

Příklady otázek – Základy umělé inteligence

Reálné otázky ze SZZ 2022–2026 (S1)

Jakub Klesa

Pokryté předměty: NAIL120 (Úvod do umělé inteligence)

Obsah

1	N-královen	2
2	Barvení grafu, DPLL	2
3	Plánování	3
4	Skrytý Markovův model	5
5	Problém splňování podmínek (CSP)	6
6	Informované prohledávání	6
7	Bayesovské sítě	7
8	Minimax	8
9	DPLL a párování	10
10	Bayesovské učení	10
11	Bellmanova rovnice užitku	11
12	Jednotahové hry	12
13	CSP	14

Poznámka. Tento dokument obsahuje reálné otázky z minulých termínů SZZ (2022–2026) zařazené do okruhu S1. U otázek s obrázkem je grafika oříznutá z originálního PDF (viz odkaz na zdroj). Text, číselná data a tabulky jsou přepsané věrně. Zadání i oficiální náčrty řešení pocházejí ze sad zveřejněných na webu MFF UK:

<https://www.mff.cuni.cz/cs/studenti/bakalarske-studium/statni-zaverecne-zkousky/bakalarske-statni-zkousky-studijniho-programu-informatika>

Náčrty označené jako doplněné jsou vlastní, oficiální sada je k dané otázce neobsahuje. Vlastní doplnění k oficiálnímu náčrtu jsou zřetelně oddělena. Pojmy a značení odpovídají přednáškám předmětu NAIL120 *Úvod do umělé inteligence* (R. Barták, https://ktiml.mff.cuni.cz/~bartak/ui_intro/) a učebnici S. Russella a P. Norviga *Artificial Intelligence: A Modern Approach* (<https://aima.cs.berkeley.edu/>).

1 N-královen

(Reálná otázka: léto 2022, č. 9 – N-královen)

Zadání.

Problém N-královen (rozmístit N královen na šachovnici $N \times N$ tak, aby se neohrožovaly – žádné dvě na stejném sloupci, řádku nebo diagonále). (a) Popište vhodnou abstrakci pro prohledávací algoritmus (stavy, přechodový model, cílovou podmínku). (b) Vyberte vhodný informovaný či neinformovaný prohledávací algoritmus a zdůvodněte. (c) Ukažte, jak by se problém modeloval a řešil jako problém splňování podmínek (CSP).

Náčrt řešení.

- Abstrakce pro prohledávání.** Stav = částečné rozmístění královen, např. jedna v každém z prvních k sloupců tak, že se vzájemně neohrožují. Počáteční stav = prázdná šachovnice. Přechodový model = přidání královny do dalšího $(k+1.)$ sloupce na řádek, kde neohrožuje žádnou již položenou královnu. Cílová podmínka = rozmístěno N královen (všechny sloupce zaplněny). Tím, že na každý sloupec přijde právě jedna královna, se prohledávací strom drasticky zúží.
- Volba algoritmu.** Neinformované prohledávání do hloubky (DFS) s navracením (backtracking) je vhodné: řešení leží v hloubce N , strom je konečný a DFS je paměťově nenáročný. Informované metody (A^* , heuristiky) se obvykle nevyplatí, protože jde o splnění podmínek (constraint satisfaction), ne o hledání nejkratší cesty – všechna řešení jsou ve stejné hloubce a cena cesty není relevantní.
- Model jako CSP.** Proměnné $Q_1, \dots, Q_N$ (po jedné na sloupec), doména $\{1, \dots, N\}$ (řádek královny v daném sloupci). Podmínky pro všechna $i < j$: $Q_i \neq Q_j$ (různé řádky) a $|Q_i - Q_j| \neq |i - j|$ (nejsou na společné diagonále). Řeší se backtrackingem se šířením podmínek (forward checking / AC-3) a heuristikou výběru proměnné MRV.

(V PDF bez náčrtu řešení; náčrt doplněn k výkladu okruhu.)

2 Barvení grafu, DPLL

(Reálná otázka: podzim 2022, č. 17 – Barvení grafu, DPLL)

Zadání.

Nechť $G = (V, E)$ je neorientovaný graf s N vrcholy ($|V| = N$).

- Definujte pojem konjunktivní normální forma (CNF) formule ve výrokové logice.
- Navrhněte formuli ve výrokové logice, která je splnitelná, právě když je graf G obarvitelný k barvami. Formulí popište obecně v CNF pro libovolný graf G a libovolné $k > 0$.
- Napište formuli pro úplný graf se třemi vrcholy a pro $k = 3$. Na této formuli demonstруйте použití algoritmu DPLL.

Náčrt řešení.

- Formule je v *konjunktivní normální formě (CNF)*, je-li to konjunkce klauzulí, kde každá klauzule je disjunkce literálů a literál je výroková proměnná nebo její negace.
- Zavedeme výrokové proměnné $x_{v,c}$ pro $v \in V$ a $c \in \{1, \dots, k\}$ s významem „vrchol v má barvu c “. Klauzule:
 - Každý vrchol má alespoň jednu barvu: pro každý vrchol v klauzule $(x_{v,1} \vee \dots \vee x_{v,k})$.
 - Sousedé nemají stejnou barvu: pro každou hranu $\{u, v\} \in E$ a každou barvu c klauzule $(\neg x_{u,c} \vee \neg x_{v,c})$.

Formule je konjunkcí všech těchto klauzulí, takže je v CNF. Lze volitelně přidat i podmínku „nejvýše jedna barva na vrchol“ (pro každý vrchol v a dvojici barev $c \neq c'$ klauzuli $\neg x_{v,c} \vee \neg x_{v,c'}$), ale pro ekvivalenci splnitelnost = obarvitelnost tyto klauzule nejsou nutné: pokud vrchol dostane více barev, stačí jednu z nich ponechat.

3. Úplný graf K_3 má vrcholy 1, 2, 3 a hrany $\{1, 2\}, \{1, 3\}, \{2, 3\}$; pro $k = 3$ máme proměnné $x_{v,c}$ s $v, c \in \{1, 2, 3\}$ (celkem 9 proměnných). Pokrývací klauzule:

$$(x_{1,1} \vee x_{1,2} \vee x_{1,3}), \quad (x_{2,1} \vee x_{2,2} \vee x_{2,3}), \quad (x_{3,1} \vee x_{3,2} \vee x_{3,3}).$$

Klauzule různosti pro každou hranu a barvu, např. pro hranu $\{1, 2\}$:

$$(\neg x_{1,1} \vee \neg x_{2,1}), \quad (\neg x_{1,2} \vee \neg x_{2,2}), \quad (\neg x_{1,3} \vee \neg x_{2,3}),$$

obdobně pro hrany $\{1, 3\}$ a $\{2, 3\}$. Formule je splnitelná, neboť K_3 je 3-obarvitelný (např. barvy 1, 2, 3 vrcholům 1, 2, 3).

Běh DPLL. Rozhodneme $x_{1,1} = \text{true}$. Jednotkovou propagací přes klauzule různosti hran $\{1, 2\}$ a $\{1, 3\}$ s barvou 1 (klauzule $\neg x_{1,1} \vee \neg x_{2,1}$ a $\neg x_{1,1} \vee \neg x_{3,1}$ se stanou jednotkovými) vynutíme $x_{2,1} = \text{false}$ a $x_{3,1} = \text{false}$. Rozhodneme $x_{2,2} = \text{true}$; jednotková propagace vynutí $x_{1,2} = \text{false}$ (už splněno) a $x_{3,2} = \text{false}$. Vrcholu 3 pak v jeho pokrývací klauzuli po eliminaci $x_{3,1}$ a $x_{3,2}$ zbývá jediný literál $x_{3,3}$ (jednotková klauzule), takže $x_{3,3} = \text{true}$. Splňující ohodnocení: $x_{1,1}, x_{2,2}, x_{3,3} = \text{true}$, ostatní false, což odpovídá obarvení vrcholů 1, 2, 3 barvami 1, 2, 3. Roli zde hraje hlavně jednotková klauzule (vynucené dosazení); backtracking není při tomto pořadí rozhodnutí potřeba, protože každé rozhodnutí vede ke konzistentnímu rozšíření.

(V PDF bez náčrtu řešení; náčrt doplněn k výkladu okruhu.)

3 Plánování

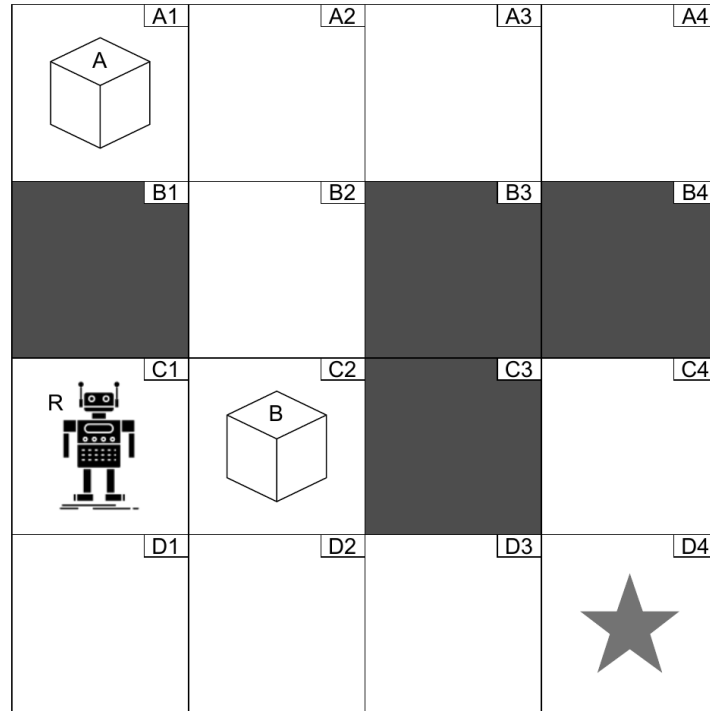
(Reálná otázka: jaro 2023, č. 5 – Plánování (robot, krabice))

Zadání.

Mějme robota, který může provádět následující akce:

- **Jdi** – přesun na sousední políčko, pokud na něm není zeď; pokud robot nese krabici, nesmí navíc na políčko s jinou krabicí.
- **Seber** – sebrání krabice na pozici robota; robot nemůže sebrat další krabici, pokud už nějakou krabici nese.
- **Polož** – položení nesené krabice na pozici robota.

Předpokládejme, že počáteční stav světa je jako na obrázku níže. Robot je značen písmenem R , dvě krabice jsou značeny písmeny A a B . Tmavá políčka označují zdi, světlá políčka jsou volná místa. Cílem je dostat krabici A na políčko označené hvězdičkou. Můžete dále předpokládat, že stav je popsán pomocí predikátů $\text{soused}(X, Y)$ vyjadřující, že X a Y jsou sousedi a na Y není zeď (tedy na Y je možné se z X přesunout), $\text{na}(X, K)$ vyjadřující, že na pozici X je krabice K , a $\text{robot}(X)$ vyjadřující, že robot je na políčku X . Značky v pravém horním rohu každého políčka uvádějí jeho jméno, které můžete použít, pokud se na konkrétní políčko potřebujete odkázat v řešení.



1. Formalizujte zadaný problém jako plánovací problém. Pro definování akcí a počátečního a cílového stavu použijte predikáty popsané výše, případně další predikáty, které si na-
definujete. Predikáty, které se nemění (*soused*), nemusíte explicitně vypisovat v definici
počátečního stavu.

*Nápověda: Budete potřebovat definovat dvě verze akce pro pohyb – podle toho, jestli robot
nese nějakou krabici nebo ne.*

2. Popište, jak funguje dopředné a zpětné plánování.
3. Napište nějaký plán, který zadaný problém řeší.

Řešení. (V PDF bez náčrtu řešení; vlastní vzorové řešení, plán ověřen prohledáním stavového
prostoru.)

(1) *Formalizace.* Pomocné fluenty: $nese(K)$ (robot nese krabici K), $volno(X)$ (na poli X neleží
krabice), $volnaruka$ (robot nic nenese). Operátory (precond $\Rightarrow$ effects):

- $Jdi(X, Y)$ – bez krabice: $robot(X), \text{soused}(X, Y), \text{volnaruka} \Rightarrow \neg robot(X), robot(Y)$.
- $JdiSK(X, Y, K)$ – s krabicí: $robot(X), \text{soused}(X, Y), nese(K), \text{volno}(Y) \Rightarrow \neg robot(X),$
 $robot(Y)$. (Nesená krabice zůstává „v ruce“ přes $nese(K)$; předpoklad $volno(Y)$ brání
vstupu na pole s jinou krabicí.)
- $Seber(X, K)$: $robot(X), na(X, K), \text{volnaruka} \Rightarrow \neg na(X, K),$
 $nese(K), \neg \text{volnaruka}, \text{volno}(X)$.
- $Poloz(X, K)$: $robot(X), nese(K), \text{volno}(X) \Rightarrow \neg nese(K), na(X, K),$
 $\text{volnaruka}, \neg \text{volno}(X)$.

Počáteční stav (jen mění se fluenty): $s_0 = \{robot(C1), \text{volnaruka}, na(A1, A), na(C2, B)\} \cup$
 $\{volno(X) : X \text{ volné}, X \neq A1, C2\}$. Cíl $g = \{na(D4, A)\}$.

(2) *Dopředné a zpětné plánování.* Dopředné (progrese) jde od s_0 vpřed: v každém stavu generuje
nástupce použitelných akcí a prohledává, dokud stav nesplní cíl; pracuje s konkrétními stavy.
Zpětné (regrese) jde od cíle g vzad přes *relevantní* akce (jejichž efekt přispívá k cíli) a počítá
jejich regresní množiny, dokud nedosáhne s_0 ; má menší faktor větvení, ale pracuje s množinami
stavů.

(3) *Plán.* Volná pole jsou A1–A4, B2, C1, C2, C4, D1–D4; kvůli zdem B1, B3, B4, C3 se mapa rozpadá
na dvě oblasti spojené *jedíným* mostem $B2 \leftrightarrow C2$, na němž leží krabice B . Robot s krabicí přes
něj nesmí, takže naivní „seber A , odnes do D4“ *neexistuje* – B je nutné nejdřív odklidit z mostu.

Validní plán (ověřeno, 16 akcí je minimum):

1. Jdi(C1, C2); 2. Seber(C2, B); 3. JdiSK(C2, B2, B); 4. JdiSK(B2, A2, B);
5. JdiSK(A2, A3, B); 6. Poloz(A3, B) (*krabice B odklizena do horní oblasti*);
7. Jdi(A3, A2); 8. Jdi(A2, A1); 9. Seber(A1, A);
10. JdiSK(A1, A2, A); 11. JdiSK(A2, B2, A); 12. JdiSK(B2, C2, A);
13. JdiSK(C2, D2, A); 14. JdiSK(D2, D3, A); 15. JdiSK(D3, D4, A); 16. Poloz(D4, A).

Výsledný stav obsahuje na(D4, A), cíl je splněn.

4 Skrytý Markovův model

(Reálná otázka: léto 2023, č. 5 – Skrytý Markovův model (HMM))

Zadání.

Definujte skrytý Markovův model (HMM z anglického Hidden Markov Model). Co je Markovův předpoklad? Jaký je rozdíl mezi filtrací (filtering), predikcí (prediction) a vyhlazováním (smoothing)? Za použití vzorce

$$P(X_t | E_{1:t}) = \alpha P(E_t | X_t) \sum_{X_{t-1}} P(X_t | X_{t-1}) P(X_{t-1} | E_{1:t-1})$$

vyřešte následující příklad:

Předpokládejme, že proměnná X_i vyjadřuje, jaké je počasí ve dne i . Každý den prší ($X_i = P$) nebo je slunečno ($X_i = S$). Tuto proměnnou nemůžeme přímo pozorovat, máme k dispozici pouze pozorování E_i , zda si náš kolega přinesl dnes deštník ($E_i = D$) nebo ne ($E_i = N$). Podmíněné pravděpodobnosti změny počasí z předchozího dne na dnešek a závislost pozorování deštníku na skutečném počasí uvádějí následující tabulky.

X_{i-1}	$P(X_i = P X_{i-1})$	$P(X_i = S X_{i-1})$	X_i	$P(E_i = D X_i)$	$P(E_i = N X_i)$
P	0.7	0.3	P	0.9	0.1
S	0.3	0.7	S	0.2	0.8

Předpokládejme, že ve dne 0 je pravděpodobnost deště $P(X_0 = P) = 0.5$ a slunečna $P(X_0 = S) = 0.5$. Jaká je pravděpodobnost deště ve druhém dni ($P(X_2 = P) = ?$), pokud první i druhý den pozorujeme, že si náš kolega přinesl deštník ($E_1 = D$ a $E_2 = D$)?

Řešení. (*V PDF bez náčrtu řešení; vlastní vzorové řešení, výpočet ověřen numericky.*)

HMM a Markovův předpoklad. Skrytý Markovův model je posloupnost skrytých stavů X_t (počasí $\in \{P, S\}$) a pozorování E_t (deštník $\in \{D, N\}$), daná priorem $P(X_0)$, přechodovým modelem $P(X_t | X_{t-1})$ a senzorovým modelem $P(E_t | X_t)$. *Markovův předpoklad:* stav závisí jen na bezprostředně předchozím stavu, $P(X_t | X_{0:t-1}) = P(X_t | X_{t-1})$; pozorování závisí jen na aktuálním stavu, $P(E_t | X_{0:t}, E_{1:t-1}) = P(E_t | X_t)$.

Filtrace, predikce, vyhlazování. Filtrace počítá rozdělení *nynějšího* stavu z dosavadních pozorování, $P(X_t | E_{1:t})$; predikce *budoucího* stavu $P(X_{t+k} | E_{1:t})$ ($k > 0$); vyhlazování *minulého* stavu $P(X_k | E_{1:t})$ ($k < t$), které využívá i pozdější pozorování, a je proto přesnější než filtrace v témže okamžiku.

Výpočet $P(X_2 = P | E_1 = D, E_2 = D)$ rekurzí (predikce + update senzorem + normalizace), zaokrouhleno na 4 desetinná místa:

- Prior $f_0 = P(X_0) = \langle P: 0,5; S: 0,5 \rangle$.
- Den 1 – predikce: $P(X_1=P) = 0,7 \cdot 0,5 + 0,3 \cdot 0,5 = 0,5$, $P(X_1=S) = 0,5$.
- Den 1 – update $E_1=D$: $\langle 0,9 \cdot 0,5; 0,2 \cdot 0,5 \rangle = \langle 0,45; 0,10 \rangle$, součet 0,55, normalizace $f_1 = \langle 0,8182; 0,1818 \rangle$.
- Den 2 – predikce: $P(X_2=P) = 0,7 \cdot 0,8182 + 0,3 \cdot 0,1818 = 0,6273$, $P(X_2=S) = 0,3727$.

• Den 2 – update $E_2=D$: $\langle 0,9 \cdot 0,6273; 0,2 \cdot 0,3727 \rangle = \langle 0,5645; 0,0745 \rangle$, součet 0,6391, normalizace $f_2 = \langle 0,8834; 0,1166 \rangle$.
Tedy $P(X_2 = P \mid E_1=D, E_2=D) \approx \boxed{0,883}$. Dva po sobě jdoucí deštníky posílí víru v déšť z 50 % na téměř 90 %.

5 Problém splňování podmínek (CSP)

(Reálná otázka: podzim 2023, č. 28 – Problém splňování podmínek (CSP, backtracking, arc consistency))

Zadání.

Definujte problém splňování podmínek (CSP z anglického Constraint Satisfaction Problem). Popište prohledávání s navracením (backtracking) jako způsob řešení CPS. Vysvětlíte pojmy „hranová konzistence“ (arc consistency), „kontrola dopředu“ (forward checking) a „pohled dopředu“ (look ahead).

Pomocí CSP modelujte problém Sudoku, tj. problém umístění čísel $1, \dots, 9$ do mřížky 9×9 takovým způsobem, že se v žádném sloupci, řádku ani boxu velikosti 3×3 žádná cifra neopakuje více než jednou. Předpokládejte, že některá čísla jsou už v mřížce zadána.

Náčrt řešení.

- Definice CSP – trojice (proměnné, domény, podmínky), obecně může být pro každou proměnnou jiná doména, podmínek je konečně mnoho. Podmínka je relace mezi proměnnými, podmnožina Kartézského součinu domén.
- Popis backtracking, arc consistency, forward checking a look ahead.
 - *backtracking* – přiřadíme hodnotu nějaké proměnné a opakujeme dokud nedostaneme validní řešení nebo spor. V případě sporu backtrackujeme a učiníme jiné rozhodnutí.
 - *arc consistency* – podmínka (arc, pokud máme pouze binární podmínky) je konzistentní, pokud pro každou hodnotu z domény existuje hodnota pro ostatní proměnné, které tuto podmínku splní. Hodnoty pro které neexistuje ohodnocení ostatních proměnných můžeme z domény vyloučit.
 - *forward checking* – po ohodnocení proměnné x zkontrolujeme ostatní proměnné, které se vyskytují v podmínkách spolu s x . Odstraníme nekonzistentní hodnoty.
 - *look ahead* – podobné jako forward check, ale kontrolujeme postupně všechny podmínky, dokud se nějaké proměnné mění doména. Tj. jde o silnější techniku než forward check. Obdobně můžeme říct, že po ohodnocení proměnné se problém udělá arc consistent.
- Model sudoku – každé políčko je jedna proměnná, každá proměnná má doménu $1, \dots, 9$. Podmínky jsou „All Different“ (případně nerovnost po dvojicích) na vhodné podmnožiny proměnných (tj. sloupce, řádky, 3×3 políčka). Typicky máme v sudoku už zadána nějaká čísla, ta můžeme přiřadit pomocí domén nebo pomocí podmínek.

6 Informované prohledávání

(Reálná otázka: jaro 2024, č. 32 – Informované prohledávání (A^* , heuristika, 8-puzzle))

Zadání.

Uvažme hlavolam na obrázku níže. Jedná se o mřížku 3×3 s 8 označenými kostičkami a jedním volným místem. Naším cílem je ze zadaného počátečního stavu uspořádat kostičky do předem daného cílového stavu, ve kterém jsou kostičky uspořádány od 1 do 8 zleva doprava, shora dolů. V každém kroku můžeme posunout právě jednu kostičku na sousední volné místo. Navíc chceme provést minimální možný počet kroků.

2	8	3
1	6	4
7		5

1. Formalizujte zadaný problém jako informovaný prohledávací problém. Speciálně popište, jak vypadá stav, jaký je přechodový model a jaký je test cílového stavu.
2. Pro vyřešení definovaného problému navrhnete optimální informovaný prohledávací algoritmus. Popište jeho průběh (například pomocí pseudokódu).
3. Navrhnete heuristickou funkci pro výše zmíněný hlavolam. Jaké vlastnosti má vaše heuristická funkce? Zaručuje použití této heuristiky s algoritmem z předchozího bodu nalezení optimálního řešení? *Nápověda: Napsali jste algoritmus jako tree search nebo graph search?*

Náčrt řešení.

1. Stav je rozložení kostiček, přechodový model je pohnutí jednou kostičkou, čímž dostanu jiné rozložení (nezapomenout na cenu akce = 1), kontrola cílového stavu zkontroluje, zda se rozložení shoduje se zadaným cílem.
2. A*, je jedno jestli graph search nebo tree search.
3. Například taxicab distance. U graph search musí být heuristika monotonní/konzistentní. U tree search stačí přípustnost/spodní odhad. Všechny rozumné heuristiky budou splňovat oboje.

Doplňující vysvětlení (nad rámec oficiálního náčrtu): Dvě klasické heuristiky pro zadaný počáteční stav: h_1 = počet kostek mimo cílovou pozici ($h_1 = 6$, správně leží jen 3 a 7) a h_2 = součet manhattanských vzdáleností kostek od cíle ($h_2 = 1 + 1 + 0 + 2 + 2 + 1 + 0 + 2 = 9$ pro kostky 1 až 8). Obě jsou přípustné (jeden tah posune jednu kostku o jedno pole, takže skutečný počet tahů nepodstřelí) i konzistentní (tahem klesne h nejvýš o 1, což je cena kroku); $h_2 \geq h_1$, tedy h_2 dominuje. Přípustnost zaručuje optimalitu A* ve stromovém prohledávání, konzistence v grafovém (s closed listem). Poznámka k instanci: počáteční stav má 11 inverzí (liché číslo), cílový 0, a posun kostky paritu inverzí nemění. Zadaná instance proto do cíle *není řešitelná*; A* (v grafové verzi) by po prohledání všech $9!/2 = 181\,440$ dosažitelných stavů ohlásil neúspěch. Otázka ale konkrétní řešení nežádá.

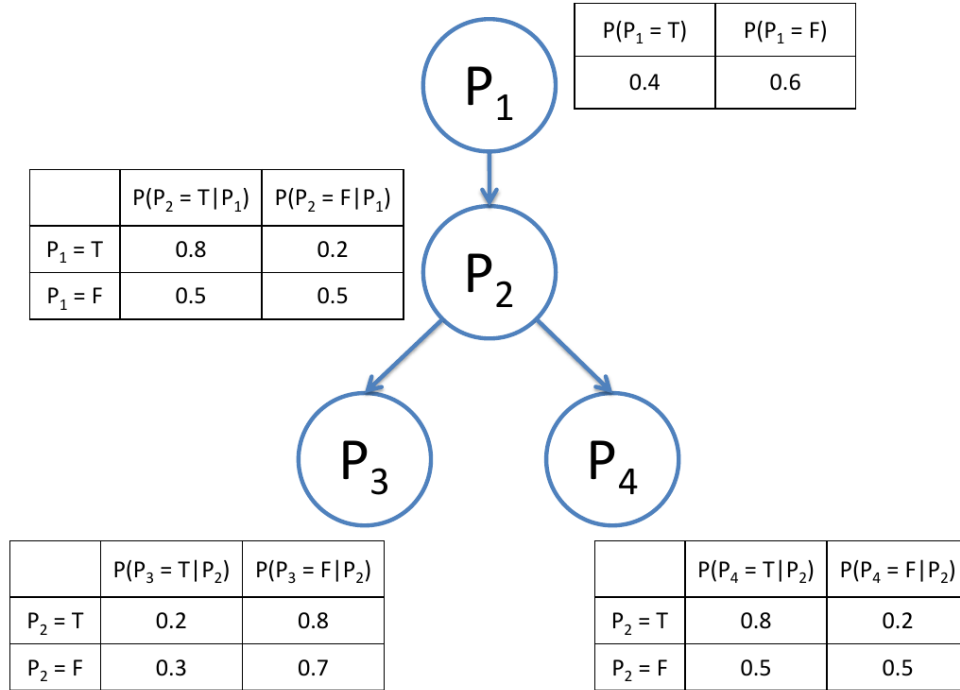
7 Bayesovské sítě

(Reálná otázka: léto 2024, č. 28 – Bayesovské sítě)

Zadání.

Definujte Bayesovskou síť a napište, jak se takové sítě používají a jak se konstruuji. Jaký je vztah mezi Bayesovskou sítí a úplnou sdruženou pravděpodobnostní distribucí?

Uvažujte níže zobrazenou Bayesovskou síť se 4 náhodnými proměnnými $P_1, \dots, P_4$ a uvedenými podmíněnými pravděpodobnostními distribucemi. Máme pozorování $P_3 = F$ a $P_4 = T$. Spočítejte, jaká je pravděpodobnost, že $P_1 = T$, tedy $P(P_1 = T \mid P_3 = F, P_4 = T)$.



Řešení. (V PDF bez náčrtu řešení; vlastní vzorové řešení, výpočet ověřen numericky.)

Definice a vztah ke sdružené distribuci. Bayesovská síť je orientovaný acyklický graf, jehož uzly jsou náhodné veličiny, hrana $X \rightarrow Y$ značí přímý vliv (rodičovství) a každý uzel nese tabulku podmíněných pravděpodobností (CPT) $P(X | \text{Parents}(X))$. Síť *plně reprezentuje* úplnou sdruženou distribuci součinem CPT:

$$P(x_1, \dots, x_n) = \prod_i P(x_i | \text{Parents}(X_i)).$$

Používá se k odpovídání na dotazy $P(X | \text{evidence})$; konstruuje se seřazením veličin (ideálně příčiny před důsledky) a výběrem minimální množiny rodičů pro každou veličinu.

Výpočet $P(P_1=T | P_3=F, P_4=T)$. Síť je řetěz $P_1 \rightarrow P_2 \rightarrow \{P_3, P_4\}$ (P_2 je rodič P_3 i P_4). Vyscítáme skrytou P_2 :

$$\begin{aligned} P(P_1=T, P_3=F, P_4=T) &= P(P_1=T) \sum_{P_2} P(P_2 | P_1=T) P(P_3=F | P_2) P(P_4=T | P_2) \\ &= 0,4 [0,8 \cdot 0,8 \cdot 0,8 + 0,2 \cdot 0,7 \cdot 0,5] = 0,4 \cdot 0,582 = 0,23280, \\ P(P_1=F, P_3=F, P_4=T) &= 0,6 [0,5 \cdot 0,8 \cdot 0,8 + 0,5 \cdot 0,7 \cdot 0,5] = 0,6 \cdot 0,495 = 0,29700. \end{aligned}$$

Normalizace ($P(P_3=F, P_4=T) = 0,23280 + 0,29700 = 0,52980$):

$$P(P_1=T | P_3=F, P_4=T) = \frac{0,23280}{0,52980} = \boxed{0,4394} \quad (\approx 43,9 \%).$$

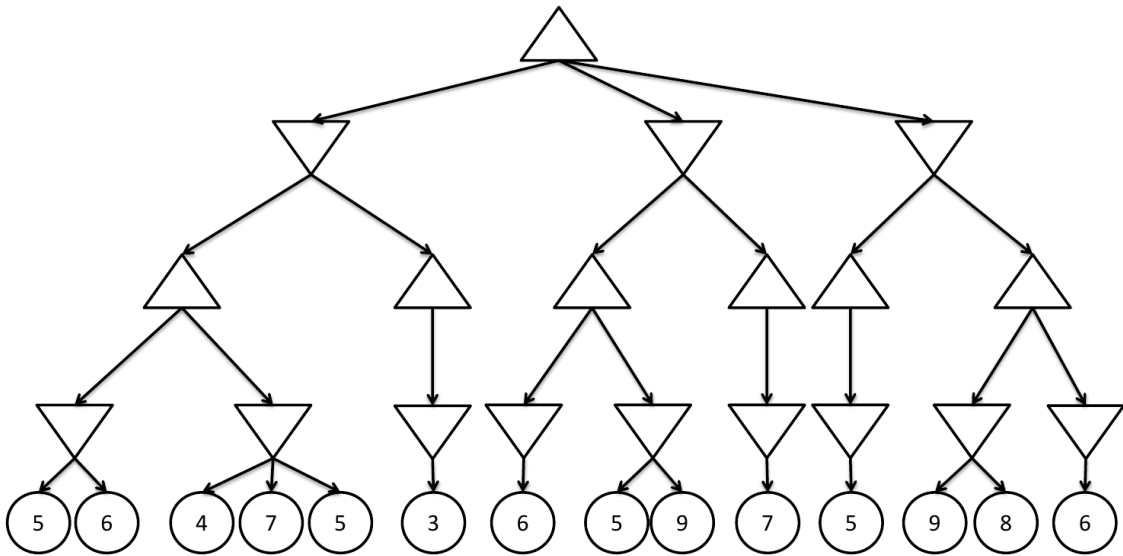
(Použito $P(P_3=F | P_2=T) = 1 - 0,2 = 0,8$ a $P(P_3=F | P_2=F) = 1 - 0,3 = 0,7$.) Pozorování P_3, P_4 tak zvedlo víru v $P_1=T$ z apriorních 0,4 na 0,439.

8 Minimax

(Reálná otázka: podzim 2024, č. 28 – Minimax (alfa-beta prořezávání))

Zadání.

1. Popište algoritmus Minimax pro hru dvou hráčů. Vysvětlete, jaké má v praxi tento algoritmus nedostatky a jak je možné je řešit.
2. Zaveďte (popište) techniku alfa-beta prořezávání.
3. Na následujícím příkladu stromu hry s nulovým součtem symbol $\triangle$ označuje tah maximalizujícího hráče (tedy hrany vedoucí směrem dolů z tohoto symbolu odpovídají možným tahům maximalizujícího hráče). Obdobně symbol ∇ označuje tah minimalizujícího hráče. Symbol $\circ$ označuje koncový stav s uvedeným ohodnocením tohoto stavu. Doplněte do nekoncových stavů, jaké ohodnocení jim přiřadí algoritmus minimax za použití alfa-beta prořezávání. Vyznačte, které části stromu budou prořezány. Předpokládejte, že možné tahy se procházejí zleva doprava.



Řešení. (V PDF bez náčrtu řešení; vlastní vzorové řešení, hodnoty i prořezání ověřeny Pythonem.)

(1) *Minimax*. Hra dvou hráčů s nulovým součtem se chápe jako prohledávání herního stromu. *Minimaxová hodnota* uzlu je užitek za předpokladu, že oba hrají optimálně: v koncovém uzlu je to jeho ohodnocení, v uzlu MAX maximum z hodnot dětí, v uzlu MIN minimum. Počítá se zdola nahoru; tah v kořeni vede do dítěte s nejvyšší hodnotou. *Nedostatek*: minimax prochází celý strom velikosti $O(b^m)$, což je u reálných her neproveditelné. *Řešení*: alfa-beta prořezávání a ořezání hloubky s náhradou koncové funkce *ohodnocovací* funkcí v neukončených listech.

(2) *Alfa-beta*. Podél cesty držíme α (nejlepší dosud zajištěná hodnota pro MAX) a β (pro MIN). V uzlu MIN, jakmile jeho průběžná hodnota klesne $\leq \alpha$, zbylé děti nezkoumáme (α -řez) – MAX nad námi má lepší alternativu. Symetricky v uzlu MAX při hodnotě $\geq \beta$ (β -řez). Alfa-beta vrací tutéž hodnotu jako minimax; při dobrém pořadí tahů zkoumá jen $O(b^{m/2})$ uzlů.

(3) *Konkrétní strom.* Listy zleva doprava: 5, 6 | 4, 7, 5 | 3 | 6 | 5, 9 | 7 | 5 | 9, 8 | 6. Hodnoty zdola nahoru: uzly MIN nad listy 5, 4, 3, 6, 5, 7, 5, 8, 6; uzly MAX 5, 3, 6, 7, 5, 8; uzly MIN pod kořenem 3, 6, 5; **kořen MAX** = $\max(3, 6, 5) = \boxed{6}$ (optimální první tah do prostředního následníka).

Procházení zleva doprava prořeže 6 ze 14 listů (prohlédne se jich 8 v pořadí 5, 6, 4, 3, 6, 5, 7, 5).
Tři řezy:

- *MIN nad listy 4, 7, 5:* sourozenec vlevo dal MAX hodnotu 5, tedy $\alpha = 5$; po přičtení listu 4 má tento MIN hodnotu $\leq 4 \leq \alpha$, zbylé listy 7, 5 se **prořežou**.
- *MIN nad listy 5, 9:* sourozenec vlevo dal MAX hodnotu 6, tedy $\alpha = 6$; po listu 5 je hodnota $\leq 5 \leq \alpha$, list 9 se **prořeže**.
- *Pravý uzel MIN pod kořenem:* kořen už má z levých větví 6, tedy $\alpha = 6$; jeho první dítě MAX dá 5, čímž je tento MIN $\leq 5 \leq \alpha$, a celý pravý podstrom MAX (listy 9, 8, 6) se **prořeže**.

9 DPLL a párování

(Reálná otázka: jaro 2025, č. 27 – DPLL a párování)

Zadání.

Nechť $G = (V, E)$ je neorientovaný graf s n vrcholy. Perfektní párování v grafu G je podmnožina hran $M \subseteq E$ taková, že každý vrchol je incidentní s právě jednou hranou z M .

1. Navrhněte formuli ve výrokové logice, která je splnitelná, právě když graf G má perfektní párování. Formuli popište obecně v konjunktivní normální formě (CNF) pro libovolný graf G .
2. Vysvětlíte pojmy *čistý literál* (Pure symbol heuristics) a *jednotková klauzule* (Unit clause heuristics) a jejich použití v rámci algoritmu DPLL pro hledání splnitelných ohodnocení CNF formulí.
3. Napište formuli z bodu 1. pro cestu na čtyřech vrcholech a, b, c, d . Na této formuli demonstруйте použití heuristik z předchozího bodu k nalezení splňujícího ohodnocení.

Náčrt řešení.

1. Formule používá výrokové proměnné m_e pro $e \in E$, které vyjadřují, že hrana e je v perfektním párování. Pro každý vrchol u a jeho incidentní hrany $e_1, \dots, e_k$ klauzule:
 - Každý vrchol je incidentní s alespoň jednou hranou $m_{e_1} \vee \dots \vee m_{e_k}$.
 - Pro každé dvě různé hrany e_i, e_j máme klauzuli: $\neg m_{e_i} \vee \neg m_{e_j}$.
2. Jednotková klauzule je klauzule s jednou proměnnou a tato klauzule vynucuje ohodnocení dané proměnné, které můžeme dosadit do zbytku formule. Čistá proměnná má buď všechny výskyty pozitivní nebo všechny výskyty negativní, a proto můžeme tuto proměnnou též z formule eliminovat.
3. Stačí zapsat formuli výše pro danou cestu. Z čistých klauzulí m_{ab} a m_{cd} musí být hrany ab a cd v párování a z klauzule $\neg m_{ab} \vee \neg m_{bc}$ po eliminaci m_{ab} dostáváme, že bc není v párování.

Doplňující vysvětlení (nad rámec oficiálního náčrtu): Klauzule m_{ab} a m_{cd} (vrcholy a a d mají jedinou incidentní hranu) jsou *jednotkové* klauzule, ne čisté; náčrt je tak míní. Celá formule pro cestu $a - b - c - d$:

$$m_{ab} \wedge (m_{ab} \vee m_{bc}) \wedge (\neg m_{ab} \vee \neg m_{bc}) \wedge (m_{bc} \vee m_{cd}) \wedge (\neg m_{bc} \vee \neg m_{cd}) \wedge m_{cd}.$$

Jednotková propagace: $m_{ab} = \text{true}$ a $m_{cd} = \text{true}$; klauzule $\neg m_{ab} \vee \neg m_{bc}$ se tím zkrátí na jednotkovou $\neg m_{bc}$, takže $m_{bc} = \text{false}$, a všechny klauzule jsou splněny bez větvení. Po dosazení m_{ab}, m_{cd} se navíc m_{bc} vyskytuje v nesplněných klauzulích už jen negativně, je tedy *čistý literál* a pravidlo čistého literálu by ho nastavilo na false také. Výsledek: perfektní párování $\{ab, cd\}$.

10 Bayesovské učení

(Reálná otázka: jaro 2025, č. 30 – Bayesovské učení)

Zadání.

Supermarket bere bedny jablek od tří dodavatelů, polovinu beden od A, třetinu od B a šestinu od C. Bedny od sebe nelze pohledem rozeznat. Jablka jednotlivých dodavatelů se mírně liší průměrem – mají 9cm, 10cm, nebo 11cm. Vybrali jsme náhodně vzorky jablek od jednotlivých dodavatelů a spočítali jsme, jak běžné jsou různé velikosti jablek u různých dodavatelů. Výsledky tohoto měření, počty jablek různých velikostí rozdělené dle jednotlivých dodavatelů, jsou v tabulce níže.

Dodavatel	9cm	10cm	11cm
A	5	4	1
B	9	8	3
C	1	8	1

1. Uveďte maximálně věrohodné odhady pravděpodobností velikosti jablka pro jednotlivé dodavatele.
2. Definujte pojem *maximálně věrohodná hypotéza*.
3. Z téže bedny jsme náhodně vzali dvě jablka a změřili je – jedno mělo průměr 10cm a druhé průměr 11cm. Která z hypotéz: „Jablka jsou od A.“, „Jablka jsou od B.“, „Jablka jsou od C“ je maximálně věrohodná? Odpověď zdůvodněte.
4. Jaká je *bayesovsky optimální predikce* pravděpodobnosti velikosti 10cm pro další jablko ze stejné bedny (při předchozích pozorováních 10cm a 11cm jablek z této bedny)? Svou odpověď zdůvodněte.

Náčrt řešení.

1. Velikosti máme v diskretních intervalech. Maximálně věrohodné odhady diskretního rozložení jsou podíl četností, tj. pravděpodobnosti jsou:

$P(\text{velikost} \mid h)$	9cm	10cm	11cm
A	.5	.4	.1
B	.45	.4	.15
C	.1	.8	.1

2. (Definice pojmu maximálně věrohodná hypotéza – hypotéza maximalizující věrohodnost dat $P(\text{data} \mid h)$.)
3. Maximálně věrohodná hypotéza po pozorování 10 a 11 cm je C dávající pravděpodobnost pozorování 8% oproti 4% A a 6% B , viz třetí sloupec v tabulce níže.
4. Spočteme aposteriorní pravděpodobnost jednotlivých hypotéz a tou vážíme jejich predikce, predikujeme 50% pro velikost 10 cm, výpočet viz tabulka:

h	$P(h)$	$P(\text{data} \mid h)$	$P(h, \text{data})$	$P(h \mid \text{data})$	predikce 10cm * $P(h \mid \text{data})$
A	.5	.04	.02	3/8	.4 * 3/8
B	1/3	.06	.02	3/8	.4 * 3/8
C	1/6	.08	.04/3	2/8	.8 * 2/8
		$P(\text{data}) = .16/3$			$\Sigma = 0.5$

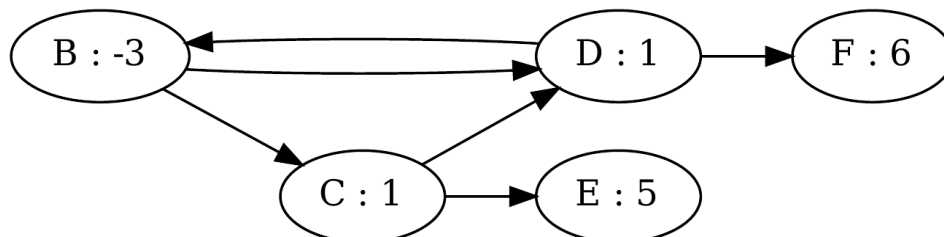
11 Bellmanova rovnice užitku

(Reálná otázka: léto 2025, č. 29 – Bellmanova rovnice užitku (MDP, iterace strategií))

Zadání.

Robot se pohybuje v konečném stavovém prostoru S . Pro každý stav $s \in S$ známe množinu možných akcí $A(s)$ a pravděpodobnost $P(s' \mid s, a)$, že se robot ze stavu s akcí $a \in A(s)$ přesune do stavu $s' \in S$. Při průchodu stavem $s \in S$ dostane robot odměnu (reward) $R(s)$. Celkový užitek ze stavu s' je diskontován faktorem $\gamma = 0.9$.

- (1) Napište rovnici pro výpočet maximálního užitku (utility) $U(s)$ ve stavu s .



Obrázek udává stavový prostor, odměnu v jednotlivých stavech a dvě možné akce z každého stavu kromě koncových stavů E a F . Pravděpodobnost, že se robot přesune do stavu, který je

daný orientovanou hranou zvolené akce, je 0.8 a ve zbylé pravděpodobnosti 0.2 se přesune do stavu, do kterého směřuje druhá možná akce.

- (2) Napište soustavu rovnic, jejichž řešení dá užítiky jednotlivých stavů, jestliže zadáme robotu akce (strategii/policy) $B \rightarrow C$, $C \rightarrow D$ a $D \rightarrow F$.
- (3) Jakým algoritmem bychom tuto soustavu vyřešili?
- (4) Jak ze spočtených užitek zjistíme, zda zadaná volba akcí dává maximální možné užítiky pro všechny stavy?
- (5) Jak bychom pokračovali, abychom našli akce pro všechny stavy maximalizující užitek?

Řešení. (V PDF bez náčrtu řešení; vlastní vzorové řešení, soustava ověřena numericky.)

Stavy a odměny: $B(-3)$, $C(1)$, $D(1)$ (nekoncové), $E(5)$, $F(6)$ (koncové). Akce: B má $\{\rightarrow C, \rightarrow D\}$, C má $\{\rightarrow D, \rightarrow E\}$, D má $\{\rightarrow F, \rightarrow B\}$; zvolená akce vede do svého cíle s pravd. 0,8, do cíle druhé akce s pravd. 0,2. Koncové užítiky $U(E) = R(E) = 5$, $U(F) = R(F) = 6$.

(1) *Bellmanova rovnice* (maximální užitek):

$$U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' | s, a) U(s').$$

(2) *Soustava pro strategii* $B \rightarrow C$, $C \rightarrow D$, $D \rightarrow F$ (pevná strategie $\Rightarrow$ odpadá max, rovnice jsou lineární):

$$U(B) = -3 + 0,9 [0,8 U(C) + 0,2 U(D)],$$

$$U(C) = 1 + 0,9 [0,8 U(D) + 0,2 \cdot 5],$$

$$U(D) = 1 + 0,9 [0,8 \cdot 6 + 0,2 U(B)].$$

(3) *Algoritmus řešení.* Jde o *vyhodnocení strategie* (policy evaluation): tři lineární rovnice o třech neznámých vyřešíme přímo (Gaussova eliminace, $O(n^3)$), případně iterativně zjednodušeným Bellmanovým updatem bez max, dokud se užítiky neustálí. Řešení:

$$\boxed{U(B) \approx 2,383, \quad U(C) \approx 6,039, \quad U(D) \approx 5,749}, \quad U(E) = 5, \quad U(F) = 6.$$

(4) *Test optimality.* Pro každý stav ověříme *Bellmanovo optimální kritérium*: zda zvolená akce dosahuje $\max_a \sum_{s'} P(s' | s, a) U(s')$, tj. zda by krok zlepšení strategie nezměnil akci. Spočítáme očekávaný užitek obou akcí (z užiteků výše): B : $\rightarrow C$ dá 5,981 vs. $\rightarrow D$ 5,807; C : $\rightarrow D$ dá 5,599 vs. $\rightarrow E$ 5,150; D : $\rightarrow F$ dá 5,277 vs. $\rightarrow B$ 3,106. V každém stavu je zvolená akce lepší, takže strategie **je optimální**.

(5) *Hledání optimálních akcí.* Obecně *iterace strategií* (policy iteration): střídáme vyhodnocení strategie (bod 3) a zlepšení strategie (v každém stavu vezmi akci s nejvyšším očekávaným užitekem, bod 4), dokud se strategie přestane měnit. Zde jsme v bodě 4 ověřili, že daná strategie je optimální, takže iterace strategií by skončila hned.

12 Jednotahové hry

(Reálná otázka: podzim 2025, č. 27 – Jednotahové hry (Nash, dominantní strategie, Pareto))

Zadání.

Uvažte jednotahové hry, kdy hráči volí tahy naráz a jejich odměna je přiřazena na základě kombinace zvolených tahů.

- (A) Pro jednotahové hry dvou hráčů napište definice pojmů *dominantní strategie*, *Nashovo ekvilibrium* a *Pareto optimální výsledek*.
- (B) V následujících maticích každé políčko obsahuje dvě čísla, která udávají zisk hráčů A a B po vykonání akcí *Right* nebo *Left*. Hráči zisk maximalizují. Pro tyto tři jednotahové hry napište všechny dominantní strategie, Nashova ekvilibria a Pareto optimální výsledky.
- (C) Pomocí těchto příkladů vysvětlete rozdíly mezi všemi dvojicemi pojmů definovaných v části (A).

		B	
		Left	Right
A	Left	$A = 10, B = 5$	$A = 7, B = 8$
	Right	$A = 0, B = 6$	$A = 15, B = 7$

Tabulka 1: Hra (1)

		B	
		Left	Right
A	Left	$A = 100, B = 100$	$A = 0, B = 0$
	Right	$A = 0, B = 0$	$A = 50, B = 50$

Tabulka 2: Hra (2)

		B	
		Left	Right
A	Left	$A = 20, B = 20$	$A = 0, B = 40$
	Right	$A = 40, B = 0$	$A = 10, B = 10$

Tabulka 3: Hra (3)

Řešení. (V PDF bez náčrtu řešení; vlastní vzorové řešení, čistá ekvilibria ověřena prohledáním všech profilů, smíšená výpočtem přes princip indiference.)

(A) *Definice.* Dominantní strategie hráče je akce, která mu dá lepší výsledek než kterákoli jiná jeho akce při každé volbě soupeře. Nashovo ekvilibrium je profil voleb, v němž si žádný hráč jednostrannou změnou své akce nepolepší (stabilní vůči jednostranné odchylce). Pareto optimální výsledek je výsledek, který nelze zlepšit všem hráčům zároveň (neexistuje jiný výsledek, jež by všichni preferovali).

(B) *Analýza tří her* (výsledky jako (zisk A, zisk B)):

- **Hra 1.** B má dominantní strategii *Right* (při $A=Left$: $8 > 5$; při $A=Right$: $7 > 6$); A dominantní strategii nemá. Jediné Nashovo ekvilibrium je *čistě*: (Right, Right) = (15, 7). *Vlastní smíšené* ekvilibrium neexistuje: B hraje svou ostře dominantní *Right* s pravd. 1 v každé rovnováze, a A na to nejlépe odpoví čistou *Right* ($15 > 7$) – ani jeden hráč tedy nerandomizuje. Pareto optimální: (15, 7) a (7, 8) (navzájem neporovnatelné – vyšší zisk A proti vyššímu zisku B).
- **Hra 2** (koordinální). Dominantní strategie nemá nikdo. Dvě *čistá* Nashova ekvilibria: (Left, Left) = (100, 100) a (Right, Right) = (50, 50). Navíc existuje *vlastní smíšené* Nashovo ekvilibrium: oba hrají *Left* s pravděpodobností $\frac{1}{3}$ a *Right* s pravděpodobností $\frac{2}{3}$. (Z principu indiference: hraje-li A *Left* s pravd. p , je B lhostejný, právě když $100p = 50(1-p)$, tj. $p = \frac{1}{3}$; symetricky pro A.) Očekávaný zisk každého je pak $100 \cdot \frac{1}{3} = 33,3$, tedy horší než v obou čistých rovnováhách. Celkem má tedy hra 2 **tři** Nashova ekvilibria (dvě čistá, jedno smíšené). Pareto optimální: jen (100, 100) – výsledek (50, 50) je Pareto-dominován (100, 100).
- **Hra 3** (struktura věznova dilematu). Oba mají dominantní strategii *Right* (A: $40 > 20$, $10 > 0$; B symetricky). Jediné Nashovo ekvilibrium je *čistě*: (Right, Right) = (10, 10); protože ostře dominantní *Right* hrají oba s pravd. 1, *vlastní smíšené* ekvilibrium neexistuje. Pareto optimální: (20, 20), (0, 40), (40, 0). Nashův výsledek (10, 10) je Pareto-dominován výsledkem (20, 20) – racionální sobecká hra vede k výsledku horšímu pro oba.

(C) *Rozdíly mezi pojmy.*

- *Dominance vs. Nash*: profil v dominantních strategiích je vždy Nash (hry 1, 3), ale Nash *nemusí* pocházet z dominance – hra 2 nemá žádnou dominantní strategii, a přesto má dvě Nashova ekvilibria. Existence Nashe tedy nevyžaduje dominanci.
- *Nash vs. Pareto*: Nash je o *stabilitě* (nikdo neodbočí), Pareto o *kolektivní dobrotě*. V hře 3 je Nash (10, 10) Pareto-dominovaný (stabilní, ale pro oba špatný); v hře 1 je Nash (15, 7) Pareto optimální. Mohou se, ale nemusí krýt.

- *Dominance vs. Pareto*: hra v dominantních strategiích nezaručuje Pareto optimum – hra 3 končí v Pareto-dominovaném (10, 10). To je jádro věžňova dilematu.
- *Čisté vs. smíšené Nashovo ekvilibrium*: ostře dominovaná strategie není v nosiči žádné rovnováhy, takže hry 1 a 3 (dominance) mají jen čistou NE. Naproti tomu koordinační hra 2 má vedle dvou čistých i smíšenou NE – konečná hra má dle Nashovy věty vždy aspoň jednu, případně právě jen smíšenou.

13 CSP

(Reálná otázka: jaro 2026, č. 27 – CSP (binární CSP, hranová konzistence, AC-3))

Zadání.

1. Definujte binární Constraint Satisfaction Problem (CSP), konkrétně vstup a výstup tohoto problému. Co znamená „hranová konzistence“ v kontextu CSP?
2. Pokud je problém hranově konzistentní, je zaručeno, že je splnitelný? Pokud problém není hranově konzistentní, může být stále splnitelný? Svou odpověď zdůvodněte nebo poskytněte protipříklady.
3. Uvažte následující binární CSP, aplikujte algoritmus AC-3 pro dosažení hranové konzistence. Součástí řešení musí být postup.

Mějme celočíselné proměnné A, B, C a D s doménami:

$$D_A = \{1\}$$

$$D_B = \{1, 2, 3\}$$

$$D_C = \{1, 2, 3\}$$

$$D_D = \{1, 2, 3\}$$

Podmínky:

$$C_1 : C + D = 4$$

$$C_2 : A + B \leq 3$$

$$C_3 : B + C \geq 4$$

Náčrt řešení.

1. CSP je definován množinou proměnných, doménou (tj. množinou hodnot) pro každou proměnnou a množinou podmínek nad proměnnými. V binárním CSP může každá podmínka obsahovat maximálně dvě proměnné. Podmínka je hranově konzistentní, pokud pro každou hodnotu z domény každé proměnné v dané podmínce existuje přípustná hodnota z domény ostatních proměnných z dané podmínky. Problém je hranově konzistentní, pokud je každá podmínka hranově konzistentní.
2. Hranově konzistentní problém nemusí být splnitelný. Například tři proměnné X_1, X_2, X_3 s binárními doménami $\{0, 1\}$ a podmínkami pro každou dvojici $X_i \neq X_j$. Hranově nekonzistentní problém může být splnitelný. Například příklad ze zadání.
3. Algoritmus AC-3 mění domény proměnných tak, aby byly všechny podmínky hranově konzistentní. Podmínky, které obsahují proměnné, kterým se změnily domény, se musí zkontrolovat znovu. Hranově konzistentní problém má domény $D_A = \{1\}$, $D_B = \{1, 2\}$, $D_C = \{2, 3\}$, $D_D = \{1, 2\}$.

Doplňující vysvětlení (nad rámec oficiálního náčrtu): Postup AC-3 (revize hrany (X, Y) vyřadí z D_X hodnoty bez opory v D_Y ; po zúžení D_X se do fronty vrátí hrany (Z, X) od ostatních sousedů Z):

1. C_2 , revize (B, A) : $A = 1$ a $A + B \leq 3$ dává $B \leq 2$, vyřadí se $B = 3$, $D_B = \{1, 2\}$. Revize (A, B) nic nemění.

2. C_3 , revize (C, B) : $B \leq 2$ a $B + C \geq 4$ dává $C \geq 2$, vyřadí se $C = 1$, $D_C = \{2, 3\}$. Revize (B, C) nic nemění ($b = 1$ i $b = 2$ má oporu $c = 3$).
3. Zúžila se D_C , do fronty se vrací (D, C) : $C + D = 4$ a $C \in \{2, 3\}$ dává $D \in \{1, 2\}$, vyřadí se $D = 3$.
4. Zúžila se D_D , do fronty se vrací (C, D) : $c = 2$ má oporu $d = 2$, $c = 3$ oporu $d = 1$, beze změny. Fronta se vyprázdní.

Výsledek odpovídá náčrtu. Protože původní domény obsahovaly hodnoty bez opory (např. $B = 3$), zadaný problém hranově konzistentní nebyl, a přesto je splnitelný (např. $A = 1$, $B = 2$, $C = 2$, $D = 2$); to je protipříklad, na který náčrt v bodě 2 odkazuje.