Uncertainty, Chances, and Utilities

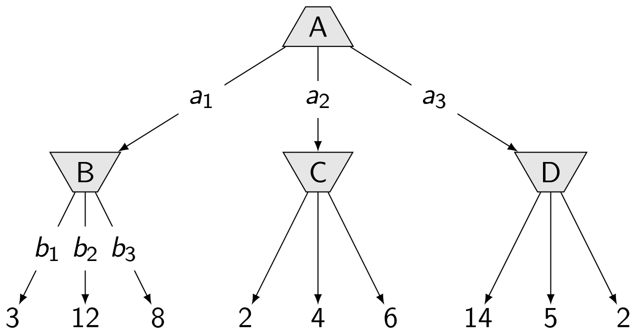
Tomáš Svoboda

Vision for Robots and Autonomous Systems, Center for Machine Perception
Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University in Prague

March 30, 2020

Notes

Deterministic opponent → stochastic environment

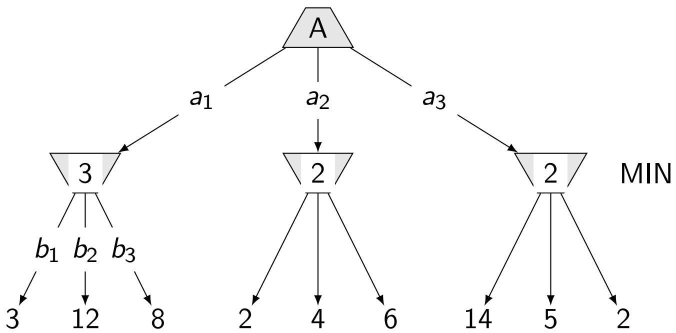


b_1, b_2, b_3 - probable branches, uncertain outcomes of a_1 action.

Notes

Stochastic environment or stochastic opponent. Simply something that is playing againts us.

Deterministic opponent → stochastic environment

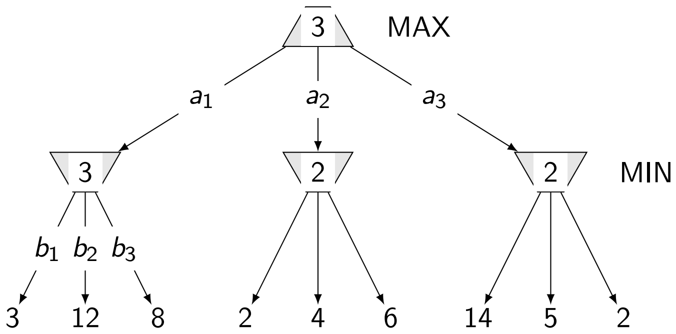


b_1, b_2, b_3 - probable branches, uncertain outcomes of a_1 action.

Notes

Stochastic environment or stochastic opponent. Simply something that is playing againts us.

Deterministic opponent → stochastic environment

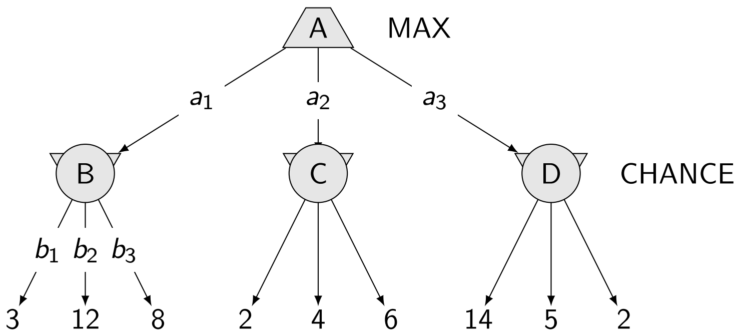


b_1, b_2, b_3 - probable branches, uncertain outcomes of a_1 action.

Notes

Stochastic environment or stochastic opponent. Simply something that is playing againts us.

Deterministic opponent → stochastic environment



b_1, b_2, b_3 - probable branches, uncertain outcomes of a_1 action.

Notes

Stochastic environment or stochastic opponent. Simply something that is playing againts us.

Why? Actions may fail, . . .

Video: Slipping robot. **Vision for Robotics and Autonomous Systems, http://cyber.felk.cvut.cz/vras**

Notes

At a certain moment, command is forward, flippers are rolling but the outcome is different, robot does not move – it is slipping a bit until it catches the grip again.

Why? Actions may fail, . . . , getting to work

A At home

tram *bike* *car*

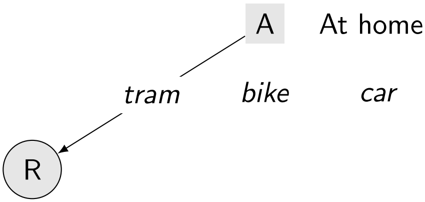
Random variable: Situation on rails R
r_1 free rails
r_2 accident
r_3 congestion

MAX/MIN depends on what the $r_?$ options and terminal numbers mean. The goal may be to get to work as fast as possible.

Notes

We talk about games. However, game model may be well used for modeling real world problems.
This is just a two-ply game/tree. But think sequentially, or, recursively.
The numbers can be seen as journey duration - then A is the MIN node - min value is the best (MAX) for me.
We can convert it to a classical MAX thinking by changing the Utilies to Working hours-delay - and we want to maximize the working hours.

Why? Actions may fail, . . . , getting to work



Random variable: Situation on rails R

r_1 free rails

r_2 accident

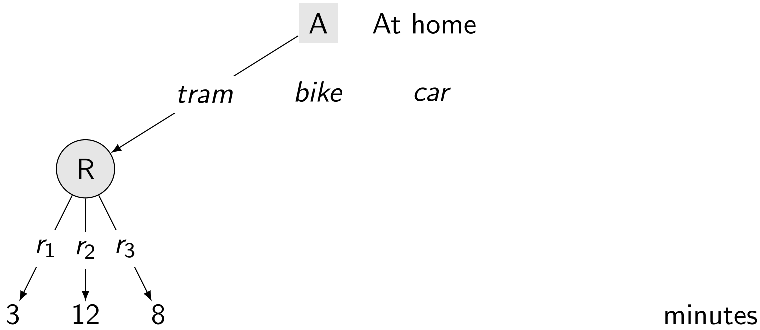
r_3 congestion

MAX/MIN depends on what the $r_?$ options and terminal numbers mean. The goal may be to get to work as fast as possible.

Notes

We talk about games. However, game model may be well used for modeling real world problems.
This is just a two-ply game/tree. But think sequentially, or, recursively.
The numbers can be seen as journey duration - then A is the MIN node - min value is the best (MAX) for me.
We can convert it to a classical MAX thinking by changing the Utilies to Working hours-delay - and we want to maximize the working hours.

Why? Actions may fail, . . . , getting to work



Random variable: Situation on rails R

r_1 free rails

r_2 accident

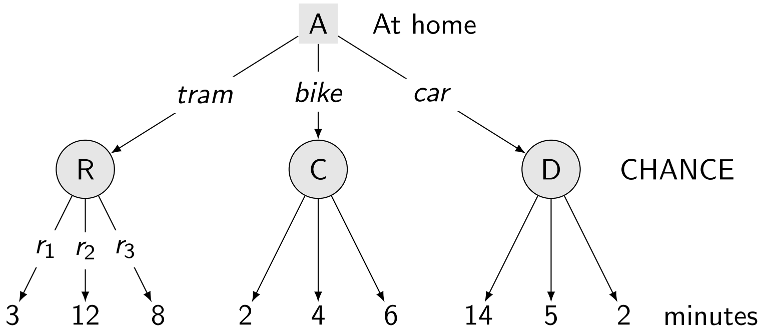
r_3 congestion

MAX/MIN depends on what the $r_?$ options and terminal numbers mean. The goal may be to get to work as fast as possible.

Notes

We talk about games. However, game model may be well used for modeling real world problems.
This is just a two-ply game/tree. But think sequentially, or, recursively.
The numbers can be seen as journey duration - then A is the MIN node - min value is the best (MAX) for me.
We can convert it to a classical MAX thinking by changing the Utilies to Working hours-delay - and we want to maximize the working hours.

Why? Actions may fail, . . . , getting to work



Random variable: Situation on rails R

r_1 free rails

r_2 accident

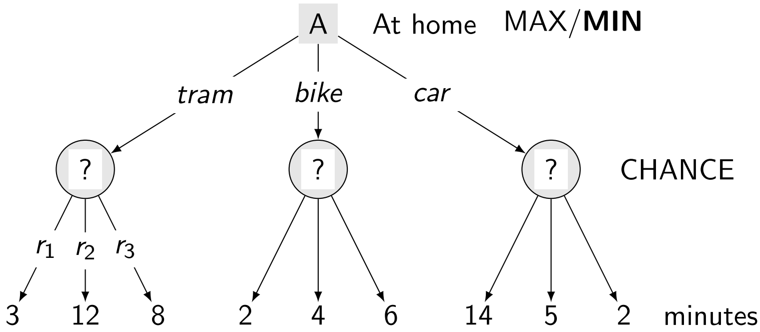
r_3 congestion

MAX/MIN depends on what the $r_?$ options and terminal numbers mean. The goal may be to get to work as fast as possible.

Notes

We talk about games. However, game model may be well used for modeling real world problems.
This is just a two-ply game/tree. But think sequentially, or, recursively.
The numbers can be seen as journey duration - then A is the MIN node - min value is the best (MAX) for me.
We can convert it to a classical MAX thinking by changing the Utilies to Working hours-delay - and we want to maximize the working hours.

Why? Actions may fail, . . . , getting to work



Random variable: Situation on rails R

r_1 free rails

r_2 accident

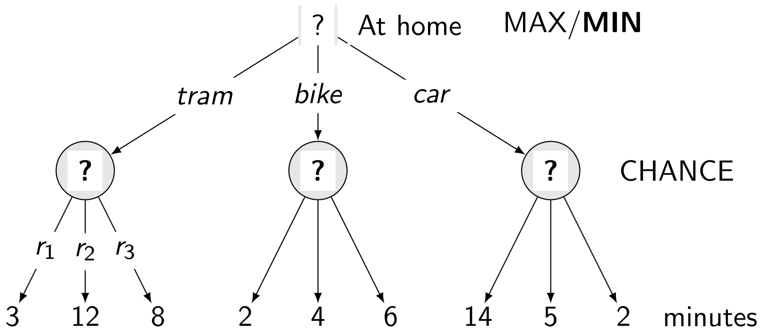
r_3 congestion

MAX/MIN depends on what the $r_?$ options and terminal numbers mean. The goal may be to get to work as fast as possible.

Notes

We talk about games. However, game model may be well used for modeling real world problems.
This is just a two-ply game/tree. But think sequentially, or, recursively.
The numbers can be seen as journey duration - then A is the MIN node - min value is the best (MAX) for me.
We can convert it to a classical MAX thinking by changing the Utilies to Working hours-delay - and we want to maximize the working hours.

Chance nodes values

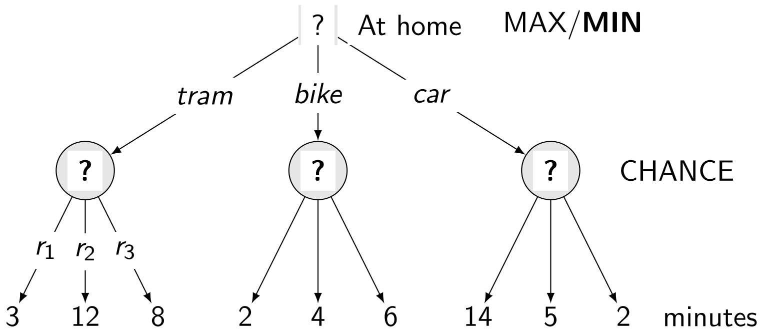


- Average case, not the worst case.
- Calculate expected utilities ...
- i.e. take weighted average (expectation) of successors

Notes

Later we will learn how to formalize all this as Markov Decision Processes.

Chance nodes values

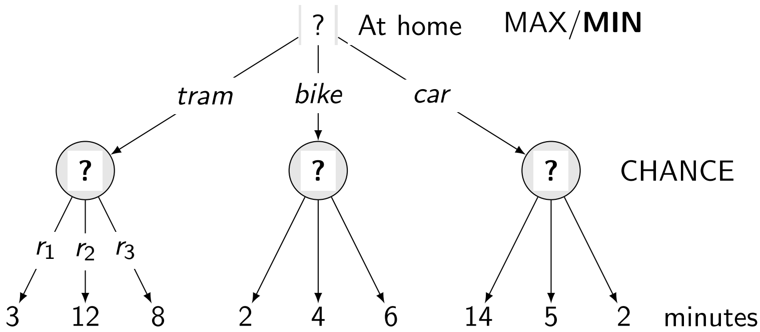


- Average case, not the worst case.
- Calculate expected utilities ...
- i.e. take weighted average (expectation) of successors

Notes

Later we will learn how to formalize all this as Markov Decision Processes.

Chance nodes values

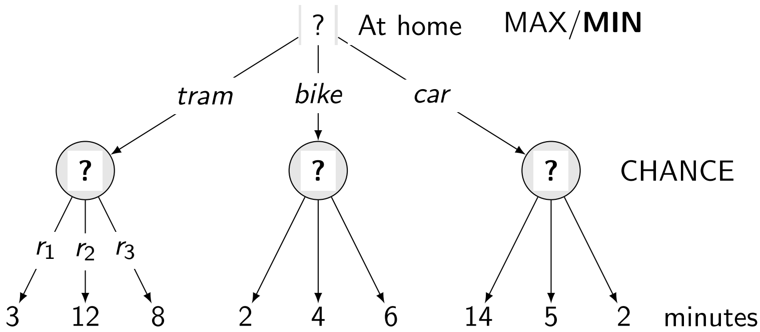


- Average case, not the worst case.
- Calculate expected utilities . . .
- i.e. take weighted average (expectation) of successors

Notes

Later we will learn how to formalize all this as Markov Decision Processes.

Chance nodes values



- Average case, not the worst case.
- Calculate expected utilities . . .
- i.e. take weighted average (expectation) of successors

Notes

Later we will learn how to formalize all this as Markov Decision Processes.

Expectimax

function EXPECTIMAX(state) **return** a value
  if TERMINAL-TEST(state): **return** UTILITY(state)
  if state (next agent) is **MAX**: **return** MAX-VALUE(state)
  if state (next agent) is **CHANCE**: **return** EXP-VALUE(state)
end function

---

function MAX-VALUE(state) **return** value v
  $v \leftarrow -\infty$
  for a **in** ACTIONS(state) **do**
    $v \leftarrow$ **max**(v, EXPECTIMAX(RESULT(state,a)))
  end for
end function

---

function EXP-VALUE(state) **return** value v
  $v \leftarrow 0$
  for all $r \in$ random events **do**
    $v \leftarrow v + P(r)$ EXPECTIMAX(RESULT(state,r))
  end for
end function

Notes

The scheme very much resembles the MINIMAX algorithm. Before, we had the deterministic opponent – MIN node.

Expectimax

function EXPECTIMAX(state) **return** a value
  if TERMINAL-TEST(state): **return** UTILITY(state)
  if state (next agent) is **MAX**: **return** MAX-VALUE(state)
  if state (next agent) is **CHANCE**: **return** EXP-VALUE(state)
end function

---

function MAX-VALUE(state) **return** value v
  $v \leftarrow -\infty$
  for a **in** ACTIONS(state) **do**
    $v \leftarrow$ **max**(v, EXPECTIMAX(RESULT(state,a)))
  end for
end function

---

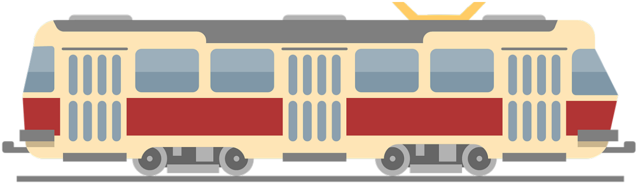
function EXP-VALUE(state) **return** value v
  $v \leftarrow 0$
  for all $r \in$ random events **do**
    $v \leftarrow v + P(r)$ EXPECTIMAX(RESULT(state,r))
  end for
end function

Notes

The scheme very much resembles the MINIMAX algorithm. Before, we had the deterministic opponent – MIN node.

Random variables, probability distribution, ...

- Random variable - an event with unknown outcome
- Probability distribution - assignment of weights to the outcomes



- Random variable: R - situation on rails
- Outcomes/events: $r \in \{\text{free rails, accindent, congestion}\}$
- Probability distribution: $P(R = \text{free rails}) = 0.3$, $P(R = \text{accident}) = 0.1$, $P(R = \text{congestion}) = 0.6$

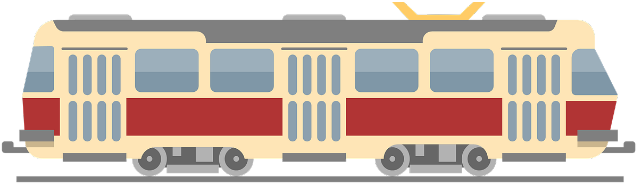
Few reminders from laws of probability, Probabilities:

- always non-negative,
- sum over all possible outcomes is equal to 1.

Notes

Random variables, probability distribution, ...

- Random variable - an event with unknown outcome
- Probability distribution - assignment of weights to the outcomes



- Random variable: R - situation on rails
- Outcomes/events: $r \in \{\text{free rails, accindent, congestion}\}$
- Probability distribution: $P(R = \text{free rails}) = 0.3$, $P(R = \text{accident}) = 0.1$, $P(R = \text{congestion}) = 0.6$

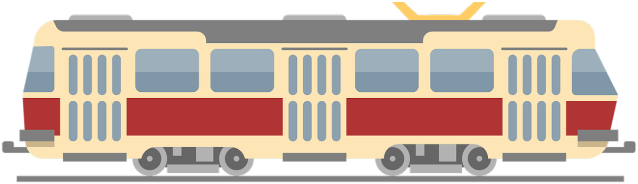
Few reminders from laws of probability, Probabilities:

- always non-negative,
- sum over all possible outcomes is equal to 1.

Notes

Random variables, probability distribution, . . .

- Random variable - an event with unknown outcome
- Probability distribution - assignment of weights to the outcomes



- Random variable: R - situation on rails
- Outcomes/events: $r \in \{\text{free rails, accindent, congestion}\}$
- Probability distribution: $P(R = \text{free rails}) = 0.3$, $P(R = \text{accident}) = 0.1$, $P(R = \text{congestion}) = 0.6$

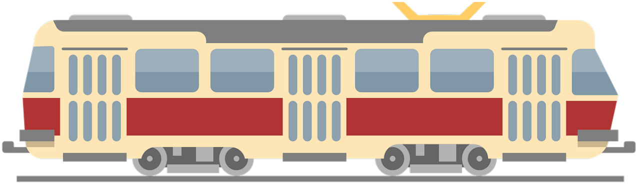
Few reminders from laws of probability, Probabilities:

- always non-negative,
- sum over all possible outcomes is equal to 1.

Notes

Random variables, probability distribution, ...

- Random variable - an event with unknown outcome
- Probability distribution - assignment of weights to the outcomes



- Random variable: R - situation on rails
- Outcomes/events: $r \in \{\text{free rails, accindent, congestion}\}$
- Probability distribution: $P(R = \text{free rails}) = 0.3$, $P(R = \text{accident}) = 0.1$, $P(R = \text{congestion}) = 0.6$

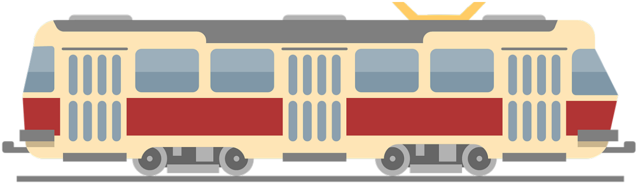
Few reminders from laws of probability, Probabilities:

- always non-negative,
- sum over all possible outcomes is equal to 1.

Notes

Random variables, probability distribution, ...

- Random variable - an event with unknown outcome
- Probability distribution - assignment of weights to the outcomes



- Random variable: R - situation on rails
- Outcomes/events: $r \in \{\text{free rails, accindent, congestion}\}$
- Probability distribution: $P(R = \text{free rails}) = 0.3$, $P(R = \text{accident}) = 0.1$, $P(R = \text{congestion}) = 0.6$

Few reminders from laws of probability, Probabilities:

- always non-negative,
- sum over all possible outcomes is equal to 1.

Notes

Expectations, . . .

How long does it take to go to work by tram?

- Depends on the random variable R - situation on rails with possible events r_1, r_2, r_3.
- What is the expectation of the time?

$$t = P(r_1)t_1 + P(r_2)t_2 + P(r_3)t_3$$

Weighted average.

Notes

The Expectation is a kind of long-horizon/many-realizations value. Think about trials/simulations.

Expectations, . . .

How long does it take to go to work by tram?

- Depends on the random variable R - situation on rails with possible events r_1, r_2, r_3.
- What is the expectation of the time?

$$t = P(r_1)t_1 + P(r_2)t_2 + P(r_3)t_3$$

Weighted average.

Notes

The Expectation is a kind of long-horizon/many-realizations value. Think about trials/simulations.

How about the Reversi game?

- Is there any space for randomness?
- Is the opponent really greedy and clever enough?

- Hope for chance when there is adversarial world – Dangerous optimism .
- Assuming worst case even if it is not likely – Dangerous pessimism .

Notes

Is there any space for randomness? Is the opponent really greedy and clever enough?
Dangerous optimism vs dangerous pessimism.

- Hoping for / assuming chance when the world is adversarial.
- Assuming the worst case – even if it is not likely

How about the Reversi game?

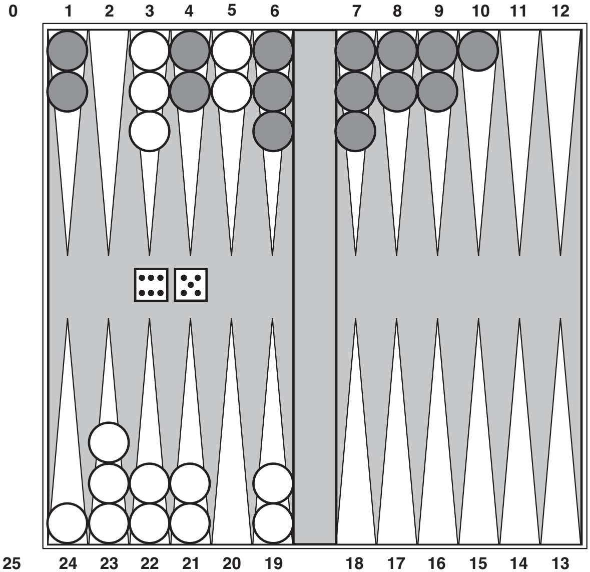
- Is there any space for randomness?
- Is the opponent really greedy and clever enough?
- Hope for chance when there is adversarial world – Dangerous optimism .
- Assuming worst case even if it is not likely – Dangerous pessimism .

Notes

Is there any space for randomness? Is the opponent really greedy and clever enough?
Dangerous optimism vs dangerous pessimism.

- Hoping for / assuming chance when the world is adversarial.
- Assuming the worst case – even if it is not likely

Games with chance **and** strategy

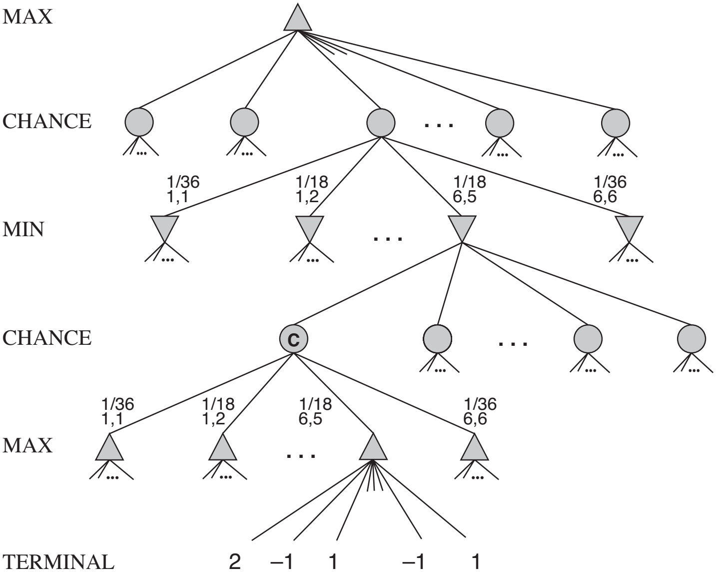


Notes

Read the rules at: https://en.wikipedia.org/wiki/Backgammon or elsewhere.
White moves clockwise - toward 25, black counterclockwise - toward 0.
Moving out from infield only after all stones are there.
No move to position where more than one opp stone.

One stone can be captured (see position 10)

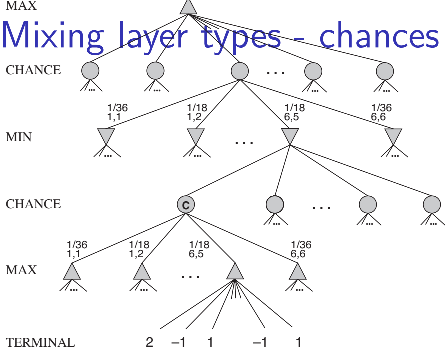
Mixing MAX, CHANCE, and MIN nodes

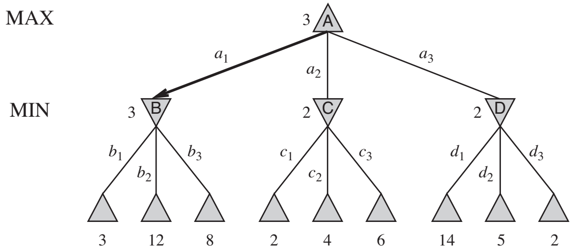


Notes

What the probabilities, what do they mean? Here, they represent solely the randomness (tossing dice).

Mixing layer types - chances inserted



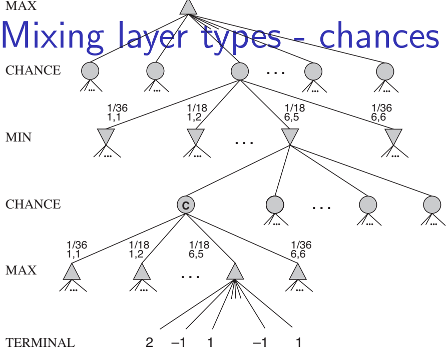


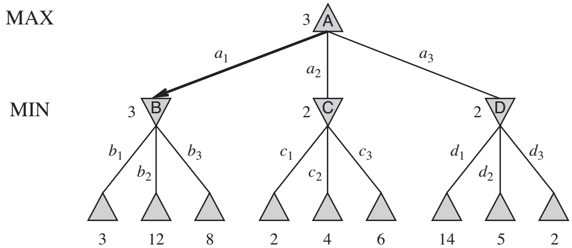
Extra random agent that moves after each MAX and MIN agent

$$\text{EXPECTIMINIMAX}(s) = \begin{cases} \text{UTILITY}(s) & \text{if } \text{TERMINAL-TEST}(s) \\ \max_a \text{EXPECTIMINIMAX}(\text{RESULT}(s,a)) & \text{if } \text{PLAYER}(s) = \text{MAX} \\ \min_a \text{EXPECTIMINIMAX}(\text{RESULT}(s,a)) & \text{if } \text{PLAYER}(s) = \text{MIN} \\ \sum_r P(r)\text{EXPECTIMINIMAX}(\text{RESULT}(s,r)) & \text{if } \text{PLAYER}(s) = \text{CHANCE} \end{cases}$$

Notes

Mixing layer types - chances inserted



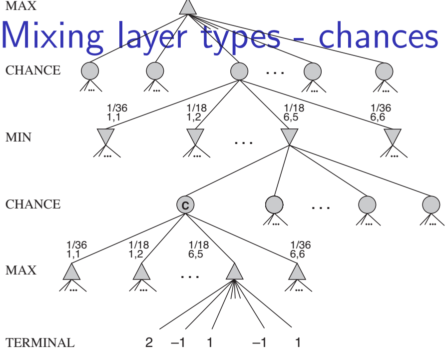


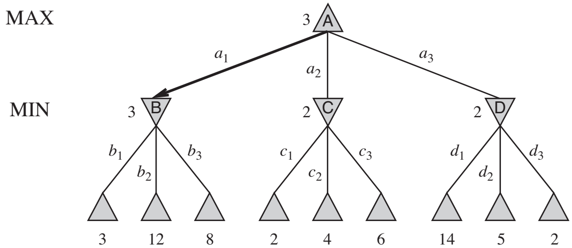
Extra random agent that moves after each MAX and MIN agent

$$\text{EXPECTIMINIMAX}(s) = \begin{cases} \text{UTILITY}(s) & \text{if } \text{TERMINAL-TEST}(s) \\ \max_a \text{EXPECTIMINIMAX}(\text{RESULT}(s,a)) & \text{if } \text{PLAYER}(s) = \text{MAX} \\ \min_a \text{EXPECTIMINIMAX}(\text{RESULT}(s,a)) & \text{if } \text{PLAYER}(s) = \text{MIN} \\ \sum_r P(r)\text{EXPECTIMINIMAX}(\text{RESULT}(s,r)) & \text{if } \text{PLAYER}(s) = \text{CHANCE} \end{cases}$$

Notes

Mixing layer types - chances inserted



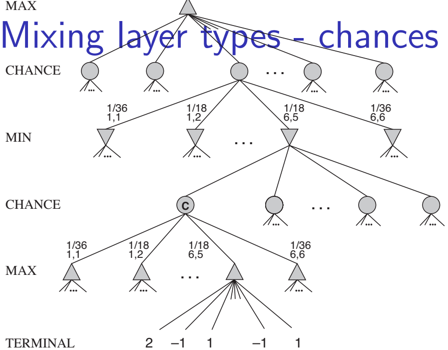


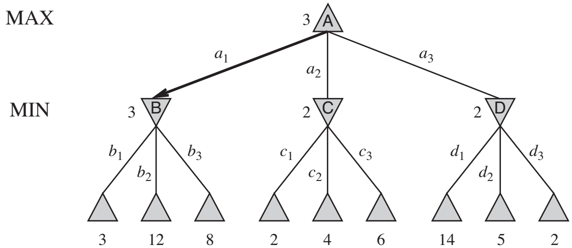
Extra random agent that moves after each MAX and MIN agent

$$\begin{aligned}
&\textsc{expectiminimax}(s) = \\
&\quad \textsc{utility}(s) \quad \text{if} \quad \textsc{terminal-test}(s) \\
&\quad \max_a \textsc{expectiminimax}(\textsc{result}(s,a)) \quad \text{if} \quad \textsc{player}(s) = \textsc{max} \\
&\quad \min_a \textsc{expectiminimax}(\textsc{result}(s,a)) \quad \text{if} \quad \textsc{player}(s) = \textsc{min} \\
&\quad \sum_r P(r)\textsc{expectiminimax}(\textsc{result}(s,r)) \quad \text{if} \quad \textsc{player}(s) = \textsc{chance}
\end{aligned}$$

Notes

Mixing layer types - chances inserted



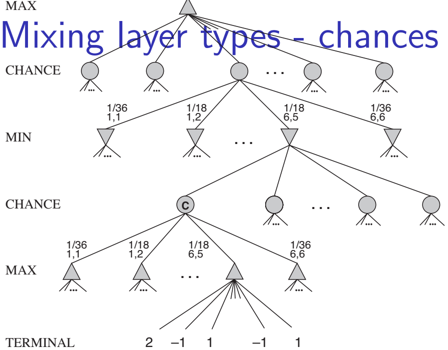


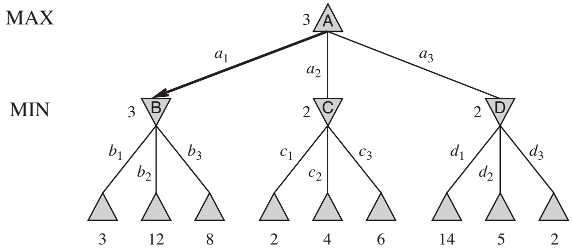
Extra random agent that moves after each MAX and MIN agent

$$
\begin{aligned}
\text{EXPECTIMINIMAX}(s) = & \\
\text{UTILITY}(s) \quad & \text{if} \quad \text{TERMINAL-TEST}(s) \\
\max_a \text{EXPECTIMINIMAX}(\text{RESULT}(s, a)) \quad & \text{if} \quad \text{PLAYER}(s) = \text{MAX} \\
\min_a \text{EXPECTIMINIMAX}(\text{RESULT}(s, a)) \quad & \text{if} \quad \text{PLAYER}(s) = \text{MIN} \\
\sum_r P(r)\text{EXPECTIMINIMAX}(\text{RESULT}(s, r)) \quad & \text{if} \quad \text{PLAYER}(s) = \text{CHANCE}
\end{aligned}
$$

Notes

Mixing layer types - chances inserted



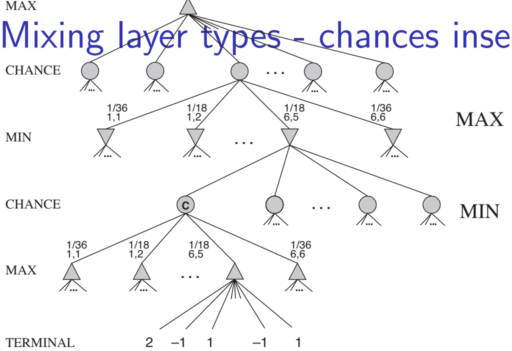


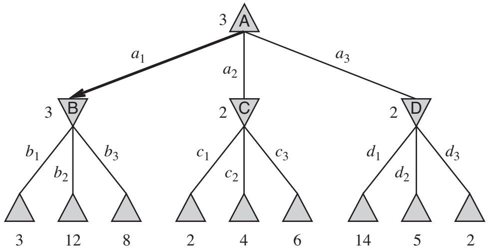
Extra random agent that moves after each MAX and MIN agent

$$
\text{EXPECTIMINIMAX}(s) = \begin{cases} \text{UTILITY}(s) & \text{if TERMINAL-TEST}(s) \\ \max_a \text{EXPECTIMINIMAX}(\text{RESULT}(s,a)) & \text{if PLAYER}(s) = \text{MAX} \\ \min_a \text{EXPECTIMINIMAX}(\text{RESULT}(s,a)) & \text{if PLAYER}(s) = \text{MIN} \\ \sum_r P(r)\text{EXPECTIMINIMAX}(\text{RESULT}(s,r)) & \text{if PLAYER}(s) = \text{CHANCE} \end{cases}
$$

Notes

Mixing layer types - chances inserted



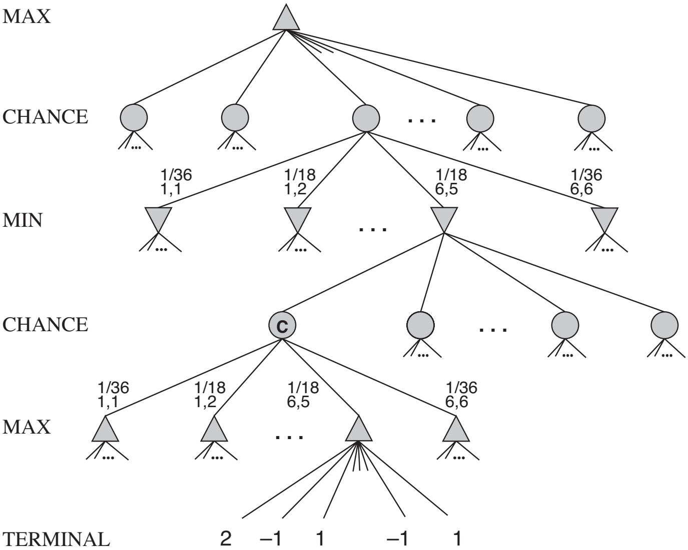


Extra random agent that moves after each MAX and MIN agent

$$
\text{EXPECTIMINIMAX}(s) = \begin{cases} \text{UTILITY}(s) & \text{if } \text{TERMINAL-TEST}(s) \\ \max_a \text{EXPECTIMINIMAX}(\text{RESULT}(s,a)) & \text{if } \text{PLAYER}(s) = \text{MAX} \\ \min_a \text{EXPECTIMINIMAX}(\text{RESULT}(s,a)) & \text{if } \text{PLAYER}(s) = \text{MIN} \\ \sum_r P(r) \text{EXPECTIMINIMAX}(\text{RESULT}(s,r)) & \text{if } \text{PLAYER}(s) = \text{CHANCE} \end{cases}
$$

Notes

Mixing chances into min/max tree, how big?



- b branching factor
- m maximum depth
- n number of distinct rolls

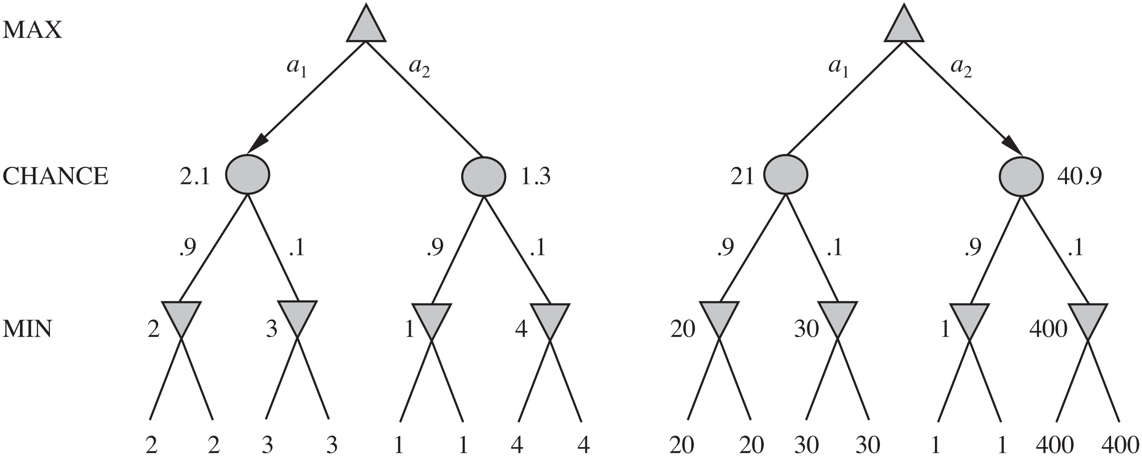
Notes

$$O(b^m n^m)$$

There are actually n^m different minimax trees. Each layer of n distinct rolls multiplies the number of min-max trees.
It is BIG! With roughly 20 legal moves in every position and 21 possible rolls of 2 dice, for expectimax search into depth = 2, we allready have:
$20 * (21 * 20)^3 = 1.2 * 10^9$ possibilities.
So we cannot get very far with search. At the same time, given the stochasticity, the fact that we cannot search so deep is less damaging.
We need an evaluation function.
Computer program for playing Backgammon – TD-Gammon, see Chapter 16.1[3] for thourough explanation. We will discuss the Reinforcement learning and learning of linear classifiers later in the course.

- depth 2
- good evaluation function + reinforcement learning
- 1st AI world champion in any game

Evaluation function

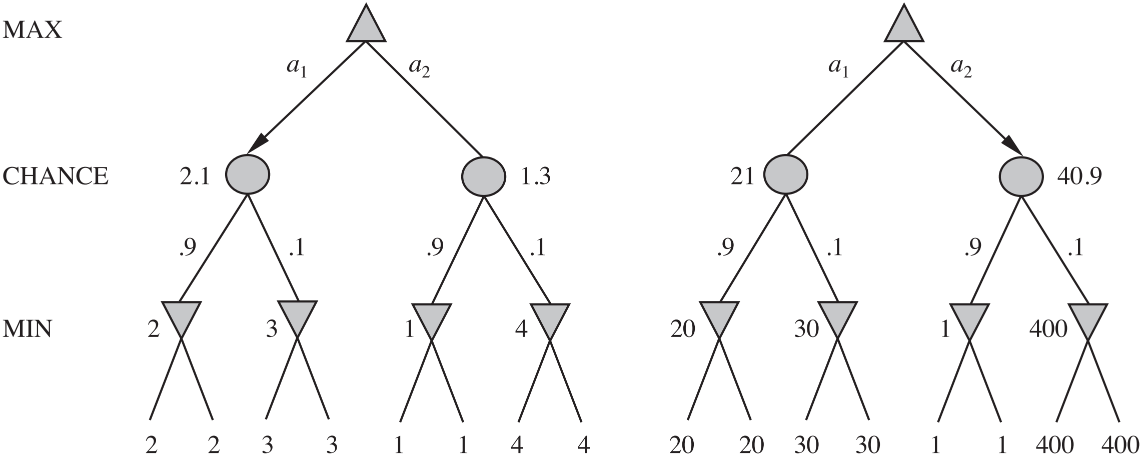


- Left: a_1 is the best. Right: a_2 is the best. Ordering of the (terminal) leaves is the same.
- Scale matters! Not only ordering.
- Can we prune the tree? (α, β like?)

Notes

About the scale, utilities, will be discussed in this lecture, later

Evaluation function

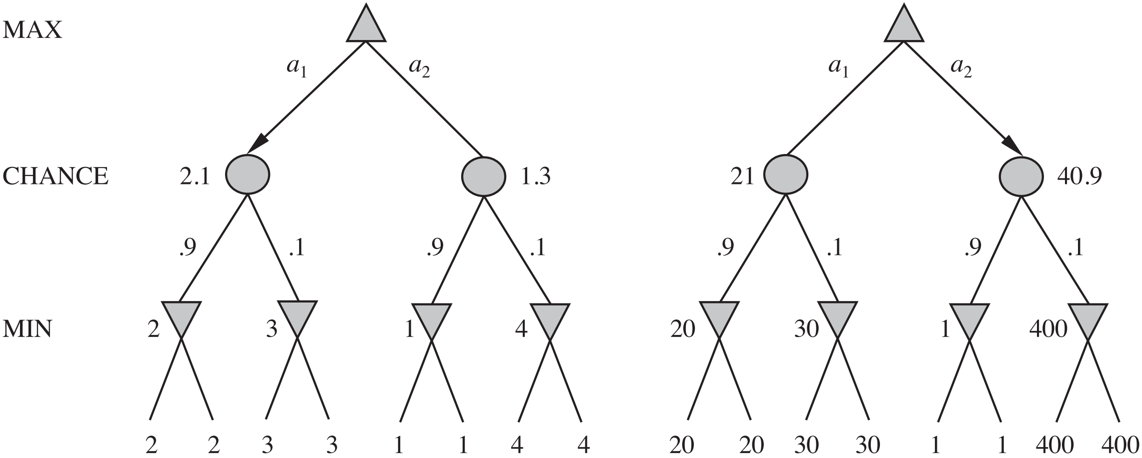


- Left: a_1 is the best. Right: a_2 is the best. Ordering of the (terminal) leaves is the same.
- Scale matters! Not only ordering.
- Can we prune the tree? (α, β like?)

Notes

About the scale, utilities, will be discussed in this lecture, later

Evaluation function

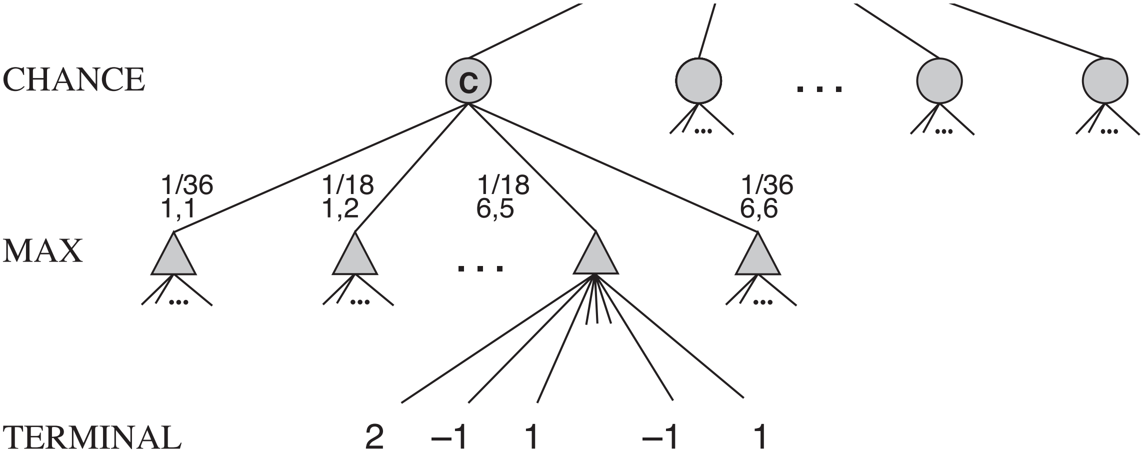


- Left: a_1 is the best. Right: a_2 is the best. Ordering of the (terminal) leaves is the same.
- Scale matters! Not only ordering.
- Can we prune the tree? (α, β like?)

Notes

About the scale, utilities, will be discussed in this lecture, later

Pruning expectiminimax tree

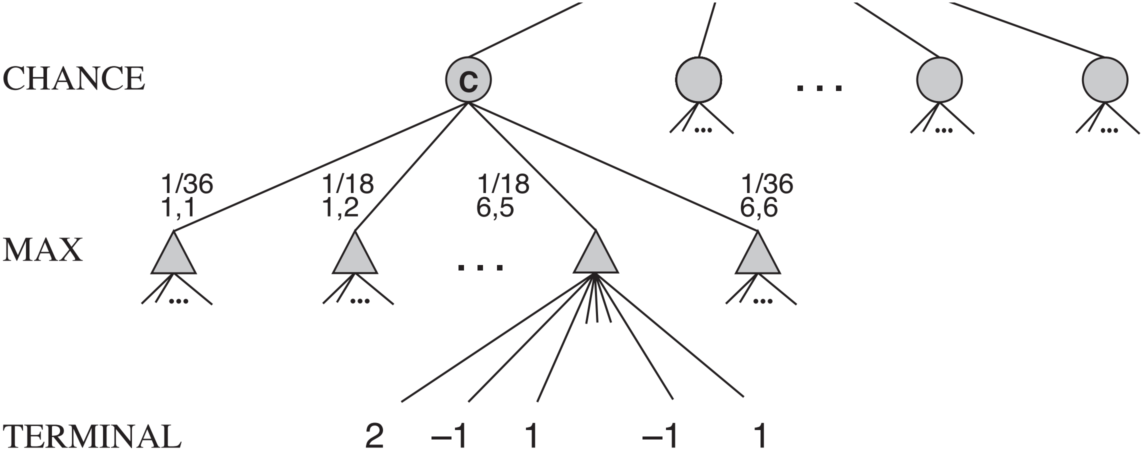


- Bounds on terminal utilities needed. Terminal values from −2 to 2.
- Monte Carlo simulation for evaluation a position (state).

Notes

Monte Carlo Simulation . From a given position play againts itself, many times, use random dice rolls. Collect results. Compute state value.

Pruning expectiminimax tree

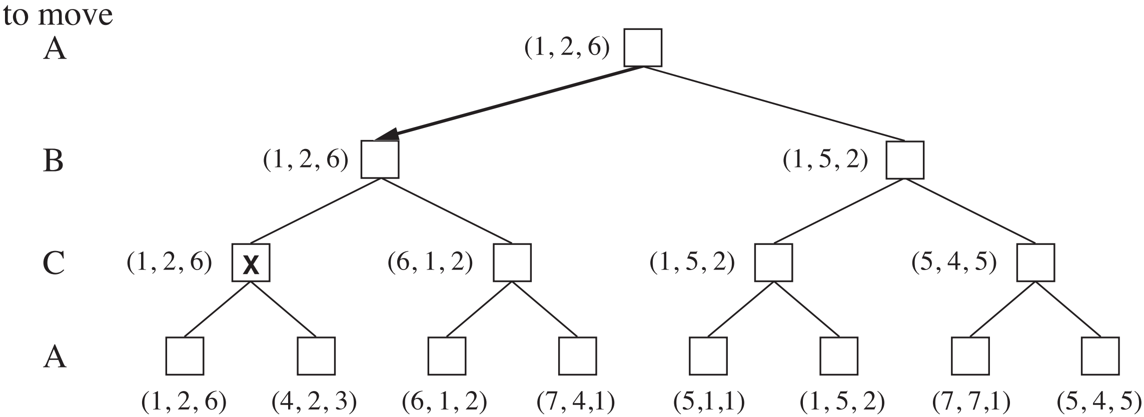


- Bounds on terminal utilities needed. Terminal values from −2 to 2.
- Monte Carlo simulation for evaluation a position (state).

Notes

Monte Carlo Simulation . From a given position play againts itself, many times, use random dice rolls. Collect results. Compute state value.

Multi-player games

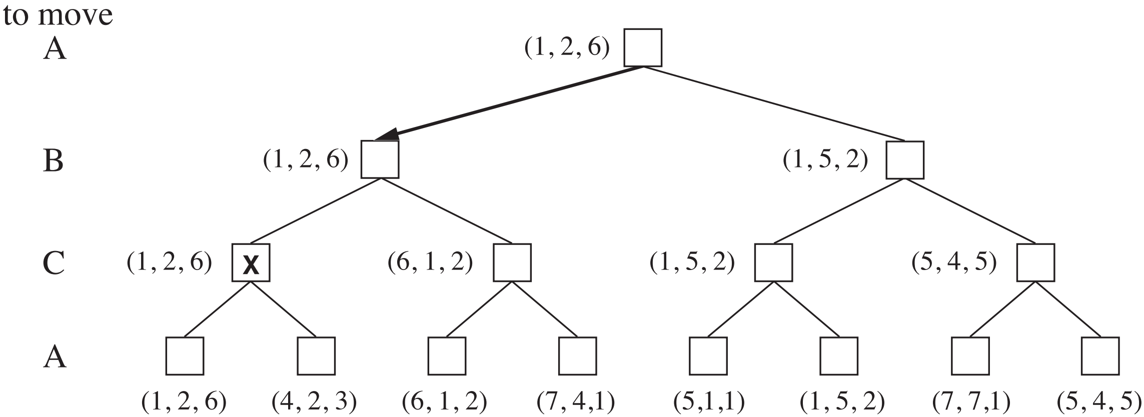


- Utility tuples
- Each player maximizes its own
- Coalitions, cooperations, competitions may be dynamic

Notes

I bet everybody remembers playing this kind of game . . . Remember the games you played when being kids.

Multi-player games



- Utility tuples
- Each player maximizes its own
- Coalitions, cooperations, competitions may be dynamic

Notes

I bet everybody remembers playing this kind of game . . . Remember the games you played when being kids.

Uncertainty recap

- Uncertain outcome of an action.
- Robot/Agent may not know the current state!

Notes

What is state for the robot?

- inner state of the robot (interoceptive measurement)
  - speed
  - inclination, orientation (N,E,S,W)
  - battery status
  - ...
- environment (exteroceptive measurement/sensing)
  - terrain profile close to robot
  - robot position within the world frame
  - ...

All of this may influence the decision about the best next action(s).

Uncertainty recap

- Uncertain outcome of an action.
- Robot/Agent may not know the current state!

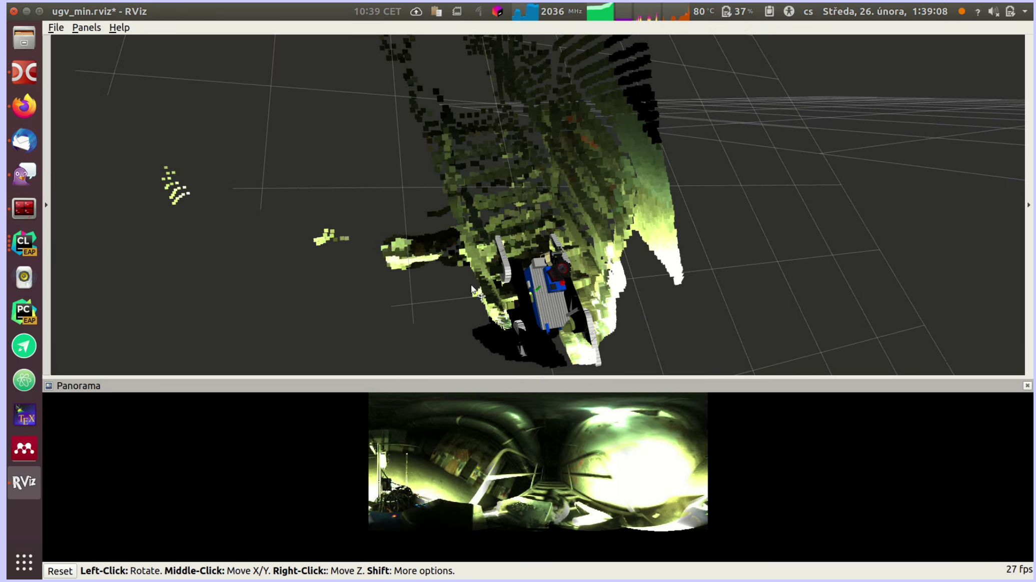
Notes

What is state for the robot?

- inner state of the robot (interoceptive measurement)
  - speed
  - inclination, orientation (N,E,S,W)
  - battery status
  - ...
- environment (exteroceptive measurement/sensing)
  - terrain profile close to robot
  - robot position within the world frame
  - ...

All of this may influence the decision about the best next action(s).

Uncertain outcome of an action



Video: Climbing stairs failure, From: http://robotics.fel.cvut.cz/cras/darpa-subt/

Notes

Climbing up, rear flipper got too weak, gave up supporting the robot and it flipped back. Reason uknown, the robot climbed up similar stairs successfully many times.

Uncertain, partially observable environment

- Currents state s may be unknown, observations $\mathbf{e}$
- Uncertain outcome, random variable $\textsc{Result}(a)$
- Probability of outcome s' given $\mathbf{e}$ is

$$P(\textsc{Result}(a) = s' | a, \mathbf{e})$$

- Utility function $U(s)$ corresponds to agent preferences.
- Expected utility of an action a given $\mathbf{e}$:

$$EU(a|\mathbf{e}) = \sum_{s'} P(\textsc{Result}(a) = s' | a, \mathbf{e}) U(s')$$

Amatrice, Italy, 2016.

Notes

See [2], Ch. 16 Making simple decisions.

Rational agent

Agent's expected utility of an action a given $\mathbf{e}$:

$$EU(a|\mathbf{e}) = \sum_{s'} P(\text{RESULT}(a) = s'|a, \mathbf{e})U(s')$$

What should a rational agent do?

Is it then all solved? Do we know all what we need?

- $P(\text{RESULT}(a) = s'|a, \mathbf{e})$
- $U(s')$

Notes

Well, obviously take the action that maximizes the expected utility.

Complete causal model is needed to compute the probabilites P, and a complete search/planning to the end required for computing the utility U. And, eh, the state space may be, and often is, infinite. Enough pessimism, we will come back to this in next lectures/courses.

Rational agent

Agent's expected utility of an action a given $\mathbf{e}$:

$$EU(a|\mathbf{e}) = \sum_{s'} P(\textrm{RESULT}(a) = s'|a, \mathbf{e})U(s')$$

What should a rational agent do?
Is it then all solved? Do we know all what we need?

- $P(\textrm{RESULT}(a) = s'|a, \mathbf{e})$
- $U(s')$

Notes

Well, obviously take the action that maximizes the expected utility.

Complete causal model is needed to compute the probabilites P, and a complete search/planning to the end required for computing the utility U. And, eh, the state space may be, and often is, infinite. Enough pessimism, we will come back to this in next lectures/courses.

Rational agent

Agent's expected utility of an action a given $\mathbf{e}$:

$$EU(a|\mathbf{e}) = \sum_{s'} P(\textsc{Result}(a) = s'|a, \mathbf{e})U(s')$$

What should a rational agent do?
Is it then all solved? Do we know all what we need?

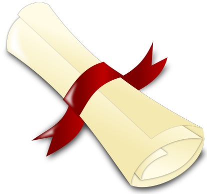
- $P(\textsc{Result}(a) = s'|a, \mathbf{e})$
- $U(s')$

Notes

Well, obviously take the action that maximizes the expected utility.

Complete causal model is needed to compute the probabilites P, and a complete search/planning to the end required for computing the utility U. And, eh, the state space may be, and often is, infinite. Enough pessimism, we will come back to this in next lectures/courses.

Utilities

- Where do utilities come frome?
- Does averaging make sense?
- Do they exist?
- What if our preferences can't be described by utilities?

Notes

Before we start solving all this let's talk about utilities, where do they come from, are they unique, ... Actually, let's talk about preferences first, we all have some preferences . Later, we will derive utilities from them.

Agent/Robot Preferences

- Prizes A, B
- Lottery: uncertain prizes $L = [p, A; (1-p), B]$

Preference, indifference, ...

- Robot prefers A over B: $A \succ B$
- Robot has no preferences: $A \sim B$
- in between: $A \succsim B$

Notes

You may use agent/robot/algorithm/. . . , according to your preferences.

Lottery can be seen as a chance node.

Agent/Robot Preferences

- Prizes A, B
- Lottery: uncertain prizes $L = [p, A; (1-p), B]$

Preference, indifference, ...

- Robot prefers A over B: $A \succ B$
- Robot has no preferences: $A \sim B$
- in between: $A \succsim B$

Notes

You may use agent/robot/algorithm/..., according to your preferences.

Lottery can be seen as a chance node.

Rational preferences

- Transitivity: $(A \succ B) \wedge (B \succ C) \Rightarrow (A \succ C)$
- Completeness: $(A \succ B) \vee (B \succ A) \vee (A \sim B)$
- Continuity: $(A \succ B \succ C) \Rightarrow \exists p\, [p, A; 1-p, C] \sim B$
- Substituability: $A \sim B \Rightarrow [p, A; 1-p, C] \sim [p, B; 1-pC]$. The same for $\succ$ and $\sim$.
- Monotonocity: $A \succ B \Rightarrow (p > q) \Leftrightarrow [p, A; 1-p, B] \succ [q, A; 1-q, B]$. Agent must prefer a lottery with higher chance to win.
- Decomposability, compressing compound lotteries into one: $[p, A; 1-p, [q, B; 1-q, C]] \sim [p, A; (1-p)q, B; (1-p)(1-q), C]$

Axioms of utility theory.
Motivation: if agent/robot violates an axiom $\Rightarrow$ irrational agent/robot.

Notes

If you think it through you will see that the properties of rational preferences are quite logical, *rational* if you want ;-)

- Transitivity: $(A \succ B) \wedge (B \succ C) \Rightarrow (A \succ C)$
- Completeness: $(A \succ B) \vee (B \succ A) \vee (A \sim B)$
- Continuity: $(A \succ B \succ C) \Rightarrow \exists p\, [p, A; 1-p, C] \sim B$
- Substituability: $A \sim B \Rightarrow [p, A; 1-p, C] \sim [p, B; 1-pC]$. The same for $\succ$ and $\sim$.
- Monotonocity: $A \succ B \Rightarrow (p > q) \Leftrightarrow [p, A; 1-p, B] \succ [q, A; 1-q, B]$. Agent must prefer a lottery with higher chance to win.
- Decomposability, compressing compound lotteries into one: $[p, A; 1-p, [q, B; 1-q, C]] \sim [p, A; (1-p)q, B; (1-p)(1-q), C]$

Any agent that breaks the rules can be shown irrational.

Rational preferences

- Transitivity: $(A \succ B) \land (B \succ C) \Rightarrow (A \succ C)$
- Completeness: $(A \succ B) \lor (B \succ A) \lor (A \sim B)$
- Continuity: $(A \succ B \succ C) \Rightarrow \exists p \, [p, A; 1-p, C] \sim B$
- Substituability: $A \sim B \Rightarrow [p, A; 1-p, C] \sim [p, B; 1-pC]$. The same for $\succ$ and $\sim$.
- Monotonocity: $A \succ B \Rightarrow (p > q) \Leftrightarrow [p, A; 1-p, B] \succ [q, A; 1-q, B]$. Agent must prefer a lottery with higher chance to win.
- Decomposability, compressing compound lotteries into one:
  $[p, A; 1-p, [q, B; 1-q, C]] \sim [p, A; (1-p)q, B; (1-p)(1-q), C]$

Axioms of utility theory.
Motivation: if agent/robot violates an axiom $\Rightarrow$ irrational agent/robot.

Notes

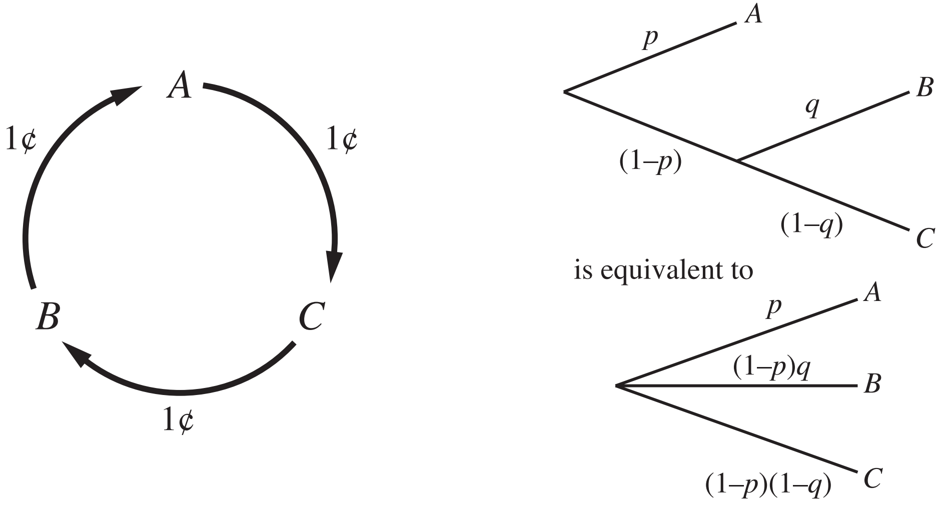
If you think it through you will see that the properties of rational preferences are quite logical, *rational* if you want ;-)

- Transitivity: $(A \succ B) \land (B \succ C) \Rightarrow (A \succ C)$
- Completeness: $(A \succ B) \lor (B \succ A) \lor (A \sim B)$
- Continuity: $(A \succ B \succ C) \Rightarrow \exists p \, [p, A; 1-p, C] \sim B$
- Substituability: $A \sim B \Rightarrow [p, A; 1-p, C] \sim [p, B; 1-pC]$. The same for $\succ$ and $\sim$.
- Monotonocity: $A \succ B \Rightarrow (p > q) \Leftrightarrow [p, A; 1-p, B] \succ [q, A; 1-q, B]$. Agent must prefer a lottery with higher chance to win.
- Decomposability, compressing compound lotteries into one:
  $[p, A; 1-p, [q, B; 1-q, C]] \sim [p, A; (1-p)q, B; (1-p)(1-q), C]$

Any agent that breaks the rules can be shown irrational.

Transitivity and decomposability

Goods A, B, C and (nontransitive) preferences of an (irrational) agent $A \succ B \succ C \succ A$.



(a) (b)

Notes

A, B, C are goods. Suppose an agent has A. As the agent prefers $C \succ A$ we offer him/her the exchange plus the gant gives one cent (the smalles currency unit). The same for $B \succ C$, and $A \succ B$. At the end of the round the agent has A again but also 3 cents less. And this can contiune until the pure agent has no money at all.

Maximum expected utility principle

Given the rational preferences (contraints), there exists a real valued function u such that:

$$u(A) > u(B) \quad \Leftrightarrow \quad A \succ B$$
$$u(A) = u(B) \quad \Leftrightarrow \quad A \sim B$$

Expected utility of a Lotery L (outcomes s_i with probabilities p_i):

$$L([p_1, S_1; \cdots; p_n, S_n]) = \sum_i p_i u(S_i)$$

Proof in [4].
Is a utility u function unique?

Notes

In other words, we can find a utility to any preferences.
No, it is not unique:

$$u'(S) = au(S) + b$$

$a > 0$ makes the agent behavior the same. Think about Fahrenheit to Celsius conversion.

Maximum expected utility principle

Given the rational preferences (contraints), there exists a real valued function u such that:

$$u(A) > u(B) \quad \Leftrightarrow \quad A \succ B$$
$$u(A) = u(B) \quad \Leftrightarrow \quad A \sim B$$

Expected utility of a Lotery L (outcomes s_i with probabilities p_i):

$$L([p_1, S_1; \cdots ; p_n, S_n]) = \sum_i p_i u(S_i)$$

Proof in [4].

Is a utility u function unique?

Notes

In other words, we can find a utility to any preferences.
No, it is not unique:

$$u'(S) = au(S) + b$$

$a > 0$ makes the agent behavior the same. Think about Fahrenheit to Celsius conversion.

Maximum expected utility principle

Given the rational preferences (contraints), there exists a real valued function u such that:

$$u(A) > u(B) \quad \Leftrightarrow \quad A \succ B$$
$$u(A) = u(B) \quad \Leftrightarrow \quad A \sim B$$

Expected utility of a Lotery L (outcomes s_i with probabilities p_i):

$$L([p_1, S_1; \cdots ; p_n, S_n]) = \sum_i p_i u(S_i)$$

Proof in [4].
Is a utility u function unique?

Notes

In other words, we can find a utility to any preferences.
No, it is not unique:

$$u'(S) = au(S) + b$$

$a > 0$ makes the agent behavior the same. Think about Fahrenheit to Celsius conversion.

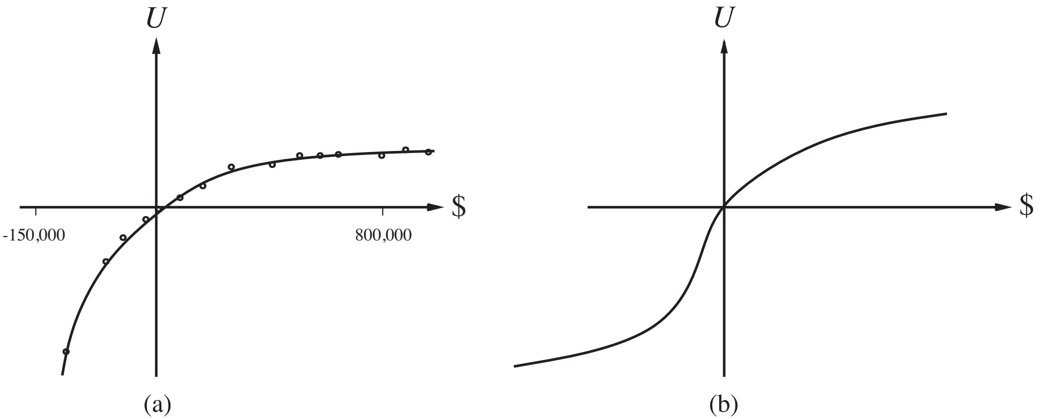
Human utilities

Notes

Utility of money

You triumphed in a TV show!

a) Take $\$1,000,000$ … or
b) Flip a coin and loose all or win $\$2,500,000$



The (human) Utility of money. (Left) Data for Mr. Beard from Grayson (1960) study. (Right) Full curve.

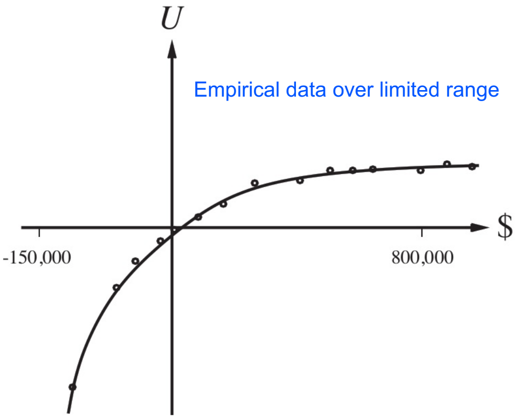
Notes

Lottery b) Expected monetary value (EMP) vs. utility. Clearly EMP(b) is bigger than EMP(a). But what about the (human) Utility?

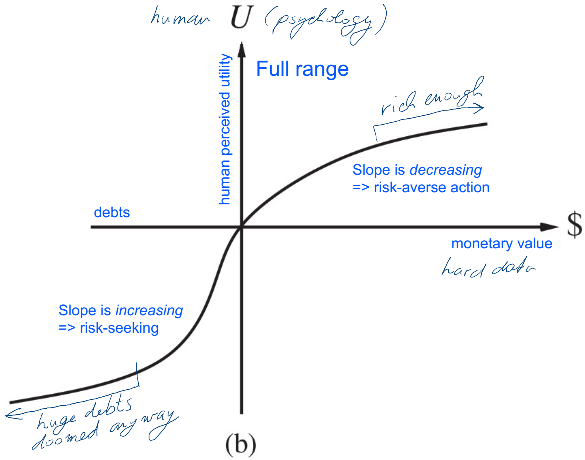
$$
\begin{aligned}
u(a) &= u(S_{k+1,000,000}) \\
u(b) &= \frac{1}{2}u(S_k) + \frac{1}{2}u(S_{k+2,500,000}),
\end{aligned}
$$

where S_k is the state of possessing $k\$$ (current wealth).
E.g., imagine $u(S_k) = 5$, $u(S_{k+1000000}) = 8$, $U(S_{k+2500000}) = 9$. Then the rational decision is to decline the gamble.

Utility of the money human psychology vs. hard data



(a)



(b)

Notes

Based on empirical studies, the human utility of money is rather logarithmic. People are in general *risk-averse*. This also motivates insurances.

References I

Some figures from [2], Chapters 5, 16. Human utilities are discussed in [1]. This lecture has been also greatly inspired by the 7th lecture of CS 188 at http://ai.berkeley.edu as it convenietly bridges the world of deterministic search and sequential decisions in uncertain worlds.

[1] Daniel Kahneman.
Thinking, Fast and Slow.
Farrar, Straus and Giroux, 2011.

[2] Stuart Russell and Peter Norvig.
Artificial Intelligence: A Modern Approach.
Prentice Hall, 3rd edition, 2010.
http://aima.cs.berkeley.edu/.

Notes

References II

[3] Richard S. Sutton and Andrew G. Barto.
Reinforcement Learning; an Introduction.
MIT Press, 2nd edition, 2018.
http://www.incompleteideas.net/book/the-book-2nd.html.

[4] John von Neumann and Oskar Morgenstern.
Theory of Games and Economic Behavior.
Princeton, 1944.
https://en.wikipedia.org/wiki/Theory_of_Games_and_Economic_Behavior, Utility theorem.

Notes