Second Workshop (W1) on AI in Adversarial Real-Time Games

https://www.cs.ualberta.ca/~mburo/aiide14ws

Co-organizers: M. Buro and S. Ontañón

4 submissions, 3 accepted, 2 invited talks

8 PC Members

14 Attendees
StarCraft
Activities

- Paper Session [1:30h]
- Invited Talks [1:40h]
- StarCraft Competition Report + Replays [1:10h] [Dave Churchill]
- Group Discussion [0:45h]
- Workgroups + Reporting [1:30h]
- Dinner
Paper and Invited Talk Summaries
Building Placement Optimization in Real-Time Strategy Games

• **Authors**
  – Nicolas Barriga, Marius Stanesu, Michael Buro

• **Premise**
  – Building placement is key in RTS games, but current bots don't do a good job

• **Approach**
  – Genetic Algorithm explores space of building placements
  – Game simulator (SparCraft) predicts the outcome of battles for given building configurations
Building Placement Optimization in Real-Time Strategy Games

• **Results**
  – Between 35 to 68% of losses turned into wins
  – Comparable to human building placement performance (from replays)
Sequential Pattern Mining in StarCraft:Brood War for Short and Long-Term Goals

• Authors
  – Michael Leece, Arnav Jhala

• Premise
  – Most AI solutions for RTS games require a significant amount of hand-crafting. Can we learn those from experts automatically?

• Solution
  – Generalized Sequential Patterns (GSP)
  – General algorithm for mining frequent patterns from sequences
  – 500 professional-level StarCraft replays
Sequential Pattern Mining in StarCraft: Brood War for Short and Long-Term Goals

• Results
  – Many interesting patterns detected: build orders, action spamming, army movement

• Next step
  – Learn patterns to be used as methods in HTN planning (into a bot)

Build Orders

1: Build(SupplyDepot)
2: Build(Barracks)
3: Build(Refinery)
4: Build(SupplyDepot)
5: Build(Factory)
6: AddOn(MachineShop)

1: Build(Pylon)
2: Build(Gateway)
3: Build(Assimilator)
4: Build(CyberneticsCore)
5: Build(Pylon)
6: Upgrade(DragonRange)
7: Build(Pylon)
High-Level Representations for Game-Tree Search in RTS Games

• **Authors**
  – **Alberto Uriarte**, Santiago Ontañón

• **Premise**
  – RTS games are too complex for game tree search
  – Can we abstract the game and use game tree search at this abstract level? Will search results still be meaningful?

• **Approach**
  – Proposed four different abstractions of the game state and used them to test game tree search (MCTSCD) in full-game scenarios.
  – Built a simulator that rolls the world forward using the abstractions
High-Level Representations for Game-Tree Search in RTS Games

• Results
  – Type of abstraction influences simulator accuracy
  – Impacts gameplay performance
  – Better than built-in AI
  – Worse than existing scripted approaches (from StarCraft competition)
Invited Talk 1

“State Evaluation and Opponent Modelling in Real-Time Strategy Games”

[Graham Erickson]

- Build order clustering from replays for game balancing and finding best response strategies
- Global RTS game state evaluation trained on replays
- Micro-skill estimation by comparing player with base-line player
Invited Talk 2

“GHOST: A Stealth Solver”  [Florian Richoux]

- Free Software C++ Constraint Satisfaction Solver Architecture
- Anytime, local search SAT solver + optimizer
- FAST!
- Applied to RTS Sub-Tasks:
  - Target Selection
  - Wall Building
  - Build order optimization
- Promising results!
Workgroup 1: Benchmark Problems

Problems
- Solutions still mainly scripted
- Only playing full-game tournaments may hinder progress on sub-problems

Solution: Sub-game competitions
=> Simpler, fosters modularity and generality

Idea: Sub-games relevant to full-game
=> Modules can be used in full-game bots
Sub-Game Candidates

- Small combat situations: $n$ vs. $m$ units
  [ regular / randomized unit stats ]
- Multiagent pathfinding: 100 zerglings vs. 4 bunkers?
- Base attack / defense
- Place buildings and survive attack waves
- Create / prevent expansion
- Faction unit/structure/techtree subsets

Will be considered for next year's competition
Workgroup 2: RTS AI History

Before StarCraft

- RL
- Influence maps
- Single-agent planning (e.g. HTN)
- Learning from demonstration
- Adversarial search and simulation
  (e.g. RandomAlphaBeta, MCPlan, RTSplan)
Workgroup 2: RTS AI History Since StarCraft

- Divide and conquer, modularizing AI
- Learning from replay data
- Build order recognition / optimization
- Tactical adversarial real-time search
  (ABCD, Portfolio Greedy Search, Combinatorial UCT, ...)
- High-level strategy selection with UCB
- High-level strategies still SCRIPTED
Workgroup 2: What should we be working on next?

**StarCraft**
Sub-games? Generalizations?

**Reactivity/Planning**
1. Plan recognition + best response
2. Holistic approach: scale up game-tree search  
   *(Two ideas presented in 10:15a session tomorrow)*

**Learning**
- Opponent modeling (in-game, from replays)
- Game mechanics from interacting with game
  => Simulators
Questions?