

1 STROJNO UCENJE

1.1 PROBLEMSKI PROSTOR, OCENJEVANJE ZNANJA

1.2 EVALVIRANJE HIPOTEZ

Pomembni kriteriji:

- **konsistentnost** hipotez z primeri (ucnini)
- **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 (najbolj informativni atribut maksimizira informacijski prispevek minimizira I_{res}:

Gain(A) = H(A) - H_{res}(A)
H_{res}(A) = - ∑_{a_i ∈ A} p(A = a_i) ∑_{c_j ∈ C} p(C = c_j | A = a_i) log₂ p(C = c_j | A = a_i)
Razmerje inofrmacijskega prispevka atributa A:
IGR(A) = $\frac{\text{Gain}(A)}{H(A)}$

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

Prdnost: 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 = $\frac{n+p_d+m}{N+m}$

m... koliko zaupam apriorni verjetnosti

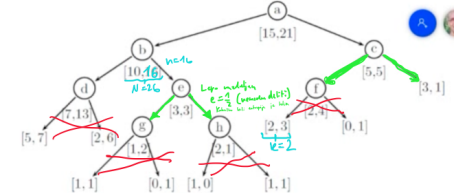
p_a apriorna verjetnost (domenski ekspert lahko pove)

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

k...stevilo vseh moznih razredov

1.4.1 MEP (MINIMAL ERROR PRUNNING)

e...staticna napaka,E...vzvratna napaka,e ≤ E → rezemo poddrevo



(Laplace)

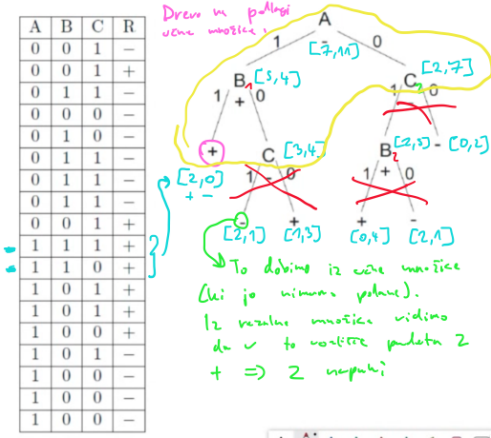
e_L(d) = 1 - t = 1 - $\frac{13+1}{20+2}$ = 0.363

E_L(d) = 12/20 · e_L(d₁) + 8/20 · e_L(d₄) = $\frac{12}{20} \cdot (1 - \frac{7+1}{12+2}) + \frac{8}{20} \cdot (1 - \frac{13+1}{20+2})$

1.4.2 REP (REDUCED ERROR PRUNNING)

Ucna mnozica: 70% za gradnjo, 30% za rezanje (z rezanjem odstranimi poddrevesa, ki niso kriticina in so redundantna tako zmansamo velikost drevesa)
G(v)=st. napacnih klasifikacij v poddrevesu - st. napacnih klasifikacij v korenu poddrevesa

G(v) ≥ 0 ⇒ rezemo podrevo



e(C) = 3, e_T = 2 + 3 = 5, G(C) = 5 - 3 = 2 ≥ 0 → rezemo

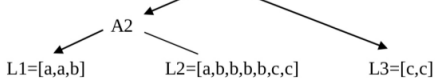
1.5 OCENJEVANJE USPEŠNOSTI MODELOV

tocnost t ... verjetnost pravilnosti klasifikacije

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

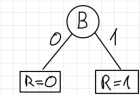
k...stevilo vseh moznih razredov

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 · 0.5 + 7/12 · 0.5 + 2/12 · 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: argmax_{c ∈ C} P(c) ∏_{i=1}ⁿ P(x_i|c)

c...razred, x_i...atributi

Verjetnost:

P(C = c | x₁, ..., 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

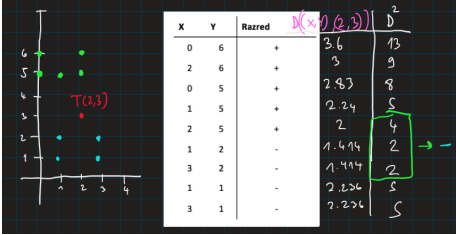
X\Y	Razred A	Razred B
p _a	P(A) = $\frac{2}{3}$	P(B) = $\frac{1}{3}$
spol	P(M A)	P(M B)
visina	P(V ≥ 175 A)	P(V ≥ 175 B)
teza	P(T ≥ 65 A)	P(T ≥ 65 B)
P(y) ∏ _{i=1} ⁿ P(x _i y)

1.6.2 NOMOGRAMI

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-NAJBILZIJH SOSEDOV



1.8 NADZOROVANO UCENJE (SUPERVISED LEARNING)

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

(x̄₁, ȳ₁), (x̄₂, ȳ₂), ..., (x̄_N, ȳ_N)

x̄₁... atributi, ȳ₁... ciljna spremenljivka

Locimo dve vrsti problemov:

1. Klasifikacijski problemi - y_j diskretna
2. Regresijski problemi - y_j zvezna

1.8.1 LOKALNO UTEZENA REGRESIJA

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

A	B	C	dolžina	d(x _i , x _j)	w _i	w _i · f(x _i)
0	0	0	9	4	1/5	0/5
0	0	0	10	4	1/5	2
0	1	1	9	2	1/3	3
0	2	0	12	2	1/3	4
0	2	1	12	1	1/2	6
1	0	0	12	3	1/4	3
1	0	0	15	3	1/4	1/4
1	1	1	11	1	1/2	1/2
1	1	1	15	1	1/2	1/2
1	1	1	9	1	1/2	3/2
1	2	0	9	1	1/2	3/2
1	2	1	12	0	1	1/2

Z lokalno utezeno regresijo želimo napovedati dolžino postvi z atributi x_j = {A = 1, B = 2, C = 1}. Pri izračunu uporabi:
• Manhattansko razdaljo za merjenje razdalj,
• jedrno funkcijo w_i = $\frac{1}{1+d_i^2}$

1.8.2 REGRESIJSKA DREVESEA

Linearna regresija je poseben primer regresijskega drevesa.

V listih regresijskega drevesa včasih napovemo kar povprečno vrednost.

1.9 NENADZOROVANO UCENJE (UNSUPERVISED LEARNING)

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

1.9.1 HIERARHICNO GRUCENJE

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

Dendrogram: drevo, ki predstavlja gruenje.

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.

1.9.2 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.

V mnozici tock A(3,1),B(1,2),C(3,4),D(5,2),E(1), manhattanska razdalja, za-

cetni vrednosti centroidov C1(4,4) in C2(5,4).

Tocka	d(X,C1)	d(X,C2)	Gruca
A	4	5	C1
B	5	6	C1
C	1	2	C1
D	3	2	C2
E	6	7	C1

V naslednji iteraciji sta koordinati centroidov:

C1 = ($\frac{3+1+3+1}{4}$, $\frac{1+2+4+1}{4}$) = (2, 2) in C2 = D = (5, 2) ...

1.10 SPODBUJEVALNO UCENJE - REINFORCEMENT LEARNING

Inteligentni agent se uci iz zaporedja nagrad in kazni

1.11 OCENJEVANJE UCENJA

k-fold, celo ucno mnozico razbij na k disjunktnih podmnozic za vsako od k podmnozic uporabi mnozico kot testno mnozico, preostalih k-1 mnozic kot ucno mnozico.

2 PREISKOVANJE

2.1 NEINFORMIRANI PREISKOVALNI ALGORITMI

2.1.1 ISKANJE V SIRINO

2.1.2 ISKANJE V GLOBINO

Izboljsave (Iskanje s sestopanjem, iterativno poglobljanje)

2.1.3 ITERATIVNO POGLOBLJANJE

problem globinsko omejenega iskanja -> nastavitve 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

2.1.4 DVOSMERNO ISKANJE

Pozenemo vzporedni iskanji od zacetka do cilja in od cilja do zacetka.

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 dvosmernem prostur ciljo vozlisce ce velja E=S (soda dolzina na isto mesto pridemo iz obeh strani) ali S->E (liha pot sosednja)

2.2 INFORMIRANI PREISKOVALNI ALGORITMI

Ideja: preiskovanje usmerjamo z dodatnim znanjem **hevrstiko** (ocenitvena funkcija za obetavnost vozlisca)

- **optimisticna/dopustna:** ∀ n : h(n) ≤ h*(n) (h* je optimalna ocena)

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

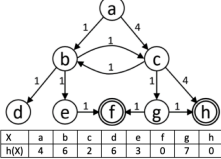
- **pesimisticna:** h(n) ≥ h*(n)

2.2.1 A*

A* is informed version of **dijkstra** (uses heuristics and pq), ce h(dopustna)=**popolna in optimalna**

Casovna zahtevnost odvisna od hevrstike: E = (h* - h)/h*, O(b^{E·d}), b-stopnja vejanja, d-globina optimalne resitve

Prostorska zahtevnost problem (hrani vsa vozlisca v spominu)



f(n) = g(n) + h(n), g(n) cena do vozlisca, h(n) hevrstika

Razvijamo dokler ne pridemo do ciljnega vozlišca

Razvijano	Generirana	Priority Queue
/	a(4)	[]
a	b(7) c(6)	[c(6),b(7)]
c	b'(11) g(12) h(8)	[b(7),h(8),b'(11),g(12)]
b	c'(4) d(8) e(5)	[c'(4),e(5),h(8),d(8),b'(11),g(12)]
...
f		

2.2.2 IDA* (ITERATIVE DEEPENING A*)

f(n) = g(n) + h(n), g(n)=cena poti do n

Meja	Razvijano	Generirana	DFS (list)
0	/	s(7)	/
7	/	s(7)	s
	s	a(8) b(7) c(7)	b, c
	b	f(6) h(5)	f h c
	f	g(7) h(9) i(11)	g h c
	g		

2.2.3 KAKOVOST HEVRISTICNIH FUNKCIJ

Kakovost h ocenimo z številom generiranih vozlišc ter učinkovitim faktorjem vejanja (N vozlišc je algoritem generiral da je na globini d našel rešitev) Hocemo imeti dopustne hevristike s čim višjimi vrednostmi in sprejemljivo ceno (casom izracuna)

Ce $h_2(n) \geq h_1(n)$, $\forall n$ potem h_2 dominira h_1

2.3 PREISKOVANJE GRAFOV AND/OR, NEDETERMINISTICNO OKOLJE

Pomagajo reševati probleme z dekompozicijo na manjše probleme Uporabnost:

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

2.3.1 AO*

- posplošitev 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) dinamična hevristična ocena Postopek:

1. Razvij najcenejše vozlišče
 - ce list in končno (oznaci), preveri 3. korak, nadaljuj v 1.
 - ce list in ni končno (oznaci) vrednost vozlišča = ∞

2. Posodobi vse predhodnike

- v AND starih, cena starsa = \sum sinov + povezava v
- v OR starih, cena starsa = min(sinovi) + povezava v

3. Končaj ko obstaja pot do zacetnega vozlišca, po kateri v AND vozlišcih po vseh sinovih prides do cilja, v OR vozlišcih v vsaj enem

2.3.2 ALGORITEM MINIMAX

- m globina - b

2.3.3 REZANJE ALFA-BETA

3 PLANIRANJE

plan zaporedje akcij, ki pripelje od zacetnega do koncnega stanja

3.1 PLANIRANJE S SREDSTVI IN CILJI (STRIPS)

Agentu opisemo svet in postavimo fizikalne omejitve.

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

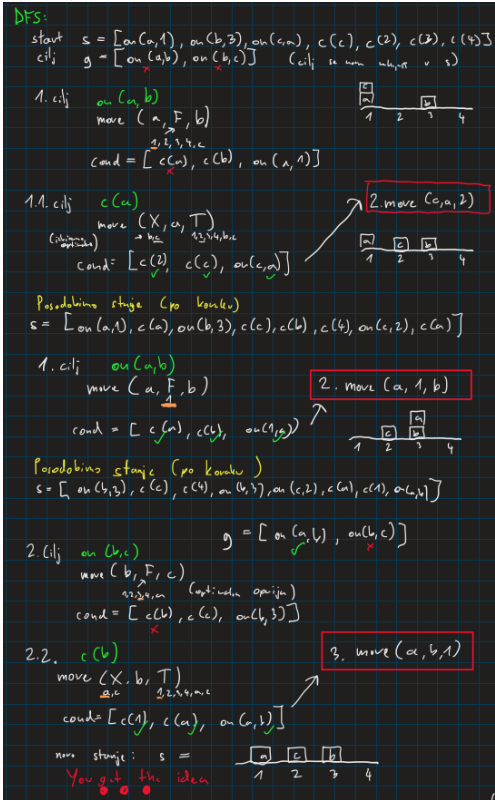
Akcija move(X, From, To)

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

Algoritem:

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

Primer dfa, zlaganje kock



3.2 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. $G_{i+1} = [\text{on}(b,c), \dots, c(c, \dots)]$)

\rightarrow zactno_stanje = $[\text{on}(a,1), \text{on}(b,a), c(b), \text{on}(c,3), c(c)]$

\rightarrow hocemo da zacetno_stanje $\subset G_i$

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

- **POGOJ**: $G_2 \cap \text{dels}(A_2) = \emptyset$ ✓
 - $G_2 = [\text{on}(b,c), c(a), c(b), \text{on}(a,1), c(c), c(b), \text{on}(b,3)]$ ✓
3. $G_2 = \dots$

3.3 RAZPOREJANJE OPRAVIL (PDDL)

Razsirimo lahko notacijo (PDDL):

Akcija1 < Akcija2: Akcija1 se mora zgoditi pred Akcija2

Resources podajo stevila razpoložljivih resursov

DURATION opredeljuje trajanje posamezne akcije

CONSUME opredeljuje (trajno) porabo določene količine resursov

USE opredeljuje (zacasno) zasedenost količine resursov med izvajanjem akcije

```
Jobs (AddEngine1 < AddWheels1 < Inspect1,
      AddEngine2 < AddWheels2 < Inspect2 )
Resources (EngineHoists(1), WheelStations(1), Inspectors(2), LugNuts(500))

Action (AddEngine1, DURATION:30,
        USE:EngineHoists(1))
Action (AddEngine2, DURATION:60,
        USE:EngineHoists(1))
Action (AddWheels1, DURATION:30,
        CONSUME:LugNuts(20), USE:WheelStations(1))
Action (AddWheels2, DURATION:15,
        CONSUME:LugNuts(20), USE:WheelStations(1))
Action (Inspect 1, DURATION:10,
        USE:Inspectors (1))
```

Metoda kritične poti

kritična pot: pot, ki je najdaljša in določa dolžino trajanja celotnega plana vsaki akciji priredimo par [ES, LS]

- **ES**: najbolj zgodnji možen začetek (Earliest Start)
 - $ES(start) = 0, ES(B) = \max_{A < B} [ES(A) + Duration(A)]$
- **LS**: najbolj pozni možen začetek (Latest Start)
 - $LS(Finish) = ES(Finish), LS(A) = \min_{A < B} [LS(B) - Duration(A)]$

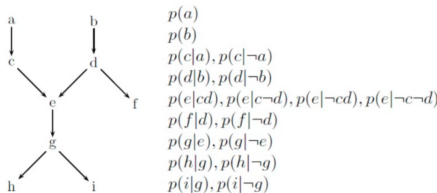
rezerva(slack)=LS-ES (**casovna rezerva**) **Algoritem** po hevristiki **minimum slack** \rightarrow na vsaki iteraciji ima prednost akcija ki ima izpolnjene predhodnike in najnižji slack, nato posodobi [ES in LS] za celotni graf in ponovi.

4 SKLEPANJE

4.1 BAYESOVSKE MREZE

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

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

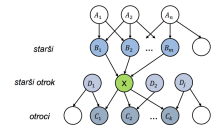


Pravila verjetnostnega sklepanja:

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) ce X **nima** staršev: $P(X | C) = P(X)$, P(X) je podan
 - (b) ce **ima** X starše S: $P(X | C) = \sum_{S \in P_X} P(X | S) P(S | C)$
7. Iz 6b zgoraj: $P(i | g) = P(i | g)$

4.2 OVOJNICA MARKOVA

X je neodvisen od vseh ostalih \Leftrightarrow podani **starsi, otroci in starsi otrok**



4.3 D-LOCEVANJE

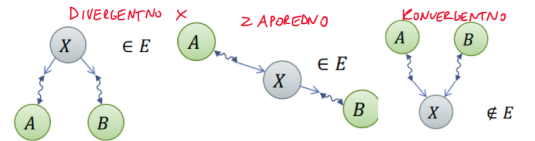
A in B v mreži sta **neodvisni** \Leftrightarrow obstaja množica vozlišc E, ki d-ločuje A in B.

potem sledi: $(P(A|E) = P(B|E))$

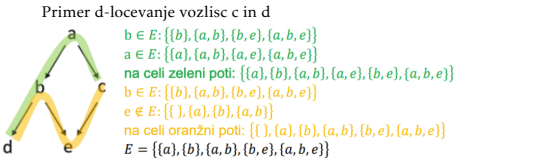
za **vsako neusmerjeno pot P** med A in B v bayesovski mreži:

za **vsako vozlišče X** na poti P:
• **analiziraj pogoj** za pripadnost X množici E glede na tip:
• **divergentno** ali **zaporedno** vozlišče: $X \in E$
• **konvergentno** vozlišče: X in nasledniki $\notin E$

S_x = množice vozlišč, ki ustrezajo pogoju za X
 $S_p = U_x(S_x)$ // množice, ki d-ločujejo samo na poti P (unija množic za vozlišča na poti)
 $E = \cap_p S_p$ // množice, ki d-ločujejo v celi mreži (presek množic za vse možne poti)



! pri konvergentnem izlocimo tudi vse naslednike X



$\rightarrow P(d|a) = P(d|a), P(d|cb) = P(d|b), P(d|cab) = P(d|ab),$
 $P(d|cbe) = P(d|be), P(d|cabe) = P(d|abe)$