EXPECTIMAX SEARCH

CSE 511A: Introduction to Artificial Intelligence

Some content and images are from slides created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley.
All CS188 materials are available at http://ai.berkeley.edu.

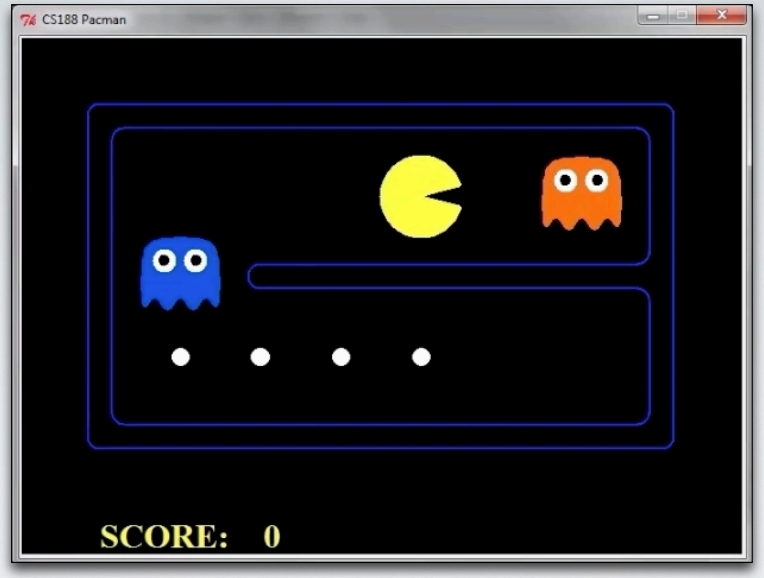
ALPHA-BETA SEARCH

	Minimax	alpha-beta
Correct the solution it finds is optimal	Yes	Yes
max nodes in memory		
Time Complexity max nodes generated	$O(b^m)$	$O(b^m)$

Key assumption in minimax and alpha-beta search:
Opponent will always take optimal actions

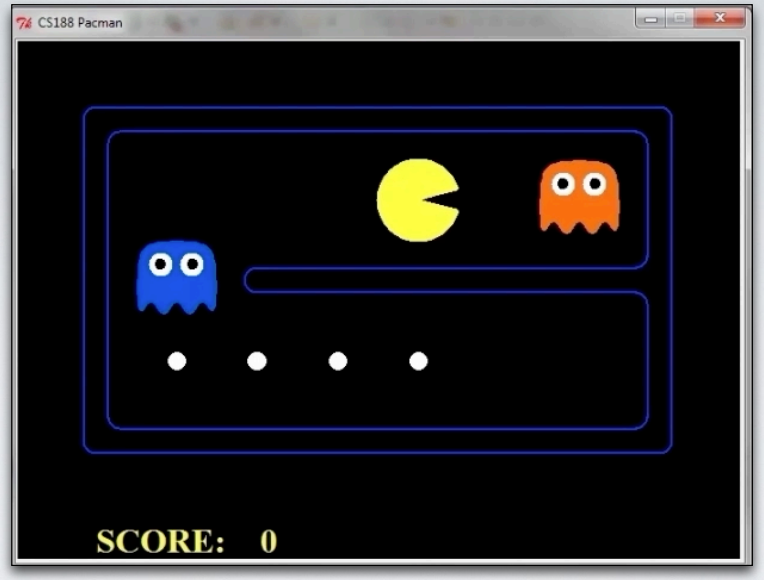
What if the opponent some times make mistakes?

branching factor b
depth of the goal d
depth of tree m



Ghosts moving optimally

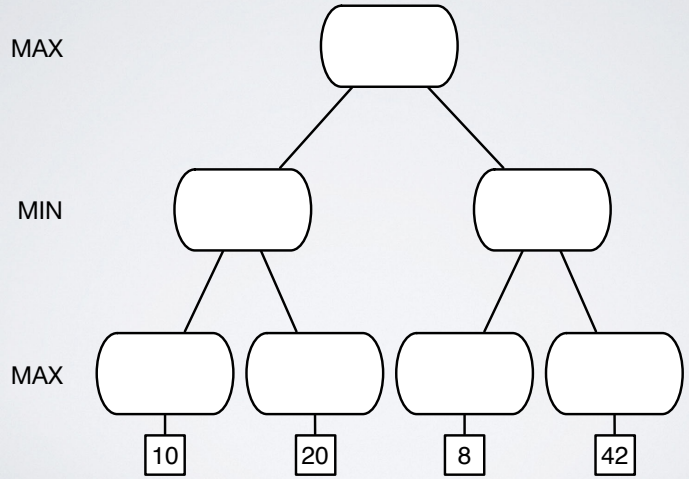
Videos are from slides created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. All CS188 materials are available at http://ai.berkeley.edu.



Ghosts moving randomly

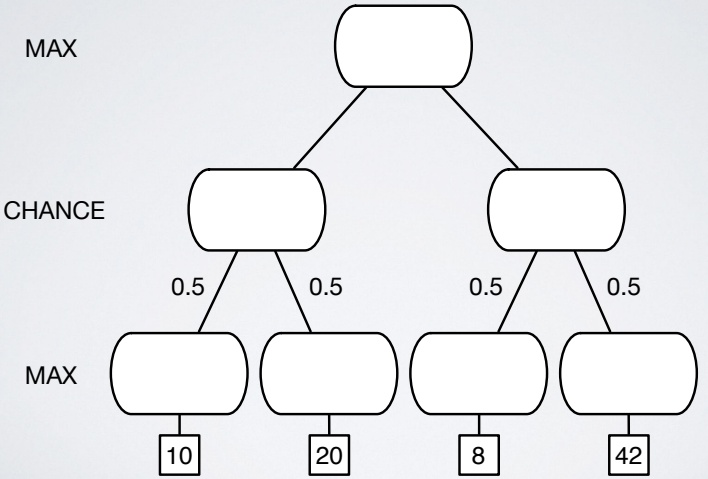
Videos are from slides created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. All CS188 materials are available at http://ai.berkeley.edu.

WORST CASE VS AVERAGE CASE



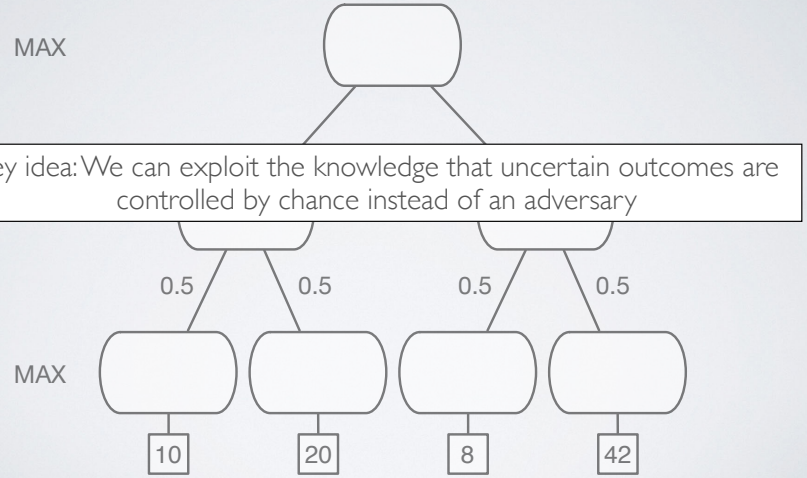
5

WORST CASE VS AVERAGE CASE



6

WORST CASE VS AVERAGE CASE



7

EXPECTIMAX SEARCH

- Previously: Values reflect worst case outcomes (minimax)
- Now: Values reflect average case outcomes (expectimax)

- Expectimax search:
  - Max nodes just like in minimax search
  - Chance nodes instead of min nodes in minimax search
    - Outcomes are uncertain
    - Calculate their expected utilities (i.e., utilities weighted by their likelihood)

8

EXPECTIMAX SEARCH

```
def value(state):
    if the state is a terminal state: return the state's utility
    if the next agent is MAX: return max-value(state)
    if the next agent is MIN: return exp-value(state)
```

```
def max-value(state):
    initialize v = -∞
    for each successor of state:
        v = max(v, exp-value(successor))
    return v
```

```
def exp-value(state):
    initialize v = 0
    for each successor of state:
        p = probability(successor)
        v += p * max-value(successor)
    return v
```

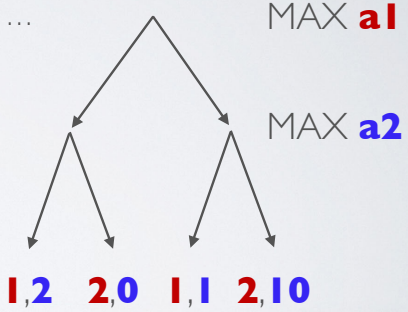

EXPECTIMAX SEARCH

	Minimax	alpha-beta	Expectimax
Correct the solution it finds is optimal	Yes	Yes	Yes
Complete it terminates	Yes	Yes	Yes
Space Complexity max nodes in memory	$O(bm)$	$O(bm)$	$O(bm)$
Time Complexity max nodes generated	$O(b^m)$	$O(b^m)$	$O(b^m)$

branching factor b
depth of the goal d
depth of tree m

OTHER GAME TYPES

- Expectiminimax search:
  - MAX, CHANCE, MIN, CHANCE, MAX, …
  - Used to model games like backgammon
- Modeling multi-agent problems
  - e.g., two Pacmans solving eat-all-dots problem
  - Utilities are now vectors, with each element representing the utility of one agent



OTHER GAME TYPES

- Expectiminimax search:
  - MAX, CHANCE, MIN, CHANCE, MAX, …
- e.g., two Pacmans solving eat-all-dots problem
- Utilities are now vectors, with each element representing the utility of one agent

How do you model problems where you can take multiple actions in one turn, and the number of actions depend on some rule?