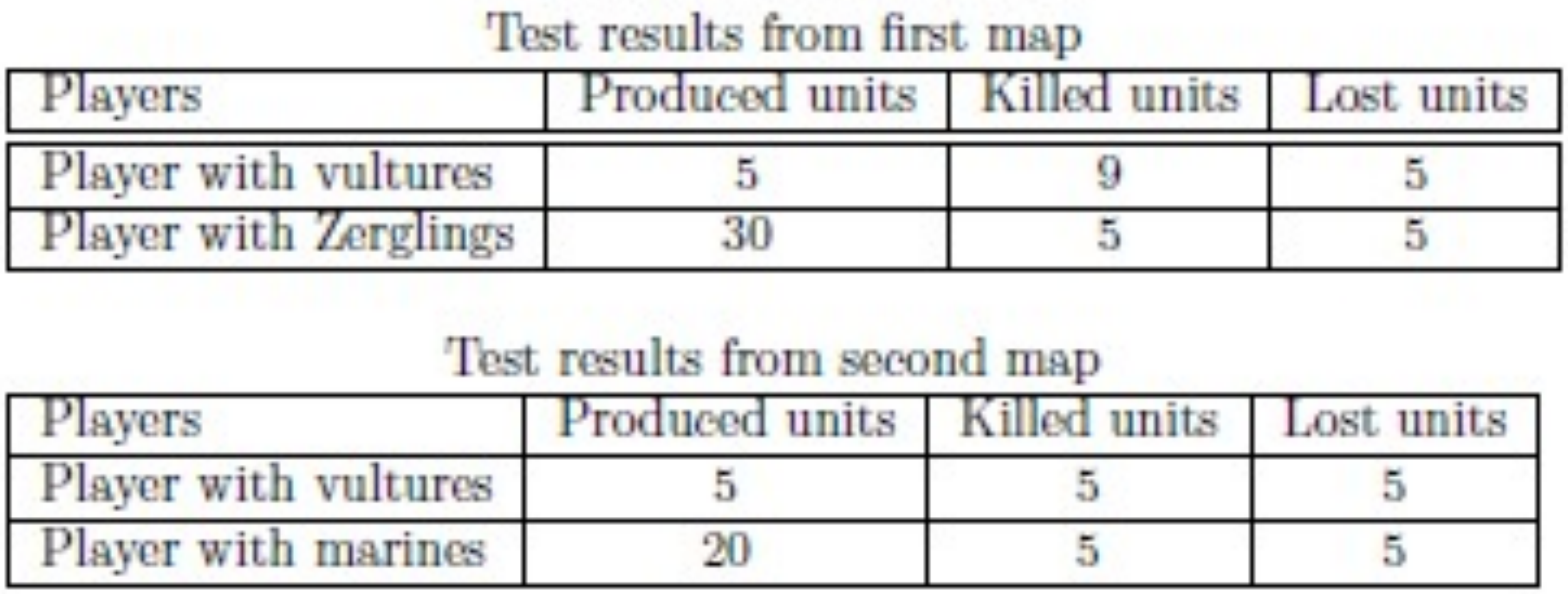


Map Pos (54, 32) | Player 1 (Red Terran) | Human | Terrain | Units: 6/19 | Dirt

Building a Test Map

Base Case

Testing without reinforcement learning or potential fields:



Test results from first map

Players	Produced units	Killed units	Lost units
Player with vultures	5	9	5
Player with Zerglings	30	5	5

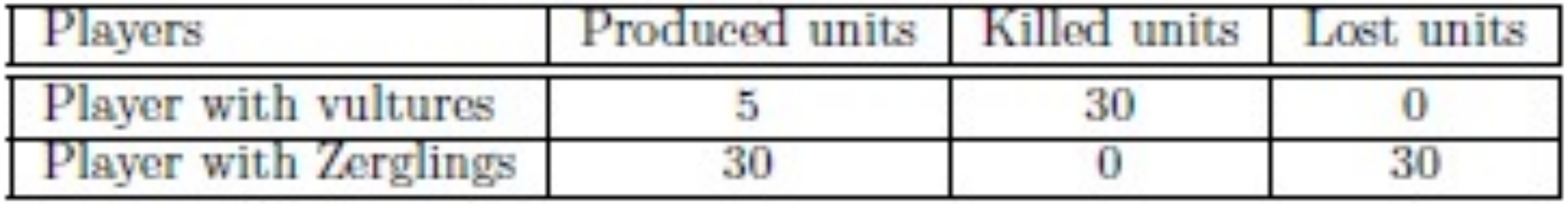
Test results from second map

Players	Produced units	Killed units	Lost units
Player with vultures	5	5	5
Player with marines	20	5	5

Base Case

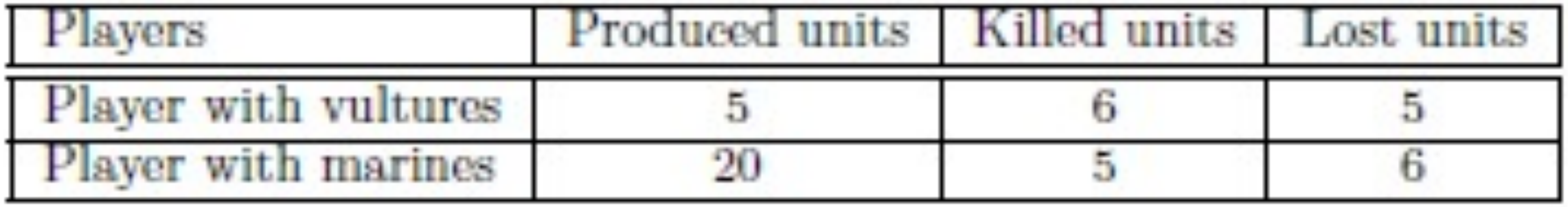
Testing with potential fields, but not reinforcement learning:

Test results from second map



Players	Produced units	Killed units	Lost units
Player with vultures	5	30	0
Player with Zerglings	30	0	30

Test results from second map



Players	Produced units	Killed units	Lost units
Player with vultures	5	6	5
Player with marines	20	5	6

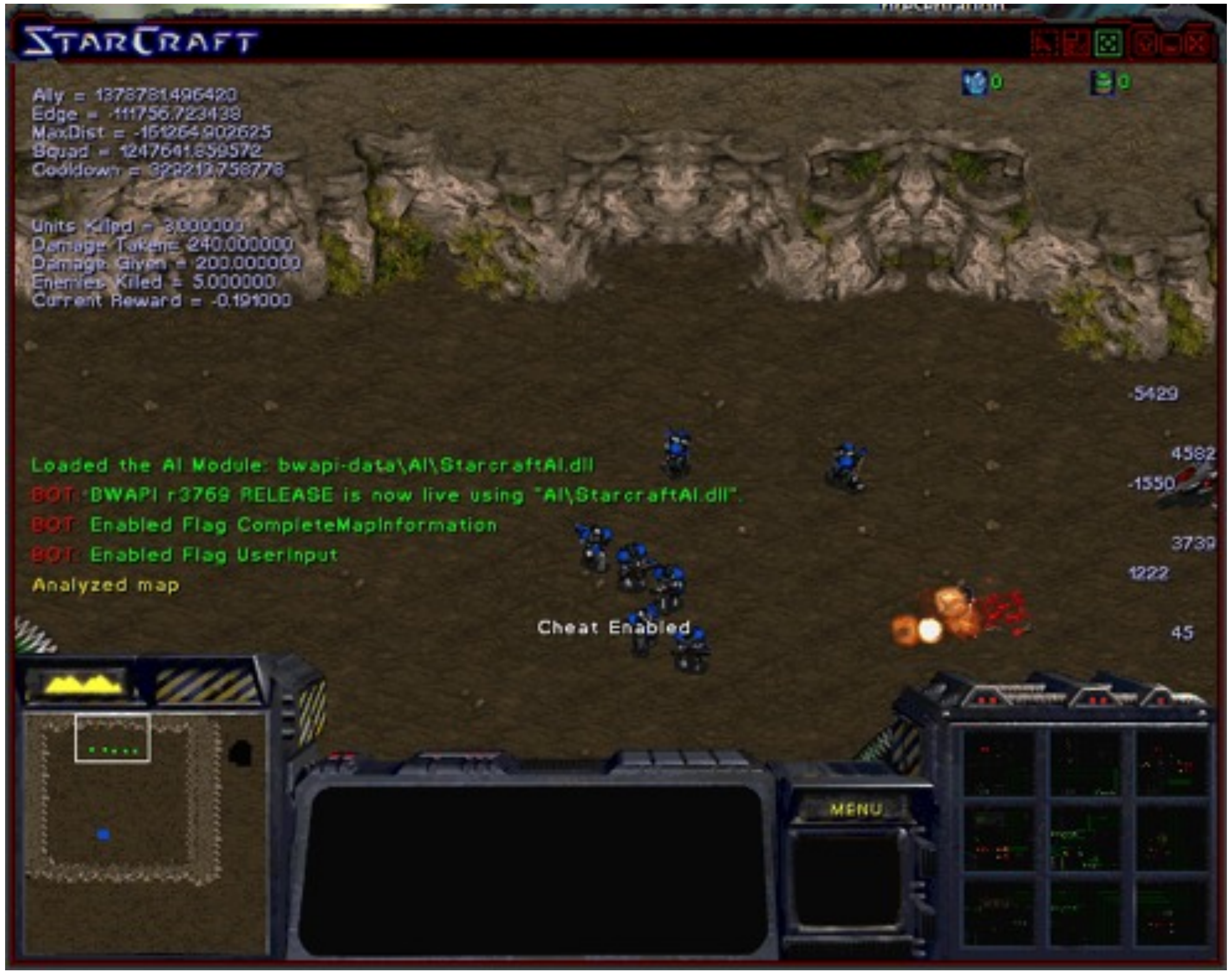
Will it converge?

How do Alpha and Gamma values affect convergence?

How many iterations are needed?

Convergence

Running tests with different values



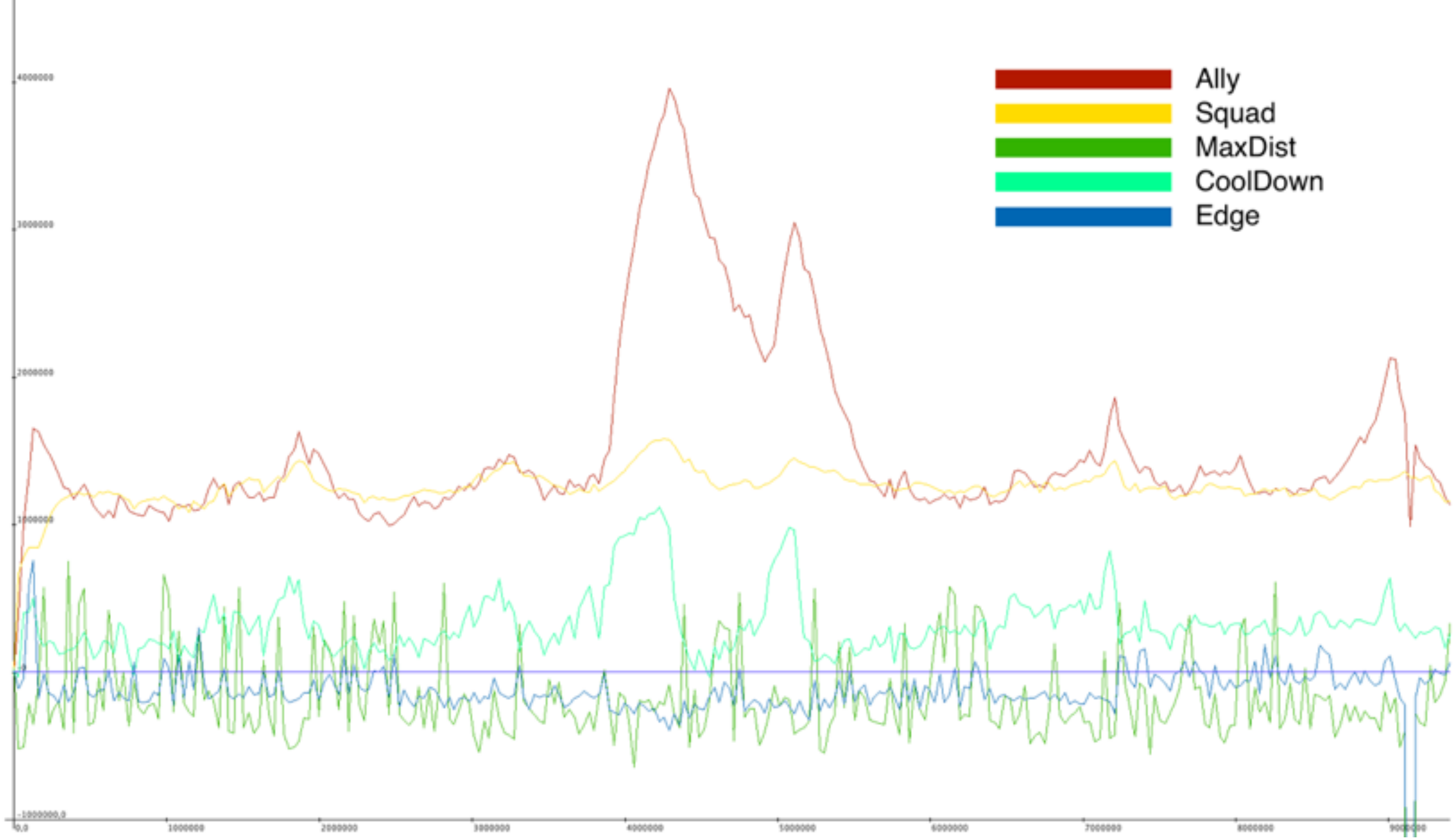
```
Ally = 1378781496420
Edge = -111756.723438
MaxDist = -151264.902625
Squad = 1247641.659572
Cooldown = 322212.758778

Units Killed = 3.000000
Damage Taken= 240.000000
Damage Given = 200.000000
Enemies Killed = 5.000000
Current Reward = -0.191000
```

```
Loaded the AI Module: bwapi-data\AI\StarcraftAI.dll
BOT: BWAPI r3769 RELEASE is now live using "AI\StarcraftAI.dll".
BOT: Enabled Flag CompleteMapInformation
BOT: Enabled Flag UserInput
Analyzed map
```

Cheat Enabled

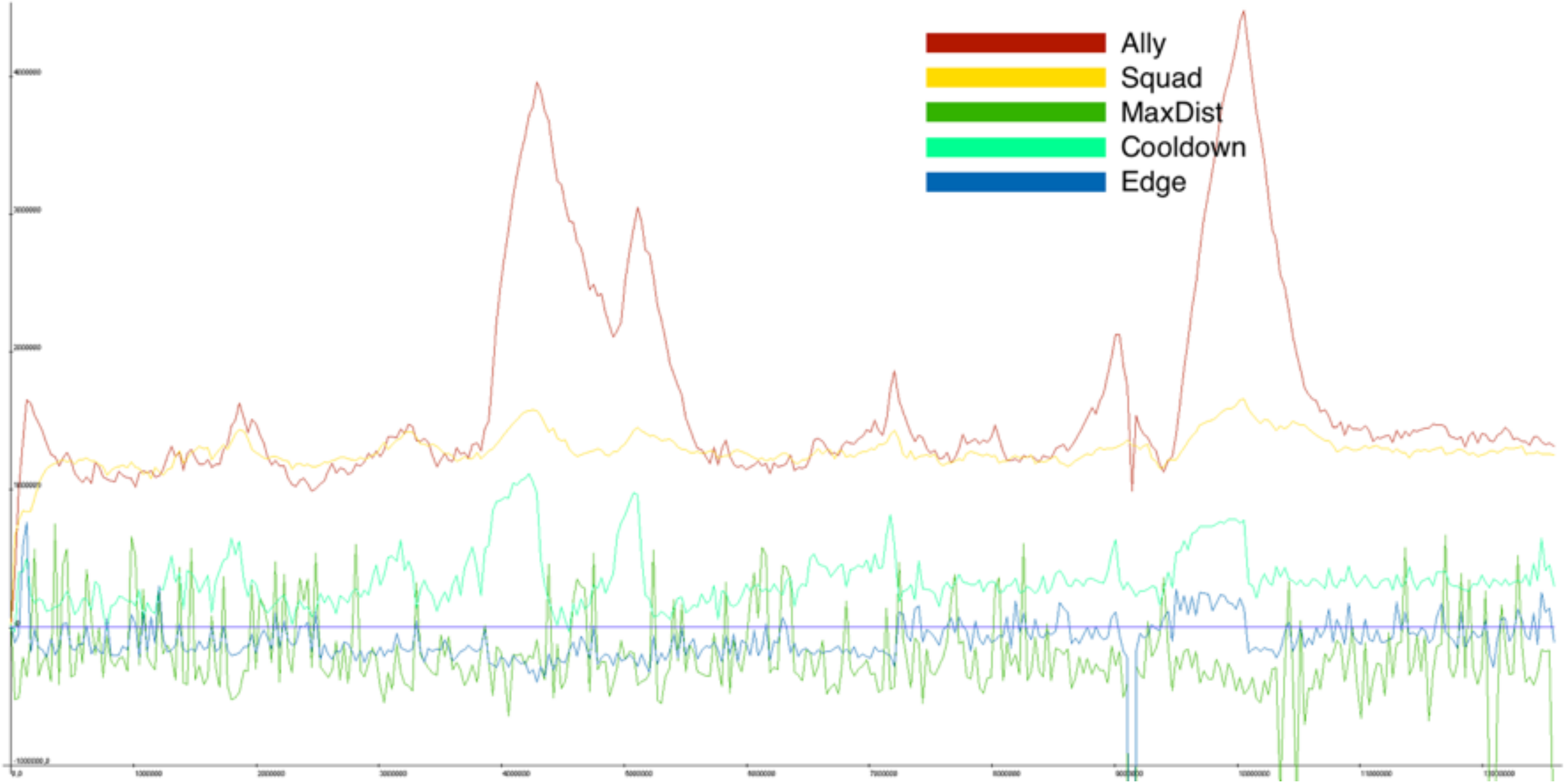
Convergence

Alpha 0.2, Gamma 0.9 (30852 iterations)



Convergence

Alpha 0.4, Gamma 0.6 (135936 iterations)



Conclusion

Identify parts of the games were machine intelligence was applicable.

Apply several concepts of machine intelligence for different circumstances.

Potential Fields + Reinforcement Learning.

Bayesian Networks.

Show improvement through testing.