Introdution

-We are

--What are we going to talk about

Analysis

Analysis

- Elements of wining:
  - Build Order
  - Information Gathering
  - Macro
  - Micro
  -

Analysis

- Terran Tactics
  - Timing Attack
  - Pushing
  - Harassment
  -

Analysis

- Unit analysis
  - Marine
  - Vulture
  - Wraiths

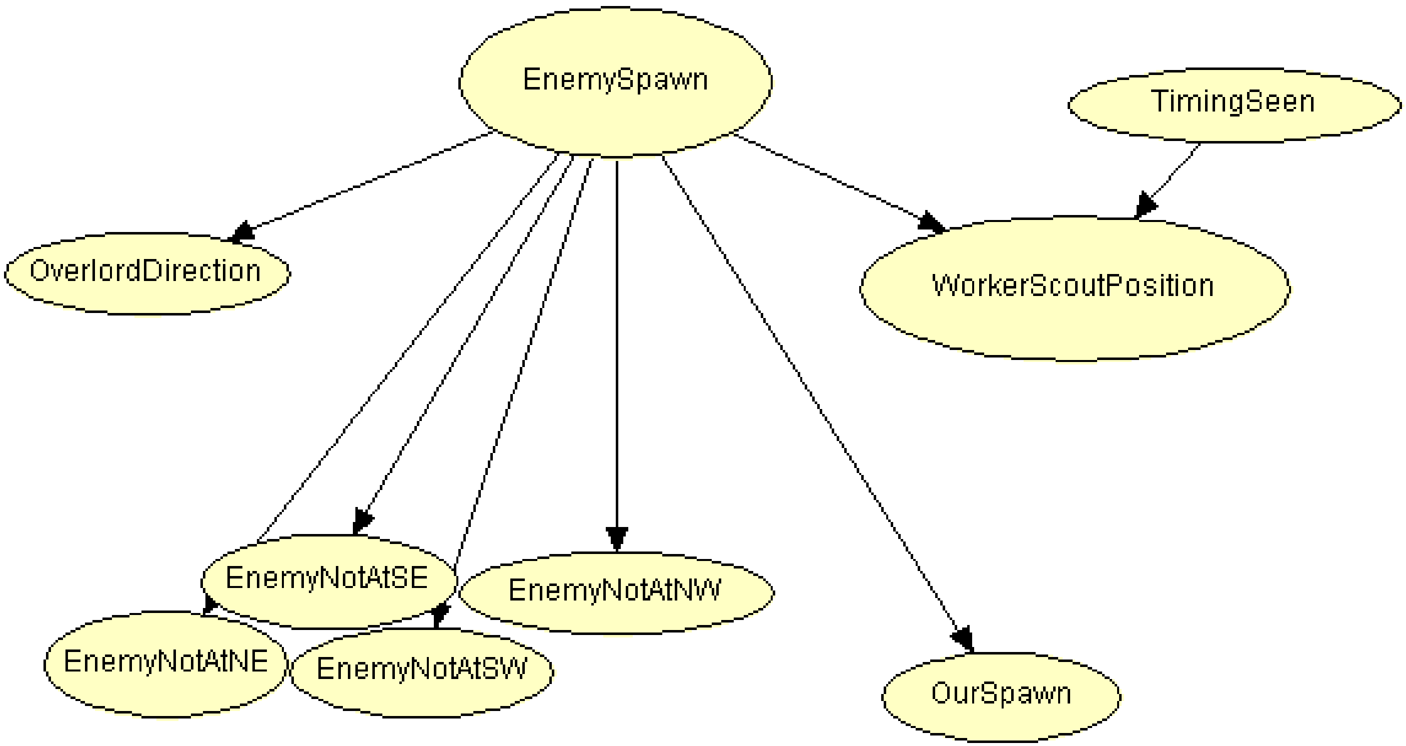
Baysian Networks

Baysian Network

- Choice of decision model
  - Bayesian Networks
  - Decision Trees

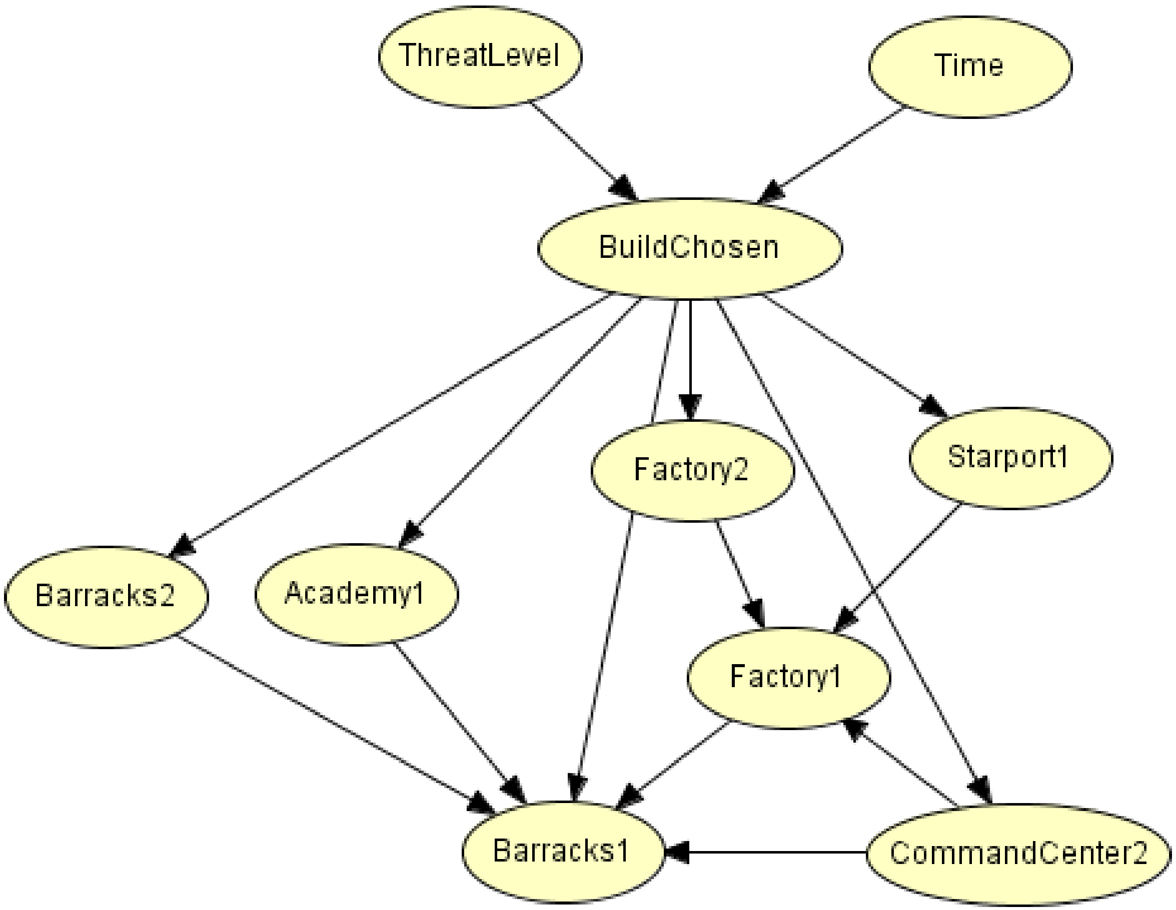
Baysian Network

- Spawn Prediction



Baysian Network

- Threat level prediction – tvt



Potential Fields

Potential Field In General

- Attractive behavior
-
-
- Repulsive behavior
-
-
- From vector to number

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

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

Our Potential Field Function

- Behavior is determined by + or –
- due vs de
- (2de - due)

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

Changed When Implementing Potential Field

- All forces are positive
- Forces are 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

Reinforcement Learning

Generalization 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

Maximum Distance Positioning

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

Ally Units

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

Edges and Cliffs

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

Squad

$$f_S \leftarrow f_S + \alpha[R(s) + \gamma(max(\hat{Q}_f(a', s'))) - \hat{Q}_f(a, s)](2ds - dsv)$$

Cooldown

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

Algorithm

- Image that i’m still doing

TEST

Building a test

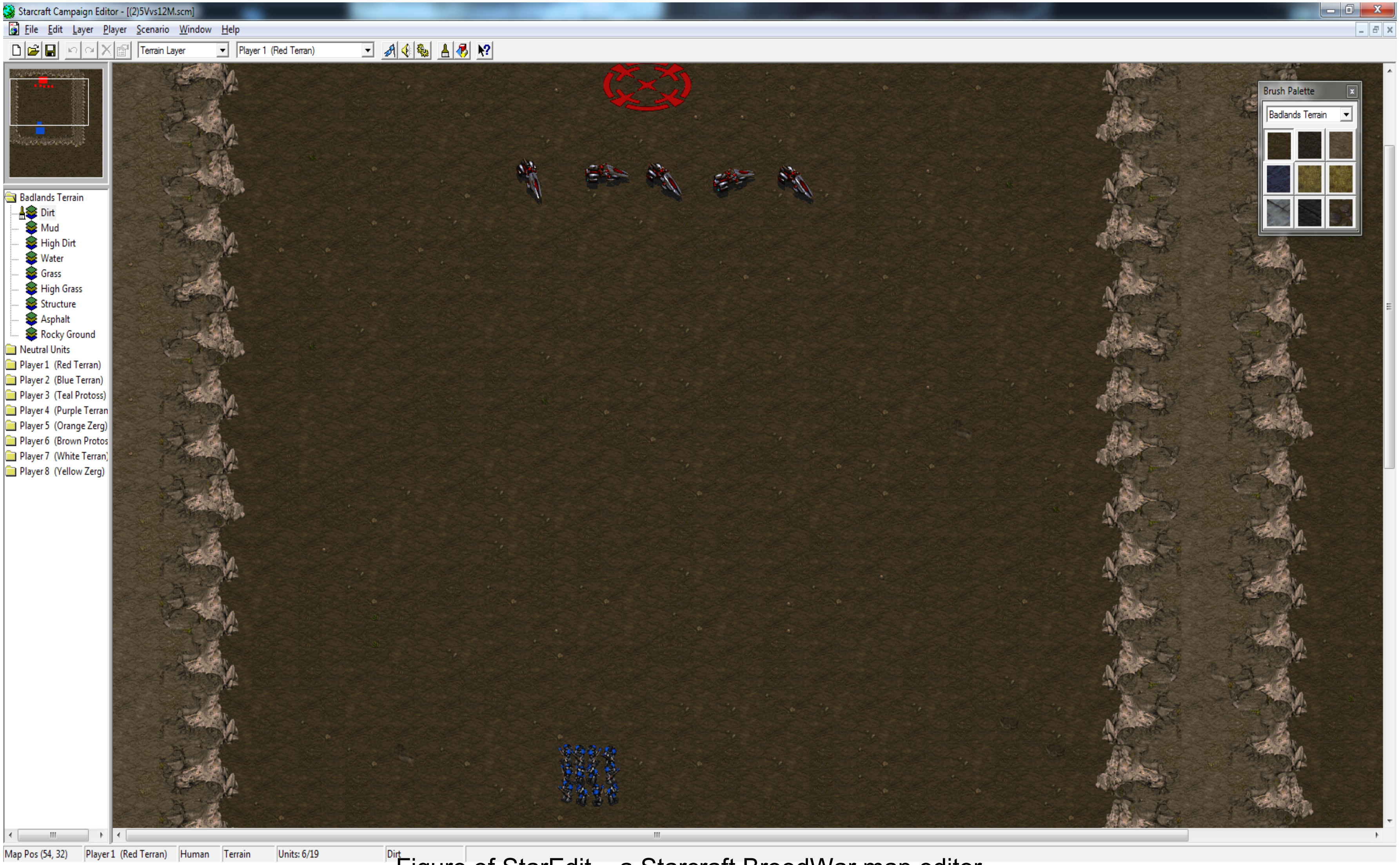


Figure of StarEdit – a Starcraft BroodWar map editor

Base case

- Testing without Reinforcement learning, and 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?

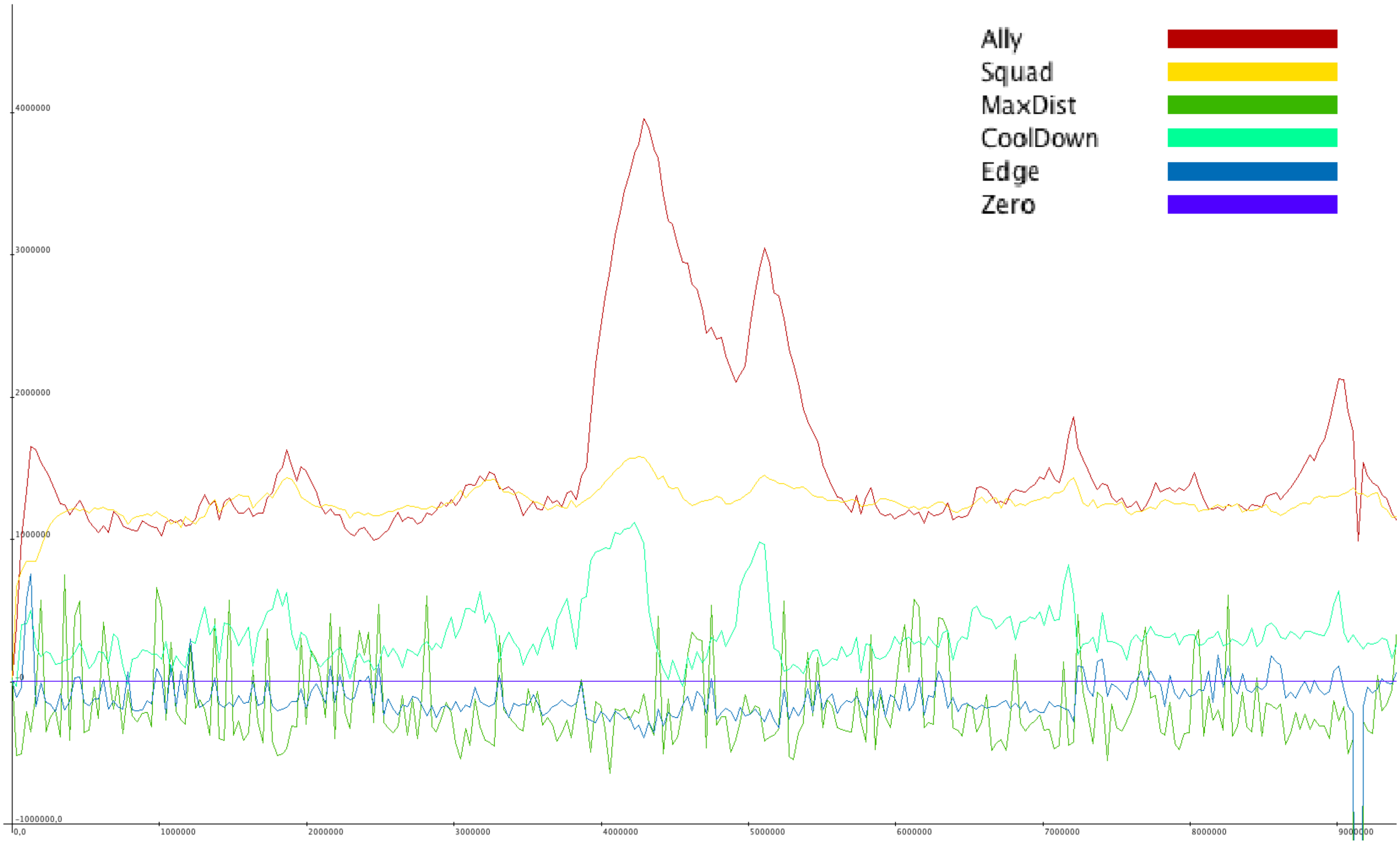
- Which Alpha and Gamma values?
- How many iterations is needed?

Will it converge?

- Running tests with different values

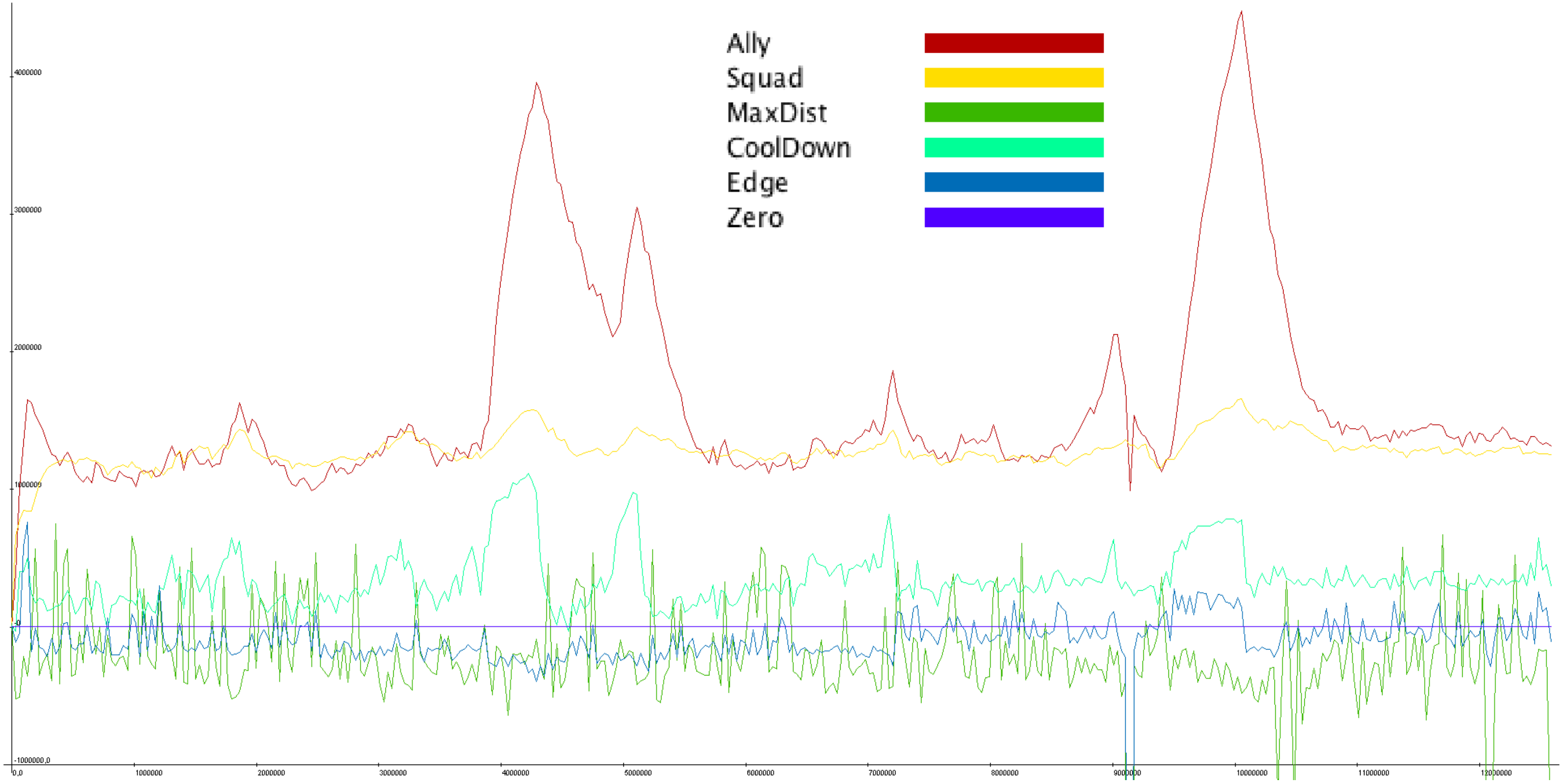
Will it converge?

- Values Alpha 0,2 and Gamma 0,9 (30852)



Will it converge?

- Values Alpha 0,4 and Gamma 0,6 (135936)



Predict spawn and build order

Conclusion