CS 440: Introduction to Artificial Intelligence

Lecture 22, Nov 30, 2015

Matthew Stone

December 1, 2015

Games–Recap

A *game* is an environment with rewards that depend on the action of more than one agent.

- Games with turns
  tic–tac–toe, checkers, chess, backgammon, go
- Games with simultaneous moves
  Rock–paper–scissors, prisoner's dilemma, sealed-bid auctions
- Approaches in the two cases are different

Games with Turns

Intuitively like planning

- Map out the future
- Anticipate that you will make good choices
- Expect that your opponent will make good choices
- Work backwards to what you should do now

Games with Turns

Wind up with a strategy

- Describes optimal play
- Given optimal (or possible) play by opponent

Games with simultaneous play

Intuitively like solving an equation

- Your move depends on your opponent's move
- Their move depends on yours
- You choose them simultaneously
- Good strategies "balance" the decisions

Example: Rock–paper–scissors

R and C choose actions jointly. C gets these payoffs:

		C		
		rock	paper	scissors
	rock	0	1	-1
R	paper	-1	0	1
	scissors	1	-1	0

Example: Rock–paper–scissors

Need to be unpredictable

- If R knows what C is going to do, R can win
- If C knows what R is going to do, C can win

If both guess any move randomly with probability $\frac{1}{3}$, neither can exploit the other

Different needs for AI techniques

Games with turns

- manage large search spaces
- develop good heuristics and approximations
- (only secondarily) learn specifically about opponent

Games with simultaneous play

- learn specifically about opponent
- (less so) develop good heuristics and approximations
- (only secondarily) manage search

Simple Games

Variant of search problem

- Two players, A and B
- Set of states S
- Initial state s_I
- Possible actions in each state
  Specifies which player moves, if any
  Players alternate
- Transition model:
  Takes state and action and gives new state
- Outcome:
  Says the score in the final state

Simple Games

Variant of search problem

- Like planning search
  Need to systematically explore states
  (want to find a win, in this case)
  work backwards from outcome to what to do next
- Twist: need to account for decisions we do not control
- Idea: assume other agent plays optimally
  Change objectives and use your own search to predict their actions

Minimax

Two players:

- A maximizes score
- B minimizes the score

Predict play and outcome recursively

Minimax

If s is a final state
stop,
outcome is the score you get
(just like decision models)

Minimax

If s is a decision for A
compute the policies and scores for the subtrees at s
take policy that gives you the best score
(just like decision models)

Minimax

If s is a decision for B
compute the policies and scores for the subtrees at s
take the policy that gives you the **worst** score
(B wants to win!)

Discussion

Minimax—pros

- Guaranteed to find winning strategy
  provided one exists, and
  algorithm converges
- Finds values for all nodes in the tree
  can play best response to any opponent

Minimax—cons

- Number of states too large to search in practice

Alpha-beta pruning

Improve minimax by discarding irrelevant possibilities

Intuitive example

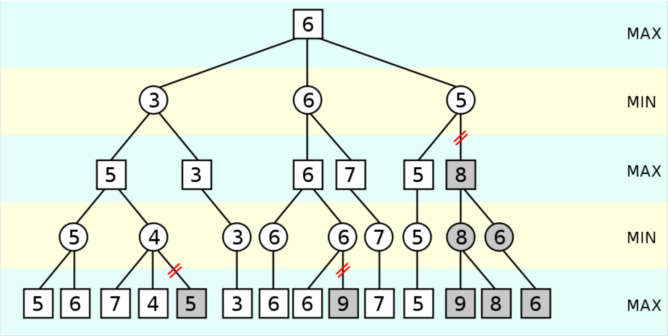

Image from Wikipedia

Alpha-beta pruning

Improve minimax by discarding irrelevant possibilities

- Keep track of lower bound α
  interested in values above α
- Keep track of upper bound β
  interested in values below β
- Discard subtrees if out of bounds

Alpha-beta pruning

Max step

```
for each child of node
    alpha := max(alpha,
                 minstep(child, alpha, beta))
    if beta <= alpha
        break
return alpha
```

Alpha-beta pruning

Min step

```
for each child of node
    beta := min(beta,
                maxstep(child, alpha, beta))
    if beta <= alpha
        break
return beta
```

Heuristics

Evaluation functions

- Cut off search at some depth
- Estimate values there
  (even though game is not done)
- Common to use features and function approximation
- Opportunity for learning

Another Example—Cooperative Games

Driving game

		C left	C right
R	left	1,1	0,0
	right	0,0	1,1

Notation: Payoffs are $U_A(s_R, s_C)$ – the utility for agent A of the outcome where R plays s_R and C plays s_C.

Equilibrium

Intuitively—fixed point for choice:

- R wouldn't change what R would do, knowing that C is going to play as C does
- C wouldn't change what C would do, knowing that R is going to play as R does

Strategy

Two notions

- Pure strategy: choice of one course of action by a player
- Mixed strategy: probability distribution over courses of action, (chosen randomly with specified probabilities)

Best response

- Pure strategy s_1^* is a *best response* for player 1 to pure strategy s_2 for player 2 if

$$U_1(s_1^*, s_2) \geq U_1(s_1, s_2)$$

for all alternative (pure) strategies s_1

Example

		C	
		left	right
R	left	1,1	0,0
	right	0,0	1,1

Strategy left for R is best response to left for C
Strategy right for R is best response to right for C
Strategy left for C is best response to left for R
Strategy right for C is best response to right for R

Best response

- Mixed strategy p_1^* is a *best response* for player 1 to mixed strategy p_2 for player 2 if

$$\sum U_1(s_1, s_2) \times p_1^*(s_1) \times p_2(s_2) \geq \sum U_1(s_1, s_2) \times p_1(s_1) \times p_2(s_2)$$

for all alternative mixed strategies p_1

Example

Best response for R when C plays .5 paper and .5 scissors

		C rock	paper	scissors
	rock	0	1	-1
R	paper	-1	0	1
	scissors	1	-1	0

Example

Best response for R when C plays .5 paper and .5 scissors

		C		
		rock	paper	scissors
	rock	0	1	-1
R	paper	-1	0	1
	scissors	1	-1	0

always play scissors

Equilibrium

A pair of strategies (p_1^*, p_2^*) is an equlibrium just in case

- p_1^* is best response for player 1 to p_2^*
- p_2^* is best response for player 2 to p_1^*

Example

		C	
		left	right
R	left	1,1	0,0
	right	0,0	1,1

Strategy left for R is best response to left for C
Strategy right for R is best response to right for C
Strategy left for C is best response to left for R
Strategy right for C is best response to right for R

Example

		C left	C right
R	left	1,1	0,0
	right	0,0	1,1

Equilibria include

- *R* plays right and *C* plays right
- *R* plays left and *C* plays left

Zero Sum Games

		C	
		heads	tails
R	heads	1,-1	-1,1
	tails	-1,1	1,-1

Zero Sum Games

		C	
		heads	tails
R	heads	1,-1	-1,1
	tails	-1,1	1,-1

Equilibrium is a mixed strategy

- *R* and *C* both play heads half the time and tails half the time

Observation

		C	
		heads	tails
R	heads	1,-1	-1,1
	tails	-1,1	1,-1

R chooses both h and t

- R is rational, so both h and t must look good
- Suppose C plays h with probability q
- R gets $-q + (1 - q) = 1 - 2q$ from h
- R gets $q - (1 - q) = 2q - 1$ from t
- Must be same payoff: $1 - 2q = 2q - 1 \Rightarrow q = 0.5$