Artificial Intelligence

Lecture 13: Game Theory

Slides, RN chap. 6.1-5 adversarial search

RN chap 17.2 non-cooperative game theory

CS4375 Artificial Intelligence Techniques
Matthijs Spaan (based on slides by Catholijn Jonker)
October 17, 2023

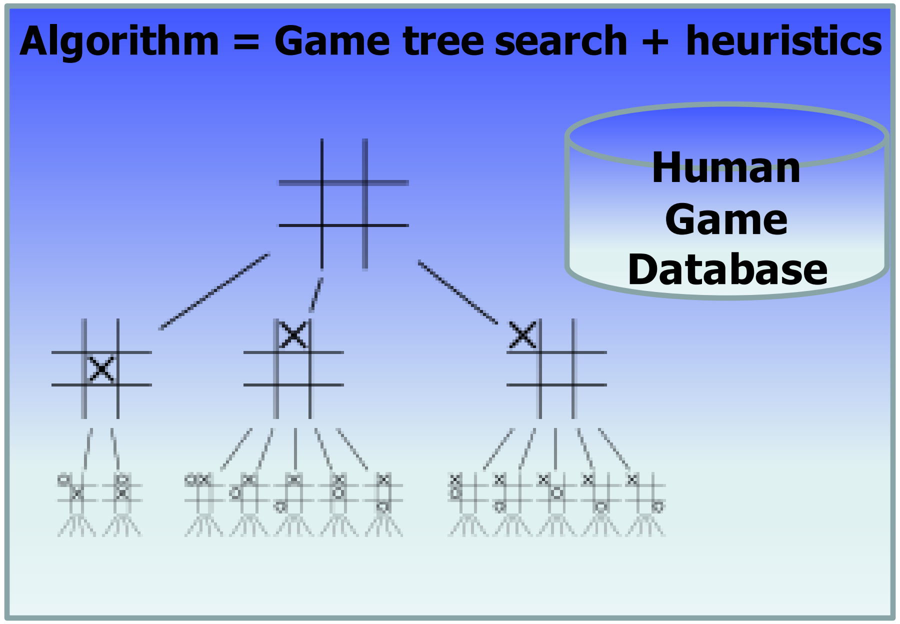
Outline

- Games
- Optimal decisions (Perfect play)
    - minimax
    - α-β pruning
- Imperfect, real-time decisions
    - Resource limits and approximate evaluation
    - Games of chance
    - Games of imperfect information
- Game Theory

AI: man versus machine

GAMES

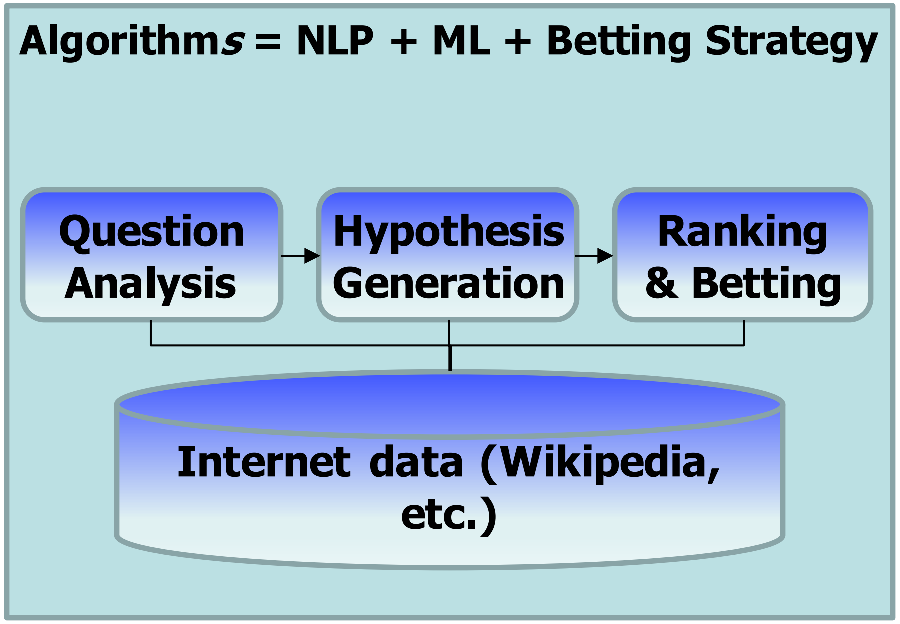
Chess (IBM Deep Blue 1997)



No human has won even a single game against a sufficiently strong computer under tournament conditions since 2005.

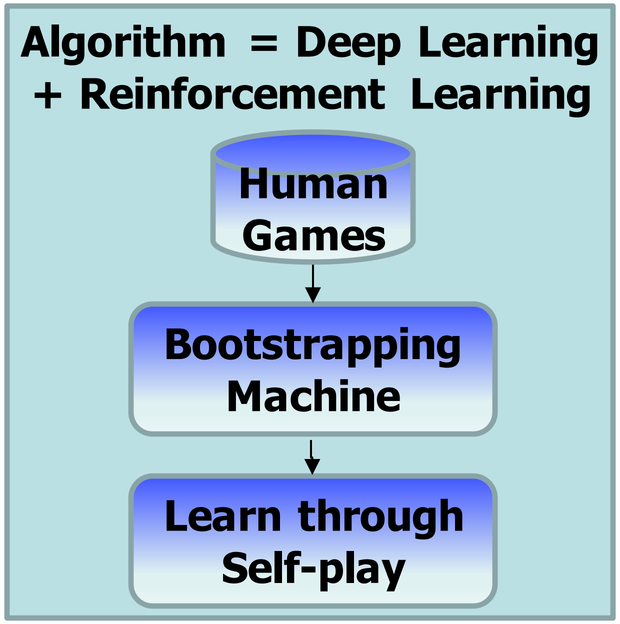
1994 first game lost by Kasparov: http://www.youtube.com/watch?v=3EQA679DFRg.

Jeopardy (IBM Watson 2011)


Algorithms = NLP + ML + Betting Strategy
Question Analysis
Hypothesis Generation
Ranking & Betting
Internet data (Wikipedia, etc.)

Humans beaten at a game they were supposed by most to outperform machines.

Go (Google's AlphaGo 2016)



Go is more difficult than chess for a machine because the game has *less structure* and the space of options is *much larger*.

TU Delft

Have Games been "Solved"?

Checkers

1994: Jonathan Schaeffer's Chinook ended 40-year-reign of human world champion Marion Tinsley. Used a precomputed endgame database defining perfect play for all positions involving 8 or fewer pieces.

2007: solved: http://www.newscientist.com/article/dn12296-checkers-solved-after-years-of-number-crunching.html#.VHzc8_mG98E

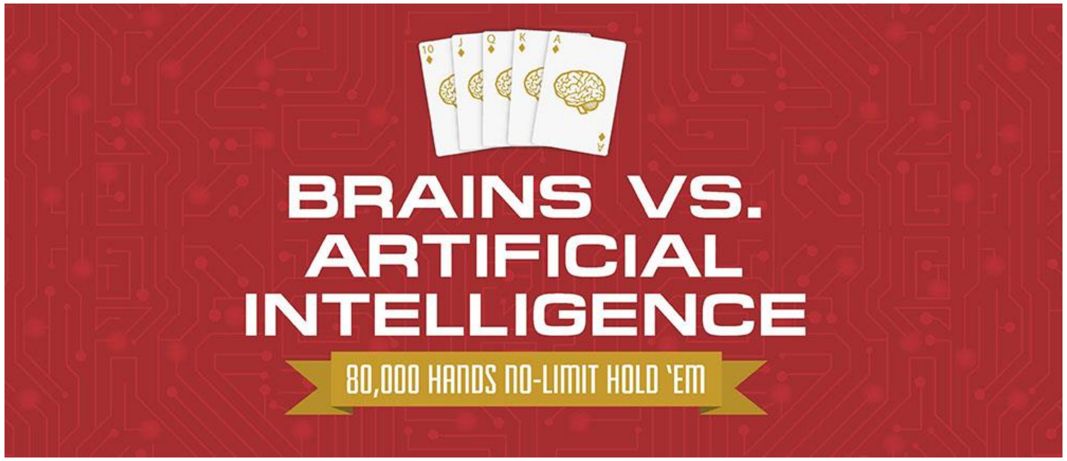
Othello

human champions refuse to compete against computers, who are **too good**.

…

Have Games been "Solved"?

Machines beat humans at **perfect** information games but not yet at **imperfect** information games.


BRAINS VS. ARTIFICIAL INTELLIGENCE
80,000 HANDS NO-LIMIT HOLD 'EM

CMU + Facebook built AI Pluribus that defeated Darren Elias (holder of most World Poker Tour titles), and Chris "Jesus" Ferguson, (won 6 World Series of Poker events) in six-player no-limit Texas hold'em poker, the world's most popular form of poker.

Noam Brown, Tuomas Sandholm. **Superhuman AI for multiplayer poker**. *Science*, 2019; eaay2400 DOI: 10.1126/science.aay2400

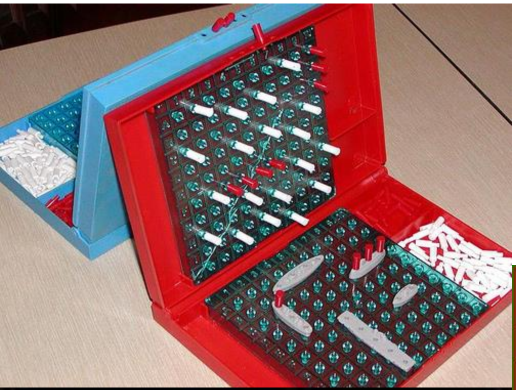
Pluribus strength and surprises

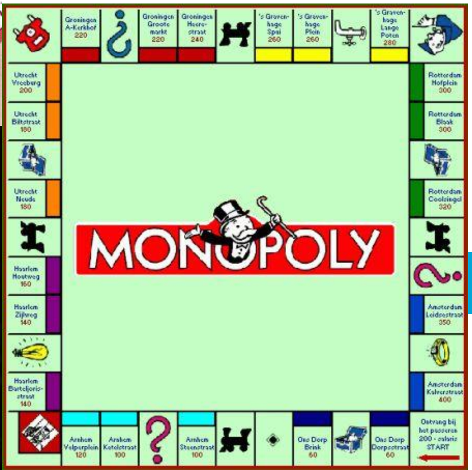
https://www.sciencedaily.com/releases/2019/07/190711141343.htm

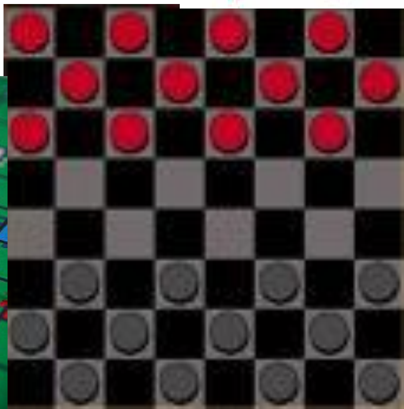
- Successful "donk betting" -- that is, ending one round with a call but then starting the next round with a bet. A weak move for humans. But Pluribus placed donk bets far more often than the professionals it defeated.
- Use of mixed strategies: -- to do this in a perfectly random way and to do so consistently. Most people just can't.
- a solid win with statistical significance, playing some of the best players in the world."
- It made several plays that humans simply are not making at all, especially relating to its bet sizing.

Types of Games

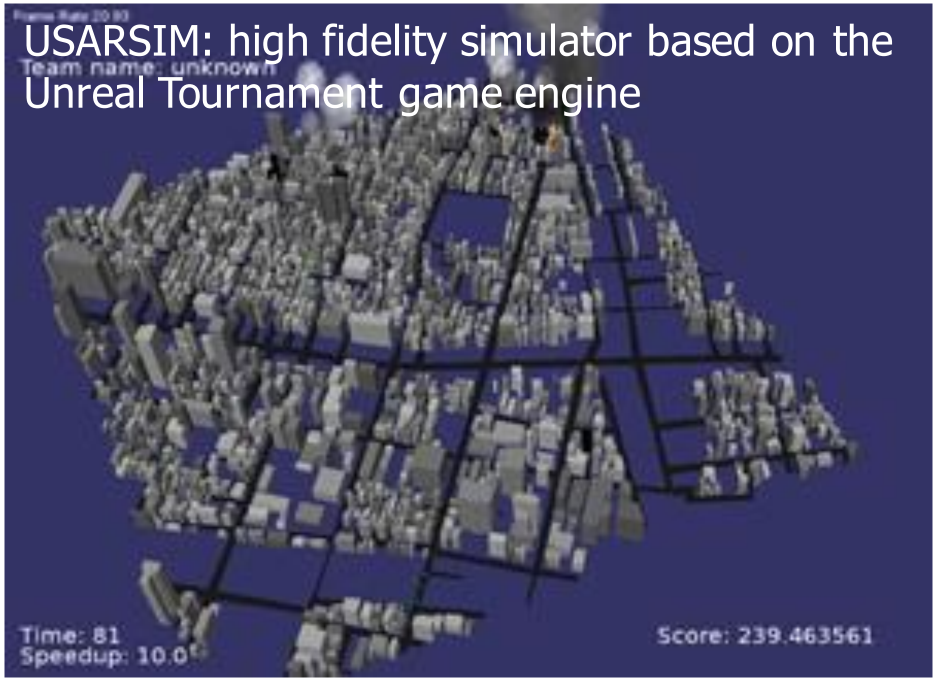
	deterministic	chance
perfect information	chess, checkers, go, othello	backgammon monopoly
imperfect information	battleships, blind tictactoe	bridge, poker, scrabble nuclear war







Serious Gaming


USARSIM: high fidelity simulator based on the Unreal Tournament game engine

Traffic simulator
Fire Simulator
Civilian Simulator

See: http://www.robocuprescue.org/simleagues.html

Real-Time Strategy Games: StarCraft

State Spaces and Game Trees

Game	State-Space	Game-Tree
Checkers	10^{18}	10^{31}
Reversi	10^{28}	10^{58}
Chess	10^{46}	10^{123}
Risk (200)	10^{47}	10^{2350}
Shogi	10^{71}	10^{226}
Risk (1000)	10^{78}	10^{2350}
Go (19 x 19)	10^{172}	10^{360}

Challenge: writing effective game-playing agents for multi-player games with **probabilistic events**.

Problem: although games like **Risk** are conceptually simpler than **Chess** the state-space is actually **bigger**.

Imperfect Information

Tackle whole problem at once
For poker this includes
- Betting strategy
- Opponent modeling
- Learning
- Performance evaluation

Challenge: writing effective game-playing agents for multi-player games with **incomplete information**.

Problem: although games like **Poker** are conceptually simpler than **Chess** the current state is **unknown**.

History of AI Gaming Research Ideas

We need to deal with "unpredictable" opponents, so need a strategy that specifies a reply for every possible opponent move, and deal with time limits, therefore must approximate best move.

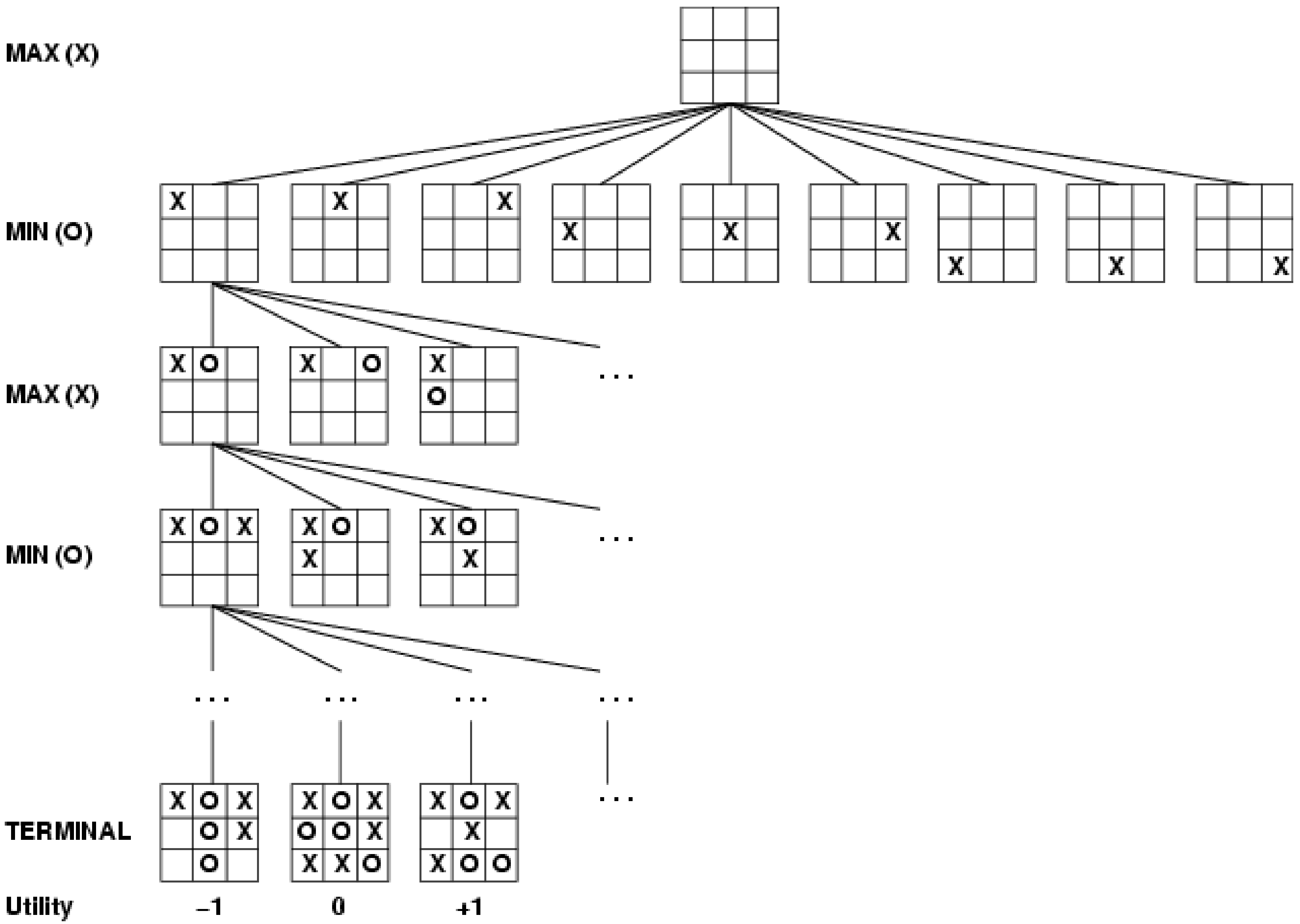
- Computer considers possible lines of play (Babbage, 1846)
- Algorithm for perfect play (Zermelo, 1912; Von Neumann, 1944)
- Finite horizon, approximate evaluation (Zuse, 1945; Wiener, 1948; Shannon, 1950)
- First chess program (Turing, 1951)
- Machine learning to improve evaluation accuracy (Samuel, 1952–57)
- Pruning to allow deeper search (McCarthy, 1956)
- Deeplearning (2010 and onward)

Minimax and alpha-beta pruning

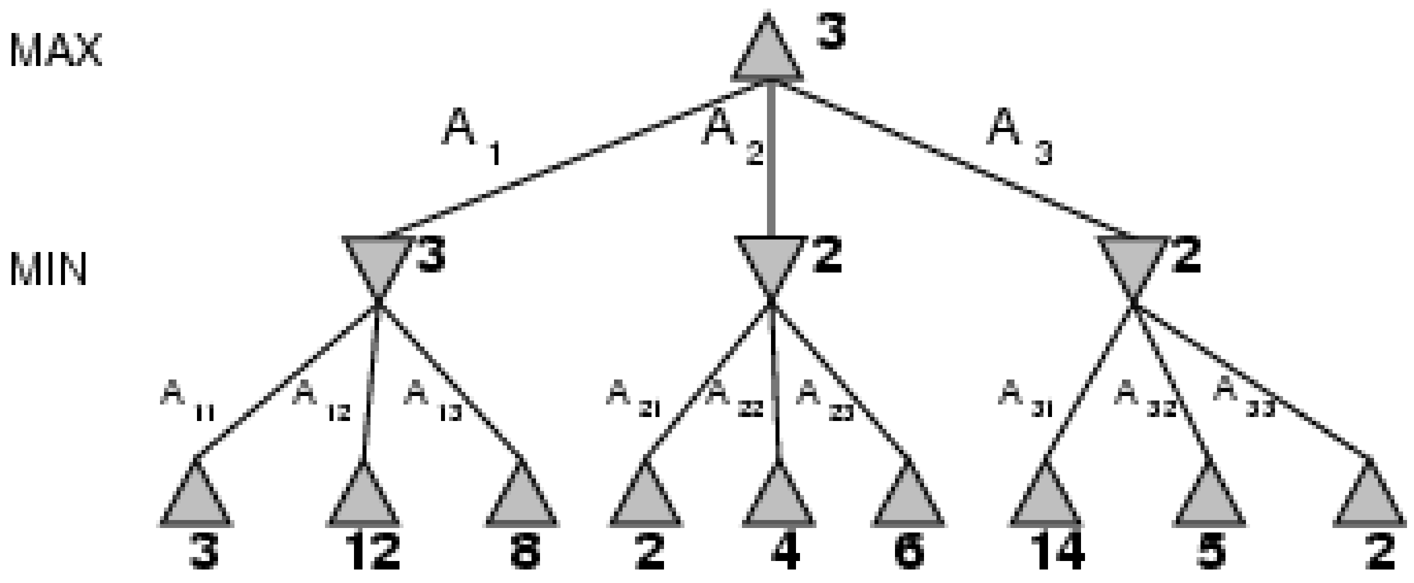
OPTIMAL DECISIONS

MINIMAX

Game tree (2-player, deterministic, turns)

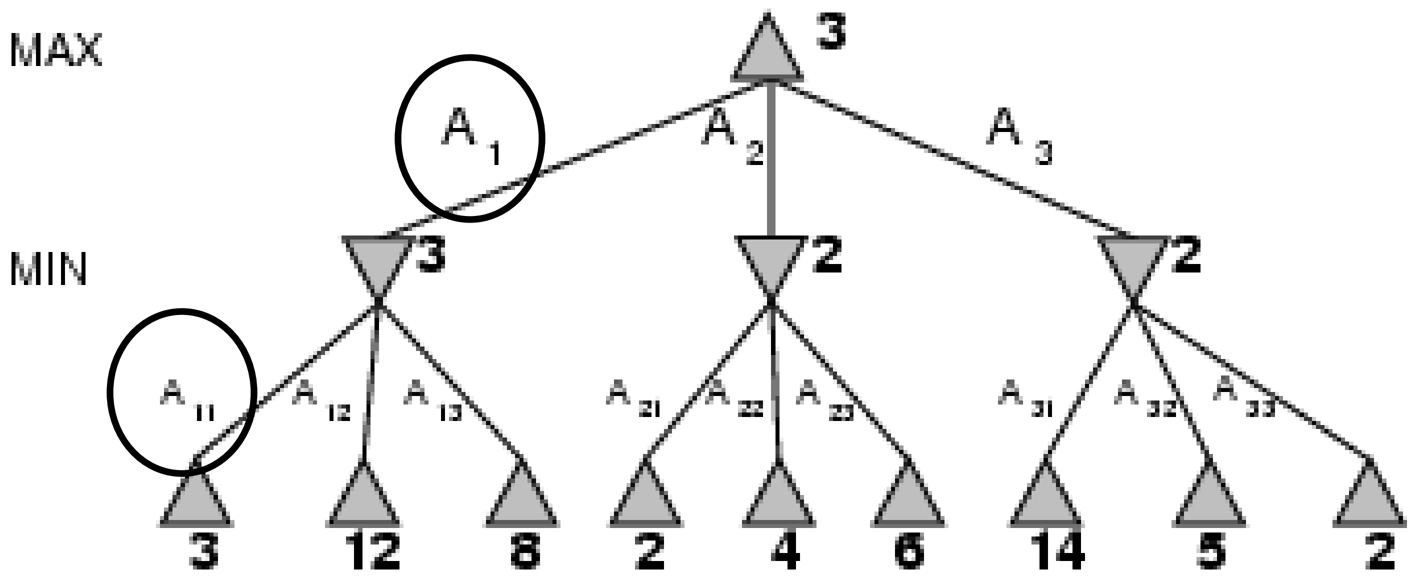


Minimax values (at nodes)



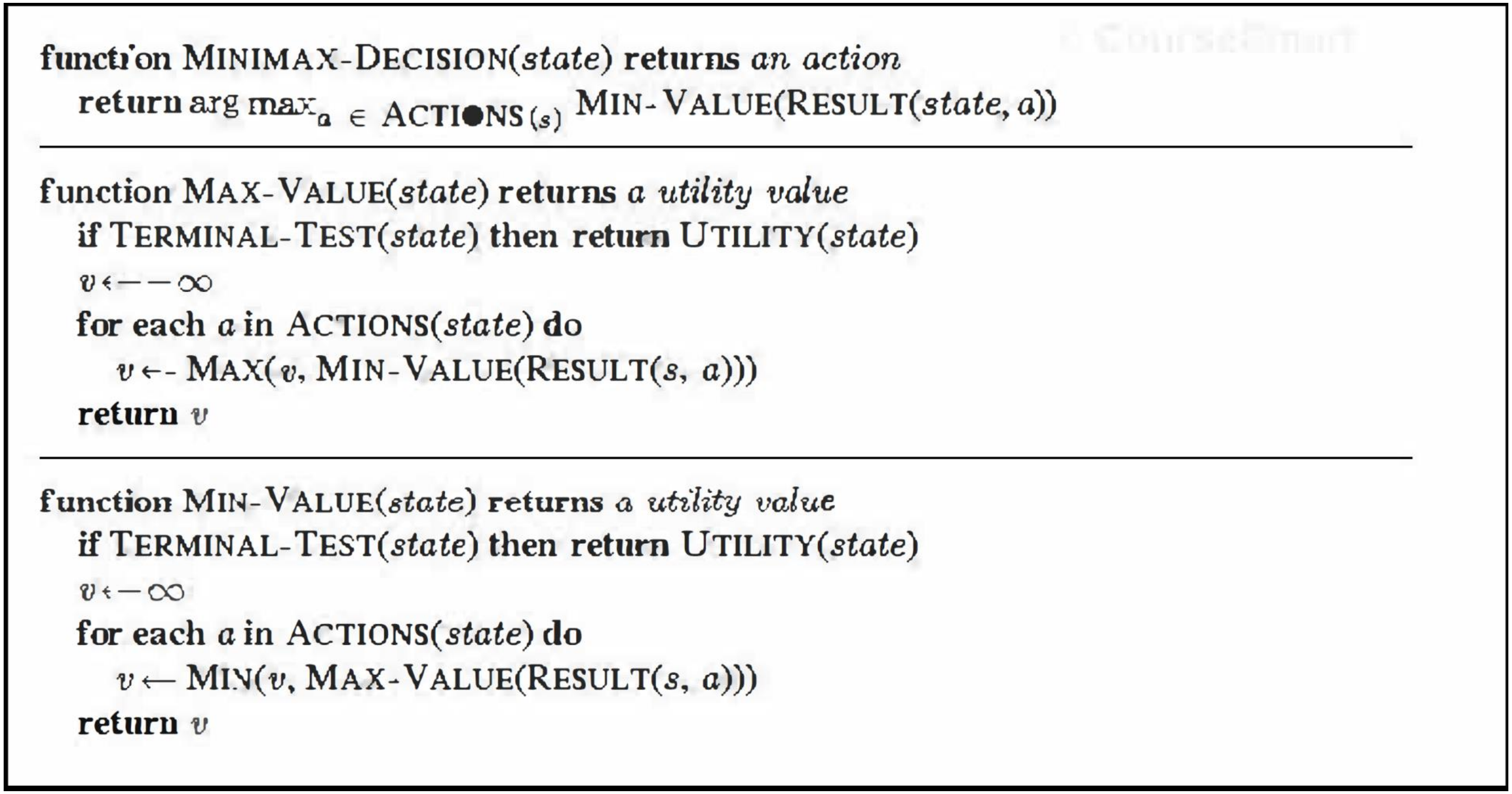
2-ply game

Minimax decision



Choose move to node with highest (MAX) or lowest (MIN) minimax value
(= perfect play for deterministic games)

Minimax algorithm



```
function MINIMAX-DECISION(state) returns an action
   return arg max_{a ∈ ACTIONS(s)} MIN-VALUE(RESULT(state, a))

function MAX-VALUE(state) returns a utility value
   if TERMINAL-TEST(state) then return UTILITY(state)
   $v \leftarrow -\infty$
   for each a in ACTIONS(state) do
      $v \leftarrow$ MAX(v, MIN-VALUE(RESULT(s, a)))
   return v

function MIN-VALUE(state) returns a utility value
   if TERMINAL-TEST(state) then return UTILITY(state)
   $v \leftarrow \infty$
   for each a in ACTIONS(state) do
      $v \leftarrow$ MIN(v, MAX-VALUE(RESULT(s, a)))
   return v
```

Properties of minimax

Complete? Yes (if tree is finite)

Optimal? Yes (against an optimal opponent)

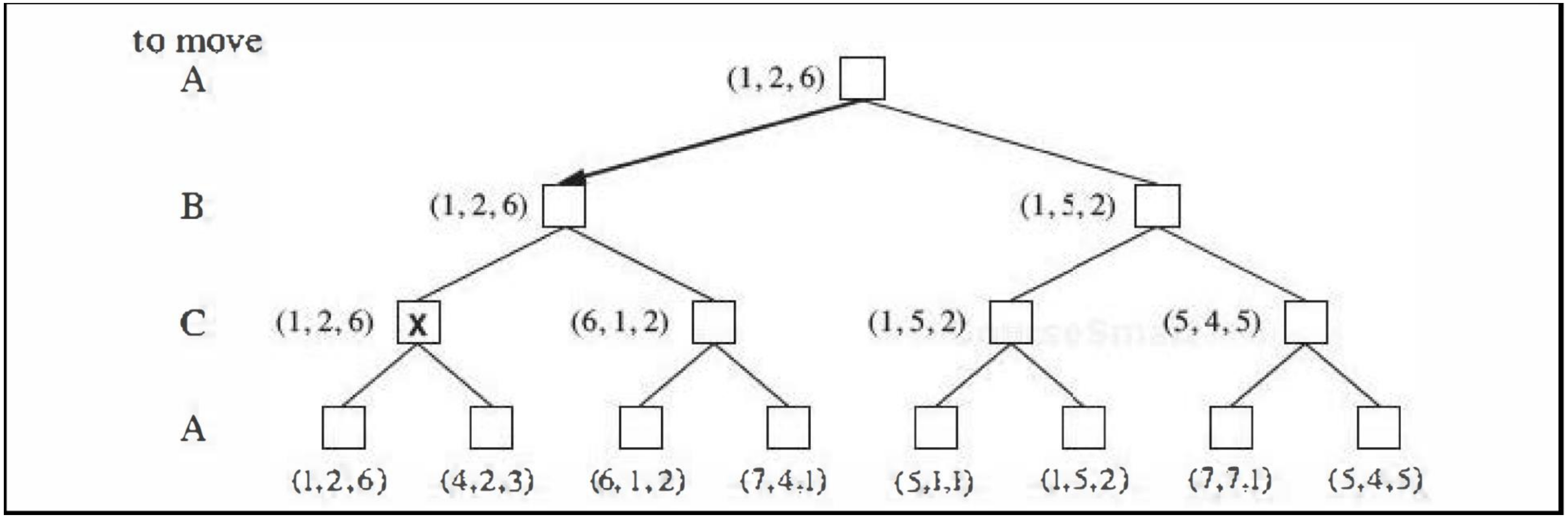
Time complexity? $O(b^m)$

Space complexity? $O(bm)$ (depth-first exploration)

For chess, $b \approx 35$, $m \approx 100$ for "reasonable" games
 → exact solution completely infeasible

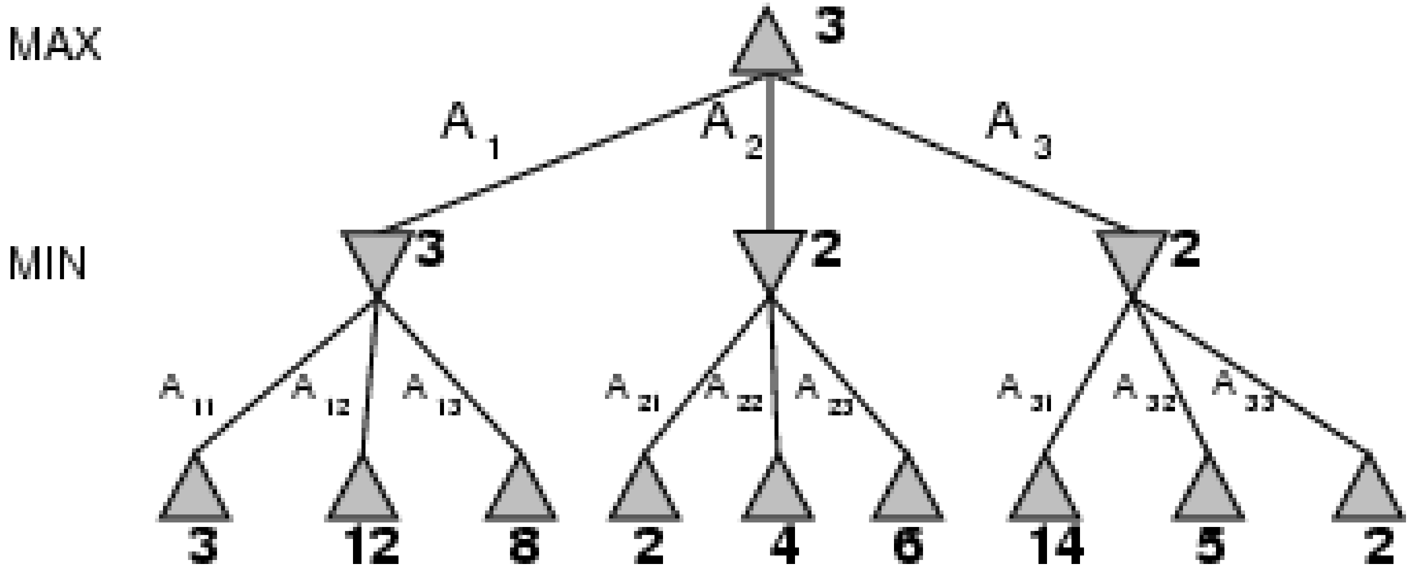
Do we need to explore every path?

More than 2 players

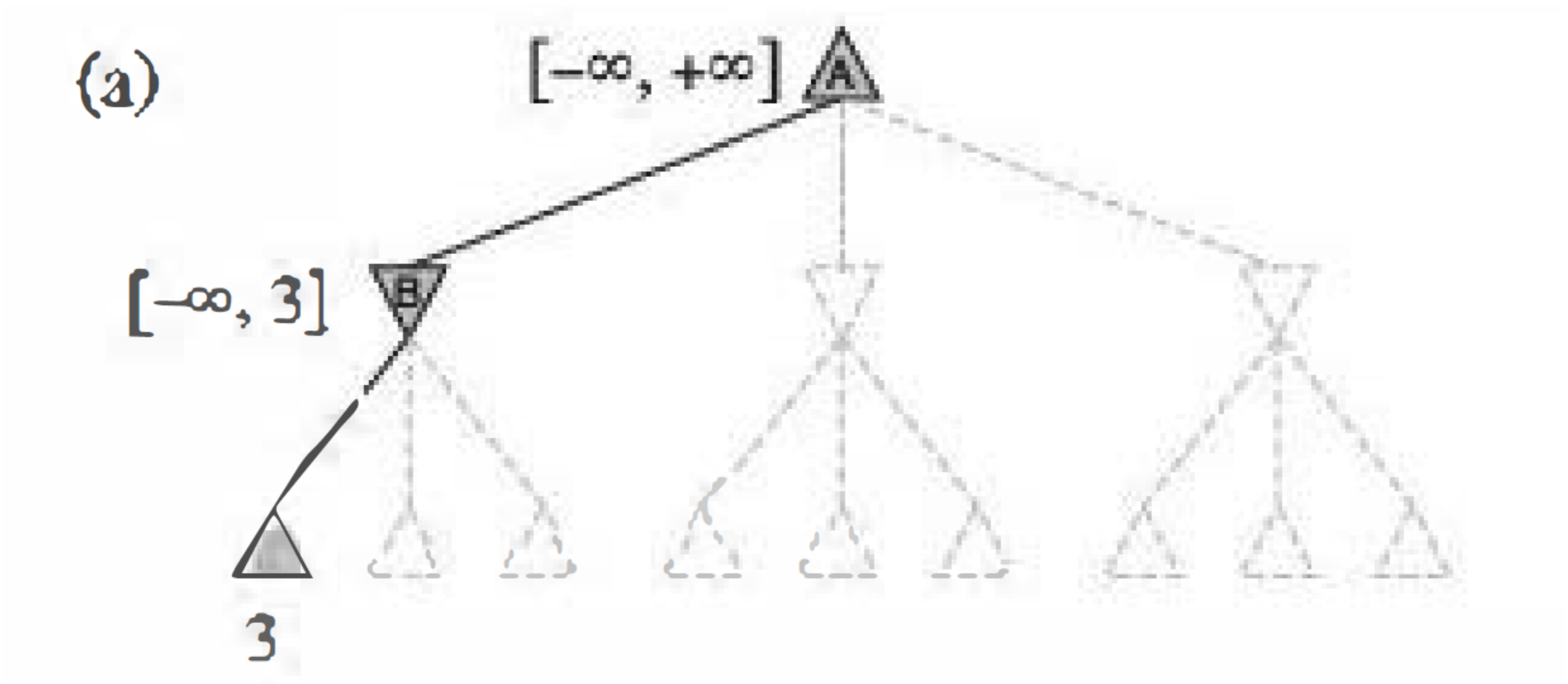


ALPHA BETA PRUNING

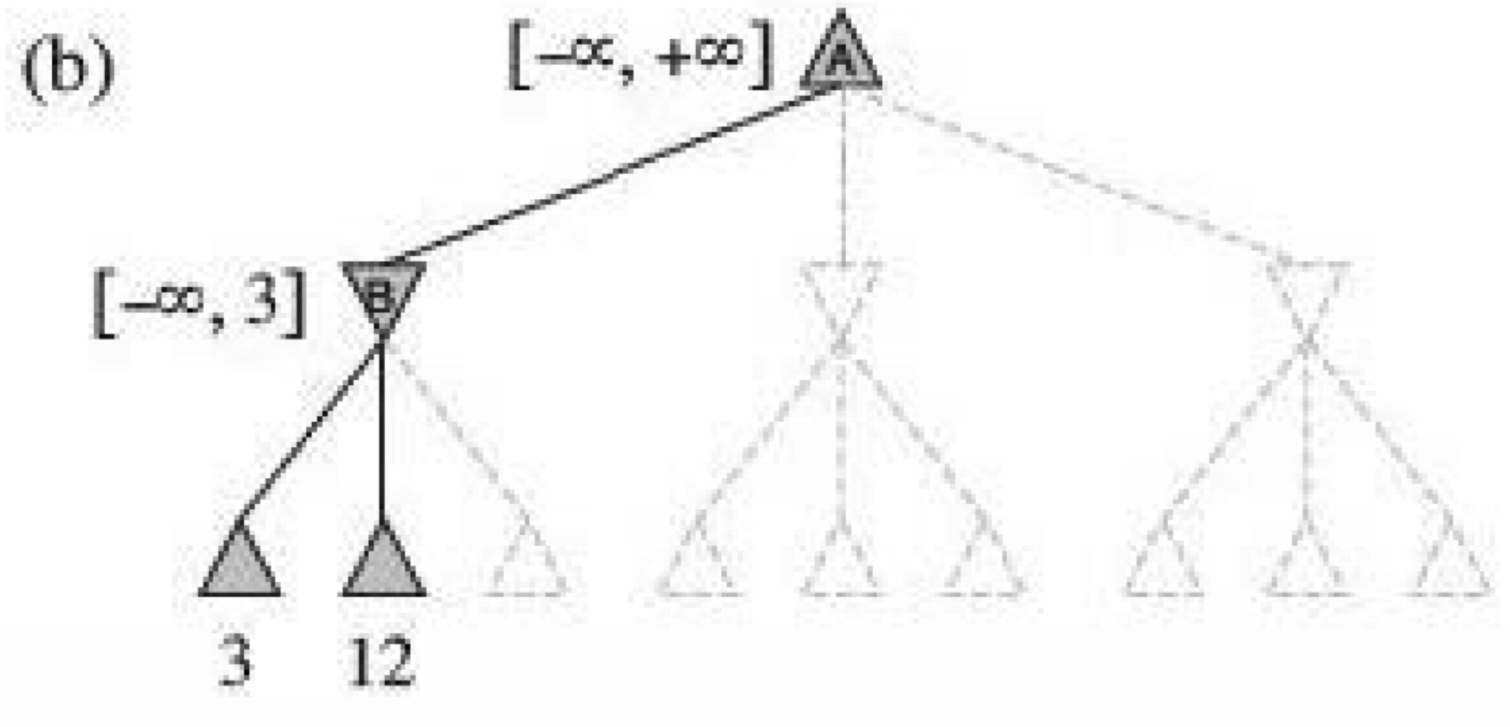
Minimax algorithm visits ALL nodes



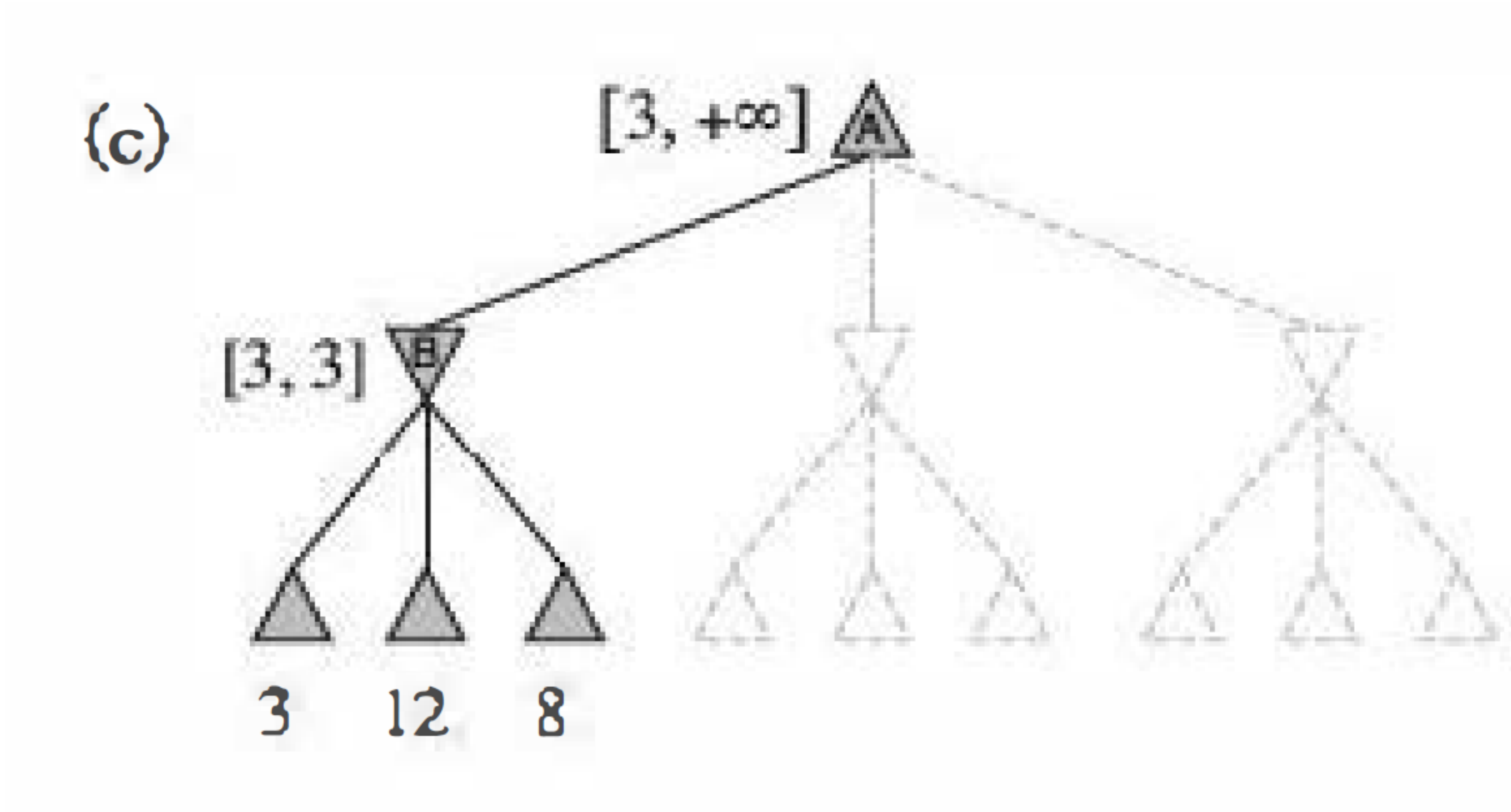
α-β pruning: example



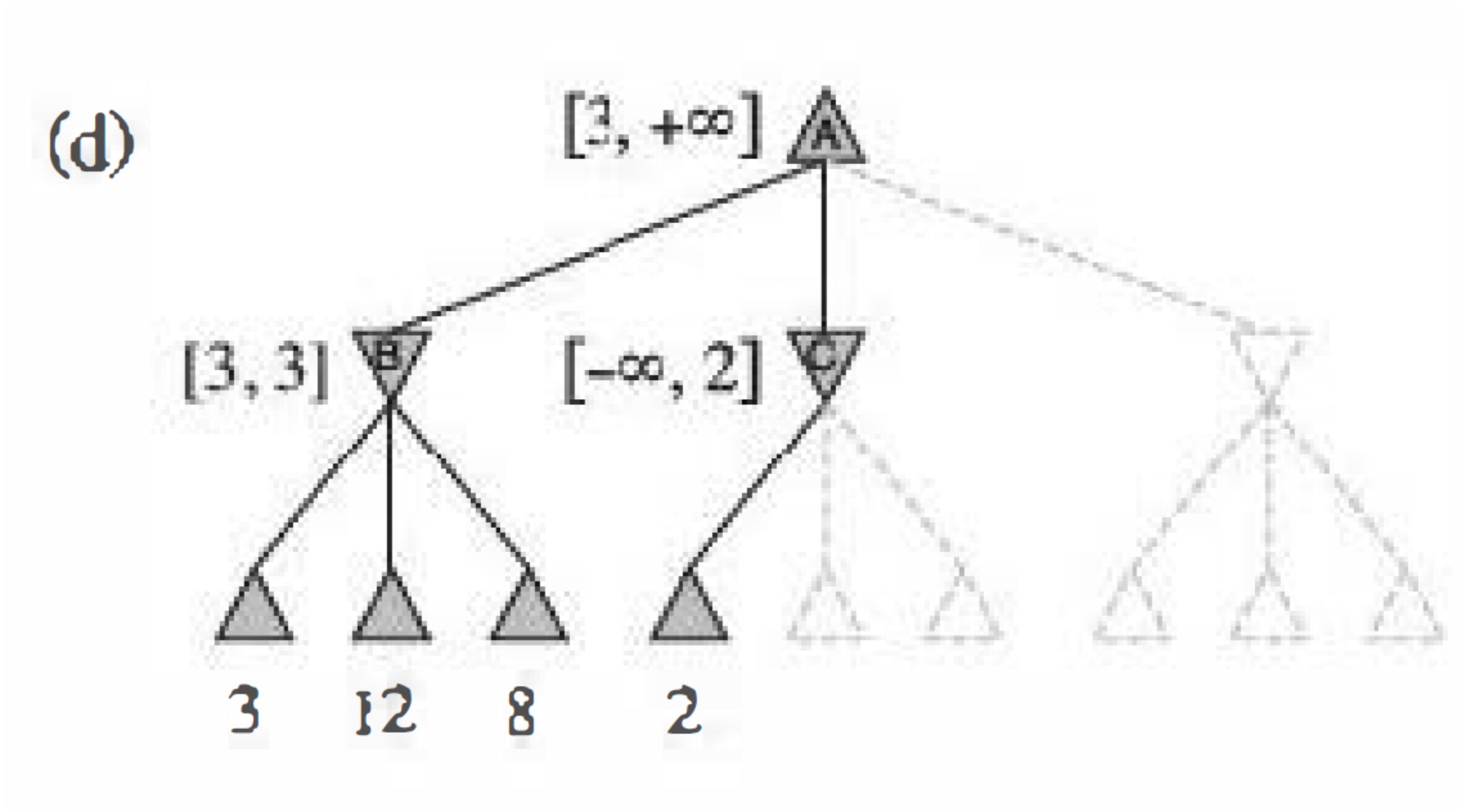
α-β pruning: example



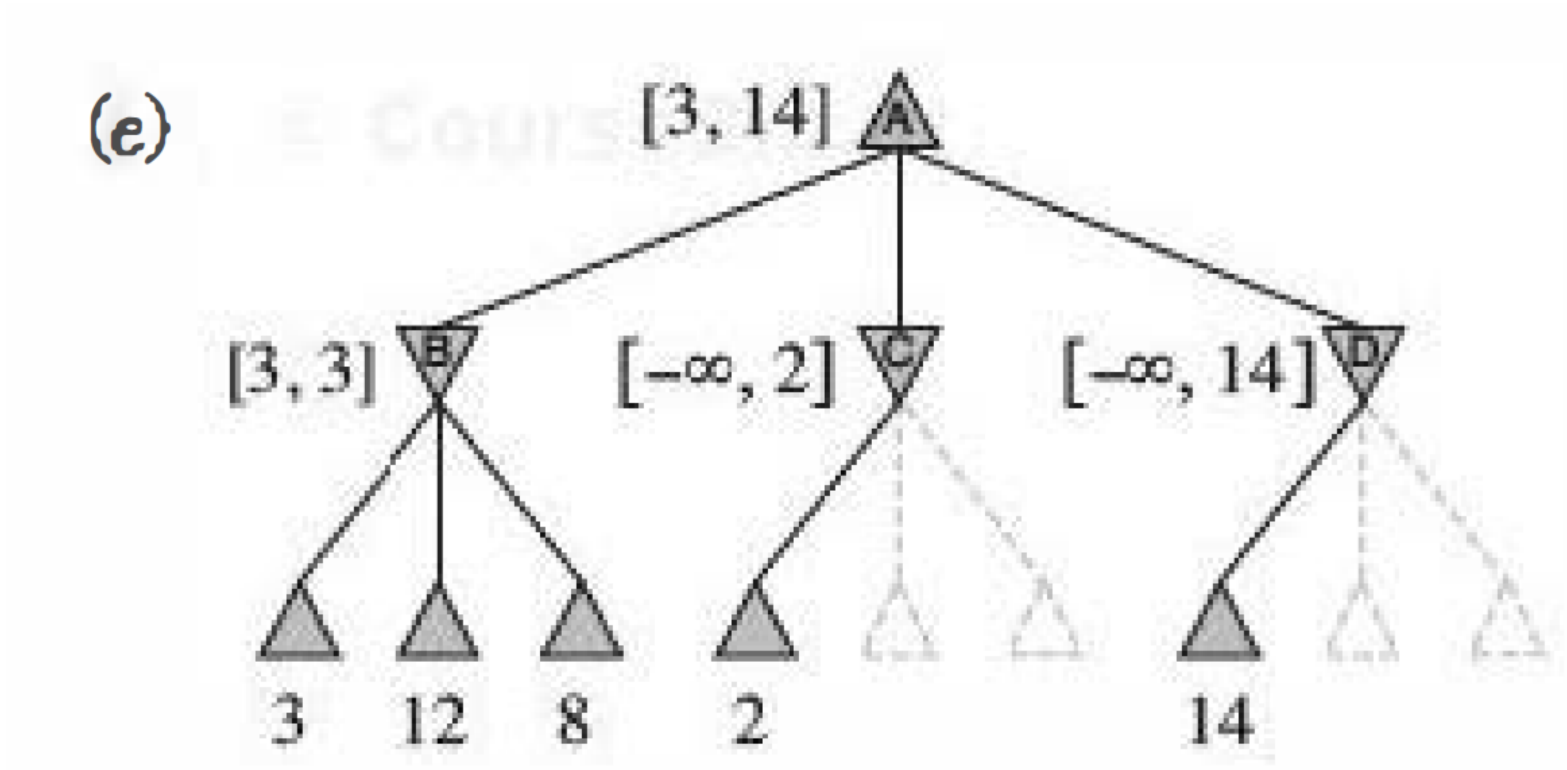
α-β pruning: example



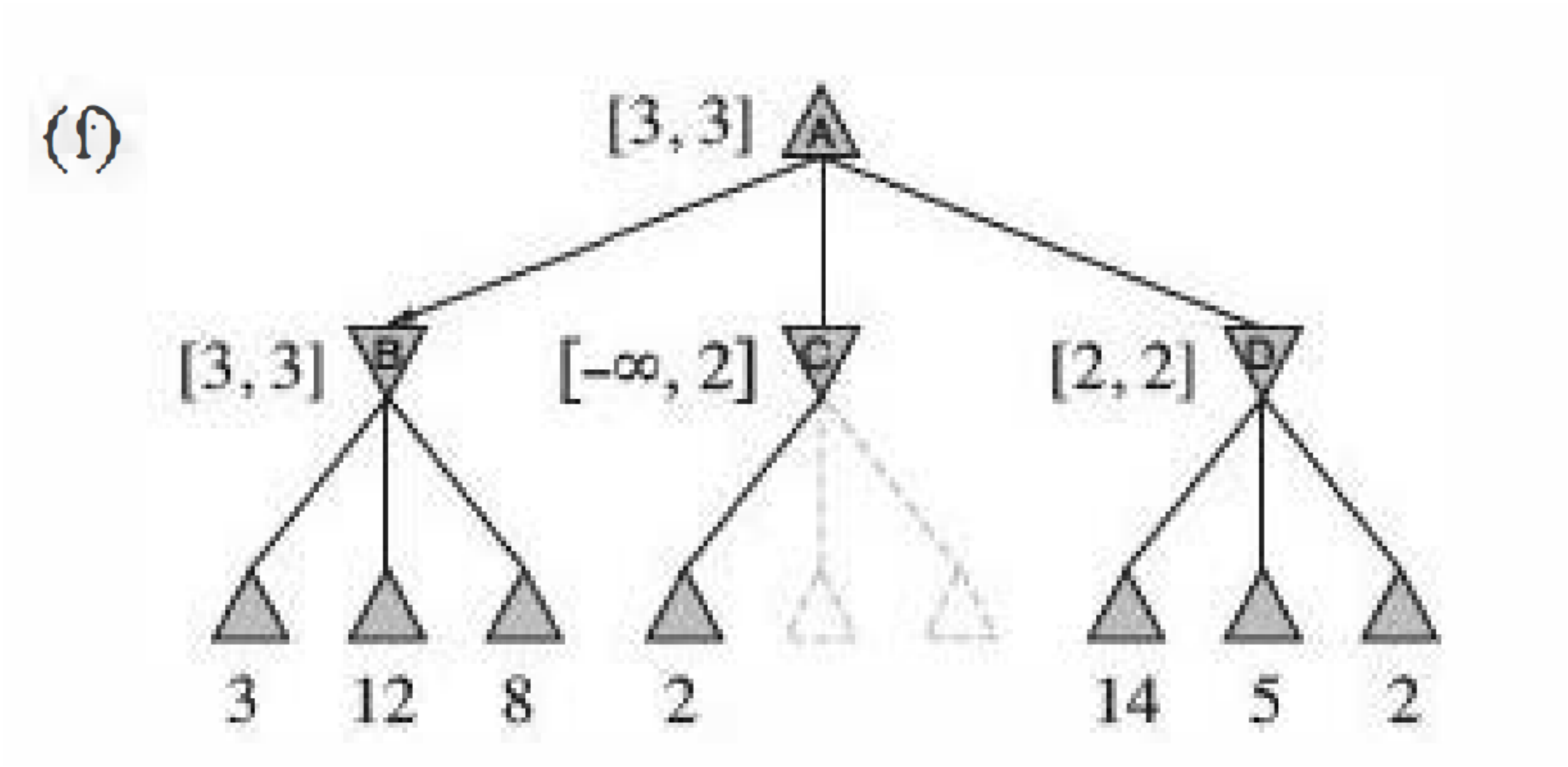
α-β pruning: example



α-β pruning: example



α-β pruning: example



Properties of α-β

- Pruning does not affect final result
- Good move ordering improves effectiveness of pruning
- With "perfect ordering," time complexity = $O(b^{m/2})$

  → doubles depth of search
- Simple example of the value of reasoning about which computations are relevant (a form of meta-reasoning)

TU Delft

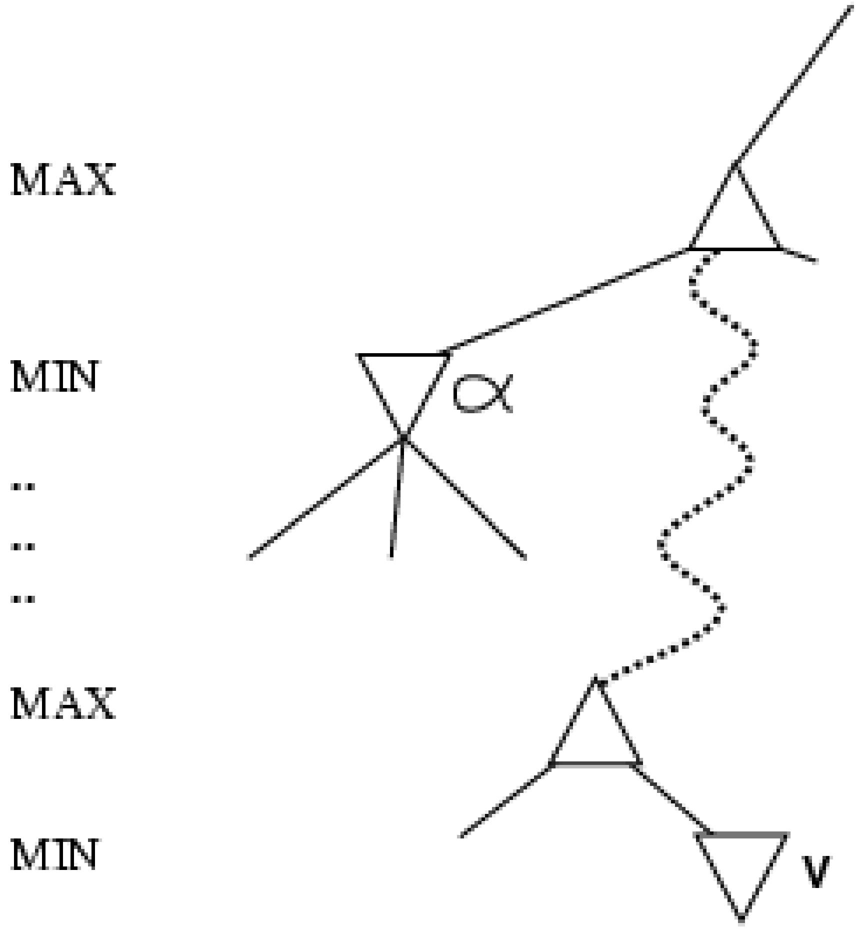
Why is it called α-β?

α is the value of the best (i.e., highest-value) choice found so far at any choice point along the path for *max*

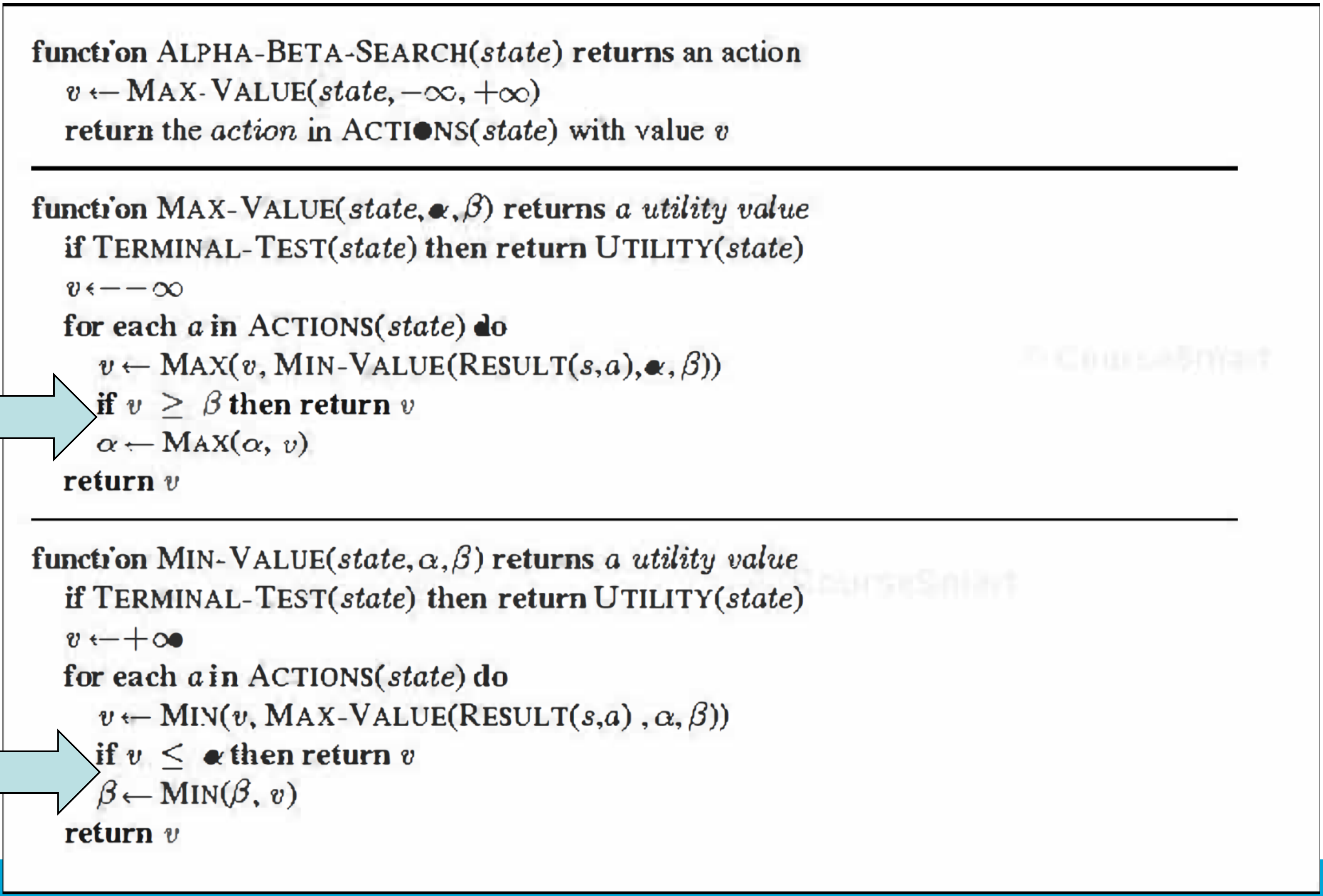
If *v* is worse than α, *max* will avoid it

→ prune that branch

Define β similarly for *min*



The α-β algorithm



```
function ALPHA-BETA-SEARCH(state) returns an action
  $v \leftarrow$ MAX-VALUE(state, $-\infty$, $+\infty$)
  return the action in ACTIONS(state) with value $v$

function MAX-VALUE(state, $\alpha$, $\beta$) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  $v \leftarrow -\infty$
  for each $a$ in ACTIONS(state) do
    $v \leftarrow$ MAX($v$, MIN-VALUE(RESULT($s$,$a$), $\alpha$, $\beta$))
    if $v \geq \beta$ then return $v$
    $\alpha \leftarrow$ MAX($\alpha$, $v$)
  return $v$

function MIN-VALUE(state, $\alpha$, $\beta$) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  $v \leftarrow +\infty$
  for each $a$ in ACTIONS(state) do
    $v \leftarrow$ MIN($v$, MAX-VALUE(RESULT($s$,$a$), $\alpha$, $\beta$))
    if $v \leq \alpha$ then return $v$
    $\beta \leftarrow$ MIN($\beta$, $v$)
  return $v$
```

IMPERFECT REAL-TIME DECISIONS

Resource limits (time, space)

- Minimax generates complete state space
- α-β pruning searches all the way to terminal nodes
- Not possible for games like chess:
  - State space is too big
  - Time to play is limited

Resource limits (time, space)

- Suppose we have 100 seconds per move and can explore 10^4 nodes/second

$\rightarrow$ 10^6 nodes per move

$\rightarrow$ approximately α-β pruning depth 8: good chess player

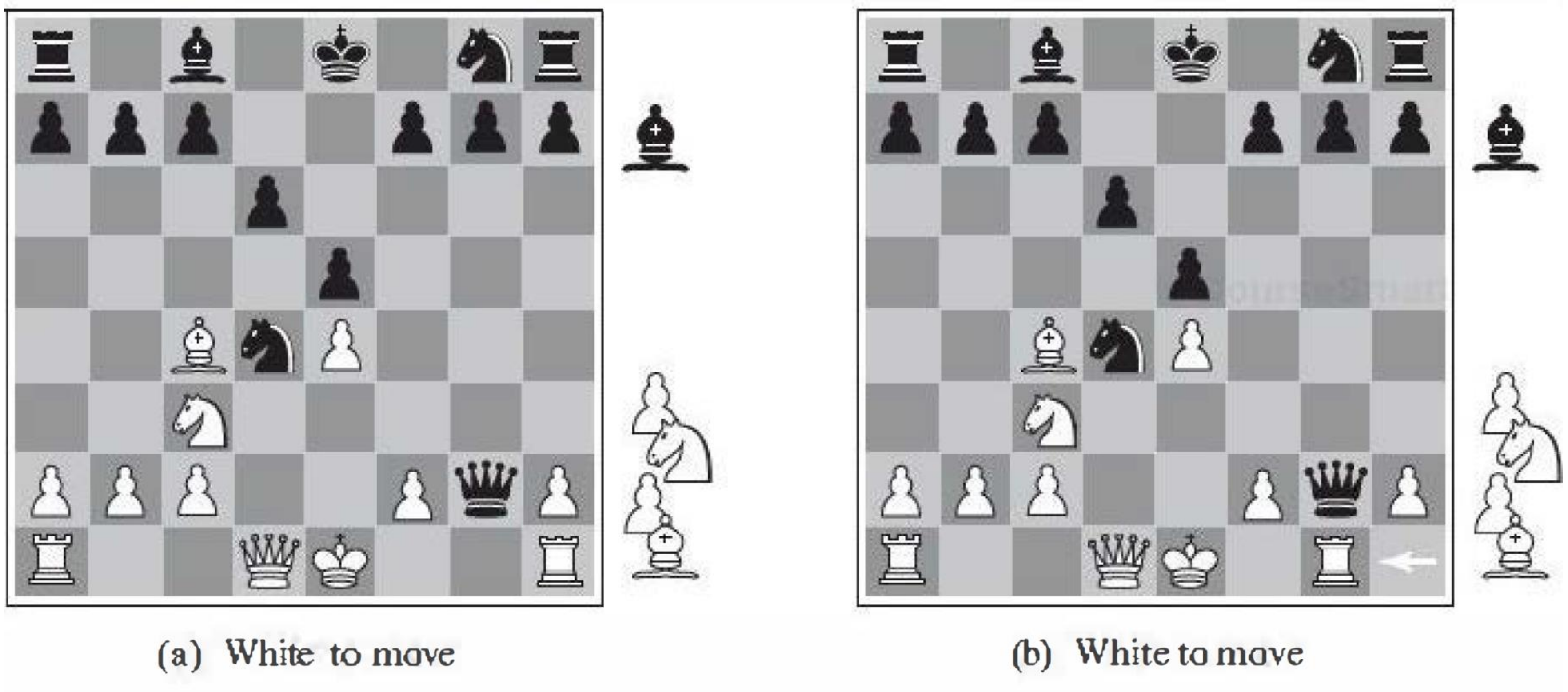
Limited game tree search

Standard approach with two ingredients:

- Cutoff test:
  e.g., depth limit (perhaps add quiescence search)

- Heuristic evaluation function
  = estimated desirability of position

Evaluation functions

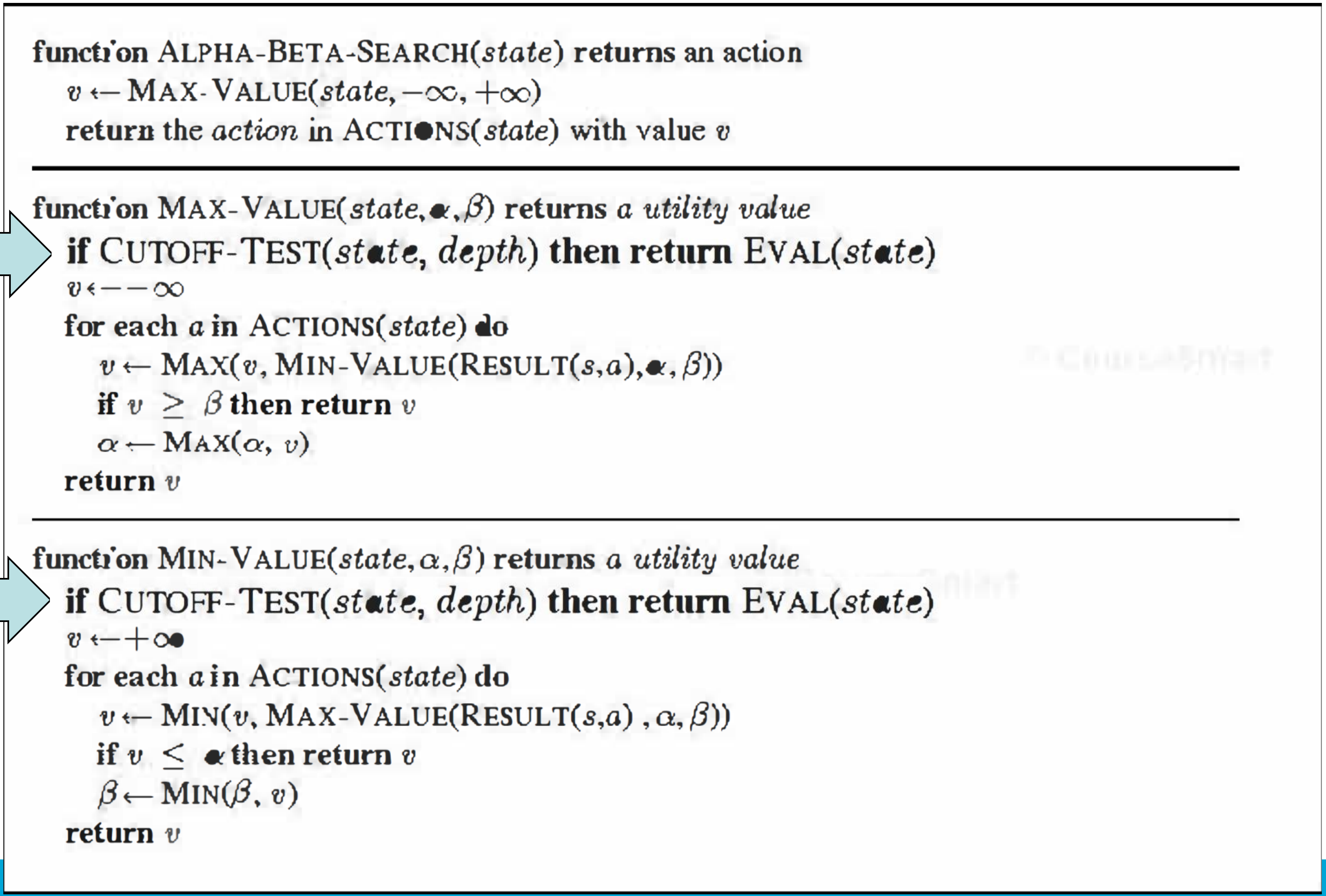


- For chess, typically linear weighted sum of features

$$Eval(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s)$$

- e.g., $w_1 = 9$ with
  $f_1(s)$ = (number of white queens) – (number of black queens), etc.

The α-β algorithm with cutoff & eval



```
function ALPHA-BETA-SEARCH(state) returns an action
  $v \leftarrow$ MAX-VALUE(state, $-\infty$, $+\infty$)
  return the action in ACTIONS(state) with value $v$
```

```
function MAX-VALUE(state, $\alpha$, $\beta$) returns a utility value
  if CUTOFF-TEST(state, depth) then return EVAL(state)
  $v \leftarrow -\infty$
  for each $a$ in ACTIONS(state) do
    $v \leftarrow$ MAX($v$, MIN-VALUE(RESULT(s,a), $\alpha$, $\beta$))
    if $v \geq \beta$ then return $v$
    $\alpha \leftarrow$ MAX($\alpha$, $v$)
  return $v$
```

```
function MIN-VALUE(state, $\alpha$, $\beta$) returns a utility value
  if CUTOFF-TEST(state, depth) then return EVAL(state)
  $v \leftarrow +\infty$
  for each $a$ in ACTIONS(state) do
    $v \leftarrow$ MIN($v$, MAX-VALUE(RESULT(s,a), $\alpha$, $\beta$))
    if $v \leq \alpha$ then return $v$
    $\beta \leftarrow$ MIN($\beta$, $v$)
  return $v$
```

Cutting off search

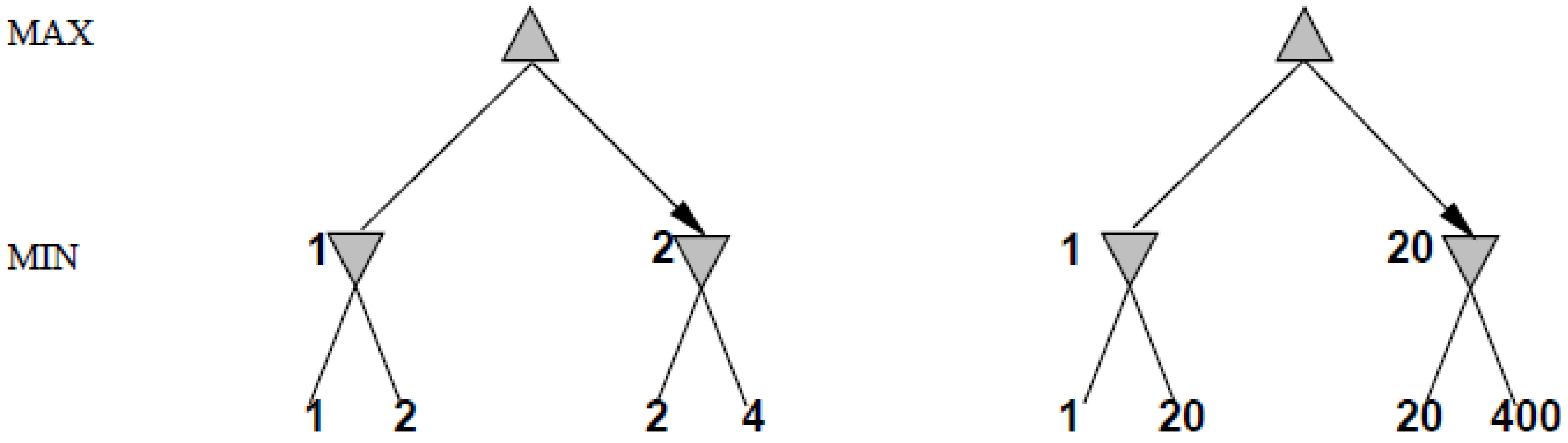
Does it work in practice?

$b^m = 10^6$, $b=35 \rightarrow m=4$

4-ply lookahead is a hopeless chess player!

- 4-ply ≈ human novice
- 8-ply ≈ typical PC, human master
- 12-ply ≈ Deep Blue, Kasparov

When exact values don't matter



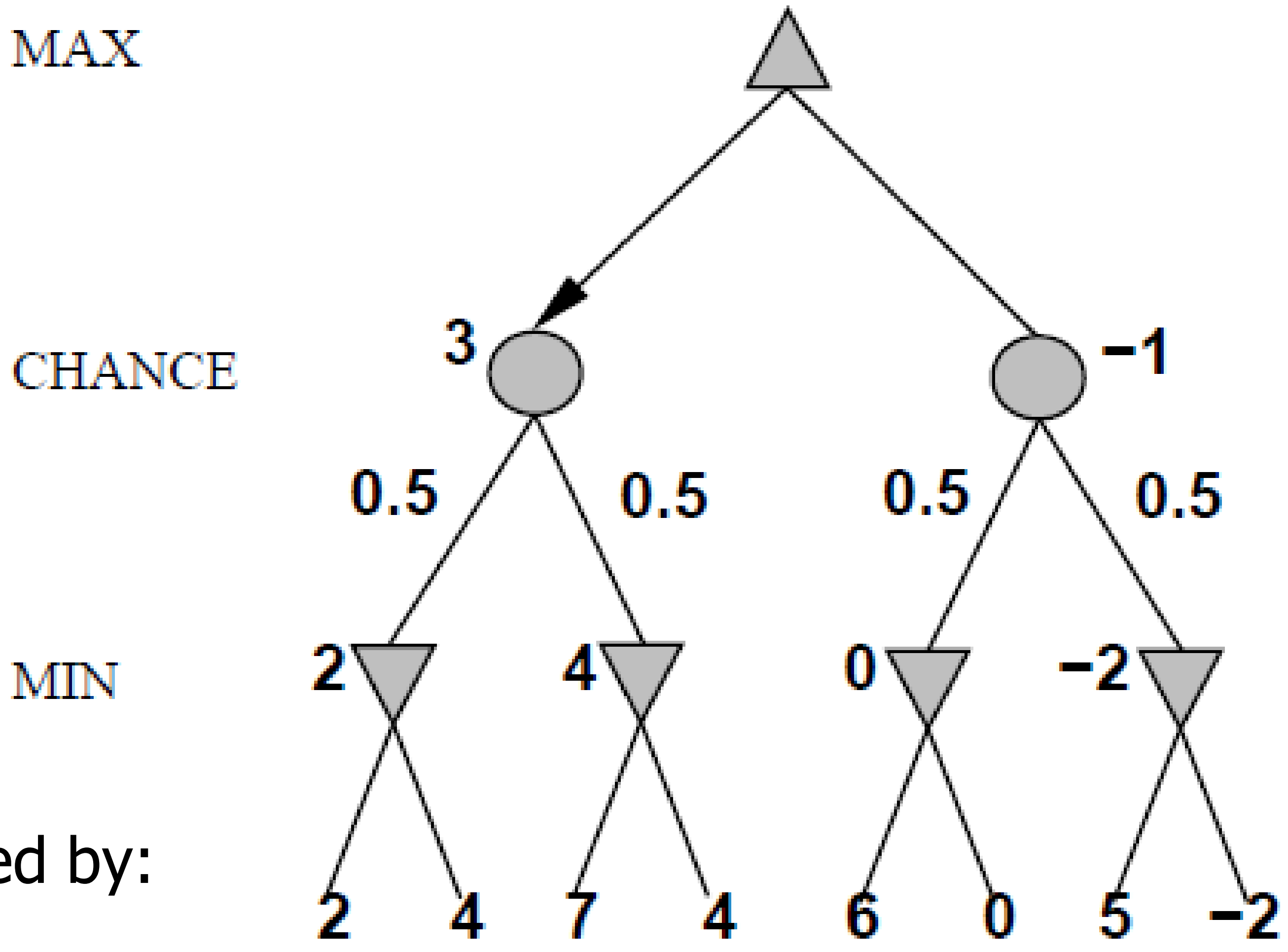
- Behaviour is preserved under **monotonic** transformations of the EVAL function
- Pay-off in deterministic games acts as an **ordinal utility** function

STOCHASTIC GAMES

Many examples

- Backgammon
- Risk
- Monopoly
- Poker
- …

Non-deterministic games in general



Chance introduced by:
- Dice
- Card shuffling
- Coin flipping
- …

Algorithm for non-deterministic games

EXPECTIMINIMAX gives perfect play

- Like MINIMIMAX
- But must handle chance nodes

…

If *state* is a

- MAX node then return the highest EXPECTIMINIMAX-VALUE of successor states
- MIN node then return the lowest EXPECTIMINIMAX-VALUE of successor states
- CHANCE node then return the average EXPECTIMINIMAX-VALUE of successor states

…

Non-deterministic games in practice

- Chance nodes increase the branching factor b
  - 21 dice rolls with two dice
  - Backgammon: approximately 20 legal moves (can be 6000 with 1-1 roll)
  - Depth 4 = $20 \times (21 \times 20)^3 \approx 1.2 \times 10^9$
  - Limited value in look-ahead for increased depth
  - α-β pruning less effective

- TD-Gammon (G. Tesauro, 1992, IBM)
  - Artificial neural net trained by a form of temporal-difference learning
  - Depth 2 search + very good EVAL: world-champion level

Games of imperfect information

Example: card games where opponent's initial cards are unknown

Today's top-level bridge programs (e.g., WBridge5, Jack) deal with this probabilistic nature by generating many samples representing the unknown hands. Each sample is generated at random, but constrained to be compatible with all information available so far from the bidding and the play.

Next, the result of different lines of play are tested against optimal defense for each sample. This testing is done using a so-called "double-dummy solver" that uses extensive search algorithms to determine the optimum line of play for both parties. The line of play that generates the best score averaged over all samples is selected as the optimal play.

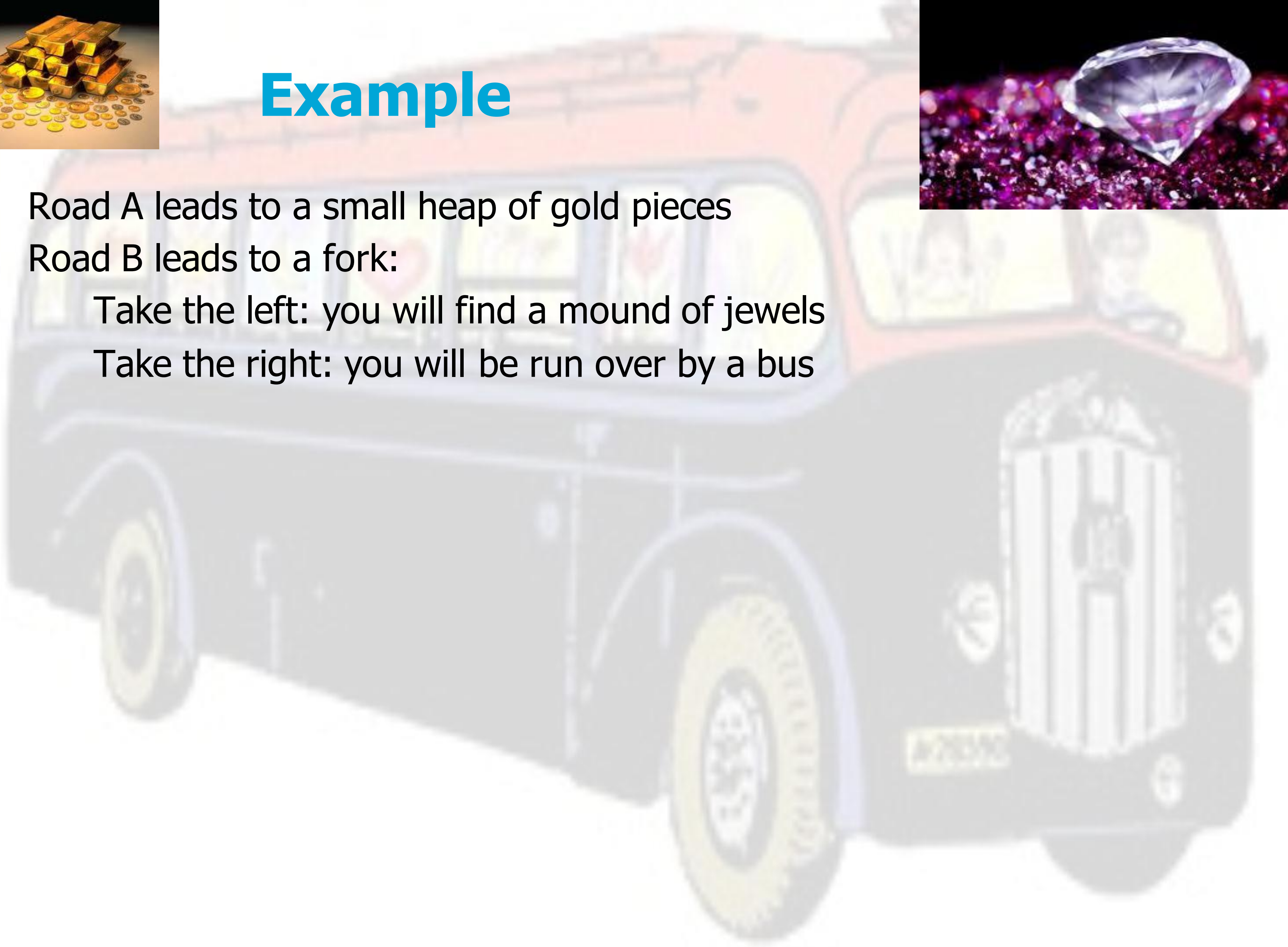
Efficient double-dummy solvers are key to successful bridge-playing programs. Also, as the amount of computation increases with sample size, techniques such as importance sampling are used to generate sets of samples that are of minimum size but still representative.

Example

Road A leads to a small heap of gold pieces

Road B leads to a fork:

- Take the left: you will find a mound of jewels
- Take the right: you will be run over by a bus



Example

Road A leads to a small heap of gold pieces

Road B leads to a fork:

- Take the left: you will find a mound of jewels
- Take the right: you will be run over by a bus

Road A leads to a small heap of gold pieces

Road B leads to a fork:

- Take the left: you will be run over by a bus
- Take the right: you will find a mound of jewels

Example

Road A leads to a small heap of gold pieces
Road B leads to a fork:
Take the left: you will find a mound of jewels
Take the right: you will be run over by a bus

Road A leads to a small heap of gold pieces
Road B leads to a fork:
Take the left: you will be run over by a bus
Take the right: you will find a mound of jewels

Road A leads to a small heap of gold pieces
Road B leads to a fork:
Which ever fork you take you stand a 0.5 chance that you will be run over by a bus, and a 0.5 chance that you find a mound of jewels

Proper analysis

- Intuition that the value of an action is the average of its value in all actual states is **WRONG**
- With partial observability, value of an action depends on the information state or belief state the agent is in
- Can generate and search a tree of information states
- Leads to rational behaviours such as
  - Acting to obtain information
  - Signalling to one’s partner
  - Acting randomly to minimize information disclosure

Summary

Games are fun to work on!

They illustrate several important points about AI

Perfection is unattainable -> must approximate…

AIMA – Chapter 17 Making complex decisions

Section 2.1-2.3 Non-Cooperative Game Theory

CS4375 Artificial Intelligence Techniques

Matthijs Spaan (based on slides by C.M. Jonker)

October 17, 2023

Decisions with Multiple Agents: Game Theory

- Game theory: analysing games with
  - Turn-taking, perfect info: Chapter 5 Adversarial Games
  - Simultaneous moves
    - One move
    - Iterated games
- Agent design
- Mechanism design

TU Delft

Game Theory

Game definition

- Players (agents)
- Actions
- Payoff matrix

Type of game

1. Constant-sum
2. Coordination

Outcome

- Pareto optimal
- Pareto domination

Agent design

- Strategy / solution
  - Pure or mixed
  - Dominated

Equilibrium

- Local optimum
- Dominant strategy
- Nash equilibrium

TUDelft

2-finger Morra

Two players O and E simultaneously display one or two fingers.

	O: one	O: two
E: one	E=2, O=-2	E=-3, O=3
E: two	E=-3, O=3	E=4, O=-4

Strategy (policy): choose one for yourself.

Strategies

- Pure: deterministic policy specifying a particular action to take in each situation.
- Mixed: Randomized policy that selects actions according to a probability distribution over actions.

  [p1:a1, p2: a2, …, pn: an]

Prisoner's dilemma

- Burglars Alice and Bob are interrogated separately.

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Prisoner’s dilemma

- Burglars Alice and Bob are interrogated separately.
- Punishment in years in prison!

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Prisoner's dilemma

- Burglars Alice and Bob are interrogated separately.
- Punishment in years in prison!

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Pure strategies: s1: testify, s2: refuse

Prisoner’s dilemma

- Burglars Alice and Bob are interrogated separately.
- Punishment in years in prison!

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Pure strategies: s1: testify, s2: refuse

Which would you choose, given the person next to you?

Prisoner's dilemma

- Burglars Alice and Bob are interrogated separately.
- Punishment in years in prison!

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Pure strategies: s1: testify, s2: refuse

Which would you choose, given the person next to you?

We are going to play the game several times with the same players.

TU Delft

Prisoner’s dilemma

- Burglars Alice and Bob are interrogated separately.
- Punishment in years in prison!

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Pure strategies: s1: testify, s2: refuse

Which would you choose, given the person next to you?

We are going to play the game several times with the same players.

Write down the results whenever you play the game.

Dominant strategy

- Strategy s1 strongly dominates s2 if the outcome for s1 is better than the outcome for s2, for every choice of strategies by the other players.
- Strategy s1 weakly dominates s2 if the outcome for s1 is better than the outcome for s2, for at least one choice of strategies by the other players, and no worse for any other.

Prisoner's dilemma

Pure strategies: s1: testify, s2: refuse

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Prisoner's dilemma

Pure strategies: s1: testify, s2: refuse

Does one dominate the other?

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Prisoner’s dilemma

Pure strategies: s1: testify, s2: refuse

Does one dominate the other?

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Play the game again with the same players.

Prisoner’s dilemma

Pure strategies: s1: testify, s2: refuse

Does one dominate the other?

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Play the game again with the same players.

Suppose Bob testifies?

Prisoner's dilemma

Pure strategies: s1: testify, s2: refuse

Does one dominate the other?

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Play the game again with the same players.

Suppose Bob testifies?

Suppose Bob refuses?

Prisoner's dilemma

Pure strategies: s1: testify, s2: refuse

Does one dominate the other?

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Play the game again with the same players.

Suppose Bob testifies?

Suppose Bob refuses?

Therefore, s1 strongly dominates s2

TU Delft

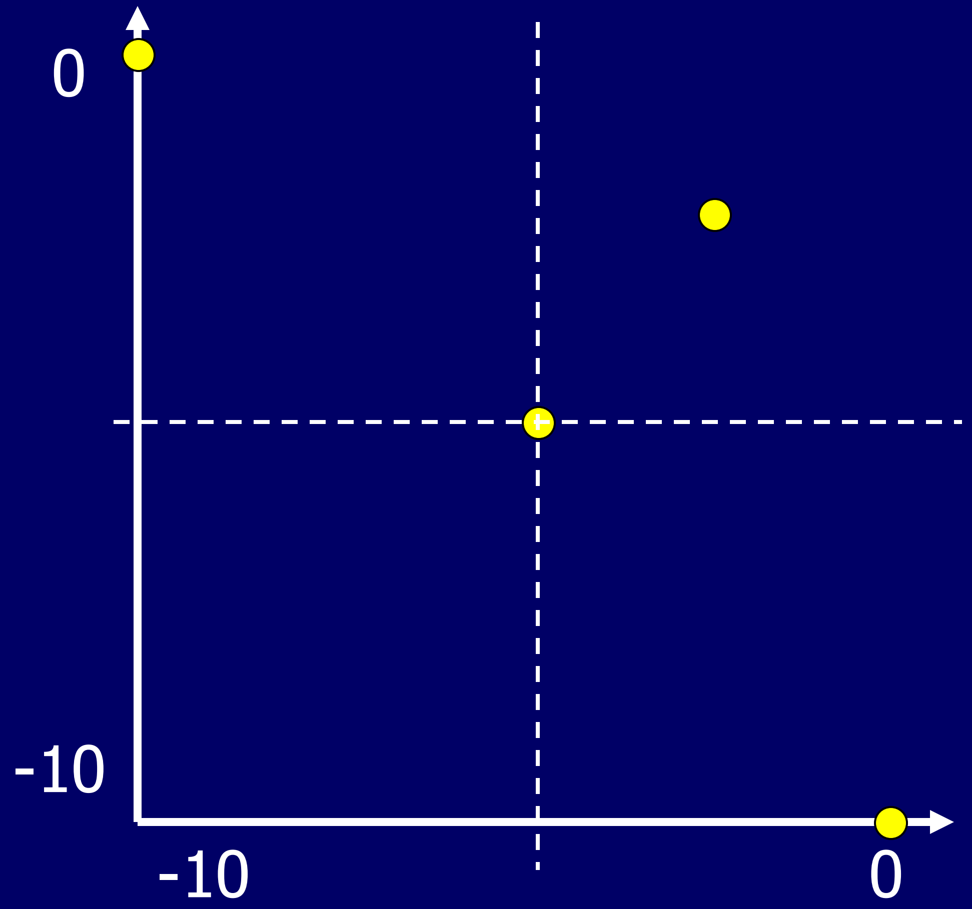
Pareto Optimal outcomes

- An outcome is Pareto optimal if there is no outcome preferred by all players.

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

- Which outcomes are Pareto optimal?
- Play the game again.

Pareto optimal outcomes



Equilibrium of strategies

- (s, s') is a Nash equilibrium if no player, knowing the strategies of all players, can benefit by switching strategies, given that every other player sticks with its current strategy.
- An equilibrium is a local optimum in the space of policies
- (testify, testify) is a Nash equilibrium for Prisoner's Dilemma.

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Dominant strategy equilibrium

- When each player pi has a dominant strategy si, then $(si)_{i \geq 1}$ is called a dominant strategy equilibrium.
- (testify, testify) is such an equilibrium
- Dilemma: the outcome is not Pareto Optimal.
- Now play the game again.

	A: testify	A: refuse
B: testify	A=-5, B=-5	A=-10, B=0
B: refuse	A=0, B=-10	A=-1, B=-1

Nash Equilibrium

- John Nash (1928 - 2015): Every game with a finite number of players in which each player can choose from finitely many pure strategies has at least one Nash equilibrium.
- A dominant strategy equilibrium is a Nash equilibrium.
- Not all games have dominant strategies.

- How can we reach the (-1, -1) outcome?
- Iterated Prisoner's Game
  - Play the Prisoner's Game
  - Repeat for an indefinite number of times

Manufacturing of video games

- Hardware manufacturer: DVD or CD

	H: DVD	H: CD
S: DVD	H=9, S=9	H=-4, S=-1
S: CD	H=-3, S=-1	H=5, S=5

Manufacturing of video games

- Hardware manufacturer: DVD or CD
- Software manufacturer: DVD or CD

	H: DVD	H: CD
S: DVD	H=9, S=9	H=-4, S=-1
S: CD	H=-3, S=-1	H=5, S=5

Manufacturing of video games

- Hardware manufacturer: DVD or CD
- Software manufacturer: DVD or CD

	H: DVD	H: CD
S: DVD	H=9, S=9	H=-4, S=-1
S: CD	H=-3, S=-1	H=5, S=5

Which strategy is dominant? DVD or CD?

Manufacturing of video games

- Hardware manufacturer: DVD or CD
- Software manufacturer: DVD or CD

	H: DVD	H: CD
S: DVD	H=9, S=9	H=-4, S=-1
S: CD	H=-3, S=-1	H=5, S=5

Which strategy is dominant? DVD or CD?

None.

Manufacturing of video games

- Hardware manufacturer: DVD or CD
- Software manufacturer: DVD or CD

	H: DVD	H: CD
S: DVD	H=9, S=9	H=-4, S=-1
S: CD	H=-3, S=-1	H=5, S=5

Which strategy is dominant? DVD or CD?

None.

Are there Nash Equilibria?

Manufacturing of video games

- Hardware manufacturer: DVD or CD
- Software manufacturer: DVD or CD

	H: DVD	H: CD
S: DVD	H=9, S=9	H=-4, S=-1
S: CD	H=-3, S=-1	H=5, S=5

Which strategy is dominant? DVD or CD?

None.

Are there Nash Equilibria?

Yes, there always is at least one. But which?

Manufacturing of video games

- Hardware manufacturer: DVD or CD
- Software manufacturer: DVD or CD

	H: DVD	H: CD
S: DVD	H=9, S=9	H=-4, S=-1
S: CD	H=-3, S=-1	H=5, S=5

Which strategy is dominant? DVD or CD?

None.

Are there Nash Equilibria?

Yes, there always is at least one. But which?
(DVD, DVD) and (CD, CD)

What if there are several Nash Equil.?

- Nash Equil.: (CD, CD) and (DVD, DVD)
- Which to choose?

	H: DVD	H: CD
S: DVD	H=9, S=9	H=-4, S=-1
S: CD	H=-3, S=-1	H=5, S=5

- Pareto Optimal solution!
  (DVD, DVD)

Pareto versus Nash

Choose the unique Pareto-optimal Nash Equilibrium

    If one exists!

Every game has at least one Pareto-optimal solution.

There can be several Pareto-optimal solutions.

Pareto-optimal solutions don't have to be Nash equilibria.

	H: DVD	H: CD
S: DVD	H=5, S=5	H=-4, S=-1
S: CD	H=-3, S=-1	H=5, S=5

What if several Pareto-optimal Nash equilibria?

- Guess
- Negotiate

Types of games

- Coordination game: game in which players need to communicate.
- Zero-sum game: the payoffs in each cell of the payoff matrix sum to zero.
- Constant-sum game: the payoffs in each cell of the payoff matrix sum to a constant c.

TUDelft

Every game has at least one Nash Equilibrium!

But, …

How about 2-finger Morra?

	O: one	O: two
E: one	E=2, O=-2	E=-3, O=3
E: two	E=-3, O=3	E=4, O=-4

Find a mixed strategy!

Von Neumann's Maximin technique

1. Force E to reveal its strategy first, followed by O.
   - Now it is a turn taking game: minimax applicable
   - Outcome for E: U_{EO}
   - This favors O, so true utility $U_E \geq U_{EO}$
2. Force O to reveal its strategy first, followed by E.
   - Apply minimax
   - Outcome for E: U_{OE}
   - This favors E, so $U_E \leq U_{OE}$

Do this with mixed strategies -> maximin equilibrium.

Every Nash equilibrium in a zero-sum game is a maximin for both players.

Example: Morra game

	O: one	O: two
E: one	E=2, O=-2	E=-3, O=3
E: two	E=-3, O=3	E=4, O=-4

Suppose E plays (one: x, two: 1-x) against O. Force O to play first:

u_E((one: x, two: 1-x), one) = 2x – 3(1-x) = 5x - 3

u_E((one: x, two: 1-x), two) = -3x + 4(1-x) = -7x + 4

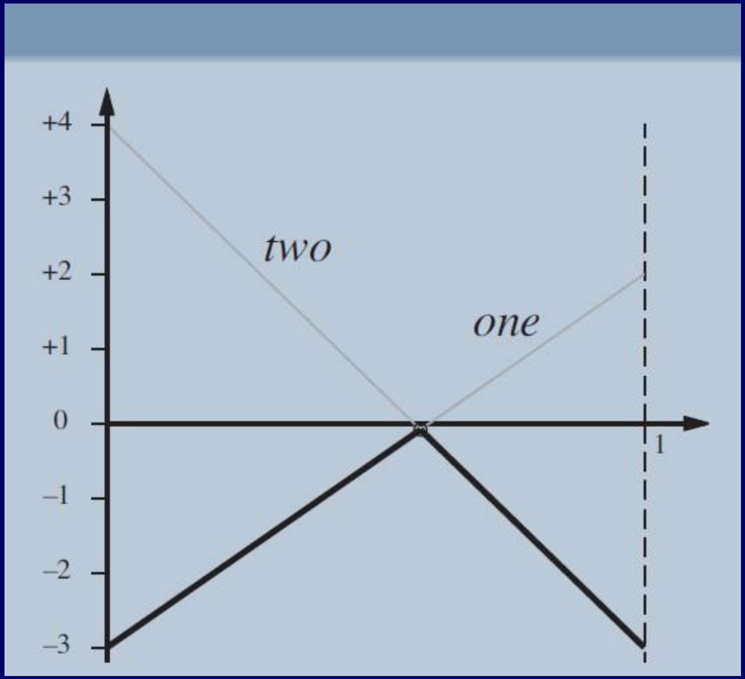
How can E maximize its worst-case payoff? Both expressions should yield the same utility, otherwise when playing for real, O would choose the action that makes it best for O and thus worst for E.

5x – 3 = -7x + 4 iff x = 7/12

So E's maximin strategy is: (one: 7/12, two: 5/12)

Which on average would yield:
u_E((one: 7/12, two: 5/12), one) = 5* 7/12 – 3 = -1/12
u_E((one: 7/12, two: 5/12), two) = -7*5/12 + 4 = -1/12

E's maximin value = -1/12



Odd: Morra game

	O: one	O: two
E: one	E=2, O=-2	E=-3, O=3
E: two	E=-3, O=3	E=4, O=-4

Do the same for player O who plays (one: y, two: 1-y) against E. Now force E to play first:

u_O((one: y, two: 1-y), one) = -2y + 3(1-y) = -5y + 3

u_O((one: y, two: 1-y), two) = 3y - 4(1-y) = 7y - 4

To maximize O's worst-case payoff:

-5y + 3 = 7y - 4 iff y = 7/12

So O's maximin strategy is (one: 7/12, two: 5/12).

However, O's maximin value is +1/12, so O is better off than E.

The unique Nash Equilibrium is ((one: 7/12, two: 5/12), (one: 7/12, two: 5/12))

Knowing about the other's strategy

Maximin equilibrium strategy for 2-finger Morra is
[7/12: one, 5/12: two] for both players.

The expected utility for E is -1/12.

E can use a random number generator or E can decide for either of the pure strategies:
one or two.

Unilaterally choosing a particular action does not harm one's expected payoff, but if the other knows about it, then he can adjust his strategy and thus influence the payoff!

Iterated Prisoner’s Dilemma

- How did you do in your repeated games?
- Which strategy did you use?
- Would it matter if you knew that the next game would be the last game?
- In general, what happens if you know that you will have to play the game c times?
- If c=100, then the 100th game is not an iterated game: it’s outcome will not affect the next iteration. Therefore: testify!
- But then the 99th game cannot affect the 100th iteration. Testify! …

Iterated Prisoner's Dilemma

Suppose: the number of rounds is uncertain

- Neither knows for sure which round will be the last

Strategies for more cooperative behavior possible:

- Perpetual punishment: *refuse* unless the other ever played *testify*.
- Tit-for-tat: start with *refuse*, after that play the same action the other played in the previous iteration.

- What was your strategy?

Tragedy of the commons

- If global utility = sum of individual agent utilities
- Each agent pursues maximal expected utility
- Then, global utility will be maximal. Right?

Commons:

- Individual farmers bring livestock to graze for free on the commons
- If everybody does so, then the commons are destroyed, and global utility is down.
- Every agent will rationally bring all his livestock, reasoning that the others will do so anyway, and his refraining from doing so would not help.

Other examples?

Literature

- Russel, S. and Norvig, P. (2010). Articial Intelligence: a Modern Approach. Prentice Hall, 3 edition. Chapters 17.5-17.6.

More about game theory and mechanism design:

- Shoham, Y. and Leyton-Brown, K. (2009). Multiagent Systems - Algorithmic, Game-Theoretic, and Logical Foundations. Cambridge University Press.
- Osborne, M. and Rubinstein, A. (1994). A Course in Game Theory. MIT Press.