Multi-Reactive Planning for Real-Time Strategy Games

Alberto Uriarte Pérez

UAB

September 8, 2011

Advisor: Santiago Ontañón Villar
1 Motivation

2 Real-time multi agent system
   • Architecture overview
   • Working Memory Information
   • Micro management
   • Macro management

3 Experimental evaluation

4 Conclusions and Future Work
   • Conclusions
   • Future Work
Motivation

AI research has been focused on turn-based games like Chess. However, RTS games offer a more complex scenario.

- How to build a multi agent system capable of reasoning and cooperating in a real-time environment
- Concurrent and adversarial planning under uncertainty
- Spatial and temporal reasoning

Claim

Performing an immersion in the knowledge of the domain and implementing some of the latest AI techniques, we can improve the built-in AI of the game, beat other bots and be a real challenge for a human expert player
Micro management

[Image of a Starcraft game screenshot showing units and a menu]

Starcraft

Motivation
Real-time multi agent system
Experimental evaluation
Conclusions and Future Work

Architecture overview
Working Memory Information
Micro management
Macro management

Alberto Uriarte
Multi-Reactive Planning for RTS Games
Macro management
Architecture overview

- **NovaAIModule**
- **EnhancedUI**
- **Micro Agents**
  - Squad Manager
  - Squad Agent
  - Combat Agent
- **Macro Agents**
  - Build Manager
  - Worker Manager
  - Production Manager
  - Planner Manager
  - Strategy Manager
- **Information**
  - Information Manager
**Problem**
Real-time communication between agents.

**Solution**
Blackboard Architecture.
Working Memory.

![Architecture Diagram](image)
Working Memory Information

- NovaAIModule
- EnhancedUI
- Micro Agents
  - Squad Manager
  - Squad Agent
  - Combat Agent
- Macro Agents
  - Build Manager
  - Worker Manager
  - Production Manager
  - Planner Manager
  - Strategy Manager

Information Manager
Problem
Different maps with different tactical advantages.

Solution
Off-line terrain analysis
Opponent modelling

Problem
Imperfect information.

Solution
Opponent build order prediction (scouting).
Opponent’s military units tracking (threat map).

<table>
<thead>
<tr>
<th>Gathering task</th>
<th>Usefulness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial scout with a unit</td>
<td>Enemy start location</td>
</tr>
<tr>
<td></td>
<td>Detect rush strategy</td>
</tr>
<tr>
<td>Scanner sweep scanning</td>
<td>Detect target locations</td>
</tr>
<tr>
<td>Enemy air/ground DPS</td>
<td>Decide to build anti-air units</td>
</tr>
<tr>
<td>Threat map</td>
<td>Pathfinding to save location</td>
</tr>
<tr>
<td></td>
<td>Avoid dangerous regions</td>
</tr>
</tbody>
</table>
Threat map
Micro management agents

- NovaAIModule
- EnhancedUI

Micro Agents
- Squad Manager
- Squad Agent
- Combat Agent

Information
- Information Manager

Macro Agents
- Build Manager
- Worker Manager
- Production Manager
- Planner Manager
- Strategy Manager
Micro management agents

SquadManager, SquadAgent and CombatAgent follow a military organization.
Squad Agent

Problem
Effective squad movement.

Solution
Steering behaviours.

Problem:
- Effective squad movement.

Solution:
- Steering behaviours.

Diagram:
- Illustration of steering behaviours in a squad context.
Combat Agent

Problem
Target selection.

Solution
Assigning a score to each target.
Combat Agent

Problem

Range units avoiding damage from melee units.

Solution

Potential fields.
Macro management agents

- NovaAI Module
- Enhanced UI

Micro Agents:
- Squad Manager
- Squad Agent
- Combat Agent
- Information Manager

Macro Agents:
- Build Manager
- Worker Manager
- Production Manager
- Planner Manager
- Strategy Manager
Gathering resources

Problem
Workers’ tasks and income rate.

Solution
Finite State Machine.

2 workers \times \text{mineral field}
Building

Problem
Finding a build location.

Solution
Build map information with spiral-search algorithm.
**Strategies**

**Problem**
Planning strategies and reactive behaviour.

**Solution**
FSM with common states and trigger conditions.

[Diagram of State Machine]
Nova Vs Built-in AI

Results after 250 games against each race

![Nova games categorized by built-in AI's race](chart.png)
Nova Vs Bots

We tested our Nova bot in AIIDE Starcraft AI Competition

![Graph showing AIIDE competition standings]

Bot's name:
- Skynet
- UAlbertaBot
- Aiur
- ItayUndermind
- EISBot
- SPAR
- Undermind
- Nova
- BroodwarBotQ
- BTHAI
- Cromulent
- bigbrother
- Quorum

Win rate (%)
Nova Vs Bots

Nova games against bots

<table>
<thead>
<tr>
<th>Bot's name</th>
<th>Crash</th>
<th>Loss + Crash</th>
<th>Win</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undermind</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Skynet</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quorum</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPAR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cromulent</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>itayUndermind</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ETHAI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bigbrother</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alur</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESBot</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BrodwarBotQ</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UAlbertaBot</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Nova Vs Bots

![Bar chart showing Nova win rate categorized by bot's race.]

- Nova win rate categorized by bot's race
  - Opponent's race: Protoss, Zerg, Terran
  - Win rate (%): 0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
  - Terran has the highest win rate, followed by Zerg, then Protoss.
Conclusions

- Working Memory and a Blackboard has given us good results on this real-time environment.
- The crashes indicate that we need better tools for debugging.
- Potential Fields raise as an effective tool for tactical decisions.
- Unit control task can improve a lot the bot’s performance.
- Using FSM as our main strategy handler makes Nova easy to predict and less effective against undefined situations.
Future Work

- Coordinate squads to achieve joint goals.
- Exploit tactical locations to take advantage in combats.
- Use squad formations to flank the enemy and test more squad movement behaviours.
- Improve the opponent modelling.
- Test other techniques for planning like GOAP.
- Design a learning system to emerge new AI behaviours and/or strategies.