TEORIE:

name due to a lot of shaking

Co dělal AI systém Shakey?

general purpose robot, computer vision, path planning with A*

Uspořádejte následující AI systémy od nejstaršího po nejnovější:

DeepBlue
AlphaGo
Shakey
AlphaStar

Shakey
Deep Blue
AlphaGo
Alpha Star

Kde se poprvé objevilo slovní spojení "artificial intelligence"?

A. v disertační práci Allana Turinga
B. v pracích Leonarda da Vinci
C. v návrhu konání workshopu v Dartmouth

Co byl systém MYCIN?

recommendation of antibiotics (60% accuracy, which was still higher than a human)

Na jakém problému byl poprvé předveden algoritmus DQN?

A. Go
B. Jeopardy!
C. Šachy
D. Poker
E. Atari hry

Co dělal AI systém ELIZA?

chatbot, psychologist, pattern matching

Co je Turingův test a k čemu slouží?

quality of AI, CAPTCHA

Kolik položek má sdružená pravděpodobnostní distribuce nad *n* binárními proměnnými?

A. n^3
B. 2^n
C. n!
D. n
E. n^2

Které z následujících tvrzení jsou vždy pravdivé?

A. P(A, B) = P(A) * P(B)
B. P(A|B) = P(B|A)
C. P(A|B) = P(B|A) * P(A)

Které z následujících tvrzení jsou vždy pravdivé?

A. P(A, B) = P(B|A) * P(A)
B. P(A, B) = P(A) * P(B)
C. P(A|B) = P(B|A)

$$P(A|B) = \frac{P(B|A)}{P(A)}$$

Označte všechny operace, které musíme provést na STRIPS akci *a*, abychom vytvořili její relaxovanou variantu a^+.

A. Smažeme její cenu (cost)
B. Smažeme pozitivní efekty (add effects)
C. Smažeme její negativní efekty (delete effects)
D. Smažeme její předpoklady (preconditions)

$$P(A \cap B) = P(B|A)\,P(A)$$

Které z následujících tvrzení o automatickém plánování jsou pravdivé?

A. Plánování vyžaduje úplný formální popis problému.
B. Jeden z algoritmů běžně používaných v plánování je A*.
C. Korektní plán je typicky nalezen v čase polynomiálním ve velikosti popisu plánovacího problému.

Co je alfa-vektor?

A. Lineární funkce vyjadřující očekávanou pravděpodobnost pro zvolený stav světa a danou strategii v závislosti na měnícím se beliefu agenta.
B. Lineární funkce vyjadřující očekávaný zisk pro danou strategii agenta v závislosti na měnícím se beliefu agenta.
C. Lineární funkce vyjadřující očekávaný zisk pro iniciální belief agenta v závislosti na měnící se strategii

Vyznačte ta tvrzení o POMDP, která jsou pravdivá

A. Belief je pravděpodobnostní distribuce nad stavy.
B. alfa-vektor odpovídá očekávané hodnotě vždy jedné akce agenta.
C. Hodnota v iniciálním beliefu se v případě algoritmu iterace hodnot (value iteration) blíží optimální hodnotě shora.

Vyznačte ta tvrzení o POMDP, která jsou pravdivá

A. Belief je pravděpodobnostní distribuce nad akcemi.
B. Množina všech dosažitelných pravděpodobnostních distribucí, které odpovídají možným belief stavům v problému, nezávisí na strategii agenta.
C. POMDP odpovídají MDP se spojitým stavovým prostorem.

Vyznačte které z následujících tvrzení jsou pravdivé

A. Prohledávání do šířky vždy nalezne řešení, pokud toto existuje.
B. Každá konzistentní heuristika je také přípustní heuristikou.
C. Počet nesprávně seřazených čísel (tj.
D. menší číslo je za větším) je přípustnou heuristikou pro 8-puzzle problém.
E. Prohledávání do hloubky má vyšší paměťové nároky než prohledávání do šířky.
F. Předpokládejme, že všechny ceny (costs) akcí jsou celá čísla a jsou striktně pozitivní. Funkce h(x) = 1 pro všechny stavy x, kromě cíle, je přípustnou funkcí.

Vyznačte které z následujících tvrzení jsou pravdivé

A. Prohledávání do hloubky má vyšší paměťové nároky, než prohledávání do šířky.
B. Prohledávání do hloubky najde vždy optimální řešení i v nekonečném stavovém prostoru.
C. Každá konzistentní heuristika je také přípustní heuristikou.
D. Předpokládejme, že všechny ceny (costs) akcí jsou celá čísla a jsou striktně pozitivní. Funkce h(x) = 1 pro všechny stavy x, kromě cíle, je přípustnou heuristikoufunkcí.
E. Počet nesprávně umístěných čísel je přípustnou heuristikou pro 8-puzzle problém.

Vyznačte které z následujících tvrzení jsou pravdivé

A. Prohledávání do šířky (BFS) vždy expanduje méně uzlů než je prohledávání do hloubky (DFS)
B. A* musí používat konzistentní heuristiku, aby našel optimální řešení admissible
C. Za Předpokladu, že existuje řešení a všechny akce mají striktně pozitivní cenu (cost), iterativní prohledávání do hloubky IDA* s h(x)=0 pro všechny stavy vždy najde optimální řešení i v nekonečném stavovém prostoru.
D. Počet nesprávně umístěných čísel je přípustnou heuristikou pro 8-puzzle problém
E. Manhattanská vzdálenost je přípustná heuristika pro plánování robota v bludišti (na mřížce) pokud se robot může pohybovat všemi 8 směry

Vyznačte které z následujících tvrzení jsou pravdivé (dopsat)

A. Počet nesprávně seřazených čísel (tj. menší číslo je za větším) je přípustnou heuristikou pro 8-puzzle problém.
B. Každá přípustní heuristika je také konzistentní heuristikou.
C. Obousměrné (bidirectional) informované prohledávání vždy expanduje méně uzlů než jednosměrné prohledávání
D. Počet nesprávně umístěných čísel je přípustnou heuristikou pro 8-puzzle problém
E. Za předpokladu, že existuje řešení a všechny ceny (costs) za akce se rovnají 2, prohledávání do šířky vždy najde optimální řešení i v nekonečném stavovém prostoru

Question # 15 **Type: Multiple answers, Maximum score: 7, Score:4.2**

Mějme problém s vysáváním místnosti reprezentované jako obdélník s $X \times Y$ čtvercovými políčky označenými jako (x, y). Robotický vysavač musí vyčistit nečistoty na těchto polích. Množství nečistot je reprezentováno pomocí hodnoty od 0 do 100 u každého pole (x, y). Zásobník robotického vysavače má kapacitu pro nečistoty m, tento zásobník lze vysypat během návštěvy základny umístěné na poli $(0, 0)$.

Počáteční místo pro robotický vysavač je $(0, 0)$ a můžete předpokládat, že robotický vysavač je prázdný. V jednom kroku se robotický vysavač může pohybovat z aktuálního pole na sousední ve 4 směrech (nahoru, dolů, vlevo, vpravo) nebo vysát 1 jednotku nečistot.

Cílem je najít nejkratší posloupnost pohybu, která vyčistí celou místnost a vrátit vysavač do základny na políčku $(0, 0)$.

Nechť n_{ij} je množství nečistot poli (i, j). Jsou následující funkce přípustné heuristické funkce pro algoritmus A^* pro tento problém, nebo ne?

- $h = \max_{i \in X, j \in Y} n_{ij}$
- $h = \max_{i \in X, j \in Y} sgn(n_{ij}) \cdot (\max\{i, j\})$
- $h = \sum_{i \in X, j \in Y} (i + j)$
- $h = \sum_{i \in X, j \in Y} n_{ij}$
- $h = \max_{i \in X, j \in Y} sgn(n_{ij}) \cdot (i + j)$
- $h = \sum_{i \in X, j \in Y} sgn(n_{ij}) \cdot (\max\{i, j\})$
- $h = \sum_{i \in X, j \in Y} sgn(n_{ij})$

Question # 15 Type: Multiple answers, Maximum score: 7, Score:7

Mějme problém s vysáváním místnosti reprezentované jako obdélník s $X \times Y$ čtvercovými políčky označenými (x, y). k robotických vysavačů musí vyčistit nečistoty na těchto polích. Množství nečistot je reprezentováno pomocí hodnoty od 0 do 100 u každého pole (x, y).

Zásobník každého robotického vysavače má kapacitu pro nečistoty m, tento zásobník lze vysypat během návštěvy základny umístěné na poli $(0, 0)$. Počáteční místo pro každý robotický vysavač je náhodně, ale pevně dané někde v místnosti. Můžete předpokládat, že každý robotický vysavač je prázdný.

V jednom kroku se každý robotický vysavač může pohybovat z aktuálního pole na sousední ve 4 směrech (nahoru, dolů, vlevo, vpravo) nebo vysát 1 jednotku nečistot. Všechny vysavače se hýbou simultánně a na jednom políčku může být nejvýše jeden vysavač (toto neplatí pro políčko $(0, 0)$, kde může být zároveň více vysavačů).

Cílem problému je najít nejkratší posloupnost pohybu, která vyčistí celou místnost a vrátit všechny vysavače do základny na políčku $(0, 0)$.

Nechť n_{ij} je množství nečistot poli (i, j). Jsou následující funkce přípustné heuristické funkce pro algoritmus A^* pro tento problém nebo ne?

- $h = \max_{i \in X, j \in Y} n_{ij}$
- $h = \max_{i \in X, j \in Y} \frac{n_{ij}}{k} + (\max\{i, j\})$
- $h = \sum_{i \in X, j \in Y} (i + j)$
- $h = \sum_{i \in X, j \in Y} n_{ij}$
- $h = \max_{i \in X, j \in Y} \frac{n_{ij}}{k}$
- $h = \sum_{i \in X, j \in Y} sgn(n_{ij})$
- $h = \max_{i \in X, j \in Y} \frac{n_{ij}}{k} \cdot (\max\{i, j\})$

Algoritmus CSP

A. nemusí vždy najít řešení
B. hledá nejkratší cestu ze startu do cíle
C. hledá přípustné přiřazení hodnot proměnných z jejich domén

Která z následujících tvrzení jsou pravdivá?

A. Negamax prohledá stejné uzly herního stromu jako minimax.
B. Alpha-beta může prohledat méně různých uzlů herního stromu než Negascout.
C. Negascout navštíví uzly v herním stromě nejvýše dvakrát.

Která z následujících tvrzení jsou pravdivá?

A. UCT může prohledat i uzly, které alpha-beta nebude prohledávat.
B. Negascout vždy navštíví méně uzlů než minimax.
C. Negamax navštíví každý uzel v herním stromě nejvýše jednou.

Vyznačte, která z následujících tvrzení jsou pravdivé

A. Algoritmus Monte Carlo Tree Search staví vybalancovaný (z hlediska teorie grafů) herní strom
B. Negascout může navštívit některé uzly v herním stromě opakovaně
C. Alpha-beta prořezávání vždy navštíví méně uzlů herního stromu než minimax

Je následujících tvrzení pravdivé?
Pro řešení CSP můžeme využít i algoritmus A*.

A. Ano
B. Ne

Čemu odpovídá hrana v CSP prohledávacím stromě (search tree)?

A. Proměnné CSP problému.
B. Hodnotě přiřazené proměnné (uzlu, ze kterého hrana vychází)
C. Množině dostupných hodnot pro proměnnou reprezentovanou uzlem, ze kterého hrana vychází

Čemu odpovídá uzel v CSP prohledávacím stromě (search tree)?

A. Hodnotě přiřazené proměnné (uzlu, ze kterého hrana vychází)
B. Proměnné CSP problému.
C. Množině nepřiřazených proměnných

Které z následujících tvrzení o algoritmu AlphaGo jsou pravdivé?

A. Algoritmus AlphaGo nebyl zatím v hraní Go překonán žádným člověkem ani jiným AI systémem.
B. Algoritmus používal tři různé policy a jedna z nich aproximuje lidskou hru.
C. Algoritmus porazil profesionálního hráče ve hře Go.

Je následujících tvrzení pravdivé?
AC-3 algoritmus je spuštěn v rámci řešení CSP prohledáváním nejvýše jednou.

A. Ano
B. Ne

Uvažme problém CSP s dvěma proměnnými x_1, x_2, obě s doménou {a, b}, a podmínkou $x_1 \neq x_2$. Algoritmus AC-3 v tomto případě:

A. Zredukuje domény některých proměnných
B. Zjistí, že je problém neřešitelný
C. Nic neudělá

Uvažujme standardní alfa-beta prořezávání s upraveným počátečním alfa-beta intervalem nastaveným na [w, x]. Algoritmus vrátí hodnotu y < w. Které z následujících tvrzení je pravdivé?

A. Přesná hodnota hry není známa, ale je menší nebo rovna hodnotě w.
B. Hodnota hry je známá a je rovna hodnotě w.
C. Přesná hodnota hry není známa, ale je menší nebo rovna hodnotě y
D. Hodnota hry je známá a je rovna hodnotě y.

Co musíme nezbytně udělat pro transformaci epizodické úlohy do ekvivalentní pokračující úlohy ve zpětnovazebném učení?

A. Nastavit diskontní míru gamma E (0, 1)
B. Nahradit terminální stavy absorbujícími stavy (i.e., self-loops)
C. Přeškálovat odměny tak, aby byly menší než jedna

Předpokládejme MDP s A akcemi a S stavy. Které z následujících tvrzení jsou pravda?

A. Časová složitost u jednoho kroku Q-learning je $O(S^2 * A)$
B. Časová složitost u jednoho kroku Q-learning je $O(A)$
C. Prostorová složitost u tabulkového Q-learning je $O(S)$
D. Prostorová složitost u tabulkového Q-learning je $O(S^2 * A)$
E. Prostorová složitost u tabulkového Q-learning je $O(S * A)$
F. Časová složitost u jednoho kroku Q-learning je $O(S * A)$
G. Prostorová složitost u tabulkového Q-learning je $O(S * A^2)$
H. Časová složitost u jednoho kroku Q-learning je $O(S * A^2)$

Jaké vstupy musí mít agent ve zpětnovazebním učení, aby mohl najít optimální strategii (optimal policy)?

A. Množina všech možných stavů
B. Přesný popis odměnových dynamik
C. Funkce hodnot (Value function)
D. Množina všech dostupných akcí
E. Simulátor prostředí
F. Počet možných stavů
G. Konečná délka horizontu
H. Přesný popis přechodových dynamik

Uvažujme problém plánování pohybu robotické ruky v prostoru (např. pro manipulaci s předměty) s cílem vykonat co možná nejefektivnější pohyb bez nechtěné kolize s předmětem.

1. **Navrhněte formální reprezentaci, kterou by bylo možné použít pro prohledávání prostoru.** length of arms, angle of joints
2. **Navrhněte netriviální přípustnou heuristiku a zdůvodněte proč je přípustná**

arm(name, length, S)
joint(name, angle, S)
init:
arm(1, x1, init) joint(1, z1, init)
arm(n, xn, init) joint(m, zm, init)
move(x1...xn, z1...zm, S)

euclidian distance to the object (from the last joint / arm)

Který selekční algoritmus byste použili v Monte Carlo prohledávání stromu s velkým (případně nekonečným) počtem akcí?

UCT (UCB on MCTS)

4

Setřiďte fáze Monte Carlo prohledávání stromu podle počtu různých stavů hry, které jsou během každé fáze typicky navštíveny od nejmenšího.

selection 1 expansion simulation 3 back propagation 2

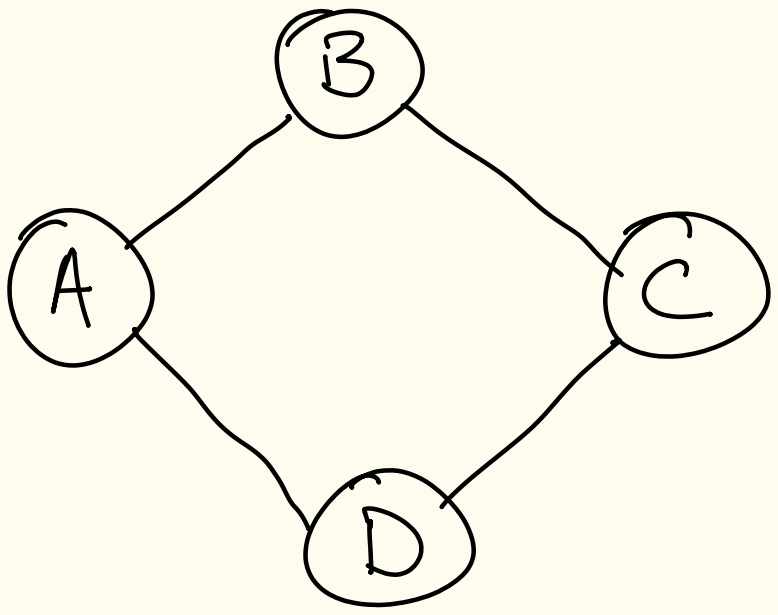
CSP prohledávání sémanticky odpovídá prohledávání

A. do šířky
B. do hloubky
C. do hloubky s iterativním prodlužováním horizontu (iterative deepening)

$a_1 = \langle \{a\}, \{b,c\}, \emptyset \rangle$
$a_2 = \langle \{b\}, \{c\}, \emptyset \rangle$
$a_3 = \langle \{c\}, \{a,b\}, \emptyset \rangle$
$a_4 = \langle \{a,b,c\}, \{d\}, \emptyset \rangle$
$G = \{d\}$ $I = \{a\}$

$\{a\} \xrightarrow{a_1} \{a,b,c\} \xrightarrow{a_4} \{d\}$
$a_1 \rightarrow \{b\} \xrightarrow{a_2} \{c\}$
$a_1 \rightarrow \{c\} \xrightarrow{a_3} \{a,b\}$

a) $h_{max} = 2$
b)



ve městě C je auto
A je zásilka

car(package, at)
package(at)
neighborTo(city, city)

init:
car(none, C)
package(A)
neighborTo(A,B), (B,A), (C,D), (D,C), (A,D), (D,A), (B,C), (C,B)

move(From, To):
neighborTo(From, To) ∧ car(package, From)
→ car(package, To)
pickup(Where)
car(none, Where) ∧ package(where)
→ car(Where, Where)
→ remove package(Where)

$1 - P(\neg J, \neg M \mid B)$

$P(J \wedge \neg M \mid B)$ $P(M \wedge \neg J \mid B)$ $P(J \wedge M \mid B)$

$P(B) \rightarrow P(E)$ → $P(A \mid B \wedge E) = 0{,}95$
→ $P(\neg A \mid B \wedge E) = 0{,}05$
$P(\neg E)$
$P(B) = 0{,}001$
$P(E) = 0{,}002$
$P(A \mid B \wedge \neg E) = 0{,}94$
$P(\neg A \mid B \wedge \neg E) = 0{,}06$

$P(J|A) = 0{,}9$ $P(\neg J|A) = 0{,}1$ $P(J|\neg A) = 0{,}05$

$P(\neg J|\neg A) = 0{,}95$

$P(M|A) = 0{,}7$ $P(\neg M|A) = 0{,}3$ $P(M|\neg A) = 0{,}01$

$P(\neg M|\neg A) = 0{,}99$

$$P(J \wedge \neg M|B) = P(B)\cdot P(E)\cdot P(A|B\wedge E)\cdot P(J|A)\cdot P(\neg M|A)$$

$$+ P(B)\cdot P(\neg E)\cdot P(A|B\wedge\neg E)\cdot P(J|A)\cdot P(\neg M|A)$$

Zmolá alespoň jeden, když je vlapsička

mark nebo John nebo oba $= 1 -$ nezmolí ani jeden

$$1 - P(\neg J|B)P(\neg M|B)$$

$$P(A|B) = P(E)P(A|E\wedge B) = 0{,}002\cdot 0{,}95$$

$$+ P(\neg E)P(A|\neg E\wedge B) = 0{,}998\cdot 0{,}94$$

$$= 0{,}94002$$

$$P(\neg J|B) = P(\neg J|A)P(A|B) + P(\neg J|\neg A)P(\neg A|B)$$

$$= 0{,}1\cdot 0{,}94002 + 0{,}95\cdot(1-0{,}94002)$$

$$= 0{,}150983 \qquad = 0{,}141386$$

$0{,}3\cdot 0{,}94002 + 0{,}99\cdot(1-0{,}94002)$

$$P(\neg M|B) = P(\neg M|A)P(A|B) + P(\neg M|\neg A)P(\neg A|B)$$

$1 - 0{,}150983\cdot 0{,}141386$

$= 0{,}948$

z.p. A

$$P(\neg J \mid A \cap B) \cdot P(\neg M \mid A \cap B) \cdot P(A \mid B) + P(\neg J \mid \neg A) \cdot P(\neg M \mid \neg A) \cdot P(\neg A \mid B)$$

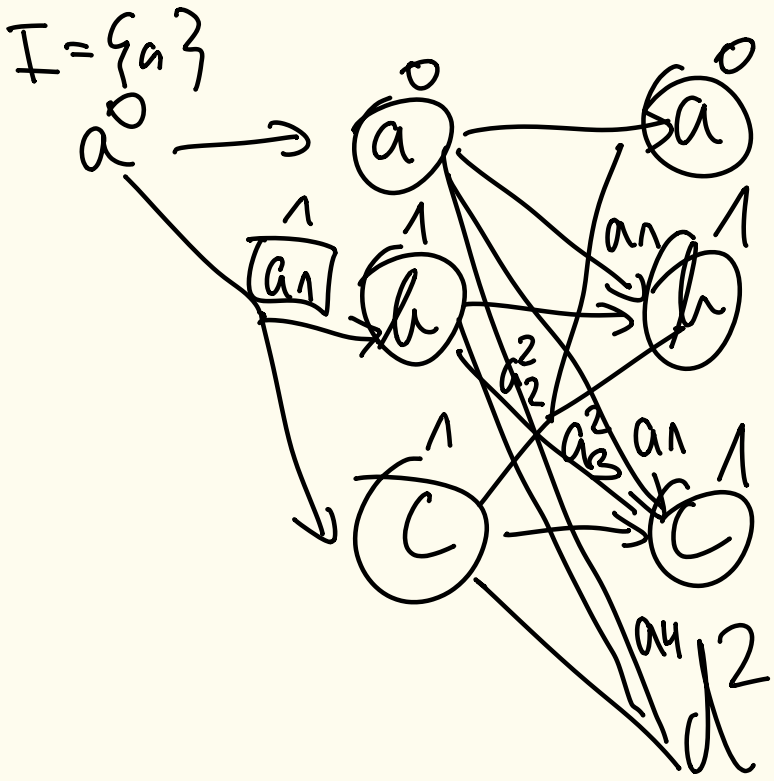
0,1 · 0,3 · 0,94002 + 0,95 · 0,99 · (1 − 0,94002)

$$1 - (\quad) = 0{,}9153$$

67) Uvažujme STRIPS problém s predikáty a, b, c, d, iniciálním stavem I = {a}, akcemi

a_1 =< {a}, {b, c}, ∅ >
a_2 =< {b}, {c}, ∅ >
a_3 =< {c}, {a, b}, ∅ >
a_4 =< {a, b, c}, {d}, ∅ >
a cílovým stavem G = {d}

(a) Jaká je hodnota heuristiky h_{max} v iniciálním stavu?
(b) Jaká je hodnota každé z akcí během výpočtu h_{max} v iniciálním stavu?



(a) $h_{max} = 2$

(b) $a_1 = 1$
$a_2 = 2$
$a_3 = 2$
$a_4 = 2$

$$2^{\frac{3^2-1}{3-1}} = 2^{\frac{9-1}{2}} = 2^4 = \underline{\underline{16}}$$

po jednom hodu:

A: 0 %

B: 0,7 · 0,9 + 0,3 · 0,05 = 0,645

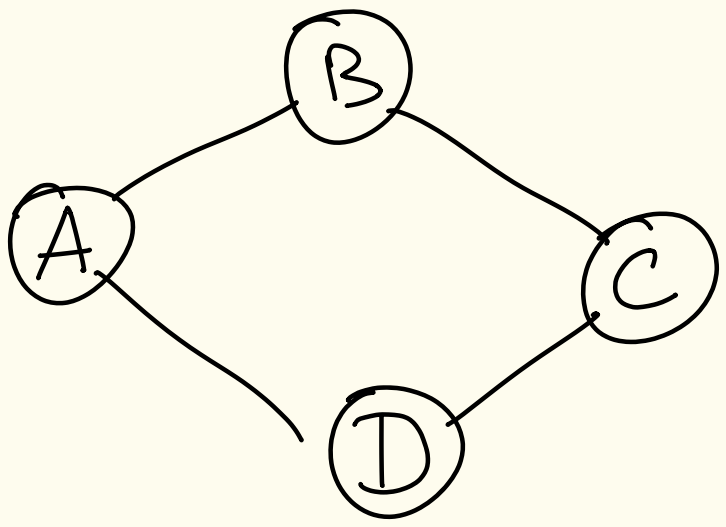
C: 0,7 · 0,05 + 0,3 · 0,05 = 0,05

D: 0,3 · 0,9 + 0,7 · 0,05 = 0,305

E: 0 %

0,7 · 0,9 · 0,9 + 0,3 · 0,05 · 0,05 = 0,567 · 10 = 5,67

− 0,2 = 5,47



A zasílka

C auto

(a) nit

(b) naložení, vyložení, přesun

Init:

{ adjacent_a_d
_d_a
_c_d
_d_c
_b_c
_c_b
_a_b
_b_a

P: { car_at_a
_b
_c
_d

box_at_a
_b
_c
_d

adjacent ...

box_loaded }

box_at_a

car_at_c }

load: ⟨{box_at_x, car_at_x}, {box_loaded}, {box_at_x}⟩

unload ⟨{box_loaded, car_at_x}, {box_at_x}, {box_loaded}⟩

move ⟨{car_at_x, adjacent_x_y}, {car_at_y},

{car_at_x} >

64) Předpokládejme deterministický gridworld, kde se robot může pohybovat jen doleva (L) nebo doprava (R). Pohyb nic nestojí a robot získá odměnu 4 pokud opustí herní plochu. Diskontní míra (discount factor) je 1. Předpokládejme, že iniciální value funkce je:

2	1	0	1	0

(a) jaké jsou hodnoty value funkce po dvou krocích value iteration algoritmu?
(b) jaké strategie by se měl držet agent s value funkcí z bodu (a) ve stavu nejvíce vlevo a proč?
(c) jak byste změnili tuto strategii aby fungovala lépe při praktické aplikaci v robotice?

→ zmínit cost za pohyb? např. −0,1 za každou akci

γ = 0,5

4 [2 1 0 1 0] 4

1: 4 [4 2 1 0 4] 4

2: 4 [4 4 2 4 4] 4

3: 4 [4 4 4 4 4] 4

S_1 = ["###" "# #" "###"] S_1 = ["###" "# #" "###"] S_1 = ["###" "# #" "###"]