EVOLVING REACTIVE MICROMANAGEMENT CONTROLLER 
FOR REAL-TIME STRATEGY GAMES

Ing. Martin Čertický, prof. Ing. Peter Sinčák CSc

Keywords: Genetic Algorithm, Evolution, Real-Time Strategy Games, StarCraft

1. Creating the Reactive Micro-Controller

Parts of the controller using parameters trained by Genetic Algorithms (GA) are highlighted blue:

1. Selecting an attack target
2. Selecting an attack position
3. Deciding when to retreat

These decisions are using simple functions to score the current game state. Parameters of these functions are optimised using our GA.

1. Attack target scoring function:
   \[ \text{Score}_{\text{AT}} = (D_e \cdot p_1) - (HP_e \cdot p_2) + (L_e \cdot p_3) \]

2. Changing the Controller's parameters

The Controller issues specific commands to each unit, trying to optimise the army performance (micro-management).

We implemented and tested the solution in a classic RTS game *StarCraft: Brood War*, which was accessed using BWMirror and BWAPI.

3. Genetic Algorithm

Genotype:

\[ P_0 \quad P_1 \quad P_2 \quad P_3 \quad P_4 \quad P_5 \quad P_7 \]

Fitness (calculated at the end of the game):

\[ \text{Score} = \sum_{i=1}^{n} HP_i - \sum_{j=1}^{m} HP_j \]

Roulette Wheel Selection (with slight elitism), Uniform Cross-over and Uniform Mutation (10% chance) were used in our GA (population of 32 individuals).

Results:

3 scenarios with different types of enemy units were chosen for the training.

Our results were compared to the built-in AI in *StarCraft* and UI Alberta both.

Interesting behavioral patterns emerged for each used scenario. Link to video:

<table>
<thead>
<tr>
<th>Scenario no.</th>
<th>Avg. Score</th>
<th>Best Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>Controller</td>
<td>124.42</td>
</tr>
<tr>
<td></td>
<td>Built-in AI of SC</td>
<td>-129.9</td>
</tr>
<tr>
<td></td>
<td>UI Alberta</td>
<td>-184.1</td>
</tr>
</tbody>
</table>