The First ORTS Game AI Competition

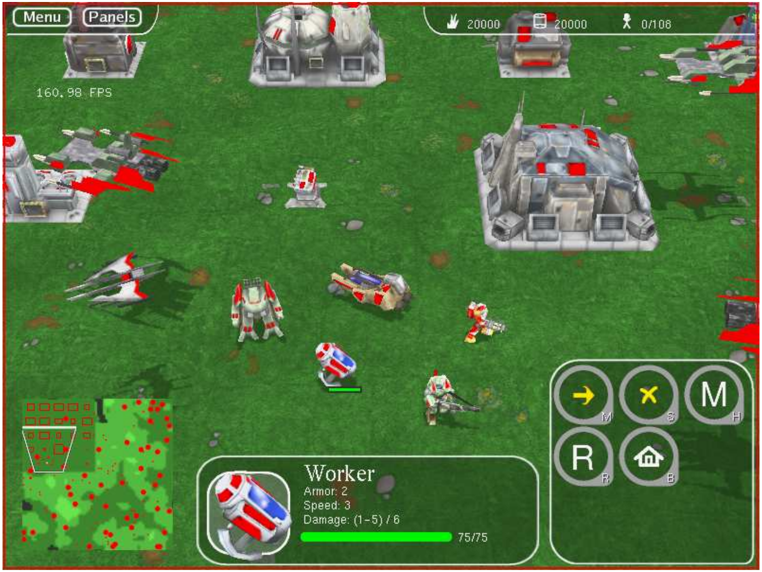
Michael Buro

GAMES Group
University of Alberta

(**G**ame-playing, **A**nalytical methods, **M**inimax search and **E**mpirical **S**tudies)

June 26, 2006

Motivation (1): Typical RTS Game User Interface



Motivation (2): Real-Time Strategy Games

Fast-paced video games modeling small-scale warfare

Players

- build economy and military infrastructure
- struggle over **resources** located on a 2d playing field
- have to wipe out opponents to win

Limited vision ("Fog of War")

Usually $\approx$ 10 simulation steps per second

Very popular!

Good testbed for studying multi-agent AI problems

Motivation (3): AI Challenges

Current AI systems for RTS games do not

reason, **plan**, nor **learn**

Human players better than machines at macro level

Large number of simultaneous actions with local effects
⇒

- Brute-force search fails
- Need **abstractions**,
- Adversarial Planning & Plan Recognition

Partial observability
⇒ need to maintain **history** and **beliefs**

Why not ...

- Choose a popular commercial RTS game,
- hook up AI software,
- ask your friends to hook up their AI software,
- and play tournaments to see whose system is better?

Doesn't work!

Solution:

ORTS

Outline

- Motivation ✓
- ORTS
- Tournament Setup
- Game 1
- Game 2
- Game 3
- Future

How ORTS relates to commercial RTS games

Feature	Typical Commercial RTS Game	ORTS
Game Specification	**fixed**	**user-definable**
Network Architecture	peer-to-peer	server-client
Network Data Rate	**low**	**low** to **medium**
Prone to Map-Revealing Hacks	**yes**	**no**
Communication Protocol	**veiled**	**open**
Unit Control	**high-level, sequential**	**low-level, parallel**
Game Interface	**fixed GUI**	**user-definable**
License	**closed software**	**free software (GPL)**

Current Projects

- Graphics
  - Hook up OGRE to ortsg
- Standard ORTS Game
  - Three Races
  - **Random Map** Generation
  - Internet ORTS Server **ORTS.NET**
- AI
  - **Group pathfinding**
  - **Constrained pathfinding**
  - **Small-scale combat**
  - **Simulation**
  - **Build-order optimization**

First ORTS Game AI Competition

Call for participation in spring

Authors install their code on machines at U. of A. – not accessible to us

Source will be made available next week

Three Game Categories

- Cooperative Pathfinding: 20 workers gather minerals
- Combat: 50 vs. 50 tanks
- A “Real” RTS Game: economy, techtree, limited view

No entry fee, but \$250 prize money for best player $\neq$ uofa in each category. Thanks Ian Davis!

Tournament Setup

31 ugrad lab machines

Athlon XP 1.5GHz, 512 MB RAM

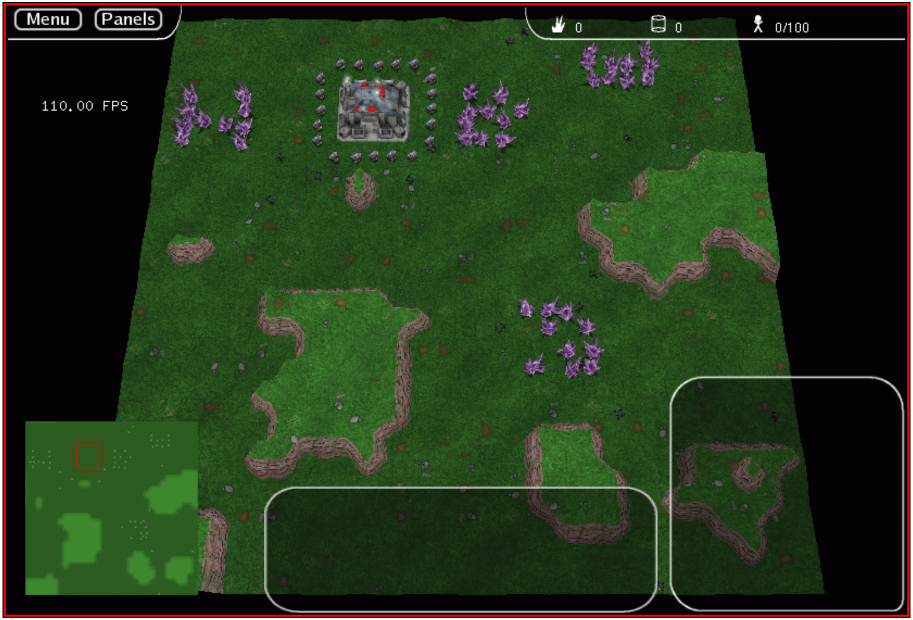
Linux 2.4.31, gcc 4.1.1

Multi-threaded tournament manager written by Krysta Mirzayans

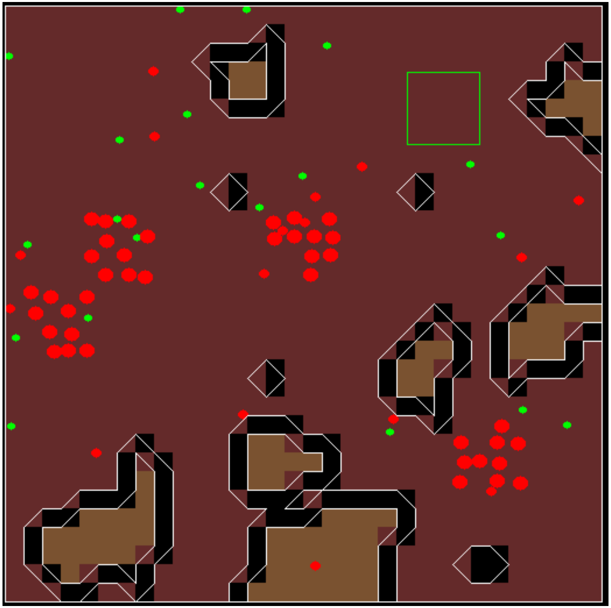
Tournament Configuration File

```
# game 1 tournament configuration
# general parameters

-564096555                    # seed (incremented for subsequent games)
1                             # game type (1: single player, 2 and 3: two-player)
16                            # >= game time in minutes + 1 (watchdog timer)
                              # important: check timeout in game spec.!
s                             # [s]ingle / [d]ouble game matches (map played twice)
225                           # matches per player pair
1                             # maximum # of jobs on each computer
                              # (one job for the server and each client)
                              # could be 6 on dual-cpu machines and 2-player games
serverout_g1                  # server output subdirectory
clientout_g1                  # client output subdirectory
replays_g1                    # replay output subdirectory in the server directory
                              # make sure it exists!
%

# server/client parameters
# name unix-id (.: none) work-directory cmd [with options]
# (cd to workdir, then run client or server)

server  .       ~/orts3  tournament/game1_orts -sdelay 10
uofa1   rts02r  ~        ~rts02s/game1
creed1  rts05r  ~        ~rts05s/game1
brzo1   rts07r  ~        ~rts07s/aiclient1
umich1  rts03r  ~        ~rts03s/game1
%

# machines
ug01
ug02
# ...
```

Game 1: Cooperative Pathfinding



Game 1: Cooperative Pathfinding



Game 1 Entries

brzo1 :	Michal Brzozowski	(Univ. of Warsaw, Poland)
creed1 :	Michal Szostakiewicz	(U. of Warsaw, Poland)
umich1 :	Joseph Xu, Sam Wintermute	(U. of Mich.)
uofa1 :	David Deutscher	(Tel Aviv University, Israel)
	Nick Wiebe	(University of Alberta, Canada)

Game 1 entry brzo1

- Discrete graph-based terrain representation
- Neighbouring vertices connected if edge traversable
- Workers are guided by an FSM:
  - **move to**, **mine**, **go back**, **drop resources**
  - **avoid** : entered when hitting a moving obstacle. Avoids obstacles by moving to the left. When it hits a static obstacle, it moves to a random direction
  - **emergency path** : when hitting a number of obstacles in the avoid state, tries to get back to original path
- A coordinator assigns workers to resources based on shortest paths. Each worker picks the closest mineral from his starting point with < 2 workers assigned already.

Game 1 entry creed1

- Each worker is assigned a **random** mineral patch and is sent to it
- Using Djikstra's shortest path algorithm
- Search graph built from nodes representing tile centers
- Edge weights depend on mobile objects close by to prevent collisions
- When colliding workers move to a random location nearby

Game 1 entry umich1

Implemented in the SOAR architecture

- Workers guided by mining manager and FSM
- Using modified standard ORTS pathfinding
- Local obstacle avoidance
- If worker exceeds estimated travel time, it requests a new route from the mining manager
- Mining manager learns which routes are bad

Game 1 entry uofa1

Multiple-resolution grid-based A*, touch target mode

Path execution module based on “force fields” for avoiding collisions. Idle units can be pushed.

Fallback to replanning a path after several repeated failures to move

Dynamic allocation of minerals to workers, based on minimizing a weighted combination of: (1) the Euclidean distance from the worker and (2) the path length between the mineral and the CC only considering terrain and minerals as obstacles.

Only one worker per mineral patch

Game 1 Results

$4\times$ 225 games played lasting 10 minutes each

20 hours into the tournament the tournament manager exceeded its disk quota

To stay on track I reduced the number of matches to 225 (down from 300)

Game 1 Results Continued

rank	name	score	matches	ratio	errors
1.	umich1	1458455.0	225	6482.02	0
2.	brzo1	1136690.0	225	5051.96	38 (*)
3.	uofa1	1136790.0	225	5052.4	43 (*)
4.	creed1	559380.0	225	2486.13	0

brzo1 was leading over uofa1 almost all the time. So, team uofa happily conceded 2nd place to it (*).

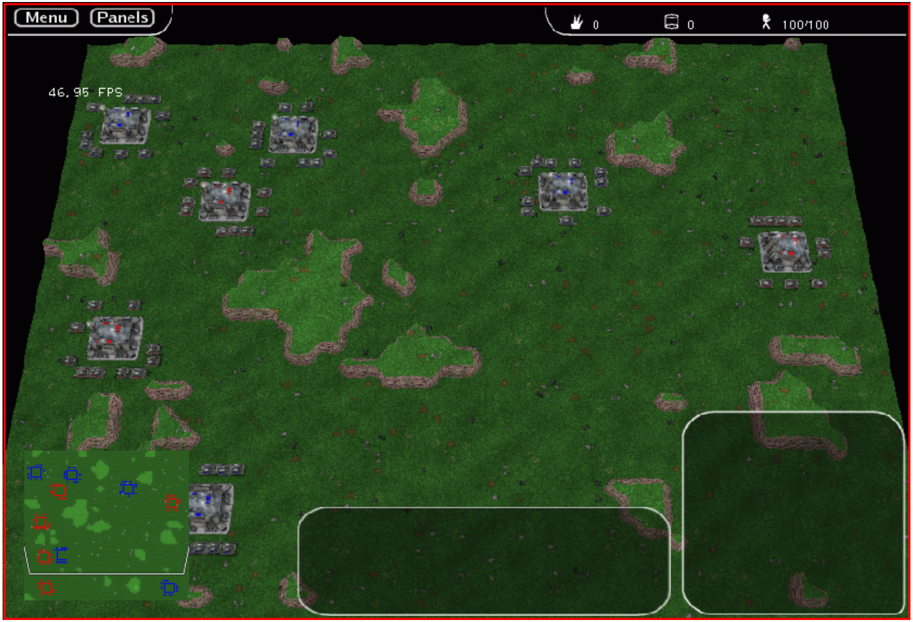
Game 1 Results Continued

Not considering the segfault games, ratios are as follows:

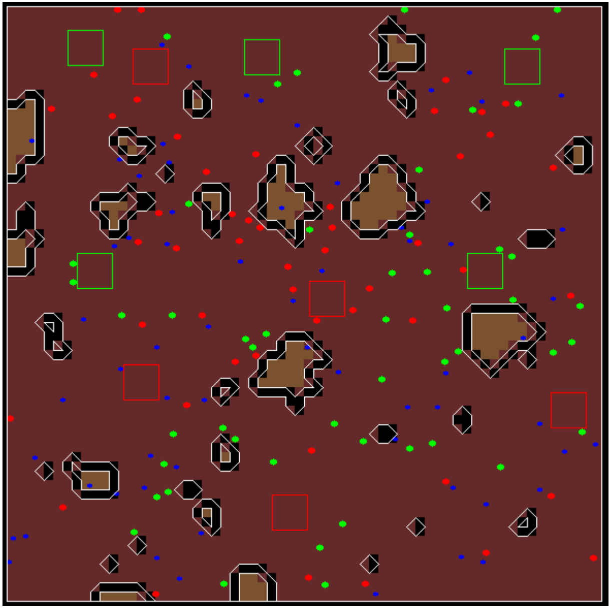
umich1	6482.02
uofa1	6246.10
brzo1	6078.56
creed1	2486.13

creed1 made the server crash several times by referring to fully mined mineral patches which had vanished

Game 2: Tank Combat



Game 2: Tank Combat



Game 2 Entries

umaas2

- Philip Kerbusch, Nyree Lemmens, Maurice Urlings, Viktor Vorsteveld (University of Maastricht, The Netherlands)

umich2

- Joseph Xu (University of Michigan)
- Sam Wintermute (University of Michigan)

uofa2

- K. Anderson, J. Bergsma, D. Demyen, T. Furtak, D. Tom, F. Sailer, N. Wiebe (U. of Alberta, Canada)
- D. Deutscher (Tel Aviv University, Israel)

Game 2 entry umaas2

Creates 5-tank-squads in column formation

Leader plans path to nearest base, others follow

When target is encountered, line or wedge formation is formed

All objects excluding opposing tanks are considered obstacles

Each tank reserves a tile

Game 2 entry umaas2 continued

Algorithm:

- Locate closest enemy base and move towards it in snake formation.
- If during this traversal, enemy tanks are encountered, attack the **weakest** of all tanks in range.
- Tanks move towards the weakest target while firing at the weakest target in range.
- If the base is destroyed, locate a new base and move towards it.
- If no more enemy tanks are in sight, resume formation and travel to the enemy base and attack it.

Game 2 entry umich2

SOAR agent attacks tanks before bases

Tanks are grouped by spatial distance, and groups of tanks will try to attack enemy tank groups that are smaller than them.

If no such enemy groups exist, smaller groups will try to regroup into larger groups and go for their target then.

Unfortunately, a bug was introduced just before the deadline, and most of this behavior was not realized in the competition.

Game 2 entry uofa2

Find a suitable spot to meet close to the average tank position

Send each tank there after joining locally first

When join is finished, start hunting and attacking the closest enemy tank with the entire group. When all tanks are destroyed, bases are attacked.

Weakest tartgets are attacked first while minimizing overkill

Game 2 Results

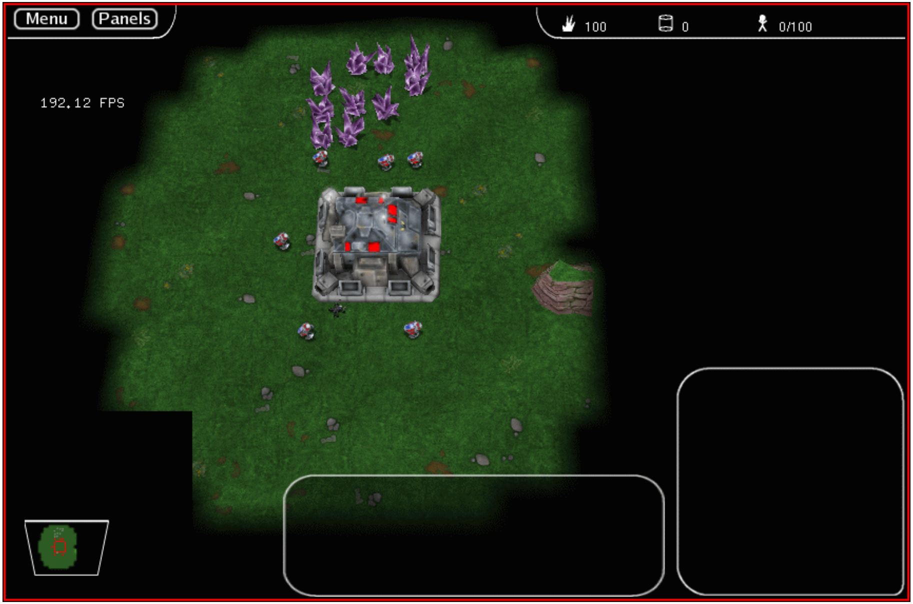
400 two-game matches played for each player pair

Each game lasted at most 15 minutes

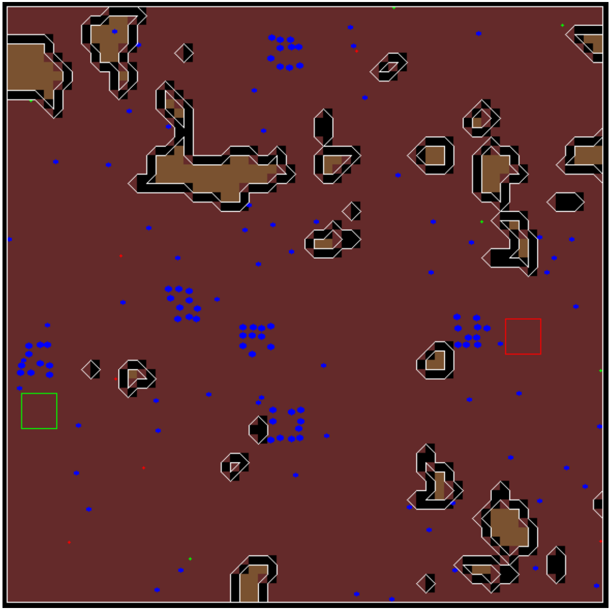
name	score	matches	ratio
uofa2	390.0	400	0.975
umaas2	210.0	400	0.525
umich2	0.0	400	0

uofa2 crashed in 20 games, but only lost 10 matches

Game 3: Simplified RTS Game



Game 3: Simplified RTS Game



Game 3 Entries

umich3

Joseph Xu	(University of Michigan)
Sam Wintermute	(University of Michigan)

uofa3

- K. Anderson, J. Bergsma, D. Tom, T. Furtak, F. Sailer, N. Wiebe (University of Alberta, Canada)
- D. Deutscher (Tel Aviv University, Israel)

Game 3 entry umich3

SOAR agent in addition to mining builds barracks and group of marines

Marines are sent to explore and attack

Defensive behaviour takes presedence, and all units are pulled into battle if the base is under attack

Game 3 entry uofa3

Turtling strategy

Create a barracks and enough workers such that each visible mineral patch is mined

Then create as many marines as we can

The squad combat AI used in game 2 controls all combat actions

An older version of the mining AI controls mining

Many crashes were due to a wrong assumption that the mining AI handles fog of war sufficiently

Game 3 Results

200 two-game matches were played

Each game lasted at most 20 minutes

name	score	matches	ratio	errors
uofa3	74.0	200	0.37	2
umich3	124.0	200	0.62	2

Server crashed twice

uofa3 crashed 30% of the time when running out of memory :-(

AI Lessons Learned

Need to set aside more time for **testing** on tournament hardware

Effective **group pathfinding** and **targeting** is important

Smart **high-level control** is crucial

AI strength can be improved a lot

No need to make games more complex at this point

Lessons Learned: ORTS Software

ORTS architecture can be used to run RTS game AI competitions on a cluster

Hooking up AI systems to ORTS isn't **trivial**

Need to **simplify** and document the ORTS client interface more

ORTS server needs to check player messages more thoroughly

Need developers who create and maintain a **native Windows ORTS build**

Future ORTS Competitions

Make this a regular AIIDE event

Now that infrastructure is in place, we can run tournaments more frequently

Keep the three categories to monitor progress

Extension Ideas

Opponent modeling: Allow programs to store information about previous encounters

Metagames: E.g. randomize numerical game parameters

More: air units, ramps, complex techtrees, upgrades

Commercial RTS Game AI Competition: TIELT?

Interested in helping, suggestions?

`mburo@cs.ualberta.ca`