

1 STROJNO UCENJE

1.1 PROBLEMSKI PROSTOR, OCENJEVANJE ZNANJA

1.2 EVALVIRANJE HIPOTEZ

Pomembni kriteriji:

- konsistentnost hipotez z primeri (ucnimi)
- splosnost (tocnost za nevidene primere)
- razumljivost hipotez

TP=true positive, TN=true negative, FP=false positive (napaka 1. tipa), FN=false negative (napaka 2. tipa)

Klasifikacijska tocnost = $\frac{TP+TN}{TP+TN+FP+FN} = \frac{TP+TN}{N}$
Obcutljivost/senzitivnost = $TPR = \frac{TP}{TP+FN}$

1.3 GRADNJA ODLOCITVENIH DREVES

Za koliko se entropija zmanjsa po delitvi z Atributom A:

Informacijski prispevek: $Gain(A) = I - I_{res}(A)$, $I=H(C)$

$I_{res} = -\sum_{v_i \in A} P v_i \sum_c P(c|v_i) \log_2 P(c|v_i)$

Razmerje inofrmacijskega prispevka atributa A:

$IGR(A) = \frac{Gain(A)}{H(A)}$

1.3.1 TDIDT (TOP DOWN INDUCTION DECISION TREE) ALGORITHM

Pozresen algoritem, ki lokalno izbira najbolsi atribut.

- kratkoviden algoritem

1.3.2 BINARIZACIJA ATRIBUTOV

Aleternativa za resevanje problematike z vecvrednostnimi atributi:

Strategije (za primer B = {Y, G, R, B}):

- [{Y}], [R, G, B]] (one-vs-all)
- [{Y, R}], [G, B]]
- vpeljava bianrnih atributov za vsako barvo

Primer B = {Y, G, R}, konstruiramo 3 nove binarne attribute:

barva	Y	G	R
Y	1	0	0
G	0	1	0
R	0	0	1

Prednost: manjse vejanje drevesa.

1.4 UCENJE IZ SUMNIH PODATKOV (REZANJE)

tocnost t...verjetnost pravilnosti klasifikacije
napaka e ... 1 - t

relativna frekvenca $p = \frac{n}{N}$

m-ocena $p = \frac{n+pa \cdot m}{N+m}$

m... koliko zaupam apriorni verjetnosti

pa apriorna verjetnost (domenski ekspert lahko pove)

Laplacova ocena verjetnosti $p = \frac{n+1}{N+k}$

k...stevilo vseh moznih razredov

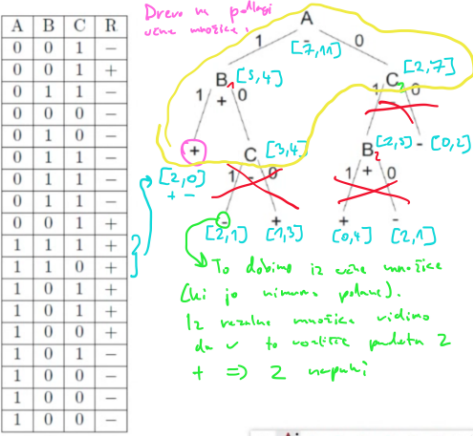
1.4.1 REP (REDUCED ERROR PRUNNING)

Dela dobro ce imamo veliko rezalno mnozico.

Obicajno uporabljamo relativno frekvenco za ocenjevanje verjetnosti.

$G(v) = \#napak_T - \#napak_v$

$G(v) \geq 0 \Rightarrow$ rezemo podrevo



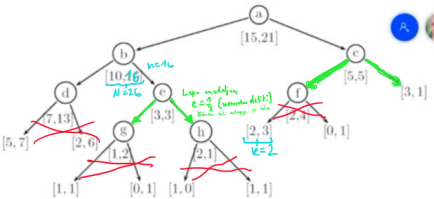
$e(C) = 3$

$e_T = 2 + 3 = 5$

$G(C) = 5 - 3 = 2 \geq 0 \rightarrow$ rezemo

1.4.2 MEP (MINIMAL ERROR PRUNNING)

e...staticna napaka,E...vzvrtna napaka, $e \in E \rightarrow$ rezemo poddrevo



(Laplace)

$e_L(d) = 1 - t = 1 - \frac{13+1}{20+2} = 0.363$

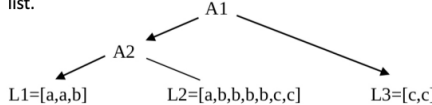
$E_L(d) = 12/20 \cdot e_L(d_1) + 8/20 \cdot e_L(d_d) = \frac{12}{20} \cdot (1 - \frac{7+1}{12+2}) + \frac{8}{20} (1 - \frac{13+1}{20+2})$

1.5 OCENJEVANJE USPEŠNOSTI MODELOV

tocnost t ... verjetnost pravilnosti klasifikacije

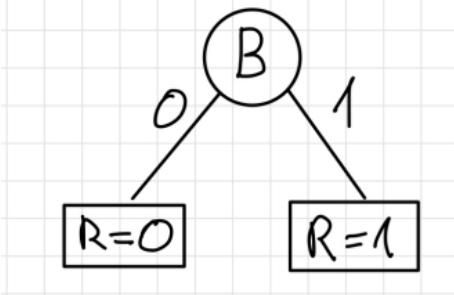
Laplacova ocena verjetnosti $p = \frac{n+1}{N+k}$

k...stevilo vseh moznih razredov list.



$t_{L1} = \frac{2+1}{3+3} = 0.5$, $t_{L2} = \frac{4+1}{7+3} = 0.5$, $t_{L3} = \frac{2+1}{2+3} = 0.6$

tocnost drevesa: $t_D = 3/12 \cdot 0.5 + 7/12 \cdot 0.5 + 2/12 \cdot 0.6 = 0.5167$



$e = 1 - (P(B=0)P(R=0|B=0) + P(B=1)P(R=1|B=1))$

1.6 OBRAVNANVA MANKAJOCIH ATRIBUTOV, NAVINI BAYESOV KLASIFIKATOR

1.6.1 NAIVNI BAYES

Ce poznamo razred, kam klasificiramo ce nepoznamo atributov:

Klasifikator: $\operatorname{argmax}_{c \in C} P(c) \prod_{i=1}^n P(x_i|c)$

c...razred, x_i ...atributi

Verjetnost::

$P(C = c|x_1, \dots, x_n) = \frac{P(C = c)P(X_1 = x_1|C = c)P(X_2 = x_2|C = c) \dots}{P(X_1 = x_1)P(X_2 = x_2) \dots}$

Primer moski: visina ≥ 175 , teza ≥ 65 , spol = M

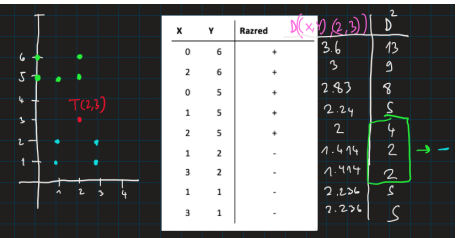
X\Y	Razred A	Razred B
pa	$P(A) = \frac{2}{3}$	$P(B) = \frac{1}{3}$
spol	$P(M A)$	$P(M B)$
visina	$P(V \geq 175 A)$	$P(V \geq 175 B)$
teza	$P(T \geq 65 A)$	$P(T \geq 65 B)$
$P(y) \prod_{i=1}^n P(x_i y)$

1.6.2 NOMOGRAMMI

Ciljni razred $C = c_T$

$X_{X_i=x_j} = \ln \left(\frac{P(X_i = x_j|C = c_T)}{P(X_i = x_j|C = \bar{c}_T)} \right)$

1.7 K-NAJBILZIJIH SOSEDOV



2 VRSTE UCENJA

2.1 NADZOROVANO UCENJE (SUPERVISED LEARNING)

Ucni primeri so podani/oznaceni kot vrednosti vhodov in izhodov.

$(\vec{x}_1, \vec{y}_1), (\vec{x}_2, \vec{y}_2), \dots, (\vec{x}_N, \vec{y}_N)$

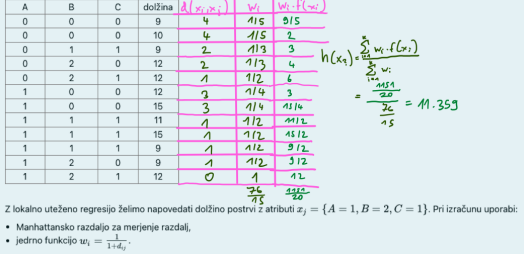
\vec{x}_i ... atributi, \vec{y}_i ... ciljna spremenljivka

Locimo dve vrsti problemov:

1. Klasifikacijski problemi - y; diskretna
2. Regresijski problemi - y; zvezna

2.1.1 LOKALNO UTEZENA REGRESIJA

$$h(\vec{x}_7) = \frac{\sum_{i=1}^k w_i \cdot f(\vec{x}_i)}{\sum_{i=1}^k w_i}, w_i(d) \dots utez$$



2.1.2 REGRESIJSKA DREVEŠA

Linearna regresija je poseben primer regresijskega drevesa.

V listih regresijskega drevesa vcasih napovemo kar povprečno vrednost.

2.2 NENADZOROVANO UCENJE (UNSUPERVISED LEARNING)

Ucni primeri niso oznaceni (nimajo ciljne spremenljivke), ucimo se vzorcev v podatkih, (npr. grucenje)

2.2.1 HIERARHICNO GRUCENJE

Poveze po podobnosti med primeri, primer zacne kot samostojna gruca, na koncu vsi primeri pripadajo eni gruci

Dendrogram: drevo, ki predstavlja grucenje.

Single-linkage: povezava med grucami je najkrajse razdalje med primeroma iz razlicnih gruc.

Complete-linkage: povezava med grucami je najdaljsa razdalja med primeroma iz razlicnih gruc.

Average-linkage: povezava med grucami je povprečna razdalja med primeroma iz razlicnih gruc.

2.2.2 K-MEANS

1. V prostor dodamo k centroidov, ki predstavljajo gruce.

2. Izracunamo ketri centroid je najblizji vsakemu primeru.

3. Izracunamo nove centre gruc $= \frac{1}{|G|} \sum_{i \in G} x_i$

4. Ponovimo korake 2 in 3 dokler se centri ne premaknejo.

2.3 SPODBUJEVALNO UCENJE - REINFORCEMENT LEARNING

Intelligentni agent se uci iz zaporedja nagrad in kazni

2.4 OCENJEVANJE UCENJA

2.4.1 PRECNO PVERJANJE

Poseben primer veckratnega ucenja in testiranja

k-kratno precno preverjanje

- celo ucno mnozico razbij na k disjunktnih podmnozic
- za vsako od k podmnozic:

- uporabi mnozico kot testno mnozico
- uporabi preostalih k-1 mnozic kot ucno mnozico
- povpreci dobljenih k ocen tocnosti v koncno oceno

Pri precnem preverjanju uporabimo vse podatke za testiranje in vse za ucenje

Metoda leave one out je poseben primer precnega preverjanja

Imamo dve hipotezi A in B. Izkase se, da A bolje napoveduje na ucnih podatkih B pa na testnih. Potem je B verjetno boljsa hipoteza.

3 PREISKOVANJE

3.1 NEINFORMIRANI PREISKOVALNI ALGORITMI

3.1.1 ISKANJE V SIRINO

3.1.2 ISKANJE V GLOBINO

Izboljsave:

- Iskanje s sestopanjem

- **depth-limited-search** (vnaprej definiramo globino l (dolocimo preko domenskega znanja))

3.1.3 ITERATIVNO POGLABLJANJE

problem gobinsko omejenega iskanja -> nastavitev meje l Mejo l postopoma povecujemo za 1, dokler ne najdemo resitve.

- **popolnost:** Da
- **optimalnost:** Da
- **casovna zahtevnost** $O(b^d)$
- **prostorska zahtevnost** $O(bd)$

Boljse od iskanja v globino/sirino

3.1.4 DVOSMERNO ISKANJE

Ideja: poghati vzporedni iskanji od zacetka do cilja in od cilja do zacetka.

Motivacija:

Implemenatcija dvosmernega iskanja

- ciljno vozlisce mora biti znano
- originalni problemski prostor preslikamo v dvosmerni prosto stanj E1, E2 dosegljiv iz E in S1,S2,S3 dosegljiv iz S (S,E) -> (S1, E1), (S1,E2), (S2, E1), (S2, E2)... Vozlisce (Si, Ei) je v dvos-mernem prostur ciljo vozlisce ce velja E=S (soda dolzina na isto mesto pridemo iz obeh strani) ali S->E (liha pot sosednja)

3.1.5 CENOVNO - OPTIMALNO ISKANJE

- posplositev iskanja v sirino (iskanje v sirino je optimalno, ce so cene vseh povezav enake 1)

- dijkstra basically (sam do zadnga noda)
- <https://stackoverflow.com/a/14587449>

3.2 INFORMIRANI PREISKOVALNI ALGORITMI

3.2.1 HEVRISTICNO PREISKOVANJE

ideja: preiskovanje usmerjamo z dodatnim znanjem (ocenitven funkcija za obetavnost vozlisca)

hevristika je ocenitvena funkcija za obetavnost vozlisca

- **optimisticna/dopustna:** $h(n) \leq h^*(n)$ (h^* je optimalna ocena)

- **optimalna:** $h(n) = h^*(n)$

- **pesimisticna:** $h(n) \geq h^*(n)$

3.2.2 POZRESNO PREISKOVANJE/ GREEDY BEST-FIRST SEARCH

$h(n)$ hevristicna ocena

vrednotenje vozlisca $f(n) = h(n)$ hevristicna ocena ... npr manhat-tan distance (zracna razdalja)

- **popolnost** (ali najde vedno resitev): Ne
- **optimalnost:** Ne
- **casovna zahtevnost** $O(b^m)$, kjer je m najveca globina drevesa

3.2.3 A*

A* is informed version of **dijkstra** (uses heuristics and pq)

Vozlisca vrednotimo: $f(n) = g(n) + h(n)$

$g(n)$ cena poti do n (znano),

$h(n)$ cena od n do najblizjega cilja (ocena)

prioritetna vrsta (max glede na f(n)) Basically dijkstra + h(n) (A* is basically an informed variation of Dijkstra.)

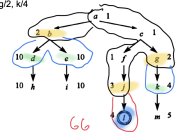
- **popolnost:** Da (ce ustreza pogoju dopustnosti)
- **optimalnost:** Da (ce ustreza pogoju dopustnosti)
- **casovna zahtevnost** $O(b^m)$, kjer je m najveca globina drevesa

3.2.4 IDA* (ITERATIVE DEEPENING A*)

Dfs with heuristics and iterative bound (value)

- primer:
 - podane so vrednost $f(n)$ ($= g(n) + h(n)$) vozlišč
 - simuliraj preiskovanje z IDA*

- generirana vozlišča
 - 1. iteracija, meja=1: a/1, b/2, c/1, f/1, j/3, g/2
 - 2. iteracija, meja=2: a/1, b/2, d/10, e/10, c/1, f/1, j/3, g/2, k/4
 - 3. iteracija, meja=3: a/1, b/2, d/10, e/10, c/1, f/1, j/3, i/4, g/2, k/4
 - 4. iteracija, meja=4: a/1, b/2, d/10, e/10, c/1, f/1, j/3, i/4



Ucinkovitost

- neucinkovit ce vozlisca raznolika $f(n)$
- prednost: ne hrani vec vseh vozlic kot A*
- optimalen: ce razvija v prioritetnem vrsntem redu, $h(n)$ mora biti **monotona|konsistentna** ($h(n)$ skos pada) (posledicno tudi dopustna)

$$h(n) \leq c(n, n') + h(n')$$

(h naslednjega vozlisca manjsi ker je blizji cilja)

- monotona \rightarrow dopustna (proti primer $h(n) = 0$)

3.2.5 KAKOVOST HEVRISTICNIH FUNKCIJ

7	2	4
5		6
8	3	1

Primer igra 8 ploscic

- h_1 : stevilo ploscic ki niso na pravem mestu (8)

- h_2 : vsota manhattanskih razdalj ploscic do pravega mesta ($3+1+2+2+2+3+3+2=18$)

Kakovost h ocenimo z:

- stevilom generirarnih vozlic

- z efektivnim faktorjem vejanja (koliko vozlic N je algoritem general da je na globini d nasel resitev)

Globina	število generiranih vozlišč			efektivni faktor vejanja		
	IDŠ	A(h)	A(h)	IDŠ	A(h)	A(h)
2	10	6	3	2,45	1,79	1,79
4	112	13	12	2,87	1,48	1,45
6	680	20	18	2,73	1,34	1,30
8	6384	39	25	2,80	1,33	1,24
10	47127	93	39	2,79	1,38	1,22
12	3644035	227	73	2,78	1,42	1,24
14	?	539	113	?	1,44	1,23
16	?	1301	211	?	1,45	1,25
18	?	3056	363	?	1,46	1,26
20	?	7276	676	?	1,47	1,27
22	?	18094	1219	?	1,48	1,28
24	?	39135	1641	?	1,48	1,26

Vidimo $h_2(n) \geq h_1(n) \forall n$ pravimo h_2 **dominira** h_1

3.3 LOKALNO PREISKOVALNI ALGORITMI

3.3.1 PLEZANJE NA HRIB

Ne pomnemo poti do cilja, ampak samo trenutno stanje

Koristni v primerih:

- ce nas zanima samo kakovost resitve (in ne pot do cilja)

- reševanje optimizacijskih problemov (kjer je podana **kriterijska funkcija** za oceno kakovosti resitve)

Prednosti:

- majhna poraba prostora

Primer 4 kraljice na sahovnici - kriterijska funkcija: maksimiziramo - (minus) stevilo kraljic, ki se medsebojno napadajo
Tezave:

- lokalni maksimumi
- "rame, plaote" (kriterijska funkcija konstantna vrednost)
- grebeni (za plezanje navzgor je potreben sestop po pobocju grebena)
 - Resevanje iz lokalnih maksimumov:
- **koraki vstran:** ce ima naslednje stanje isto vrednost kriterijske funkcije, dovolimo premik v to stanje
- **stochasticno** plezanje na hrib: iz mnozice boljsih stanj, verjetnos-tno izberemo naslednje stanje (pri cemer upostevamo da imajo boljsa stanja vecjo verjetnost izbora)
- **nakljucni ponovni zagon:** veckrat pozeni plezanje na hrib iz nakljucnih stanj dokler ne najdes resitve

3.3.2 SIMULIRANO OHLAJANJE

algoritem ki izvira iz metalurgije (ko je jeklo tekoce, so molekule v njem bolj gibljive; ko se ohlaja se strjuje in molekuele se umirjajo) Analogija:

- generiramo nakljucne sosedo trenutnega stanja
- ce najdemo **boljse stanje ga izberemo**
- ce najdemo **slabse stanje, ga izberemo z doloceno verjetnostjo**
- verjetnost izbire neoptimalnega stanja s casom pada (nizanje temperature)

3.3.3 LOKALNO ISKANJE V SNOPU

Algoritem:

- v spominu hrani k aktualnih stanj namesto enega
- izberi k optimalnih stanj od sosedov aktualnih stanj
- ponavljaj do ustavitnega pogoja

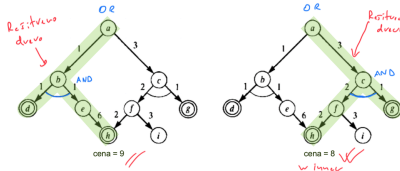
3.4 PREISKOVANJE GRAFOV AND/OR, NEDETERMINISTICNO OKOLJE

Pomagajo reševati probleme z **dekompozicijo na manjse probleme** Uporabnost:

- princip deli in vladaj
- iskanje v nedeterministicnih okoljih
- igre med dvema nasprotnikom a s popolno informacijo (sah, dama)
- ekspertno reševanje problem

Primer graf dekompozicija v dva manjsa problema skozi g in f

Resitveno drevo je resitev AND/OR grafov



3.4.1 AO*

- posplositev A* na grafe AND/OR
- **popoln in optimalen** $\Leftrightarrow h(n)$ ne precenjuje dejanske cene do cilja

$F(N)$... ocena za usmerjanje preiskovanja

$H(N)$... dinamiena hevristicna ocena

Postopek:

1. Razvij najcenejse vozlisce
 - ce list in koncno (oznaci), preveri 3. korak, nadaljuj v 1.
 - ce list in ni koncno (oznaci) vrednost vozlisca = ∞
2. Posodobi vse predhodnike
 - v AND starsih, cena starša = \sum sinov + povezava v
 - v OR starsih, cena starša = $\min(\text{sinovi}) + \text{povezava v}$
3. Koncaj ko obstaja pot od zacetnega vozlisca, po kateri v AND vozlicih po vseh sinovih prides do cilja, v OR vozlicih v vsaj

enem

3.4.2 PREISKOVANJE V NEDETERMINISTICNEM OKOLJU:

Nedeterministican akcija - ista akcija lahko obrodi razlicna ciljna stanja

Do resitve ni vec **poti** temvec **drevesa** (uporbljamo AND/OR grafe) Vozsilca OR **mozne akcije**, vozlisca AND **vejanja v mozna stanja**, ki so rezultat nedeterministicnih akcij

3.5 PREISKOVANJE BREZ INFORMACIJ O STANJU

Okolja smo razdelili na **transparent** (agent lahko zazna popolno informacija) in **netransparentna** (brez informacije o stanju) Kej ce imamo opravka z netraspranetim okoljem?

- izvajamo preiskovanje prostora **verjetnih** stanj in ne prostora **dejanskih** stanj
- izvajamo s postokopom omejevanja moznostzi kandidatnih stanj

3.6 IGRANJE IGER

3.6.1 PREDSTAVITEV PROBLEMA

3.6.2 ALGORITEM MINIMAX

- m globina - b

3.6.3 REZANJE ALFA-BETA

4 PLANIRANJE

plan zaporedje akcij, ki pripelje od zacetnega do koncnega stanja

4.1 PLANIRANJE S SREDSTVI IN CILJI (STRIPS)

Agentu opisemo svet in postavimo fizikalne omejitve.

Ne zagotovlja optimalne resitve, obravnavamo le en cilj naenkrat (ko ga dosežemo, se lahko ostali izgubijo) = Sussmanova anomalija

Akcija move(X, From, To)

- pogoj: **cond**=[clr(X), on(X,F), clr(T)] \rightarrow pogoji za izvajanje akcije,
- poz. ucinki: **adds**=[on(X, T), clr(F)] \rightarrow nova stanja,
- neg. ucinki: **dels**=[on(X, F), clr(T)] \rightarrow izbrisana stanja,
- omejitve: **constr**=[F \neq T, X \neq F, X \neq T, block(X)] \rightarrow omejitve akcij (fizikalne omejitve),

Algoritem:

1. Izberi se neresen cilj iz mnozice CILJEV
2. Izberi akcijo, ki izbrani cilj doda v stanje
3. Omogoci izbrano akcijo (izpolni pogoje)
4. Izvedi akcijo (ki izpolni največ pogojev)
5. Ce obstajajo nereseni cilji \Rightarrow 1.

Primer dfs, zlaganje kock

DFS:

start $s = [on(a,1), on(b,2), on(c,a), c(c), c(2), c(3), c(4)]$
 cilj $g = [on(a,b), on(b,c)]$ (cilj se mora učitati v s)

1. cilj $on(a,b)$
 move (a, F, b)
 cend = $[c(a), c(b), on(a,1)]$

1.1. cilj $c(a)$
 move (X, a, T)
 cend = $[c(2), c(c), on(c,a)]$

Poslednje stanje (po koncu)
 $s = [on(a,1), c(a), on(b,2), c(c), c(b), c(4), on(c,2), c(a), on(c,3), c(a)]$

1. cilj $on(a,b)$
 move (a, F, b)
 cend = $[c(a), c(b), on(a,1)]$

Poslednje stanje (po koncu)
 $s = [on(b,3), c(c), c(4), on(b,1), on(c,2), c(a), c(1), on(c,3)]$

2. cilj $on(b,c)$
 move (b, F, c)
 cend = $[c(b), c(a), on(b,3)]$

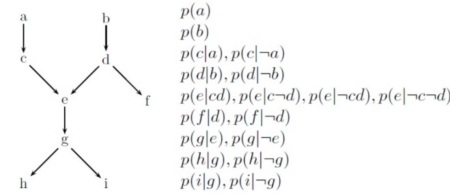
2.2. $c(b)$
 move (X, b, T)
 cend = $[c(a), c(a), on(c,1)]$

novi stanje: $s = [on(a,1), c(a), c(b), on(c,3)]$
 You got the idea

5.1 BAYESOVSKE MREZE

Baye. mreza = Usmerjen graf, kjer so podane zahtevane verjetnosti:

- Za vozlišča **brez starsev** verjetnosti $P(v_i)$
- Za vozlišča z **starsi** pogoje verjetnosti vseh kombinacij starsev



Pravila verjetnostnega sklepanja:

1. **Konjunkcija:** $P(X_1 X_2 | C) = P(X_1 | C)P(X_2 | X_1 C)$
2. **Gotov dogodek:** $P(X | \dots X \dots) = 1$
3. **Nemogoc dogodek:** $P(X | \dots \bar{X} \dots) = 0$
4. **Negacija:** $P(\bar{X} | C) = 1 - P(X | C)$
5. Če je Y naslednik od X in je Y vsebovan v pogojnem delu:
 $P(X | YC) = P(X | C) \cdot \frac{P(Y|XC)}{P(Y|C)}$
6. Če pogojni del ne vsebuje naslednika od X:
 (a) če X **nima** starsev: $P(X | C) = P(X)$, $P(X)$ je podan
 (b) če **ima** X starše: $P(X | C) = \sum_{S \in P_X} P(X | S)P(S | C)$
7. Iz 6b zgoraj: $P(i | gc) = P(i | g)$

4.2 PLANIRANJE Z REGRESIRANJEM CILJEV (STRIPS)

Resitev za sussmanovo anomalijo

Zacnemo v ciljih, regresiramo do zacetka ($G_i \subset S_0$):

1. $G_{i+1} = G_i \cup \text{cond}(A) - \text{adds}(A)$
2. **POGOJ:** $G_i \cap \text{dels}(A) = \emptyset$
3. Preveri da ni protislovja (npr. $G_{i+1} = [on(b,c), \dots, c(c), \dots]$)

PRIMER:

→ zactno_stanje = $[on(a,1), on(b,a), c(b), on(c,3), c(c)]$

→ hocemo da zacetno_stanje $\subset G_i$

1. $G_0 = [on(a,b), on(b,c)]$
 - **on(a,b):** $A_0 = \text{move}(a, \text{From}, b)$
 - From = 1
 - **POGOJ:** $G_0 \cap \text{dels}(A_0) = \emptyset$ ✓
 - $G_1 = [on(a,b), on(b,c), c(a), c(b), on(a,1)] - [c(1), on(a,b)]$ ✓
2. $G_1 = [on(b,c), c(a), c(b), on(a,1)]$
 - **c(a):** $A_1 = \text{move}(X, a, To)$
 - X = c, To = 2
 - **POGOJ:** $G_1 \cap \text{dels}(A_1) = \emptyset$ ✓
 - $G_2 = [on(b,c), c(a), c(b), on(a,1), \underline{c(c)}, c(2), on(c,a)] - [c(a), on(c,2)]$ ✗ (protislovje)
 - **on(b,c):** $A_2 = \text{move}(b, \text{From}, c)$
 - From = 3
 - **POGOJ:** $G_2 \cap \text{dels}(A_2) = \emptyset$ ✓
 - $G_2 = [on(b,c), c(a), c(b), on(a,1), c(c), c(b), on(b,3)]$ ✓
3. $G_2 = \dots$

4.3 RAZPOREJANJE OPRAVIL (PDDL)

5 SKLEPANJE