State Evaluation and Opponent Modelling in Real-Time Strategy Games

Graham Erickson

University of Alberta

September 23, 2014
1. Introduction

2. Build-Order Clustering

3. State Evaluation
Game AI

- Enabling computers to play games
AI for Video Games

- Why work on video games?
- Tools for balancing
- More interesting opponents
- Dynamic game elements
Real-Time Strategy

- Combat simulation games
- Manage resources, build units, engage in battles
- Simultaneous moves
- Real-time
- Imperfect information
StarCraft

- Blizzard Entertainment
- 1998
- Commercial and critical success
- Three factions
  - Protoss
  - Terran
  - Zerg
- Known to be well-balanced
Why StarCraft?
- Large online community
- Professional players
- Replays from various ladders freely available
- BWAPI (Brood War API)
  - C++
  - Can test against programs and humans
StarCraft AI Tournaments

- AI against AI
- Nowhere near human skill
- AIIDE
- CIG
- SSCAI
- UAlbertaBot won in 2013
Terminology

- **Macro**
  - Managing resources
  - Build-orders
  - High-level plans

- **Micro**
  - Controlling units in battles
  - Path finding
Why study RTS?

- Well-defined environments
- Can be broken into sub-tasks
- Areas in AI [2]
  - Adversarial planning under uncertainty
  - Learning and opponent modeling
  - Spatial and temporal reasoning
- Abstraction is necessary
Complexity of shooting game

- PSPACE-hard [3]
Introduction

Build-Order Clustering

State Evaluation

Layout

1. Introduction

2. Build-Order Clustering

3. State Evaluation
What is a Build-Order?

- Sequence
- Embody high-level strategy
- String of characters
Why Cluster Build-Orders?

- Hard coded rules
  - Expert knowledge
- Game balancing is an extensive process
  - Starcraft was patched continuously for 11 years
Why Cluster Build-Orders?

- novel strategies
- avoid expert knowledge
- empirical basis for responses
Why Cluster Build-Orders?
Why Cluster Build-Orders?
Why Cluster Build-Orders?

60% Win Rate
Jeff Long’s Master’s Thesis

- 100 WarCraft III replays [4]
- Hand-labeled build-orders
- Classification problem
- Sequence alignment
- Populate payoff matrices
Sequence Alignment

abba
ba
Sequence Alignment

abba

_b_a
Sequence Alignment

\[ \sum_{i=0}^{n} S(A_i, B_i) \]
Introduction
Build-Order Clustering
State Evaluation

Needleman-Wunsch Sequence Alignment

- Dynamic greedy algorithm
- $O(nm)$
- Score and alignment
Edit or Levenshtein Distance

\[ S(a, b) = \begin{cases} 
0 & \text{if } a = b \\
-1 & \text{if } a \neq b 
\end{cases} \]
Sequence Alignment for StarCraft

- Unit Similarity
- We introduce supply
- Strongly reward matches
- Additional penalties for mis-matches
Mis-Match Penalty II

- Ground
  - Ranged
    - Dragoon
  - Melee
    - Zealot
  - Spellcaster
    - Archon
    - High Templar

Graham Erickson  MSc Seminar
Now that we have a similarity metric we can cluster build-orders.

We want a technique that works using just a similarity matrix.
Hierarchical Clustering

Diagram showing the hierarchical clustering of the letters 'abcd'. The letters are clustered into groups, with 'a' and 'b' forming one group, and 'c' and 'd' forming another group. These two groups then combine to form the final group 'bcd'.
Agglomerative Hierarchical Clustering

Input: Similarity matrix $S$
Compute proximity matrix $P$ from $S$

\[\textbf{while} \ |P| > 1 \ \textbf{do}\]

- Merge clusters $i$ and $j$ where $P_{ij}$ is maximized
- Update $P$

\[\textbf{end while}\]
Proximity Measures

(a) MIN (single link.)

(b) MAX (complete link.)

(c) Group average.
Proximity Measures

\[ \text{Max}(C_1, C_2) = \max(\{S_{ij} | i \in C_1, j \in C_2\}) \]

\[ \text{Min}(C_1, C_2) = \min(\{S_{ij} | i \in C_1, j \in C_2\}) \]

\[ \text{Average}(C_1, C_2) = \frac{\sum_{i \in C_1} \sum_{j \in C_2} S_{ij}}{|C_1| \ast |C_2|} \]
Choosing a Proximity Measure

- CoPhenetic Correlation Coefficient
Data collected by Gabriel Synnaeve [5]
- Protoss versus Protoss (∼400 games)
- Protoss versus Terran (∼2000 games)
- Replays are taken from major amateur ladders
- Replays are already parsed
## CPCC values for PvT

<table>
<thead>
<tr>
<th>Linkage Policy</th>
<th>Protoss CPCC</th>
<th>Terran CPCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min</td>
<td>0.68256</td>
<td>0.77136</td>
</tr>
<tr>
<td>Max</td>
<td>0.18612</td>
<td>0.16551</td>
</tr>
<tr>
<td>Average</td>
<td>0.83518</td>
<td>0.85562</td>
</tr>
<tr>
<td>Ward</td>
<td>0.61552</td>
<td>0.54474</td>
</tr>
</tbody>
</table>
Cohesion and Separation

(a) Cohesion.

(b) Separation.
Choosing Partitional Clustering PvT Protoss
Protoss Clusters

- Cluster 1 (*rush*)
  - Short
  - Zealots and Dragoons
  - Probably rushes

- Cluster 2 (*drop*)
  - Small
  - Reaver drops

- Cluster 3 (*drag*)
  - Mid-length
  - Dragoons

- Cluster 4 (*big*)
  - Very large
  - Tough to see high-level coherence

<table>
<thead>
<tr>
<th></th>
<th>rush</th>
<th>tanks</th>
<th>drop</th>
<th>big</th>
</tr>
</thead>
<tbody>
<tr>
<td>rush</td>
<td>0.07 (15)</td>
<td>-0.09 (33)</td>
<td>0 (0)</td>
<td>-1.0 (1)</td>
</tr>
<tr>
<td>drop</td>
<td>0 (0)</td>
<td>1.0 (2)</td>
<td>-1.0 (1)</td>
<td>0.33 (3)</td>
</tr>
<tr>
<td>drag</td>
<td>0.5 (4)</td>
<td>0.30 (158)</td>
<td>0.11 (9)</td>
<td>-0.03 (203)</td>
</tr>
<tr>
<td>big</td>
<td>0 (0)</td>
<td>1.0 (4)</td>
<td>1.0 (1)</td>
<td>0.17 (1655)</td>
</tr>
</tbody>
</table>
Terran Clusters

- **Cluster 1 (rush)**
  - Short
  - Marines

- **Cluster 2 (tanks)**
  - Mid-length
  - Marines → Vultures and Tanks or just Tanks

- **Cluster 3 (drop)**
  - Varying lengths
  - Goliaths and Dropships

- **Cluster 4 (big)**
  - Very large
  - Tough to see high-level coherence

---

<table>
<thead>
<tr>
<th></th>
<th>rush</th>
<th>tanks</th>
<th>drop</th>
<th>big</th>
</tr>
</thead>
<tbody>
<tr>
<td>rush</td>
<td>0.07 (15)</td>
<td>-0.09 (33)</td>
<td>0 (0)</td>
<td>-1.0 (1)</td>
</tr>
<tr>
<td>drop</td>
<td>0 (0)</td>
<td>1.0 (2)</td>
<td>-1.0 (1)</td>
<td>0.33 (3)</td>
</tr>
<tr>
<td>drag</td>
<td>0.5 (4)</td>
<td>0.30 (158)</td>
<td>0.11 (9)</td>
<td>-0.03 (203)</td>
</tr>
<tr>
<td>big</td>
<td>0 (0)</td>
<td>1.0 (4)</td>
<td>1.0 (1)</td>
<td>0.17 (1655)</td>
</tr>
</tbody>
</table>

Graham Erickson | MSc Seminar
Conclusion

- Build-orders are just sequences of characters
- Sequence alignment for developing similarity metrics
- Agglomerative hierarchical clustering
- Clusters show some cognitive coherence
- Future work
  - Different clustering techniques
  - Experimenting with custom cost functions
  - Using payoff matrices to influence in-game decision making
1 Introduction

2 Build-Order Clustering

3 State Evaluation
Problem

- Given a state, predict the winner
- Perfect information
- Identify important features
- Estimate player skill
Motivation

- Search algorithms that require evaluation have had success in other games
- Part of a research initiative into hierarchical search systems [6]
- Used for pruning and rule-based decision making
Objectives

- Present a feature set
- Micro skill estimation metric
- Show effectiveness of technique across time-steps
Data

- Synnaeve data-set
- Protoss versus Protoss
- Parser developed
  - Some errors with destruction events
  - Control over battle detection
SCFeatureExtractor

- https://github.com/gkericks/SCFeatureExtractor
- C++
- BWAPI
- Computes feature vectors and writes them to file periodically
Battles

- Isolated skirmishes
- Micro game
Battle Extraction

- Identify battles
- Log unit info when the battle starts
  - Time
  - Health
  - Location
- Let units enter at different times
- Battles time out or end by rout
Battle Example I
Battle Example II
Battle Example III

Graham Erickson

MSc Seminar
Battle Example IV
Battle Example V
Battle Example VI
Battle Example VII
Battle Example IX
Battle Example XI
Battle Example XII
Battle Example XIII
Features

- Feature vectors are extracted every 10 seconds
- Feature values are in terms of differences
  - Player $A$ has $D_A$ Dragoons
  - Player $B$ has $D_B$ Dragoons
  - Feature is $D_A - D_B$
- Two feature vectors are added for each state
  - Symmetric match-up
Economic Features

- Average unspent resources
- Income
Military Features

- Unit and building counts
- Units that are ammo not included
- Research/Upgrades not included
Map Coverage Feature
Micro Skill Estimation

- Combat game
- Specific type of skill
- Skill estimate can be used as a feature
Baseline

- Work done in Poker [7]
- Play out same situation using a baseline player
- Compare agent and baseline outcome
SparCraft

- https://code.google.com/p/sparcraft/
Scripted Player

- No-OverKill Attack-Value (NOK-AV)
- Targets highest damage-per-frame to hit-point ratio
- Buildings are just obstacles
Battle Skill Metric

1. Battle Input
2. Simulator
3. Battle Results
4. Sim Results
5. Skill Metric
Macro Skill

- High-level decision making
- Number of frames that supply is maxed out for
- Number of idle production facilities ($PF$)
- Number of units queued ($Q$)
Learning

- Logistic Regression
- Learn feature weights
## Breakdown of Time Intervals

<table>
<thead>
<tr>
<th>Time (min)</th>
<th>Games</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-5</td>
<td>391</td>
<td>23418</td>
</tr>
<tr>
<td>5-10</td>
<td>386</td>
<td>22616</td>
</tr>
<tr>
<td>10-15</td>
<td>364</td>
<td>19836</td>
</tr>
<tr>
<td>15-20</td>
<td>289</td>
<td>14996</td>
</tr>
<tr>
<td>20-</td>
<td>211</td>
<td>31060</td>
</tr>
</tbody>
</table>
Feature Set Evaluation

Training and Testing on [k,l]

Accuracy (%) vs. Time Interval (minutes)

- Map Control
- Economic
- Military
- Macro Skill
- Micro Skill
- All Features
With Larger Training Sets

Training on \([k, \infty]\), Testing on \([k, l]\)

Accuracy (%)

Time Interval (minutes)

- Map Control
- Economic
- Military
- Macro Skill
- Micro Skill
- All Features
Problems with Micro Skill Estimator

- Low number of repeat players in data-set
- Late game units
- No external ranking
2013 AIIDE StarCraft AI Tournament

- 8 AI systems
- 10 maps
- Each bot plays each other bot 20 times on each map
- Ranked by win percentage
## Ranking from AIIDE 2013 StarCraft Competition

<table>
<thead>
<tr>
<th>Team</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>UAlbertaBot</td>
<td>82.43%</td>
</tr>
<tr>
<td>Skynet</td>
<td>72.77%</td>
</tr>
<tr>
<td>Aiur</td>
<td>60.29%</td>
</tr>
<tr>
<td>Ximp</td>
<td>55.29%</td>
</tr>
<tr>
<td>Xelnaga</td>
<td>49.96%</td>
</tr>
<tr>
<td>ICEStarCraft</td>
<td>47.82%</td>
</tr>
<tr>
<td>Nova</td>
<td>27.47%</td>
</tr>
<tr>
<td>BTHAI</td>
<td>3.93%</td>
</tr>
</tbody>
</table>
### Ranking using Micro Skill Averaged

<table>
<thead>
<tr>
<th>Team</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nova</td>
<td>7.65</td>
</tr>
<tr>
<td>UAlbertaBot</td>
<td>3.30</td>
</tr>
<tr>
<td>Aiur</td>
<td>1.01</td>
</tr>
<tr>
<td>ICEStarCraft</td>
<td>-0.026</td>
</tr>
<tr>
<td>Ximp</td>
<td>-1.91</td>
</tr>
<tr>
<td>BTHAI</td>
<td>-3.03</td>
</tr>
<tr>
<td>Skynet</td>
<td>-4.61</td>
</tr>
<tr>
<td>Xelnaga</td>
<td>-5.60</td>
</tr>
</tbody>
</table>
## Ranking using Micro Skill with Variance Control

<table>
<thead>
<tr>
<th>Team</th>
<th>Micro Skill</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nova</td>
<td>7.59</td>
</tr>
<tr>
<td>UAlbertaBot</td>
<td>1.97</td>
</tr>
<tr>
<td>Aiur</td>
<td>0.85</td>
</tr>
<tr>
<td>ICEStarCraft</td>
<td>0.01</td>
</tr>
<tr>
<td>Ximp</td>
<td>-1.79</td>
</tr>
<tr>
<td>Xelnaga</td>
<td>-2.99</td>
</tr>
<tr>
<td>BTHAI</td>
<td>-3.13</td>
</tr>
<tr>
<td>Skynet</td>
<td>-4.51</td>
</tr>
</tbody>
</table>
Conclusion

- Predict game outcome
- Noisy problem
- Feature set has > 70% accuracy in the later stages of a match
- Average unspent resources
- Income
- Map control
- Skill estimation through simulation
How hard is RTS?

- Huge state space
  - Average map size is 128*128 tiles
  - Approx. 50 unit types
  - Approx. 200 units
  - $10^{1685}$ possible unit locations
  - Ignoring Health, Attacks, Resources etc.
Needleman-Wunsch sequence alignment algorithm [8]

Sequences $A$ and $B$ can have gaps inserted to make aligned sequences $A'$ and $B'$

Take $S(a, b)$ to be the similarity between two characters $a$ and $b$

Take $S(−, a)$ to be the gap penalty for some character $a$

Maximize alignment score:

$$
\sum_{i=0}^{n} S(A'_i, B'_i)
$$
Example

\[ S(a, b) = \begin{cases} 
0 & \text{if } a = b \\
-1 & \text{if } a \neq b 
\end{cases} \]

\[ \text{abba} \]

\[ \text{ba} \]

\[ S(a, b) \] is the Levenshtein or edit distance [9]
Needleman-Wunsch Sequence Alignment

- Dynamic program
- Populates a matrix \( M \)
- \( M_{ij} \) is the score of an optimal alignment between the first \( i \) characters in \( A \) and the first \( j \) characters in \( B \)

<table>
<thead>
<tr>
<th></th>
<th>-</th>
<th>A</th>
<th>T</th>
<th>C</th>
<th>G</th>
<th>A</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>0</td>
<td>-4</td>
<td>-8</td>
<td>-12</td>
<td>-16</td>
<td>-20</td>
<td>-24</td>
</tr>
<tr>
<td>C</td>
<td>-4</td>
<td>-3</td>
<td>-7</td>
<td>-3</td>
<td>-7</td>
<td>-11</td>
<td>-15</td>
</tr>
<tr>
<td>A</td>
<td>-8</td>
<td>1</td>
<td>-3</td>
<td>-7</td>
<td>-6</td>
<td>-2</td>
<td>-6</td>
</tr>
<tr>
<td>T</td>
<td>-12</td>
<td>-3</td>
<td>6</td>
<td>2</td>
<td>-2</td>
<td>-6</td>
<td>-5</td>
</tr>
<tr>
<td>A</td>
<td>-16</td>
<td>-7</td>
<td>2</td>
<td>3</td>
<td>-1</td>
<td>3</td>
<td>-1</td>
</tr>
<tr>
<td>C</td>
<td>-20</td>
<td>-11</td>
<td>-2</td>
<td>-1</td>
<td>0</td>
<td>-1</td>
<td>8</td>
</tr>
</tbody>
</table>
Needleman-Wunsch Initialize M

\[ T = 0 \]
\[ \text{for } i \in [1...n] \text{ do} \]
\[ M_{i0} = T + S(\_, A_i) \]
\[ T = M_{i0} \]
\[ \text{end for} \]
\[ T = 0 \]
\[ \text{for } j \in [1...m] \text{ do} \]
\[ M_{0j} = T + S(\_, B_j) \]
\[ T = M_{0j} \]
\[ \text{end for} \]
\[ M_{00} = 0 \]
for \( i \in [1...n] \) do
  for \( j \in [1...m] \) do
    match = \( M_{i-1,j-1} + S(A_i, B_j) \)
    gapA = \( M_{i-1,j} + S(-, A_i) \)
    gapB = \( M_{i,j-1} + S(-, B_j) \)
    \( M_{i,j} = \max(\text{match}, \text{gapA, gapB}) \)
  end for
end for
Goal is to cluster sequences

In [10] sequence alignment is used to define sequence similarity

\[
\text{dis}(A, B) = M_{0,0}
\]

\[
S'(a, b) = \begin{cases} 
S(a, b) & \text{if } a = b \\
0 & \text{if } a \neq b
\end{cases}
\]

\[
\text{dis}_{\text{correct}}(A, B) = \sum_{i=0}^{n} S'(A_i, B_i)
\]

\[
\text{Sim}_{\text{align}}(A, B) = \frac{\text{dis}(A, B)}{\text{dis}_{\text{correct}}(A, B)}
\]
Similarity Metric

\[ \text{cor}(a, b) = \begin{cases} 
1 & \text{if } a = b \\
0 & \text{if } a \neq b 
\end{cases} \]

\[ \text{Num\_Correct}(A, B) = \sum_{i=0}^{n} \text{cor}(A_i, B_i) \]

\[ \text{Sim}_{\text{Significance}}(A, B) = \frac{\text{Num\_Correct}(A, B)}{n} \]

\[ \text{Sim}(A, B) = \text{Sim}_{\text{align}}(A, B) \times \text{Sim}_{\text{Significance}}(A, B) \]
General Cost Function

$$\rho(a, b) = \begin{cases} 
k \times c(a) & \text{if } a == b \\
|c(a) - c(b)| - \phi(a, b) & \text{otherwise}
\end{cases}$$

- $c$ is some attribute of a unit type
- $k$ is a constant
- $\phi$ is a mis-match penalty.
Recall that Needleman-Wunsch uses a character similarity function $S(a, b)$.

Also called a cost function.

More interesting to design a domain specific cost function.

Let $\rho$ be a custom cost function of character $a$ and $b$. 
General Cost Function

\[
\rho(a, b) = \begin{cases} 
  k \ast c(a) & \text{if } a == b \\
  |c(a) - c(b)| - \phi(a, b) & \text{otherwise}
\end{cases}
\]

- \(c\) is some attribute of a unit type
- \(k\) is a constant
- \(\phi\) is a mis-match penalty.
For $c$ we introduce supply
Supply is for limiting unit counts
Jeff Long used $k = 16$
Strongly reward matches
Defining $\phi(a, b)$

Larger when $a$ and $b$ are more different

We propose a unit *Ontology*

- Hierarchical Categorization
- Penalties are assigned depending on at what level the units differ
- More essential differences have higher penalties
Cluster Evaluation

What clusterings should we choose?
Choosing a Cluster Proximity Measure

- Recall: Agglomerative Hierarchical clustering requires a proximity measure
- How to choose the best one for the data?


Similarity matrix $S$ and proximity matrix $P$

During clustering there will be an iteration where two elements $x$ and $y$ are first members of the same cluster.

The proximity of the two clusters at that iteration is the CoPhenetic distance for $x$ and $y$.

Populate a matrix $P'$ of CoPhenetic distances.

CPCC is correlation between $P'$ and $S$. 

CoPhenetic Correlation Coefficient
### CPCC values for PvP data

<table>
<thead>
<tr>
<th>Linkage Policy</th>
<th>CPCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min</td>
<td>0.62337</td>
</tr>
<tr>
<td>Max</td>
<td>0.21094</td>
</tr>
<tr>
<td>Average</td>
<td>0.76905</td>
</tr>
<tr>
<td>Ward</td>
<td>0.56441</td>
</tr>
</tbody>
</table>
## CPCC values for PvT

<table>
<thead>
<tr>
<th>Linkage Policy</th>
<th>Protoss CPCC</th>
<th>Terran CPCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min</td>
<td>0.68256</td>
<td>0.77136</td>
</tr>
<tr>
<td>Max</td>
<td>0.18612</td>
<td>0.16551</td>
</tr>
<tr>
<td>Average</td>
<td>0.83518</td>
<td>0.85562</td>
</tr>
<tr>
<td>Ward</td>
<td>0.61552</td>
<td>0.54474</td>
</tr>
</tbody>
</table>
Choosing a Partitional Clustering

- Have a metric for a partitional clustering [11]
- Chose the clustering that optimizes the metric
Cohesion

\[ \text{Cohesion}(C) = \sum_{i \in C} \sum_{j \in C} S_{i,j} \]
\[ \text{Sep}(C, C') = \sum_{i \in C} \sum_{j \in C'} S_{i,j} \]

\[ \text{Separation}(C) = \sum_{C' \in \kappa \text{ and } C' \neq C} \text{Sep}(C, C') \]
Combining Cohesion and Separation

\[ \text{Sep}_{\text{and}} \text{Co}(\kappa) = \sum_{C \in \kappa} \frac{\text{Separation}(C)}{\text{Cohesion}(C)} \]
Sep\_and\_Co versus the number of clusters for the hierarchical clustering of the PvP dataset
Choosing Partitional Clustering PvP II

**Sep and Co** versus the number of clusters for the hierarchical clustering of the PvP dataset normalized by number of clusters
Choosing Partitional Clustering PvP III

Sep_and_Co versus the number of clusters for the hierarchical clustering of the PvP dataset normalized by number of clusters on the domain of [2,100]
Choosing Partitional Clustering PvT Protoss

Sep_and_Co versus the number of clusters for the hierarchical clustering of the PvT dataset normalized by number of clusters on the domain of [2,100] just using Protoss players.
Choosing Partitional Clustering PvP Terran

$Sep\_and\_Co$ versus the number of clusters for the hierarchical clustering of the PvP dataset normalized by number of clusters on the domain of $[2,100]$ just using Terran players.
Building Payoff Matrices

- Quantify how build-orders in each cluster perform against each other
- Game balance
- Strategy response
- Populated via replay data
Building Payoff Matrices II

\[ G_{ij} = \frac{w_{ij} - l_{ij}}{t_{ij}} \]

- \( i \) is the row player
- \( j \) is the column player
- \( w_{ij} \) is the number of wins for cluster \( i \) against cluster \( j \)
- \( l_{ij} \) is the number of losses for cluster \( i \)
- \( t_{ij} = w_{ij} + l_{ij} \)
### Payoff Matrix from PvP Data

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0 (0)</td>
<td>-1.0 (1)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>2</td>
<td>1.0 (1)</td>
<td>0.0 (640)</td>
<td>0.2 (25)</td>
</tr>
<tr>
<td>3</td>
<td>0 (0)</td>
<td>-0.2 (25)</td>
<td>0 (60)</td>
</tr>
</tbody>
</table>
Problems

- Diagonal is uninteresting
  - PvP is a symmetric match-up
- Diagonal has most of the examples
- Cluster 1 is very small
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0 (2)</td>
<td>0 (0)</td>
<td>1.0 (2)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>2</td>
<td>1.0 (1)</td>
<td>-1.0 (1)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>3</td>
<td>-1.0 (1)</td>
<td>0 (0)</td>
<td>-0.09 (33)</td>
<td>0.07 (15)</td>
</tr>
<tr>
<td>4</td>
<td>0.15 (1858)</td>
<td>0.2 (10)</td>
<td>0.32 (60)</td>
<td>0.5 (4)</td>
</tr>
</tbody>
</table>

There are still small clusters! These might not represent a coherent strategy.
## Payoff Matrix from PvT 1

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0 (2)</td>
<td>0 (0)</td>
<td>1.0 (2)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>2</td>
<td>1.0 (1)</td>
<td>-1.0 (1)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>3</td>
<td>-1.0 (1)</td>
<td>0 (0)</td>
<td>-0.09 (33)</td>
<td>0.07 (15)</td>
</tr>
<tr>
<td>4</td>
<td>0.15 (1858)</td>
<td>0.2 (10)</td>
<td>0.32 (60)</td>
<td>0.5 (4)</td>
</tr>
</tbody>
</table>

- There are still small clusters!
- These might not represent a coherent strategy
Let $C$ be the topmost cluster of the hierarchical clustering
Let $T$ be a threshold size
Let $P$ be kept clusters
Split $C$ into two clusters
Larger cluster is added to $P$
If smaller cluster is $\geq T$ it is added to $P$, discarded otherwise
Synnaeve’s Extraction Algorithm

- Similar
- No buildings
- Starts with unit destruction events
  - Ours start with attacks
- Only start and end timestamps
  - Ours has timestamps for when units enter the battle
Some replays contained strange activity
  - Giving up when clearly ahead
  - AFK

Winner not always clearly marked
Determining the Winner

- Flags
  - `isWinner`
  - `playerLeft` (with time-stamp)
- If `isWinner` is present, use that
- Otherwise `playerLeft` and game score are used
Determining the Winner

- If no `playerLeft` flag
  - game score is used, if scores are close ($\Delta < T$) replay is discarded

- If one `playerLeft` flag
  - Opposite player is winner
  - Unless that conflicts with game score (type A)

- Two `playerLeft` flags
  - Player who left second is chosen
  - Unless that conflicts with game score (type B)
## Breakdown of Discarded Replays

<table>
<thead>
<tr>
<th>Games</th>
<th>Number of Games</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>447</td>
</tr>
<tr>
<td>Kept</td>
<td>391</td>
</tr>
<tr>
<td>No Status Close Score</td>
<td>30</td>
</tr>
<tr>
<td>Conflict Type A</td>
<td>24</td>
</tr>
<tr>
<td>Conflict Type B</td>
<td>1</td>
</tr>
<tr>
<td>Corrupt</td>
<td>1</td>
</tr>
</tbody>
</table>
Economic Features

- Let $R_{cur}$ be current unspent resources
- Let $R_{tot}$ be current unspent resources
- Let $T$ be current frame number
- Average unspent resources:
  \[ U = \frac{\sum_{t \leq T} R_{cur}}{T} \]

- Income:
  \[ I = \frac{R_{tot}}{T} \]
Map Coverage Feature

- Map divided into grid (2-by-2 build tiles)
- Ratio of occupied to total tiles
- Units to walkable space

\[
MC(p) = \sum_{pos \in P} f(pos, p)
\]

\[
f(pos, p) = \begin{cases} 
1 & \text{if } pos \text{ is occupied by } p \\
0 & \text{otherwise}
\end{cases}
\]
• Life-time damage:

\[ \text{LTD2}_{\text{start}}(U) = \sum_{u \in U} \sqrt{\text{HP}(u)} \cdot \text{DMG}(u) \]

• \( U \) is the set of units for a player
• Favours having multiple units to single units given equal summed health
• Rewards keeping units alive that can deal greater damage quicker.
- Units can enter battles at varying times
- Let $T$ be the length of the battle
- Let $st(u)$ be the time unit $u$ entered the battle

$$LTD2_{end}(U) = \sum_{u \in U} \frac{T - st(u)}{T} \cdot \sqrt{HP(u) \cdot DMG(u)}$$
• $P_s$ and $O_s$ are unit sets for player and opponent at the start of the battle

• $P_{out}$ and $O_{out}$ are unit sets for player and opponent at the end of the battle

\[ V^P = (\text{LTD2}_{\text{end}}(P_{out}) - \text{LTD2}_{\text{end}}(O_{out})) - (\text{LTD2}_{\text{start}}(P_s) - \text{LTD2}_{\text{start}}(O_s)) \]

• $P_\beta$ and $O_\beta$ are the unit sets from the baseline player

\[ V^\beta = (\text{LTD2}_{\text{end}}(P_\beta) - \text{LTD2}_{\text{end}}(O_\beta)) - (\text{LTD2}_{\text{start}}(P_s) - \text{LTD2}_{\text{start}}(O_s)) \]
\[ \beta_{tot} = \sum_{i=1}^{n} (V_i^p - V_i^\beta) \]

\[ \beta_{avg} = \frac{\beta_{tot}}{n} \]
Battle Skill Metric

\[ \beta_{var} = \frac{1}{n} \sum_{i=1}^{n} (V_i^p - \frac{\widehat{\text{Cov}}[V_i^p, V_i^\beta]}{\widehat{\text{Var}}[V_i^p]} \cdot V_i^\beta) \]
Macro Skill

- High-level decision making
- Number of frames that supply is maxed out for

\[ SF = \sum_{t \leq T} f(t) \]

\[ f(t) = \begin{cases} 
1 & \text{if } S_{cur} = S_{max} \text{ at time } t \\
0 & \text{otherwise} 
\end{cases} \]

- Number of idle production facilities \((PF)\)
- Number of units queued \((Q)\)
Logistic Regression

Matrix $X$ with $n$ examples (rows) and $k$ features (columns)

Corresponding response vector $Y$

Gives $k$ weights $K$ such that

$$X \cdot K = Y'$$

$$T(g(Y')) \approx Y$$

$$g(s) = \frac{1}{1 + e^{-s}}$$

$T$ is a threshold function
10-fold cross validation by games

- Reporting accuracy
  - Proportion of correct predictions
  - Responses threshold at 0.5

- Average Log-Likelihood

\[ L(y, r) = y \cdot \log(r) + (1 - y) \cdot \log(1 - r) \]
### Feature Set Evaluation

<table>
<thead>
<tr>
<th>Features</th>
<th>0-5</th>
<th>5-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{cur}, I, U$</td>
<td>54.42 (-0.686)</td>
<td>57.76 (-0.672)</td>
</tr>
<tr>
<td>$UC$</td>
<td>51.96 (-0.712)</td>
<td>57.84 (-0.682)</td>
</tr>
<tr>
<td>$MC$</td>
<td>51.27 (-0.693)</td>
<td>55.20 (-0.685)</td>
</tr>
<tr>
<td>$\beta_{var}$</td>
<td>50.23 (-0.693)</td>
<td>53.25 (-0.690)</td>
</tr>
<tr>
<td>$SF, PF, Q$</td>
<td>51.26 (-0.695)</td>
<td>49.96 (-0.695)</td>
</tr>
<tr>
<td>$A$</td>
<td>53.91 (-0.708)</td>
<td>58.81 (-0.680)</td>
</tr>
<tr>
<td>$B$</td>
<td>54.05 (-0.708)</td>
<td>58.66 (-0.681)</td>
</tr>
<tr>
<td>$C$</td>
<td>53.81 (-0.710)</td>
<td>58.72 (-0.681)</td>
</tr>
</tbody>
</table>

- A = economic/military features $R_{cur}, I, U, UC$
- B = A + map control feature $MC$
- C = B + skill features $\beta_{var}, SF, PF, Q$
## Feature Set Evaluation

<table>
<thead>
<tr>
<th>Features</th>
<th>10-15</th>
<th>15-</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{cur}, I, U$</td>
<td>62.98 (-0.647)</td>
<td>64.17 (-0.625)</td>
</tr>
<tr>
<td>$UC$</td>
<td>66.67 (-0.705)</td>
<td>66.46 (-0.644)</td>
</tr>
<tr>
<td>$MC$</td>
<td>61.45 (-0.657)</td>
<td>71.39 (-0.561)</td>
</tr>
<tr>
<td>$\beta_{var}$</td>
<td>55.09 (-0.690)</td>
<td>52.82 (-0.690)</td>
</tr>
<tr>
<td>$SF, PF, Q$</td>
<td>51.75 (-0.694)</td>
<td>54.97 (-0.709)</td>
</tr>
<tr>
<td>A</td>
<td>66.36 (-0.712)</td>
<td>69.22 (-0.613)</td>
</tr>
<tr>
<td>B</td>
<td>66.44 (-0.712)</td>
<td>69.87 (-0.608)</td>
</tr>
<tr>
<td>C</td>
<td>66.41 (-0.708)</td>
<td>72.59 (-0.587)</td>
</tr>
</tbody>
</table>

- **A** = economic/military features $R_{cur}, I, U, UC$
- **B** = A + map control feature $MC$
- **C** = B + skill features $\beta_{var}, SF, PF, Q$
With Larger Training Sets

<table>
<thead>
<tr>
<th>Feature Set</th>
<th>0-5</th>
<th>5-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{cur}, I, U$</td>
<td>53.75 (-0.6875)</td>
<td>58.85 (-0.6708)</td>
</tr>
<tr>
<td>$UC$</td>
<td>52.03 (-0.6936)</td>
<td>58.43 (-0.6735)</td>
</tr>
<tr>
<td>$MC$</td>
<td>51.27 (-0.6943)</td>
<td>55.20 (-0.6872)</td>
</tr>
<tr>
<td>$\beta_{var}$</td>
<td>50.23 (-0.6931)</td>
<td>53.25 (-0.6896)</td>
</tr>
<tr>
<td>$SF, PF, Q$</td>
<td>52.02 (-0.6925)</td>
<td>50.74 (-0.6939)</td>
</tr>
<tr>
<td>A</td>
<td>53.19 (-0.6917)</td>
<td>58.74 (-0.6726)</td>
</tr>
<tr>
<td>B</td>
<td>52.60 (-0.6916)</td>
<td>58.56 (-0.6727)</td>
</tr>
<tr>
<td>C</td>
<td>52.73 (-0.6914)</td>
<td>58.70 (-0.6669)</td>
</tr>
</tbody>
</table>

- If time interval is $[k, l]$ training is done on examples in $[k, \infty)$ and tested on examples in $[k, l]$
With Larger Training Sets

<table>
<thead>
<tr>
<th>Feature Set</th>
<th>10-15</th>
<th>15-20</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{cur}, I, U$</td>
<td>$62.82 (-0.6510)$</td>
<td>$60.23 (-0.6562)$</td>
</tr>
<tr>
<td>$UC$</td>
<td>$65.76 (-0.6329)$</td>
<td>$63.96 (-0.6516)$</td>
</tr>
<tr>
<td>$MC$</td>
<td>$61.45 (-0.6588)$</td>
<td>$64.02 (-0.6385)$</td>
</tr>
<tr>
<td>$\beta_{var}$</td>
<td>$55.24 (-0.6899)$</td>
<td>$56.14 (-0.6868)$</td>
</tr>
<tr>
<td>$SF, PF, Q$</td>
<td>$52.82 (-0.6916)$</td>
<td>$55.21 (-0.6857)$</td>
</tr>
<tr>
<td>$A$</td>
<td>$65.28 (-0.6367)$</td>
<td>$63.58 (-0.6612)$</td>
</tr>
<tr>
<td>$B$</td>
<td>$64.89 (-0.6377)$</td>
<td>$63.99 (-0.6617)$</td>
</tr>
<tr>
<td>$C$</td>
<td>$65.77 (-0.6267)$</td>
<td>$65.23 (-0.6510)$</td>
</tr>
</tbody>
</table>

If time interval is $[k, l]$ training is done on examples in $[k, \infty)$ and tested on examples in $[k, l]$.
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.07 (15)</td>
<td>-0.09 (33)</td>
<td>0 (0)</td>
<td>-1.0 (1)</td>
</tr>
<tr>
<td>2</td>
<td>0 (0)</td>
<td>1.0 (2)</td>
<td>-1.0 (1)</td>
<td>0.33 (3)</td>
</tr>
<tr>
<td>3</td>
<td>0.5 (4)</td>
<td>0.30 (158)</td>
<td>0.11 (9)</td>
<td>-0.03 (203)</td>
</tr>
<tr>
<td>4</td>
<td>0 (0)</td>
<td>1.0 (4)</td>
<td>1.0 (1)</td>
<td>0.17 (1655)</td>
</tr>
</tbody>
</table>
Protoss Clusters

- Cluster 1
  - Short (in length) build-orders
  - Zealots and Dragoons
  - Probably rushes

- Cluster 2
  - Small
  - Scouts, Shuttles, Reavers, and Carriers
  - Reaver drops

- Cluster 3
  - Mid-length
  - Dragoons

- Cluster 4
  - Very Large
  - Tough to see high-level coherence
Terran Clusters

- Cluster 1
  - Short
  - Mostly just Marines
- Cluster 2
  - Mid-length
  - Start with Marines
  - Move to Vultures and Siege Tanks or just Siege Tanks
- Cluster 3
  - Varying lengths
  - Goliaths and Dropships
- Cluster 4
  - Long
  - Very large cluster
  - Tough to see high-level coherence


