Search and Abstraction in Real-Time Strategy Games

Michael Buro

with

N. Barriga, D. Churchill, D. Demyen, G. Erickson,
T. Furtak, M. Lanctot, I. McCarten,
S. Orsten, S. Ontañón, A. Saffidine, F. Sailer,
D. Schneider, M. Stanescu, N. Sturtevant

UNIVERSITY OF ALBERTA

Collaborators

Douglas

Abdallah

Sterling

Frano

Doug

Santi

Graham

Nicolas Dave Marius

Marc

Nathan

Isabel

Tim

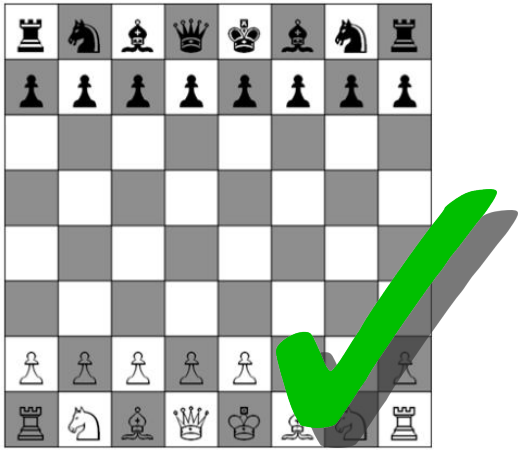
AI / ML Group @ University of Alberta

Edmonton, Canada

Heuristic Search, **Domain Independent Planning**, **State and Action Abstractions**, **Opponent Modelling**, **Solving Games**, **Path Planning**, **Real-Time Search**, etc.

- Interested in World-class AI or ML research and spending time in Canada ?
- **We are looking for graduate students !**

Human vs. Machine



Real-Time Strategy (RTS) Games

RTS Game Properties

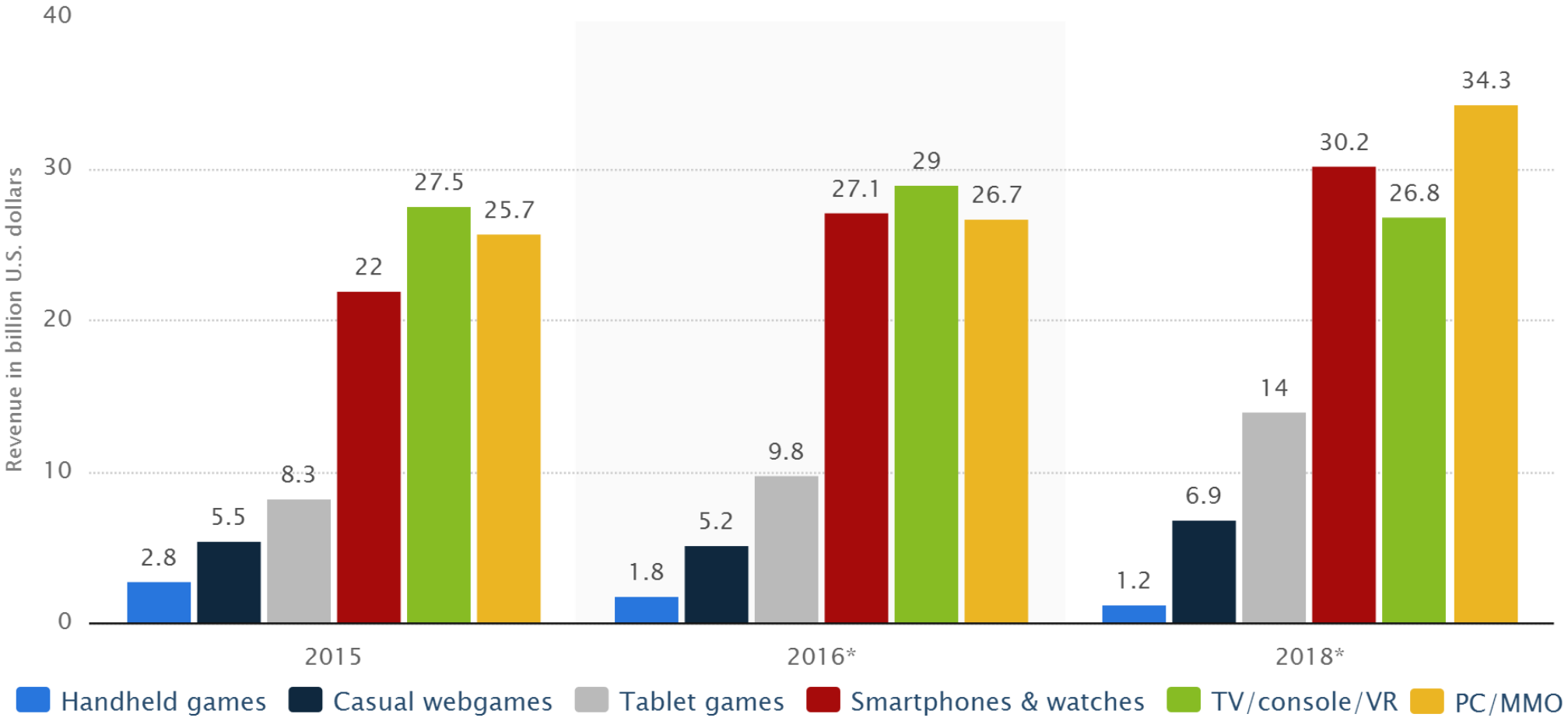
- Fast paced war simulations
- Real-time:
  - simulation continues even when players don’t act
- Imperfect information
  - Simultaneous moves + “Fog of War”
- Game mechanics:
  - Gather resources
  - Build towns and armies
  - Combat with enemies
  - Last player standing wins
- Huge state and action spaces

E-Sports

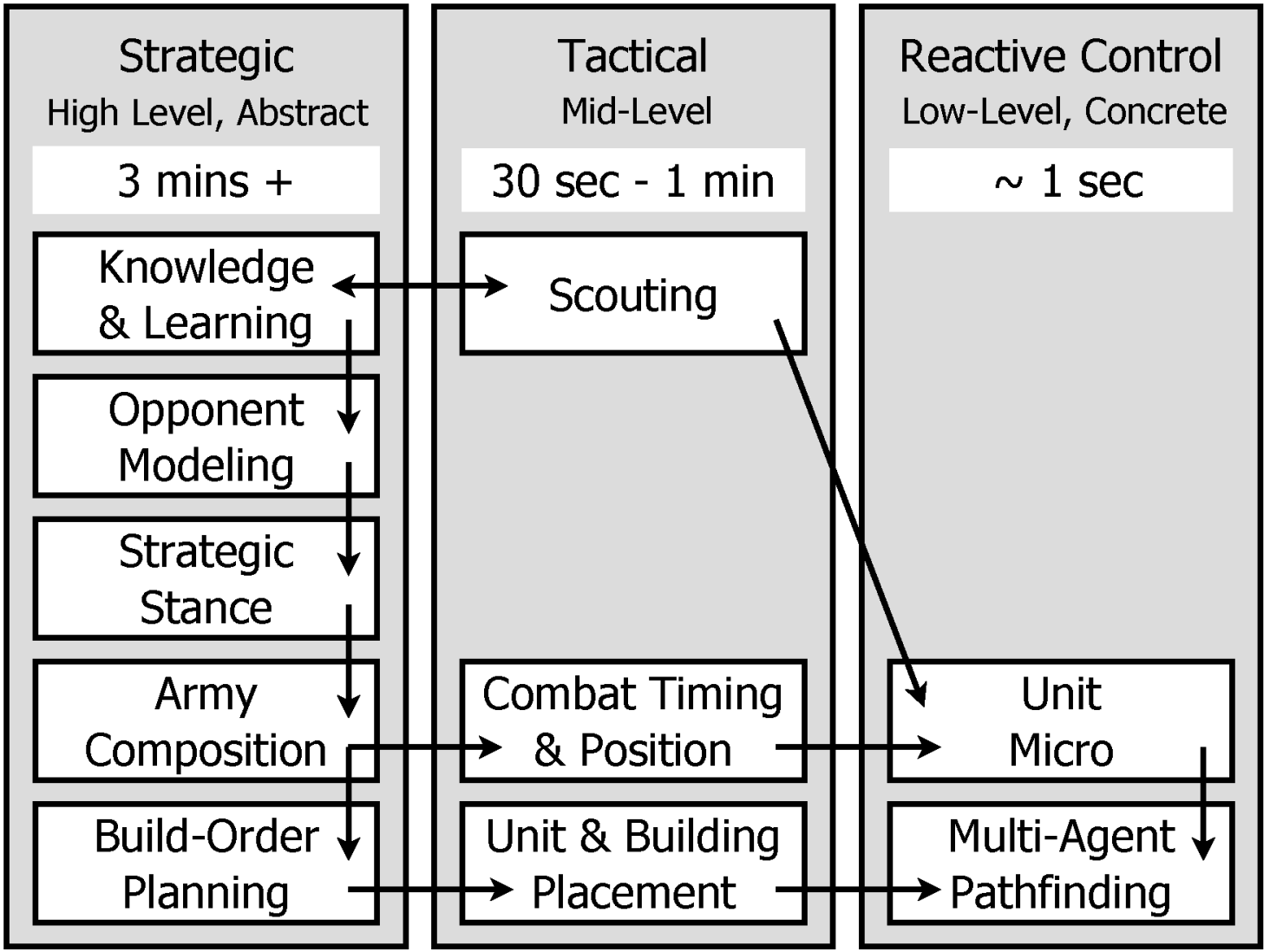
gettyimages
Bloomberg
SK planet
StarCraft
Season1
ProLeague
147618187


2014 WCS
StarCraft
WORLD CHAMPIONSHIP
SERIES

Video Game Industry



RTS Game Strategy Decomposition



Variations on Search and Abstraction in RTS Games

1. Finding Paths Quickly

2. Building Things Quickly

3. Destroying Things Quickly

4. Master of Puppets

5. Ω-StarCraft ?

Variation 1: Finding Paths Quickly

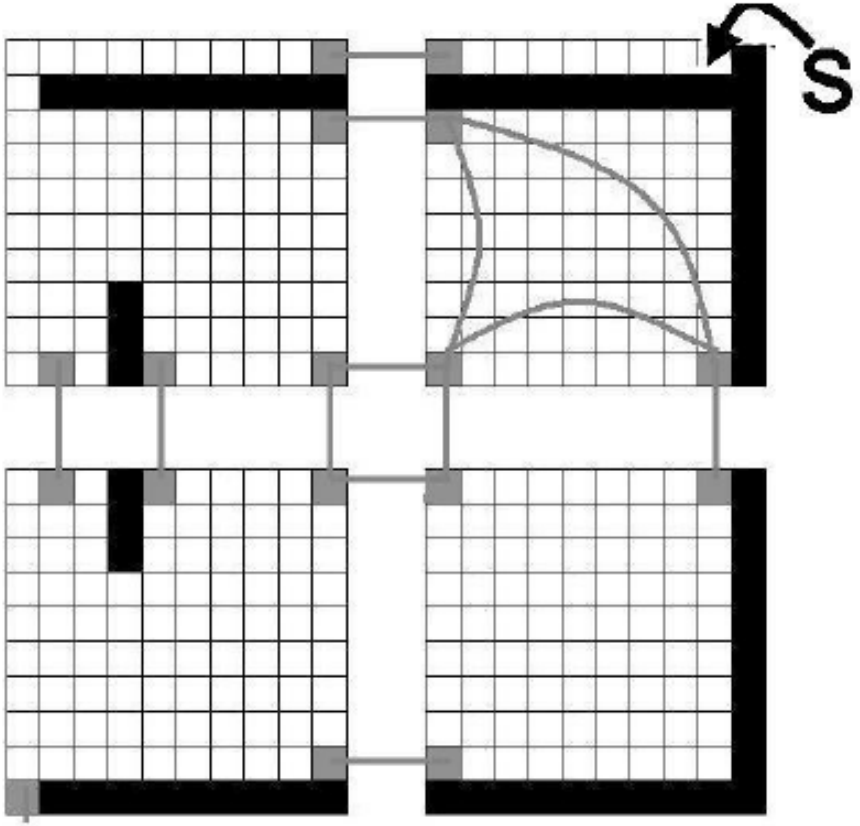
Challenges

- Large maps (often 4096 x 4096 walk tiles or more)
- Real-time constraints
  - Generate near-optimal paths for dozens of units within milliseconds
  - If more time is needed, start moving even before optimal path is found
- Dynamic obstacles
- Collaborative and/or adversarial

Hierarchical Path Finding A* (HPA*)

(Botea, Müller, Schaeffer 2004)

- Sector based, select sector entrances
- Optimal intra-sector paths between entrances
- Run top-level A*
- Path smoothing



- Enhancements:

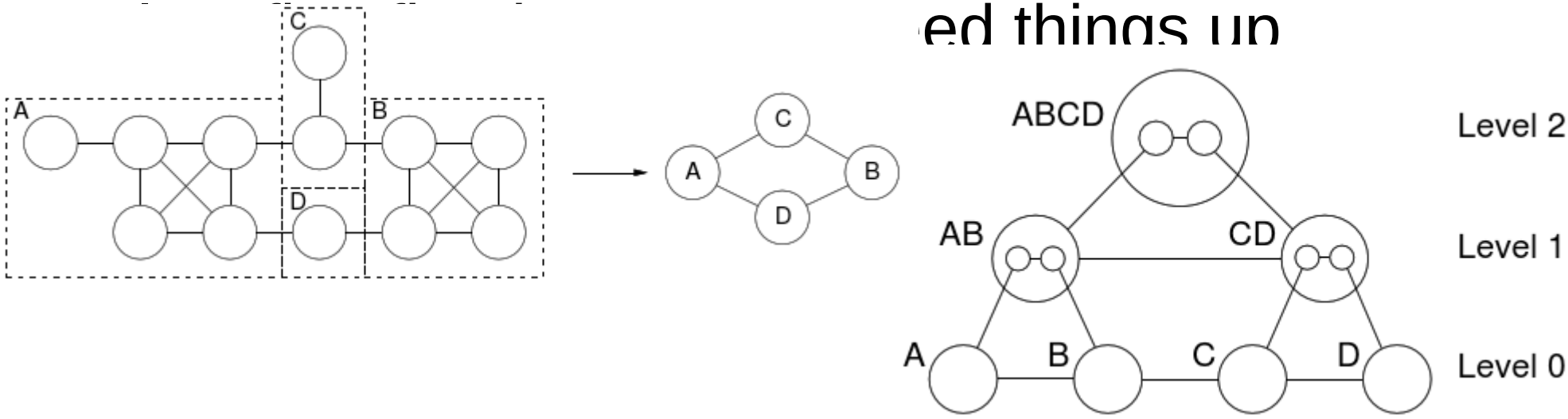
  DHPA* and SHPA*

  (Kring et al., 2010)

Path Refinement A* (PRA*)

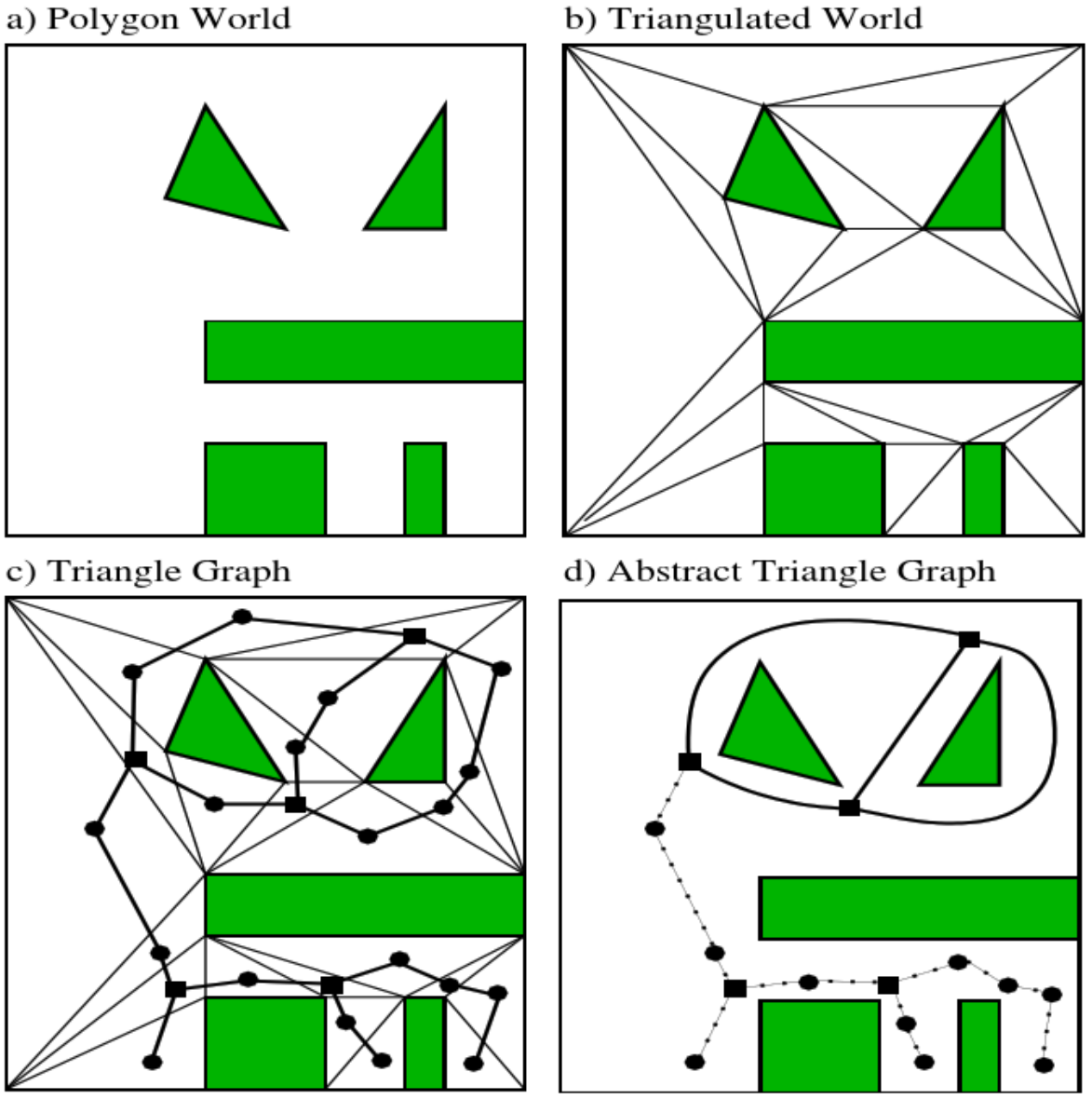
(Sturtevant and Buro, 2005)

- Use clique abstraction to create abstraction "pyramid"
- Pick abstraction level
- Map start/goal points to their abstract counterparts
- Use A*, project path down, restrict A* to corridor, ...

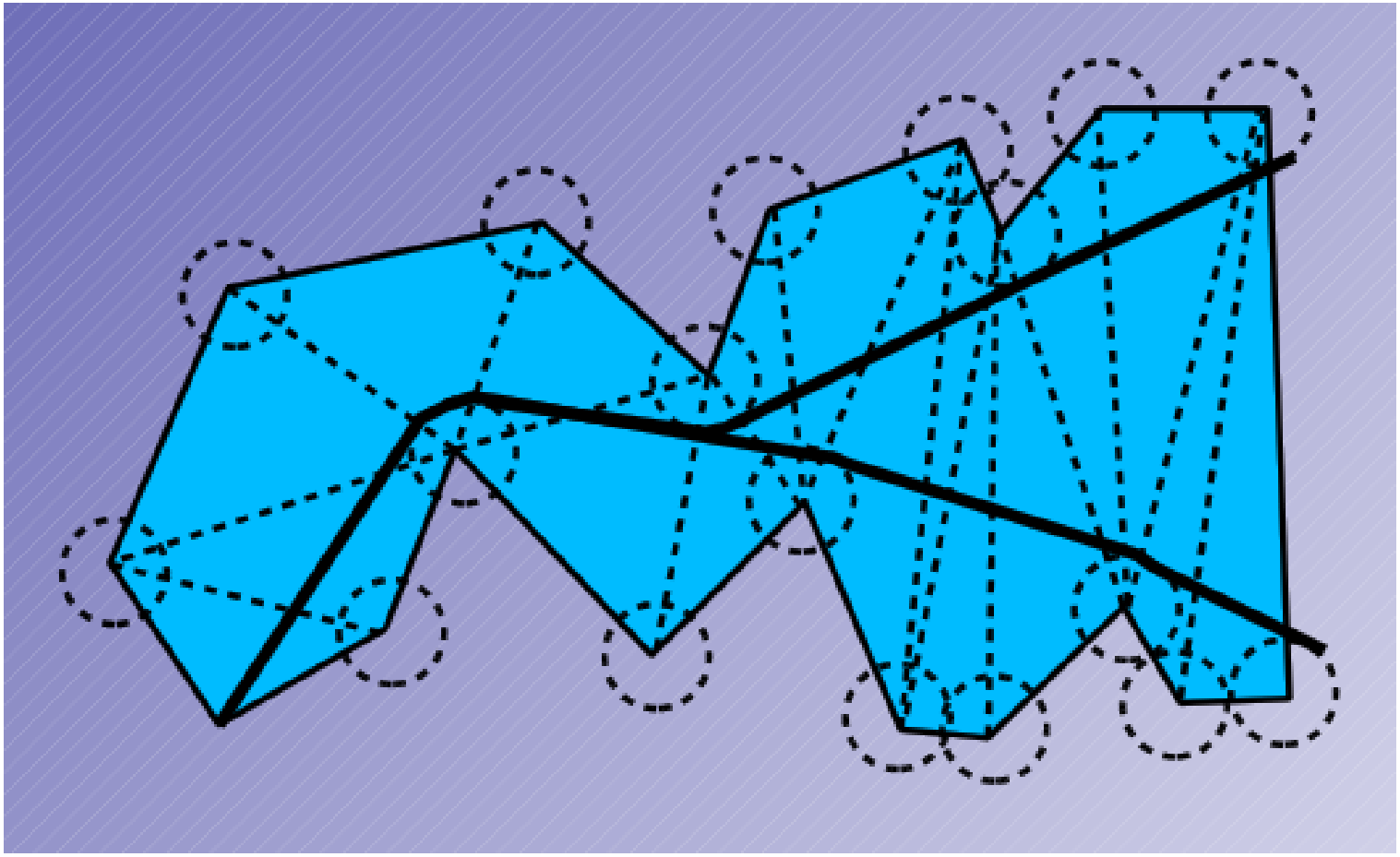


Triangulation (Reduction) A*

(Demyen and Buro, 2006)



Modified Funnel Algorithm



TA* and TRA* in Practice

- Any-angle path finding for circular objects in large polygon soups
- TRA* ~50 times faster than A* on 512 x 512 tile game maps, near optimal path length
- Blizzard Entertainment uses TA* in StarCraft 2

Other Selected Work on Path Finding

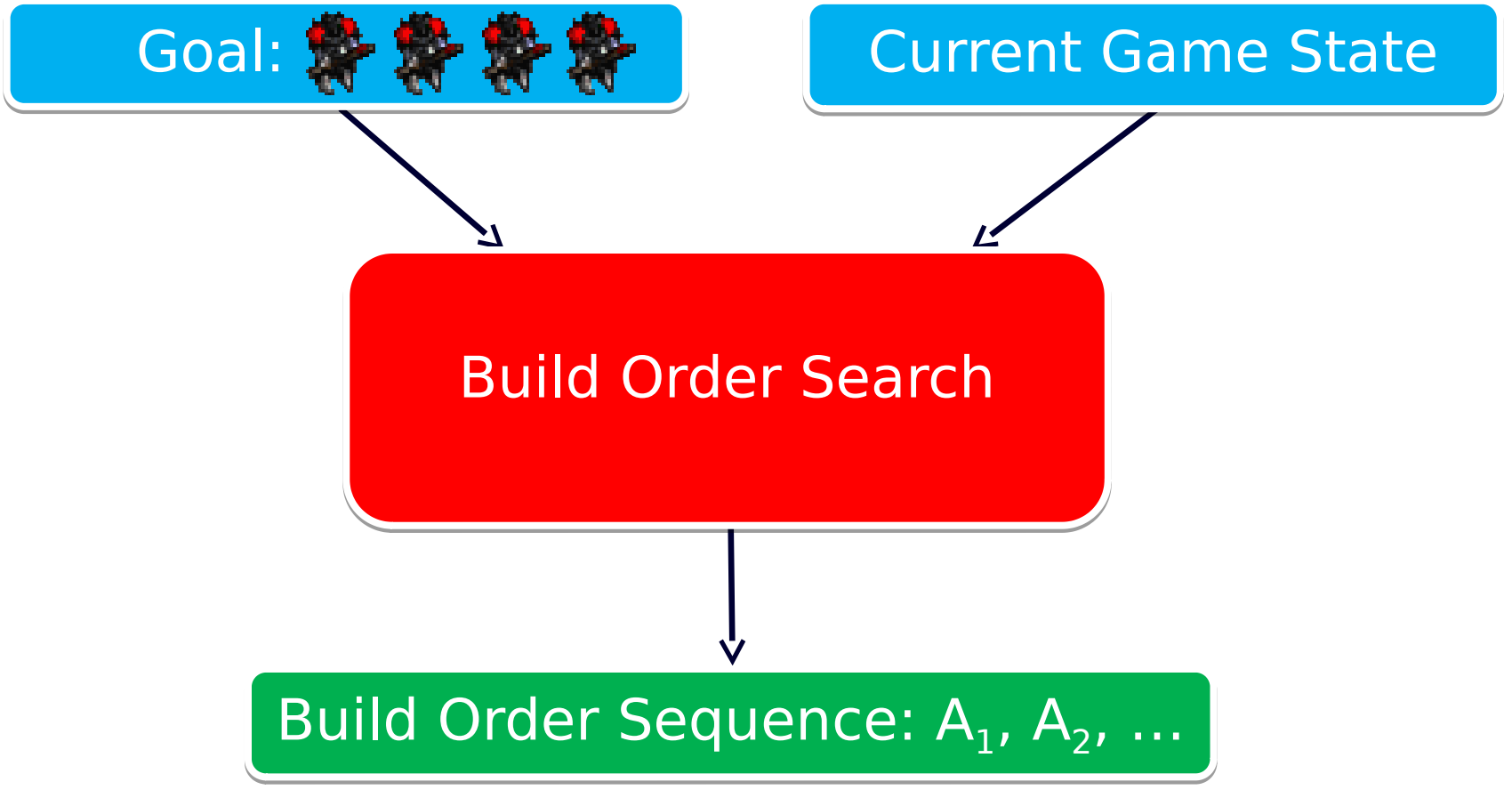
- Optimal Any-Angle Pathfinding in Practice

  (Harabor et al., JAIR 2016)

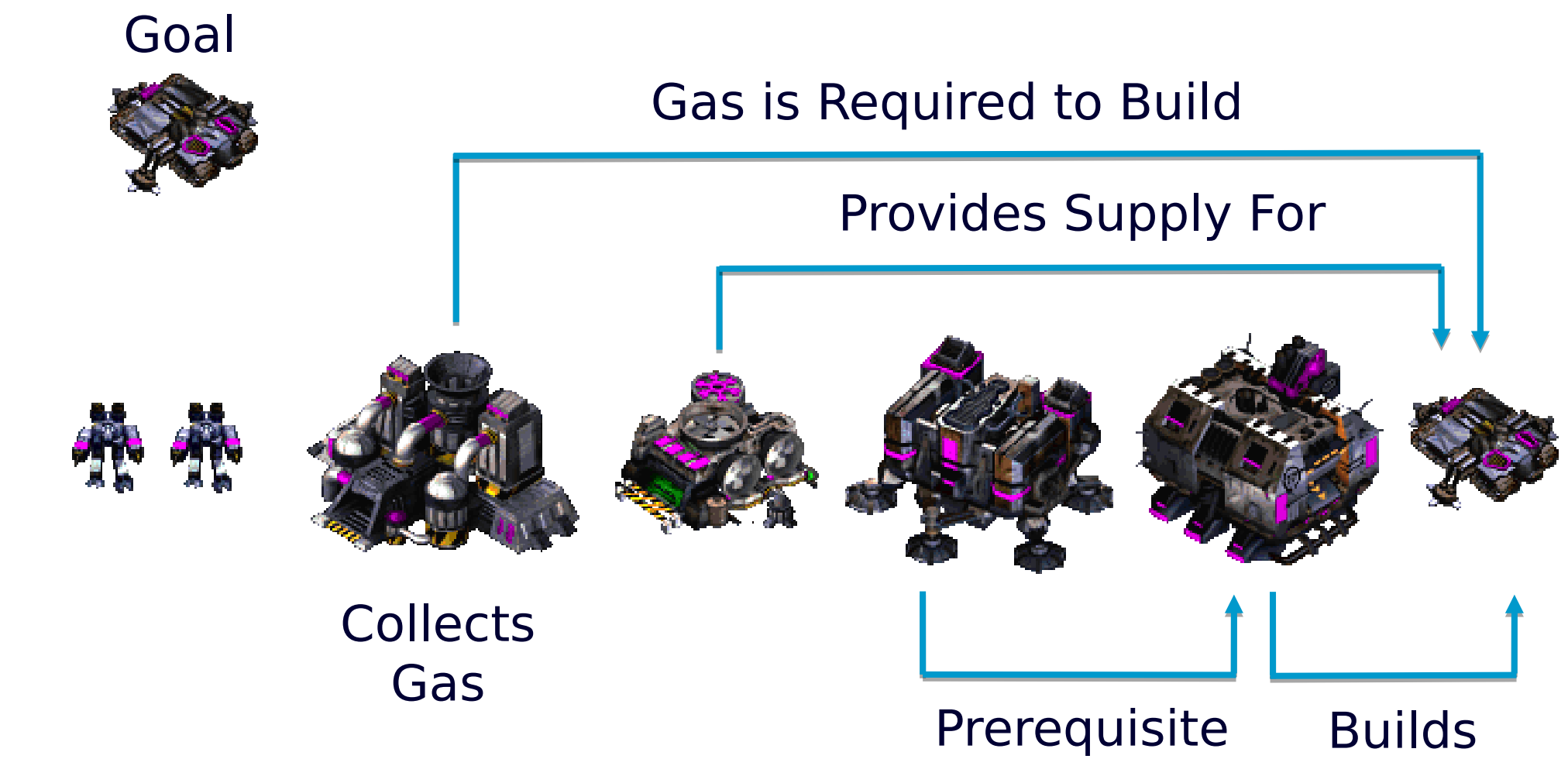
- Ultra-fast Optimal Pathfinding without Runtime Search

  (Botea, AIIDE 2011)

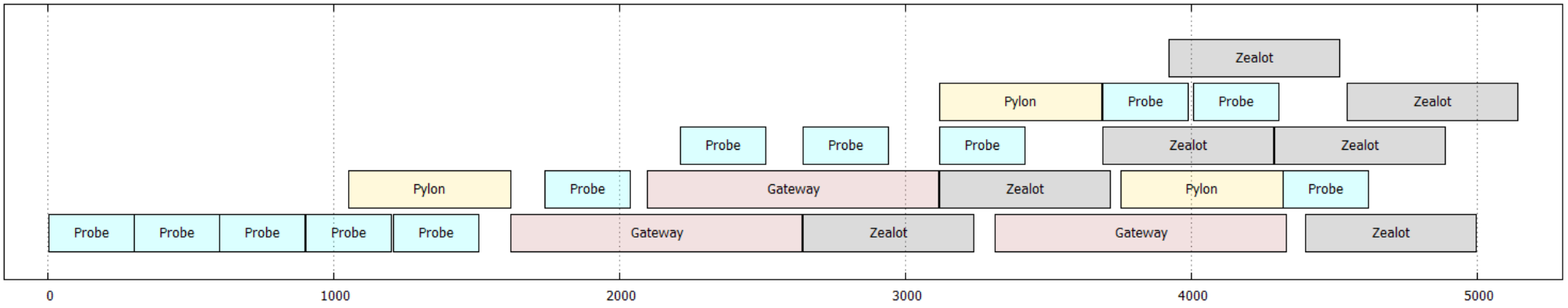
Variation 2: Build Things Quickly



Build Orders



Concurrent and Durative Actions

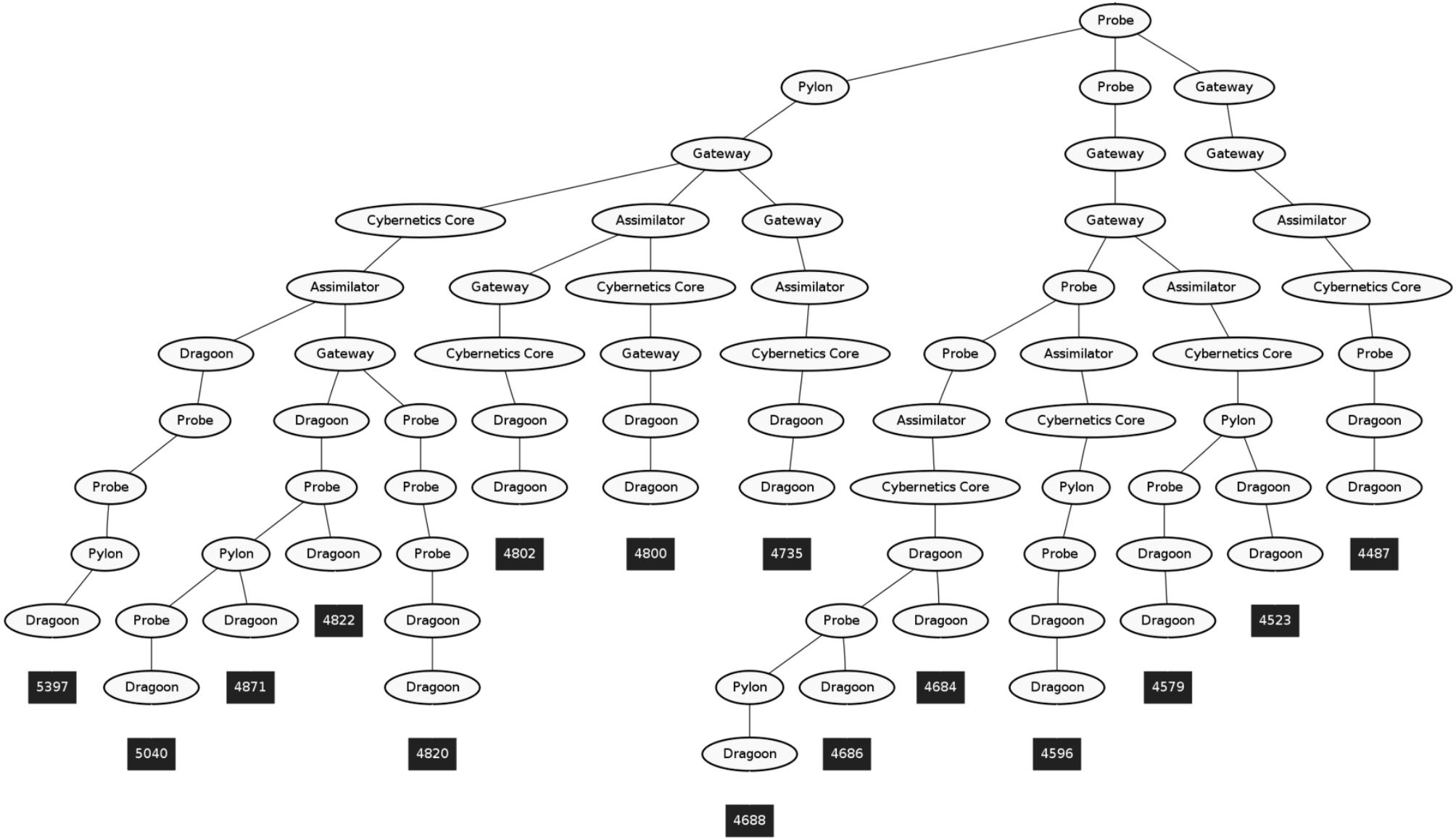


Build Order Optimization

- For given state and production goal, minimize makespan
- Use depth-first branch and bound algorithm
  - Low memory usage
  - Any-time computation
  - Pause / resume search easy
  - Make use of upper and lower bound heuristics

(Churchill and Buro, AIIDE 2011)

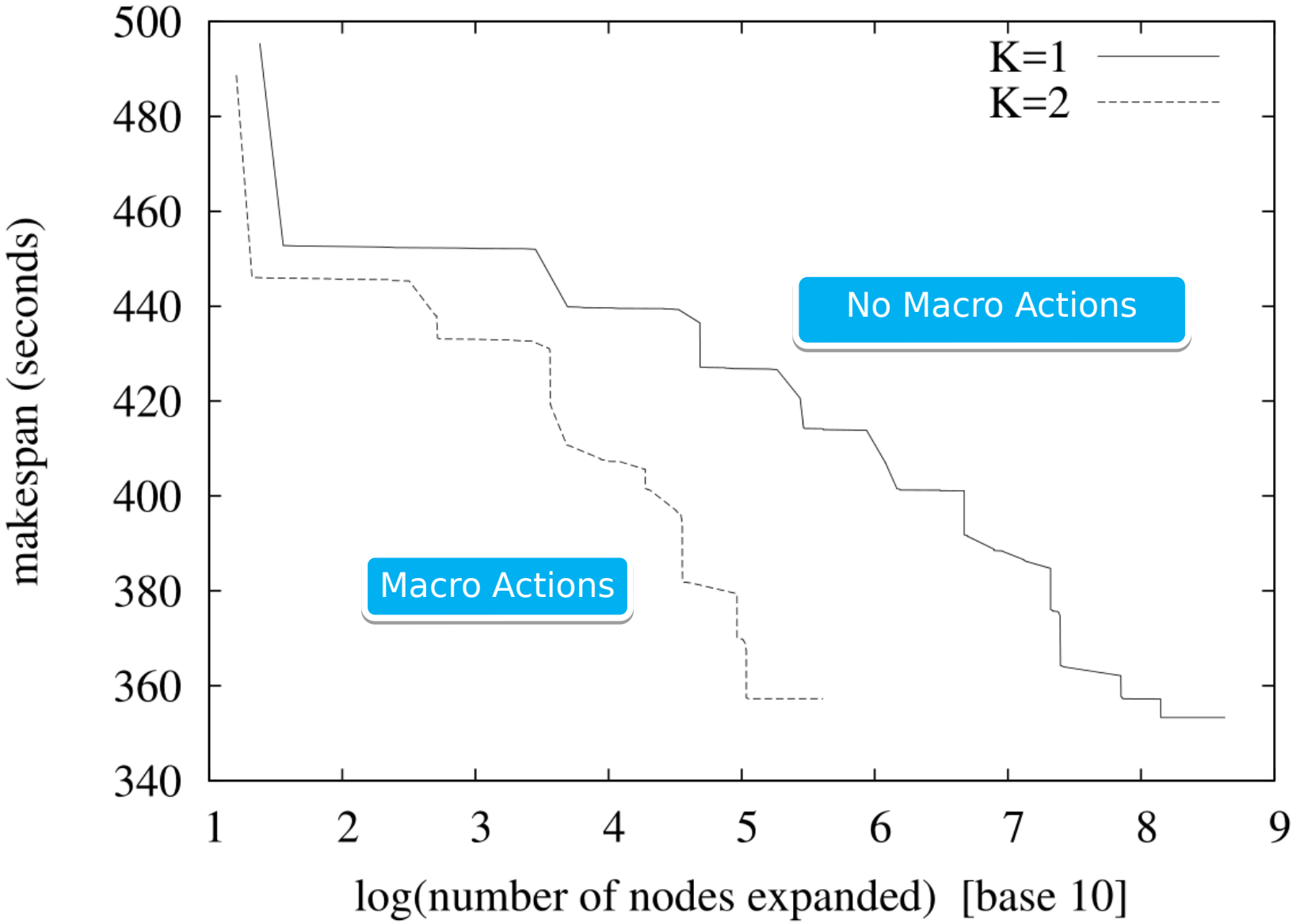
Sample Run


Probe
Pylon
Probe
Gateway
Gateway
Gateway
Gateway
Cybernetics Core
Assimilator
Gateway
Gateway
Assimilator
Assimilator
Gateway
Cybernetics Core
Assimilator
Probe
Assimilator
Cybernetics Core
Dragoon
Gateway
Cybernetics Core
Gateway
Cybernetics Core
Probe
Assimilator
Cybernetics Core
Probe
Probe
Dragoon
Probe
Dragoon
Dragoon
Dragoon
Assimilator
Cybernetics Core
Pylon
Dragoon
Probe
Probe
Probe
Dragoon
Dragoon
Dragoon
Cybernetics Core
Pylon
Probe
Dragoon
Dragoon
Pylon
Pylon
Dragoon
Probe
4802
4800
4735
Dragoon
Probe
Dragoon
Dragoon
4487
Dragoon
Probe
Dragoon
4822
Probe
Dragoon
Dragoon
Dragoon
Dragoon
4523
5397
Dragoon
4871
Dragoon
Dragoon
Pylon
Dragoon
4684
Dragoon
Dragoon
4579
5040
4820
Dragoon
4686
4596
4688

Optimizations and Heuristics

- Actions executed as soon as they are legal
  - Resource hoarding not benefiting economy
- Build actions don’t invalidate other build actions
- Fix ordering of concurrent actions
- Model resource gathering: C * time * workers
- Use makespan of sequential plan as upper bound
- Use tech-tree landmarks as lower bound
- Use gathering goal resources as lower bound
- Limit unit numbers
  - (e.g. building 2 tanks requires at most 2 factories)
- Macro Actions
  - Repeat certain actions k times

Effect of Macro Actions



Comparison with Professional Players

Search Without Macro Actions (120s Interval)	
Problems solved (total)	100%
Median Search/Pro makespan ratio	0.964
Median Search Time (as % of makespan)	4.1%

Search With Macro Actions (120s Interval)	
Problems solved (total)	100%
Median Search/Pro makespan ratio	0.957
Median Search Time (as % of makespan)	1.4%

Variation 3: Destroy Things Quickly

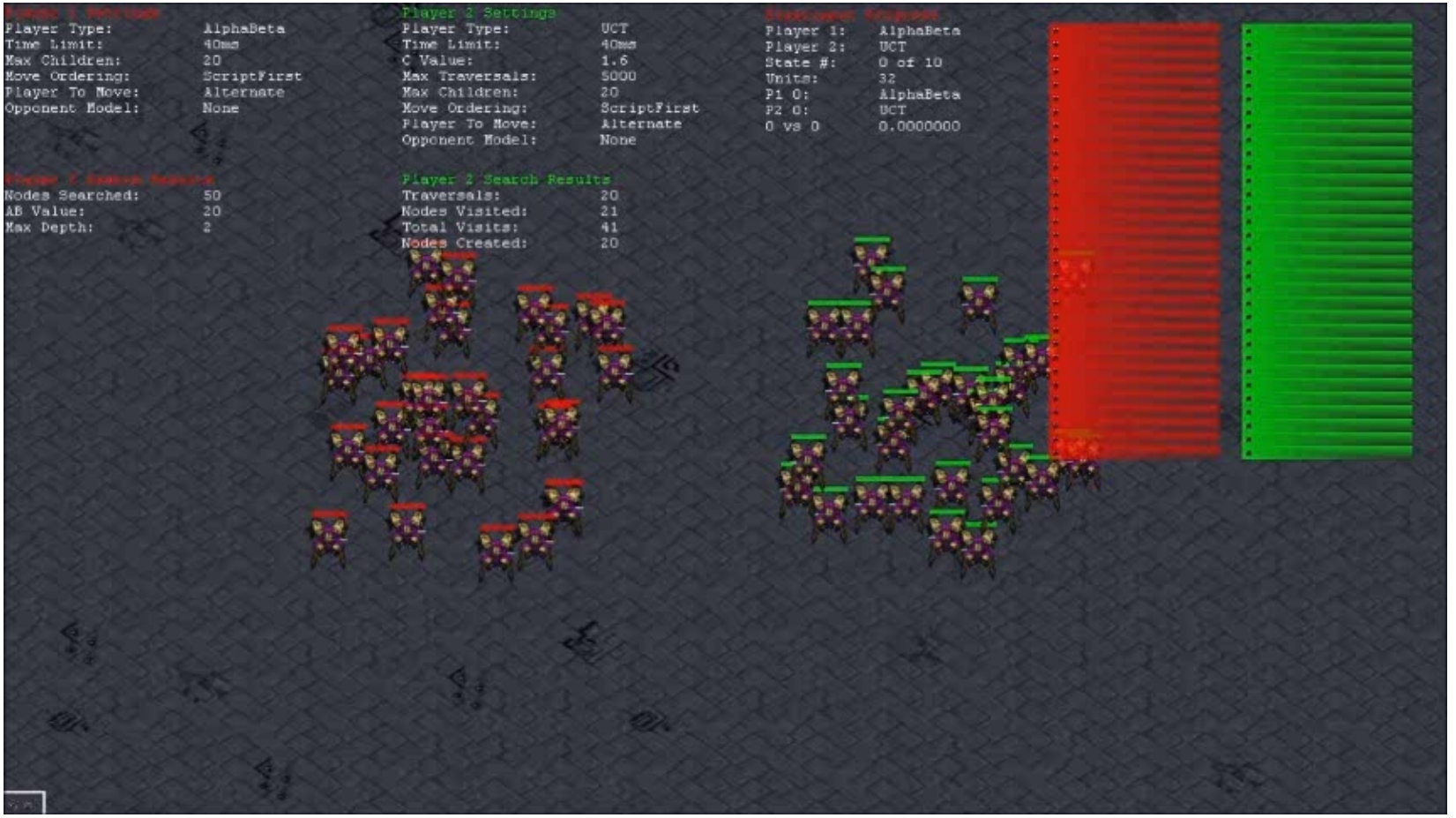
Combat Model

- Two groups of combat units fight each other
- Simultaneous attack / move actions ("stacked matrix games")
- Each unit
  - can possibly move
  - can attack others within weapon's range
  - has health points from which attack values are deducted
  - dies when health points reach 0
- Weapons have attack values and cool down periods and possibly area effects
- Combat games end when one party is eliminated

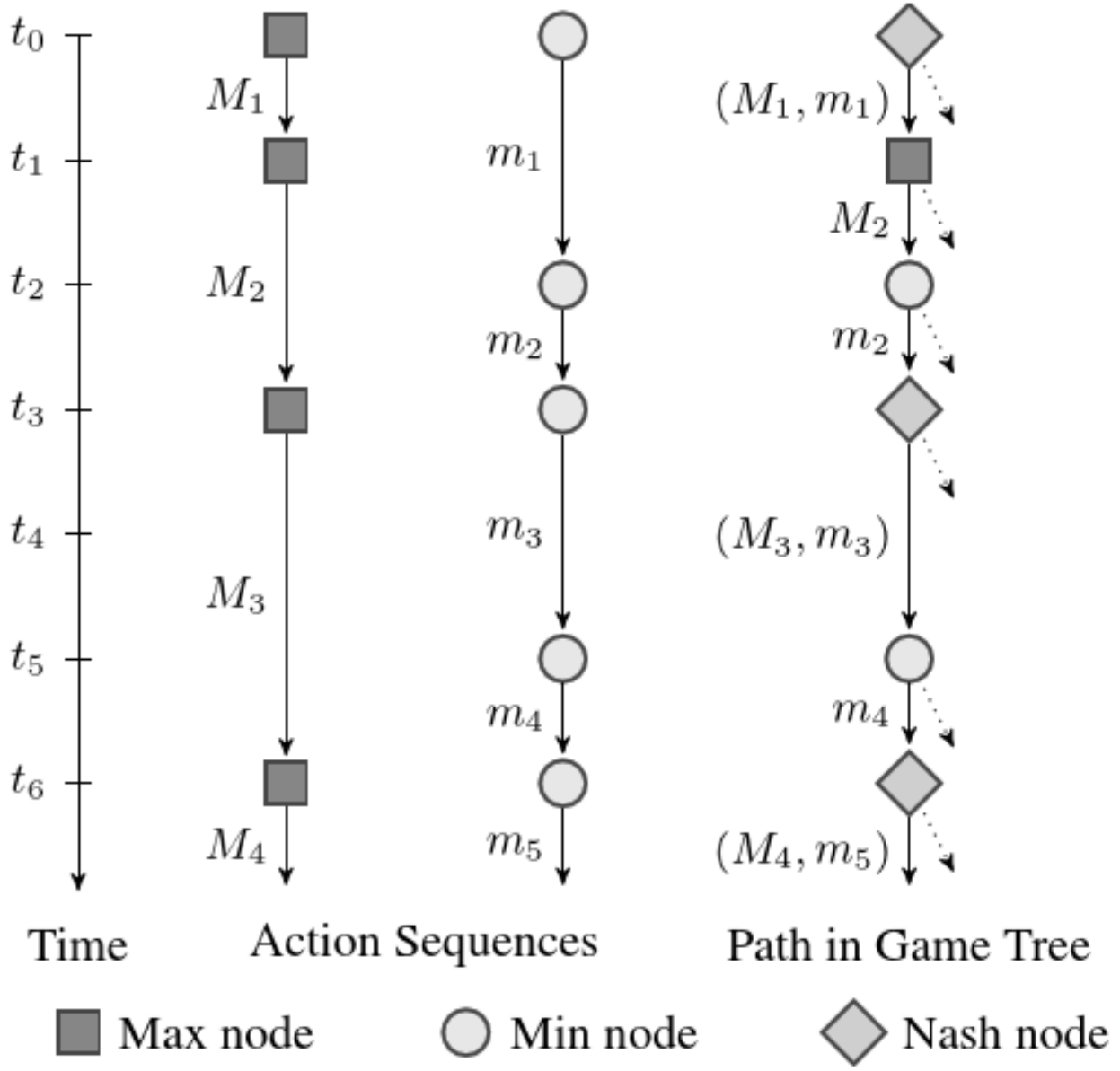
Combat AI Challenges

- Hard combinatorial optimization problem (PSPACE hard even without unit motion, Furtak and Buro, AIIDE 2010)
- Real-time, simult. moves
- Multi-unit control: huge branching factors
- State evaluation
- No access to game engine source code

SparCraft Combat Simulator



Alpha-Beta Considering Durations (ABCD) Algorithm (Churchill et al. AIIDE 2012)



Approximations and Heuristics

- Transform stacked matrix games into serialized perfect information game
- Use ABCD or UCTCD
- Reduce action sets by using scripts
- Use static evaluations or scripted playouts to evaluate leaf nodes

Portfolio Greedy Search
(Churchill and Buro, CIG 2013)

- Use scripts to propose individual unit's actions
  - e.g. "attack closest unit", "attack weakest unit"
- Assign default actions to units
- Avoid overkill
- **Improve unit actions greedily** in turn by evaluating current unit's script actions using playouts
- Iterate until time runs out

Soundly defeats ABCD and UCTCD for large unit groups (32+ units)

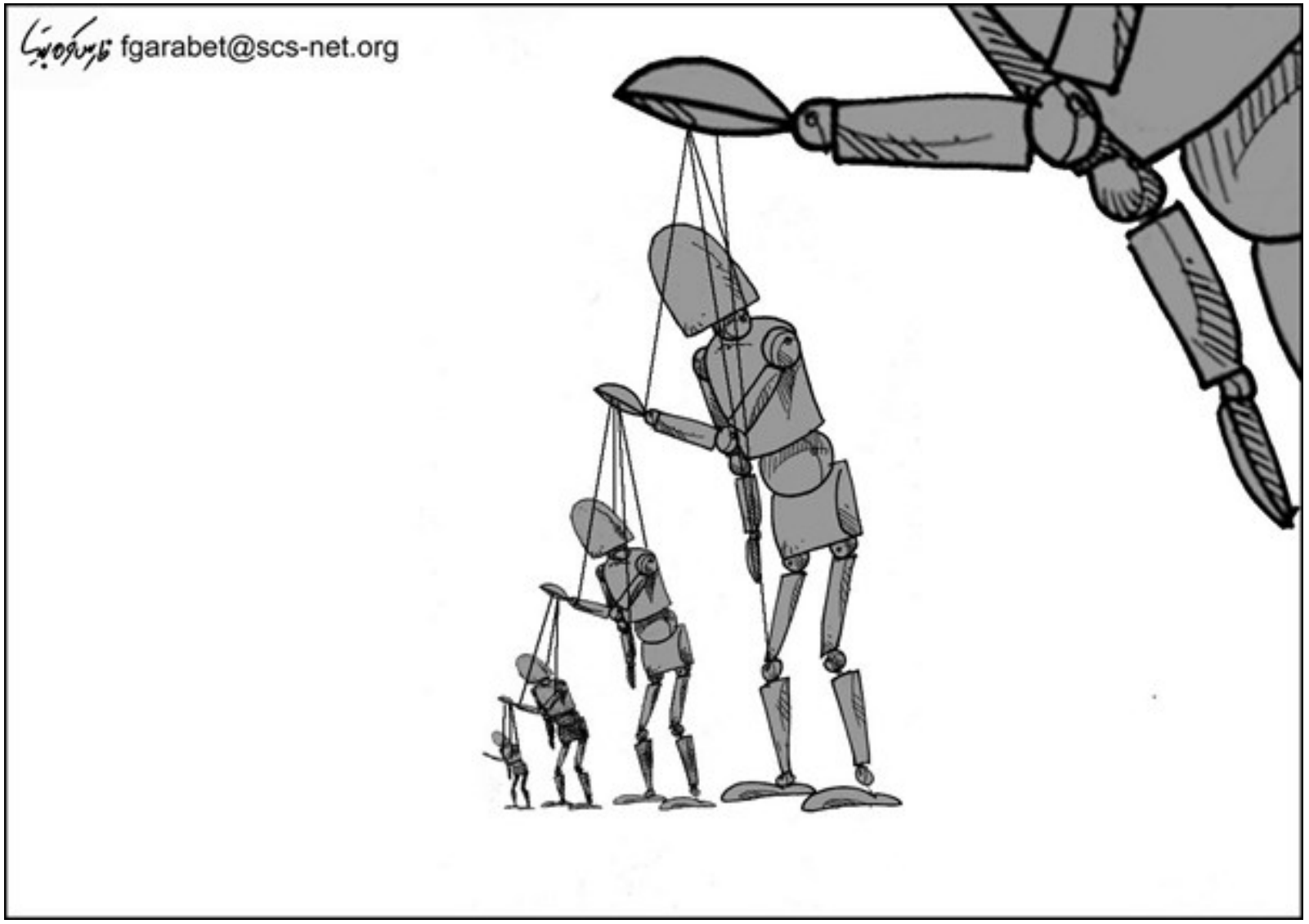
Combat Search in Practice

- Simulated combat results used for **attack / retreat decision** in StarCraft bots
- Not yet used for actual combat – simulation doesn’t handle motion and unit collisions well enough
- Work in progress ...

Other Selected Combat AI Work

- Attrition Games Played on Graphs
  (Furtak and Buro, AIIDE 2010)
- Kiting using Influence Maps
  (Uriarte and Ontañón, AIIDE 2012)
- The Combinatorial Multi-Armed Bandit Problem
  (Ontañón, AIIDE 2013)

Variation 4: Master of Puppets



Heuristic Search for Full RTS Game

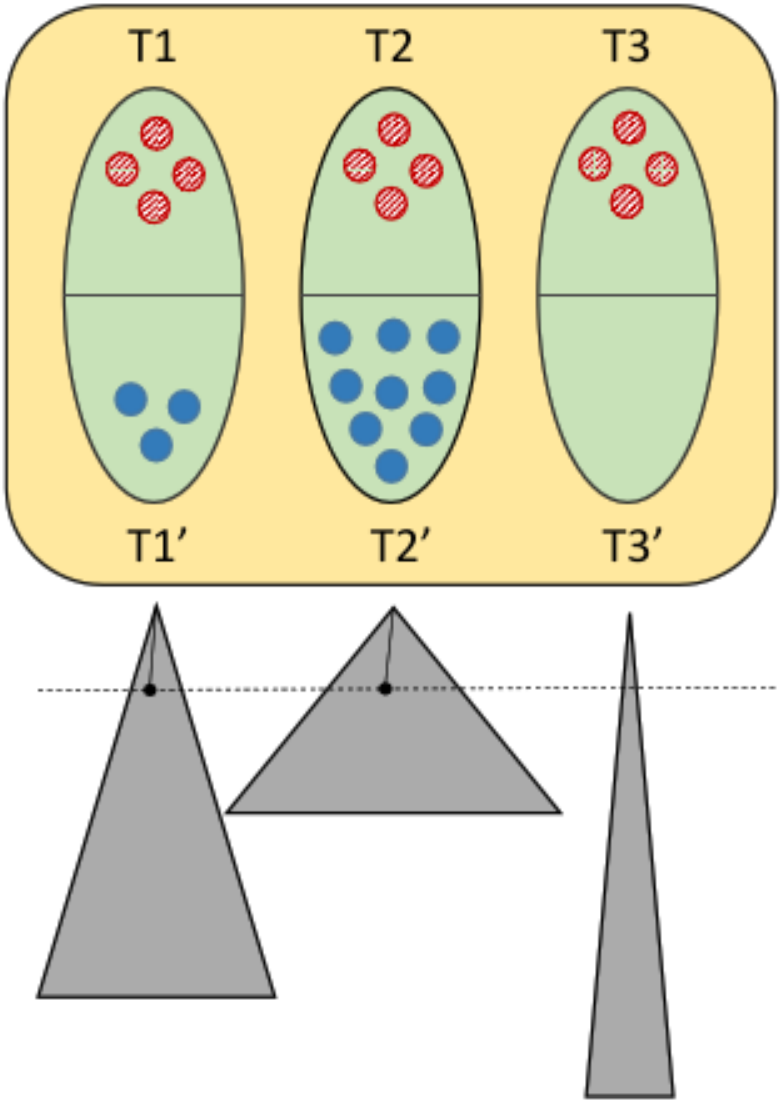
How to apply conventional adversarial search methods to real-time games with huge action sets and large depths ?

Idea: Hierarchical Strategy Decomposition

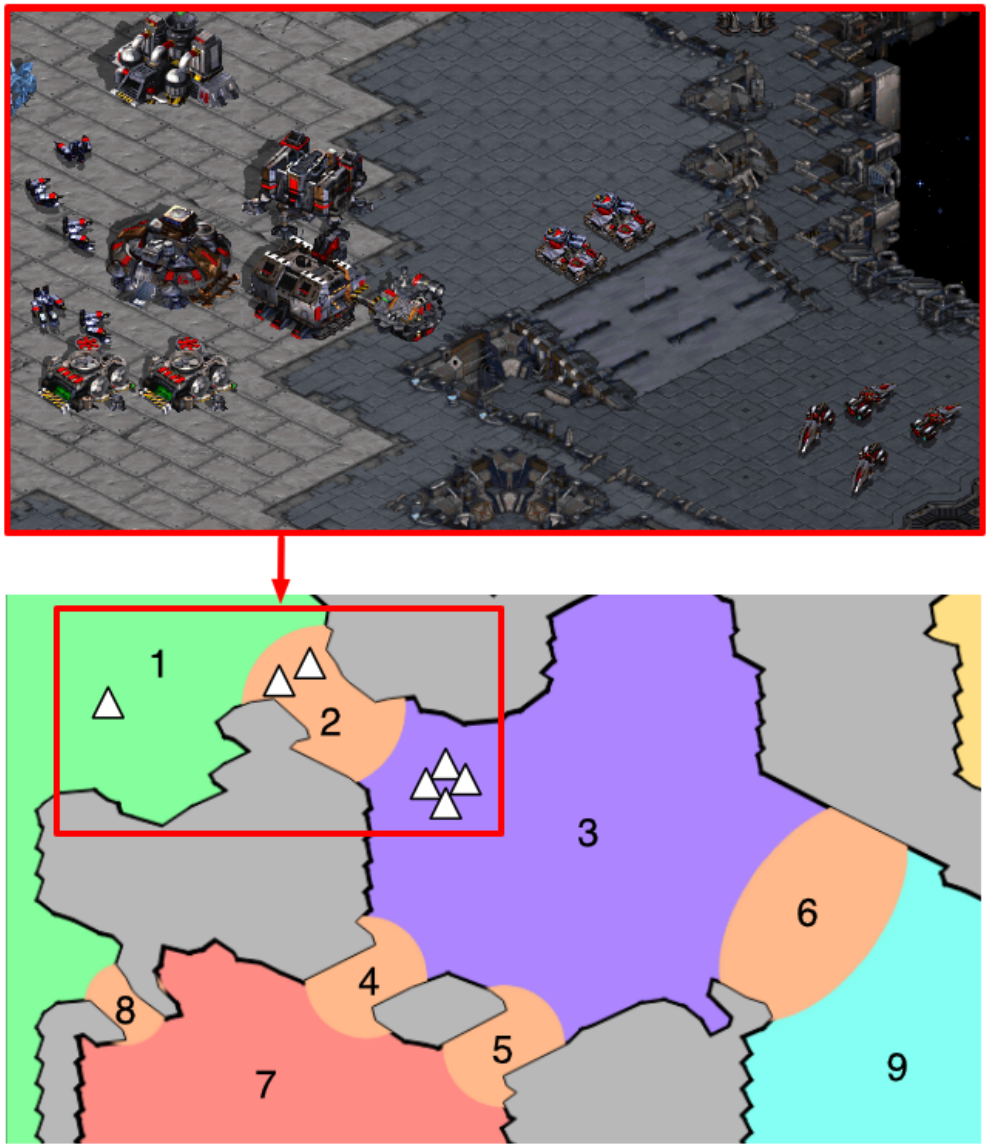
similar to military command & control

Hierarchical Adversarial Search

(Stanescu et al., AIIDE 2014)

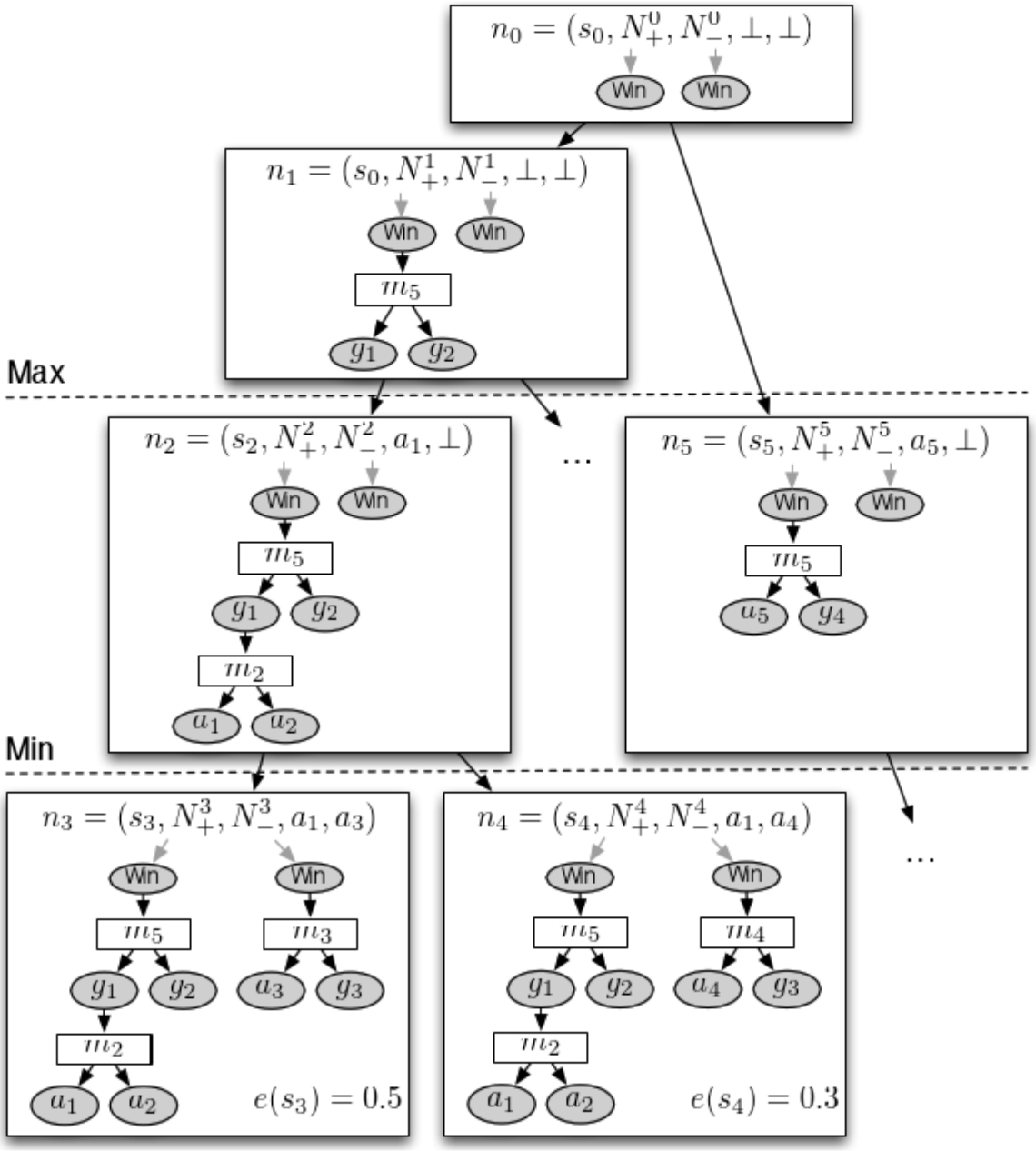


Game-Tree Search Over High-Level Game States (Uriarte and Ontañón, AIIDE 2014)



Adversarial HTNs

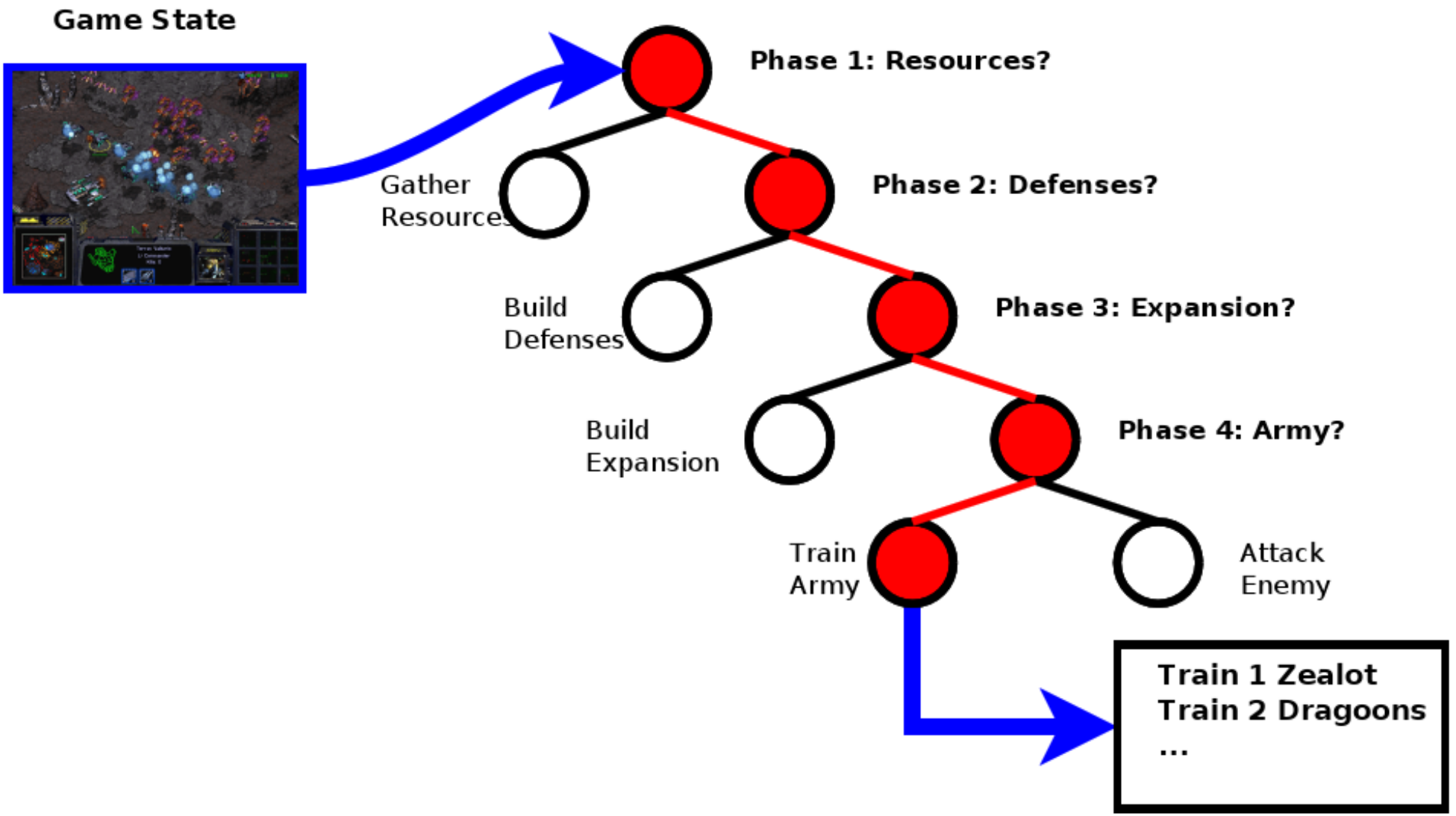
(Ontañón and Buro, IJCAI 2015)



“Puppet Search”
(Barriga et al., AIIDE 2015)

- Consider full-game scripts (“puppets”) with choice points (“joints”)
  - “Should I expand now?”
  - “Should I attack now?”
  - “Should I upgrade my weapons now?”
- Those choices are usually hard-coded
- **Idea: Let search decide what course of action is best**
- I.e., run MiniMax search on transformed game

Script in Form of Decision Tree



Puppet Search Discussion

- Imitates human thought process
- Scripts reduce branching factors drastically
- Allows to look far ahead

- Requires scripts to cover opponent strategies to be effective
- How to write scripts? What are effective choice points?

Puppet Search in Practice

- Applied Puppet Search to 8 by 8 micro-RTS games
- Initially chose 4 rush scripts with one choice point at the top deciding what script to execute
- Puppet Search won more games against 6 StarCraft bots than each individual script
- Work in progress:
  - Add choice points for middle game
  - Scale to bigger games – eventually StarCraft

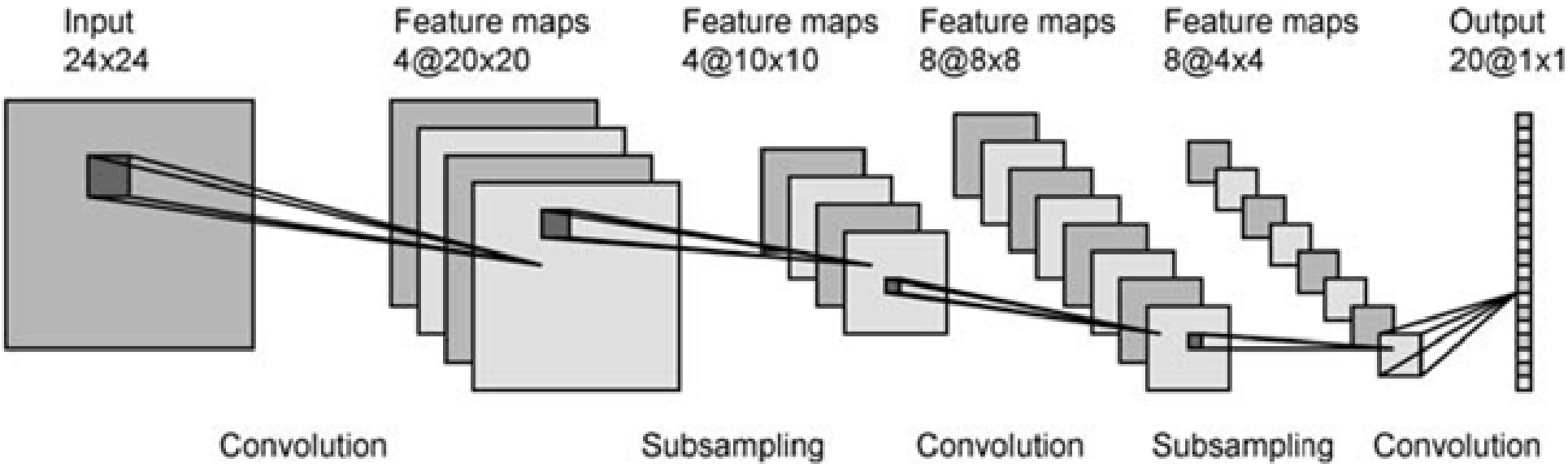
Variation 5: Ω-StarCraft ?

March 15, 2016
Lee Sedol (9-dan) loses 1:4

TPU Racks

What if ...

… CNNs could also play StarCraft ?



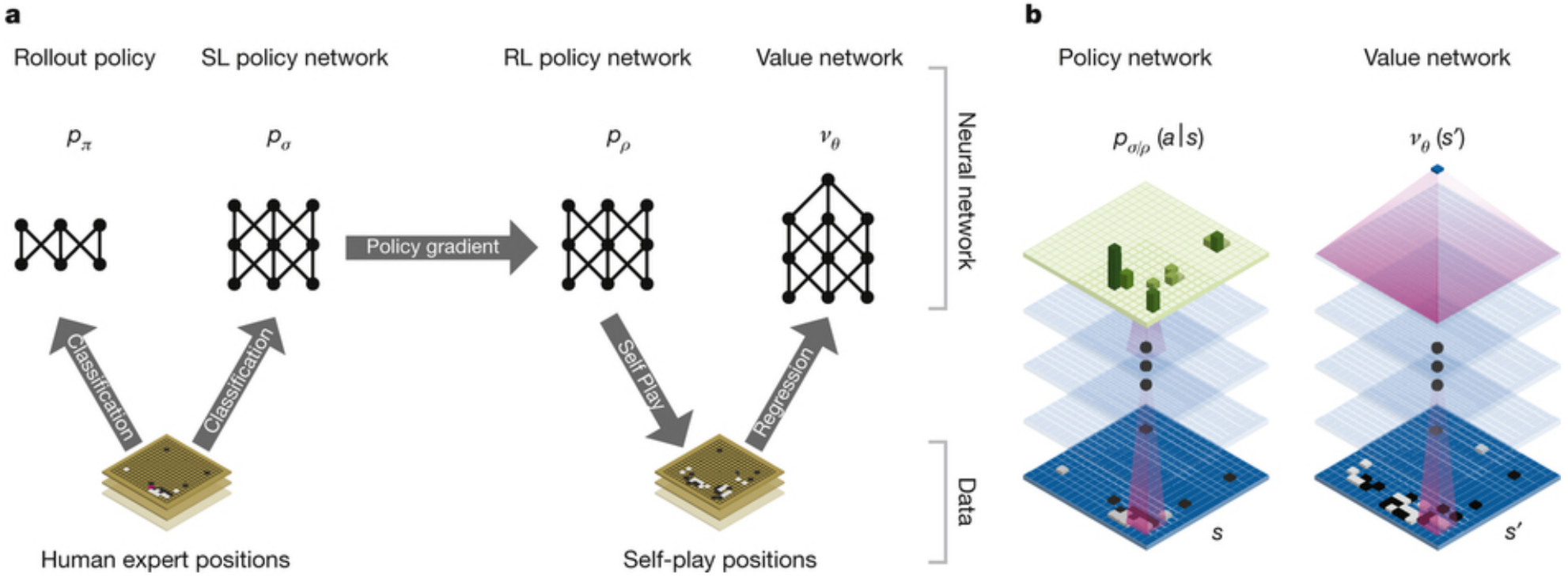
Go

- ≤ 361 moves
- ~ 300 turns
- Perfect information

StarCraft

- max ≥ 4**100 moves
- Often ≥ 28800 frames
- Imperfect information

CNNs for RTS Games



Constructing value networks not that hard (Stanescu et al., CIG 2016)

Policy networks and search are challenging

We are working on it … and Google, perhaps, too

Free Software on GitHub
written by Dave Churchill

- UAlbertaBot
  - Modular StarCraft bot that plays all three races
  - Basis of several other StarCraft bots
  - Also used in game AI courses
  - BOSS (Build Order Search System)
  - SparCraft (Combat Simulator)
    - Scripts / Alpha-Beta / UCT / Greedy Portfolio
- StarCraft Tournament Manager

Conclusion

- Abstraction and search in RTS games shown to be effective for sub-problems
- Macro policies still mostly scripted
- Some progress in high-level search

Dethroning world-class players may take a while …


KEEP
CALM
AND
ASK ME
QUESTIONS