REINFORCEMENT LEARNING IN GRAPHICAL GAMES

By Cosmin Hanganu, Priyanka Deo, Ben Campbell, and Larry Muhlstein
Outline

- Intro to RL
- Scratch
- Harry Potter “Snitch” Game
- Copter Game
- Q Learning
- Optimizing the Algorithm
- Conclusion
Intro to Reinforcement Learning

Diagram:
- Agent
- Environment
- State $s_t$, Reward $r_t$, Action $a_t$, Next State $s_{t+1}$

Diagram illustrates the interaction between the agent and the environment, showing the flow of state, reward, and action.
State: Hand not in contact with the stove

Reward: None

New Reward: Hurt hand

New State: Hand in contact with the stove

Action: Touching the stove
Scratch

- Developed at MIT in 2007
- Very easy to work with
- Detachable blocks of code can be customized very quickly and while code is running
- Used primarily in schools and other educational organizations
when I clicked
delete all of Hits
delete all of Misses
set explore to 1
set Sum_Hits to 0
set Sum_Miss to 0
set Avg_Hits to 0
set Avg_Miss to 0

forever

set distance to abs(position of seeker - position of seeker)

if pick random 1 to 10 = explore or explore = 0 and distance > Avg_Hits - 10 and distance

else

broadcast grab and wait

if grabbed = 1

insert distance at abs of Hits

set Sum_Hits to Sum_Hits + abs of item of Hits

set Avg_Hits to sum Hits / length of Hits

else

insert distance at abs of Misses

set Sum_Miss to Sum_Miss + abs of item of Misses

set Avg_Miss to sum Miss / length of Misses

if length of Hits = 3

set explore to 0
Outline

- Intro to RL
- Scratch
- Harry Potter “Snitch” Game
- Copter Game
- Q Learning
- Optimizing the Algorithm
- Conclusion
Snitch
Outline

- Intro to RL
- Scratch
- Harry Potter “Snitch” Game
- Copter Game
- Q Learning
- Optimizing the Algorithm
- Conclusion
Q-Learning

\[ Q[s, a] := (1 - \alpha)Q[s, a] + \alpha \left( r + \gamma \max_{a'} Q[s', a'] \right) \]

Q= \(1- \alpha\)*known experience + \(\alpha\)*discounted new experience

- **Q[s,a]** : Q-value of state-action space
- **\(\alpha\)**: learning rate
- **\(\gamma\)**: discount factor
- **r**: reward
Optimization
Redefined wall relative Y values to wall absolute Y values

Relative States
Optimization

Absolute states allow the agent to determine helicopter and wall position independently.
Optimization

- Adjusting Reward Function
  - Floor/Ceiling = -20
  - Wall = -10
  - Initial value of State = 1

- Cooling Alpha
  - Agent at first values new experience, then bases actions on old experience.
Thick Wall vs. Thin Wall
Conclusion

- Proper state space and action space definitions are critical
- Computation time vs. learning accuracy tradeoff
- RL can be applied innovatively in several fields with many languages
Acknowledgements

- Carlos Diuk, Thomas Walsh, Prof. Michael Littman
- Jameslevi Maximilian Schmidt
- Dean Donald M. Brown and Blase Ur
- Dr. Yogesh Jaluria (Outgoing Interim Dean) and Dr. Thomas Farris (Dean)
- NJ Governor’s School Board of Overseers

Sponsors:
- Rutgers University
- Rutgers University School of Engineering
- Motorola Foundation
- Morgan Stanley
- PSEG
- Silver Line Building Products
- The families of 2001-2008 program alumni
Questions and Comments

- Feel free to ask!