Asocijativna analiza

Šta je asocijativna analiza?

- Asocijativna analiza sastoji se u identifikovanju jakih asocijativnih pravila u datom skupu podataka
  - Brojne su varijante osnovnog problema
- Originalna primjena: analiza BP iz oblasti trgovine
- Asocijativno pravilo je implikacija oblika Body $\rightarrow$ Head [support, confidence]
  - buys(x, “diapers”) $\rightarrow$ buys(x, “beers”) [0.5%, 60%]
  - major(x, “CS”) ^ takes(x, “DB”) $\rightarrow$ grade(x, “A”) [1%, 75%]

Asocijativna analiza, definicije

- Neka je I konačan skup
  - Elementi skupa I se nazivaju itemi
  - Ma koji podskup K od I naziva se itemset
  - Itemset sa k elemenata je k-itemset
- Transakciona baza podataka za skup I je funkcija $T: \{1, ..., n\} \rightarrow P(I)$, gdje je $P(I)$ partitivni skup za I.
- Pravila povezuju primjerke na način da prisustvo jednih implicira prisustvo drugih

Asocijativna analiza, definicije 2

- Frequent (large) itemset
- Support count σ za itemset K je broj transakcija koje sadrže K
  - npr. σ({Milk, Bread, Diaper})=2
- Supprt s za itemset K je s = σ / n, n je ukupan broj transakcija
  - npr. s({Milk, Bread, Diaper})=2/5
- Itemset K je frequent ako je s(K)≥minsup

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Asocijativna analiza, definicije 3

- Asocijativno pravilo je implikacija oblika $X \rightarrow Y$, gdje su X i Y itemsetovi i $X \cap Y = \emptyset$
- Support za $X \rightarrow Y$, $s(X \rightarrow Y) = s(X + Y)$
- Confidence za $X \rightarrow Y$, $c(X \rightarrow Y) = s(X + Y) / s(X)$

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

$$\{\text{Milk}, \text{Diaper}\} \Rightarrow \text{Beer}$$

$$s = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{|\text{T}|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{\sigma(\text{Milk}, \text{Diaper})} = \frac{2}{3} = 0.67$$

Zadatak asocijativne analize

- Za dati skup transakcija T, zadataka je naći jaka asocijativna pravila, tj. pravila koja zadovoljavaju
  - support ≥ minsup
  - confidence ≥ minconf
- Algoritam grube sile
  - Generisati sva moguća pravila
  - Izračunati support i confidence
  - Eliminisati pravila sa support < minsup i confidence < mincon
  - Složenost: za d itema ukupan broj pravila je $3^d-2^{d+1}+1$

Strategija za asocijativnu analizu

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Pravila:

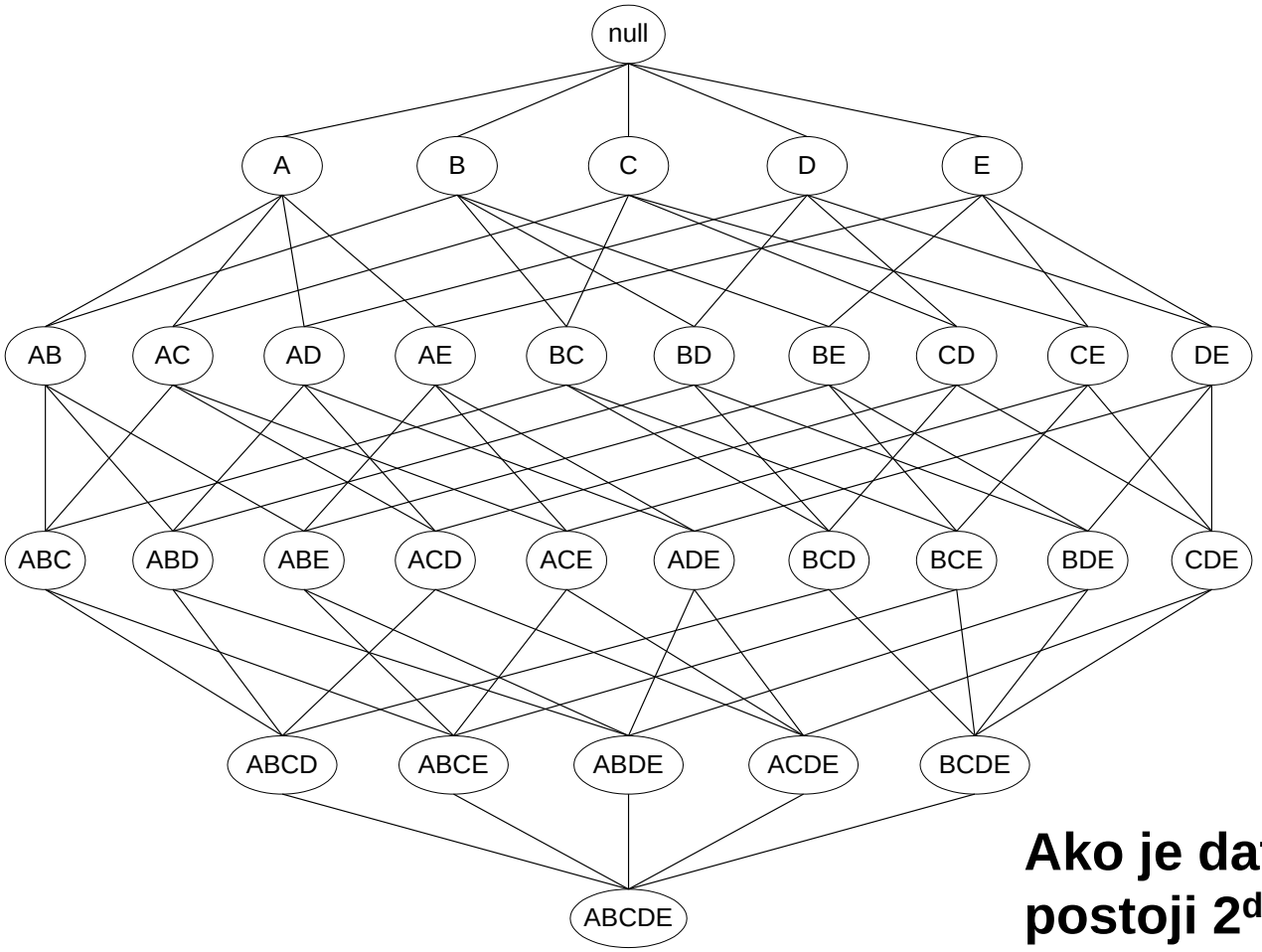
{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

- Sva pravila nastala su iz itemseta {Milk, Bread, Diaper} i imaju jednaku vrijednost za support
- Razdvojiti uslove za minsup i monconf

Strategija za asocijativnu analizu 2

- Problem se dijeli na dva
  - Generisanje frequent itemsetova
  - Generisanje pravila
    - Svako pravilo je particija frequent itemseta
- Drugi korak ne utiče na ukupne performanse algoritama, pa je glavni zadatak generisanje frequent itemsetova

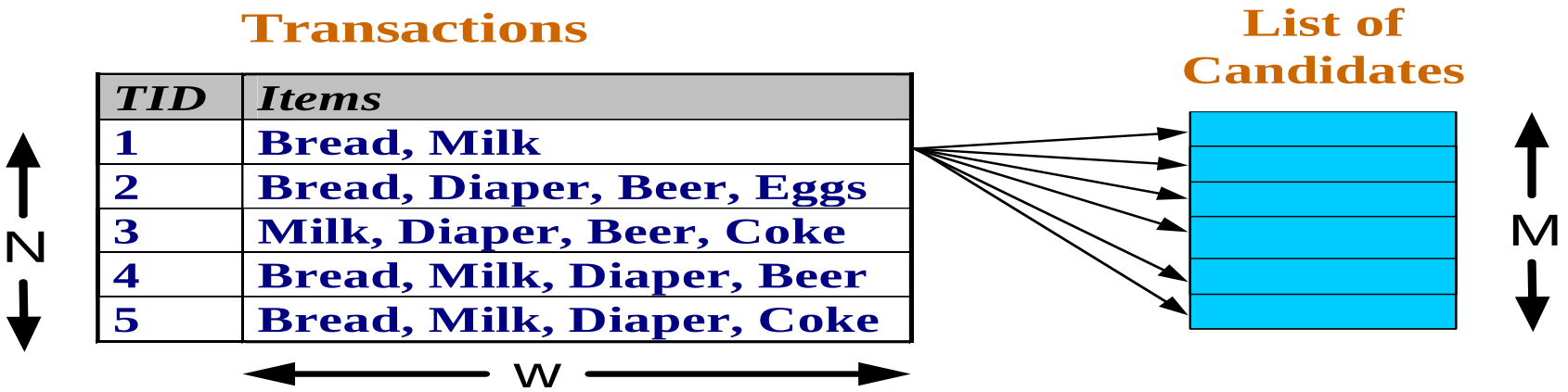
Generisanje frequent itemsetova



Ako je dato d itema, postoji 2^d mogućih frequent itemsetova.

Generisanje frequent itemsetova 2

- Algoritam grube sile
  - Svaki itemset je kandidat za frequent itemset
  - Računanje podrške za svaki itemset



  - Upoređuje svaku transakciju sa svakim itemsetom, složenost O(NMw) = eksponencijalna jer je $M = 2^d - 1$

Strategije za generisanje frequent itemsetova

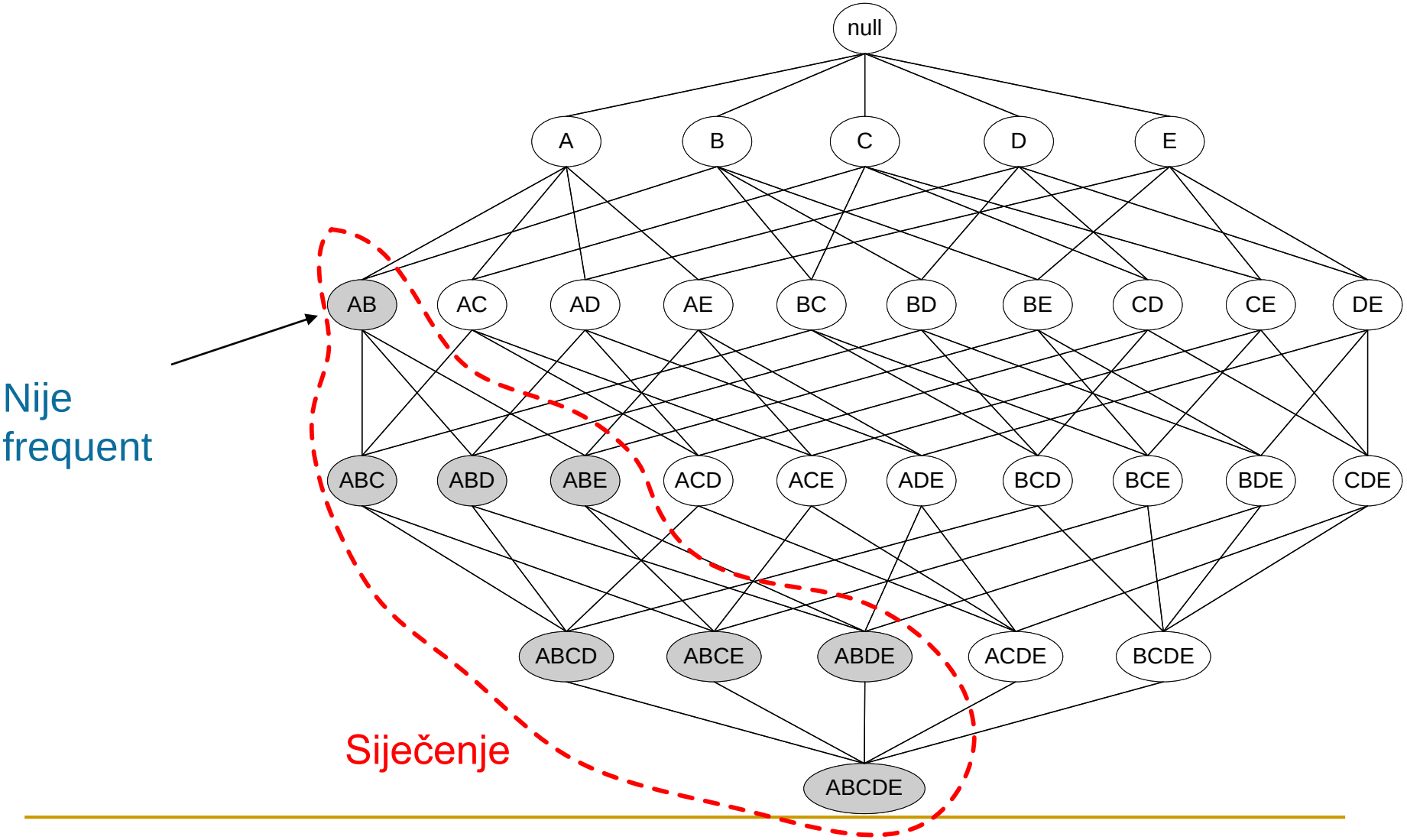
- Smanjenje broja kandidata M
- Smanjenje broja transakcija N
- Smanjenje broja poređenja upotrebom specijalnih struktura podataka za čuvanje kandidata i/ili transakcija, tako da nije potrebno upoređivati svaku transakciju sa svakim itemsetom

Smanjenje broja kandidata

- Apriori princip
  - Ako je za neki itemset support > minsup, tada i za svaki njegov podskup važi support > minsup
- Apriori princip je zasnovan na svojstvu antimonotonosti za support mjeru

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

Apriori princip, primjer



Apriori princip, primjer 2

Item	Count
Bread	**4**
Coke	**2**
Milk	**4**
Beer	**3**
Diaper	**4**
Eggs	**1**

1-itemsets

Itemset	Count
{Bread,Milk}	**3**
{Bread,Beer}	**2**
{Bread,Diaper}	**3**
{Milk,Beer}	**2**
{Milk,Diaper}	**3**
{Beer,Diaper}	**3**

2-itemsetovi

(ne generišemo kandidate koji sadrže Coke i Eggs)

Minimum Support = 3

3-itemsetovi

Itemset	Count
{Bread,Milk,Diaper}	**3**

Bez Apriori principa broj kandiadta je:

$${}^6C_1 + {}^6C_2 + {}^6C_3 = 41$$

Sa Apriori principom:

$$6 + 6 + 1 = 13$$

Apriori algoritam

- Algoritam

C_k: svi kandidatski k-itemsetovi
F_k : svi frequent k-itemsetovi

1. k=1
2. Generiši frequent 1-itemsetove
3. WHILE L_k !=∅
   1. C_{k+1} se generišu iz L_k
   2. Siječenje kandidata koji sadrže podskup od k elemenata koji nije u L_k
   3. Čitanje baze podataka i određivanje podrške za preostale kandidate
   4. L_{k+1} = kadidati iz C_{k+1} sa podrškom većom od minsup

Generisanje kandidata

- L_{k-1} je uređen skup, spajanje $C_k = L_{k-1} \times L_{k-1}$

  insert into $\boldsymbol{C_k}$

  select $\boldsymbol{p.item_1, p.item_2, \ldots, p.item_{k-1}, q.item_{k-1}}$

  from $\boldsymbol{L_{k-1}\ p, L_{k-1}\ q}$

  where $\boldsymbol{p.item_1 = q.item_1, \ldots, p.item_{k-2} = q.item_{k-2},}$

  $\boldsymbol{p.item_{k-1} < q.item_{k-1}}$

Generisanje kandidata 2

- siječenje po Apriori principu

  forall ***itemsets c in*** C_k do

    forall ***(k-1)-subsets s of c*** do

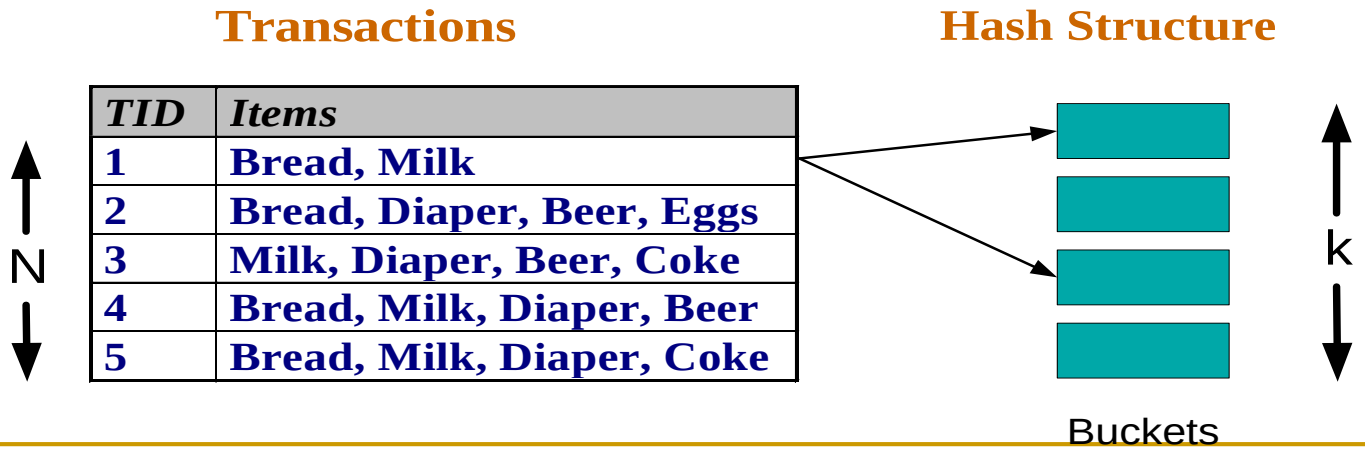
      if *(s is not in* L_{k-1}*)* **then delete** *c* **from** C_k

Generisanje kandidata 3

- L_3={*abc, abd, acd, ace, bcd*}
- Spajanje $L_3 x L_3$
  - *abcd* from *abc* and *abd*
  - *acde* from *acd* and *ace*
- Siječenje:
  - *acde* se briše jer *ade* nije u L_3
- C_4={*abcd*}

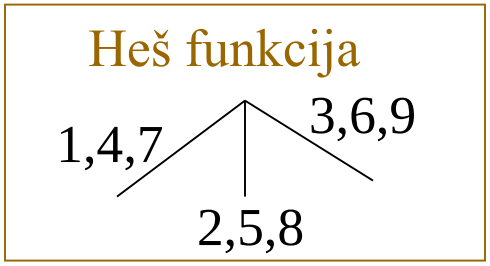
Smanjenje broja poređenja

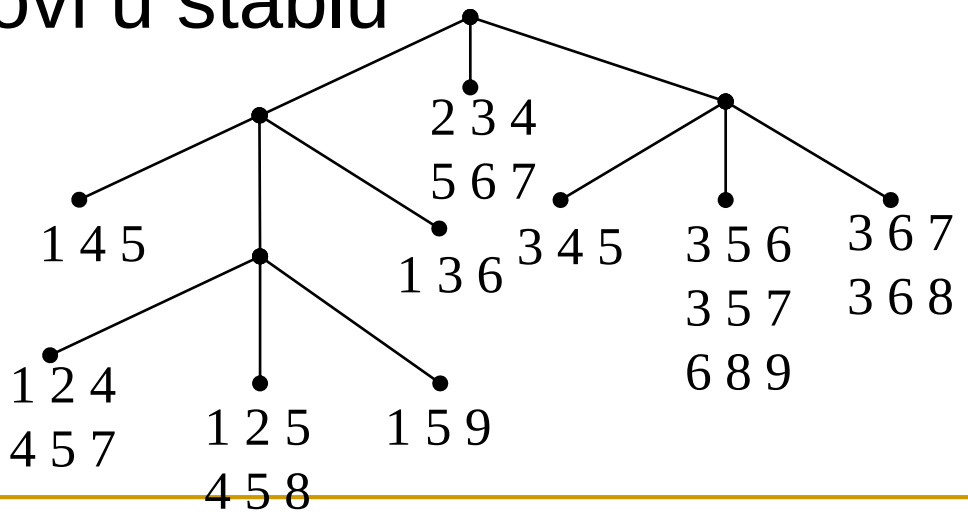
- Računanje podrške za kandidate
  - Čitanje baze podataka i određivanje podrške za svakog kandidata
  - Za smanjenje broja poređenja, kandidati se čuvaju u heš stablu; transakcija se ne poredi sa svakim kandidatom, već samo sa onima u odgovarajućim baketima



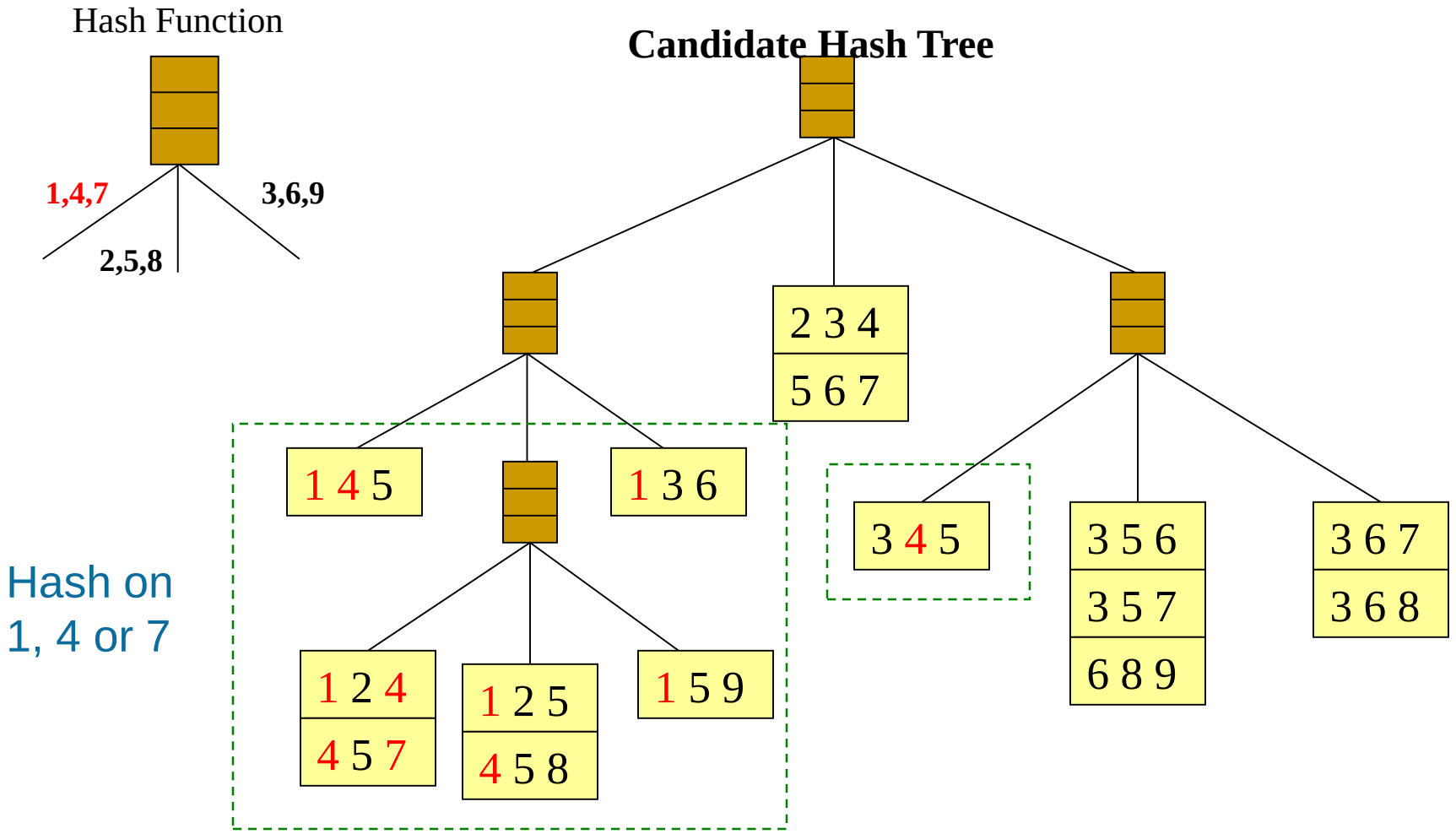
Kreiranje heš stabla

- Neka je C_3 = **{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}**
- Potrebno je definisati heš funkciju i kapacitet baketa koji su listovi u stablu

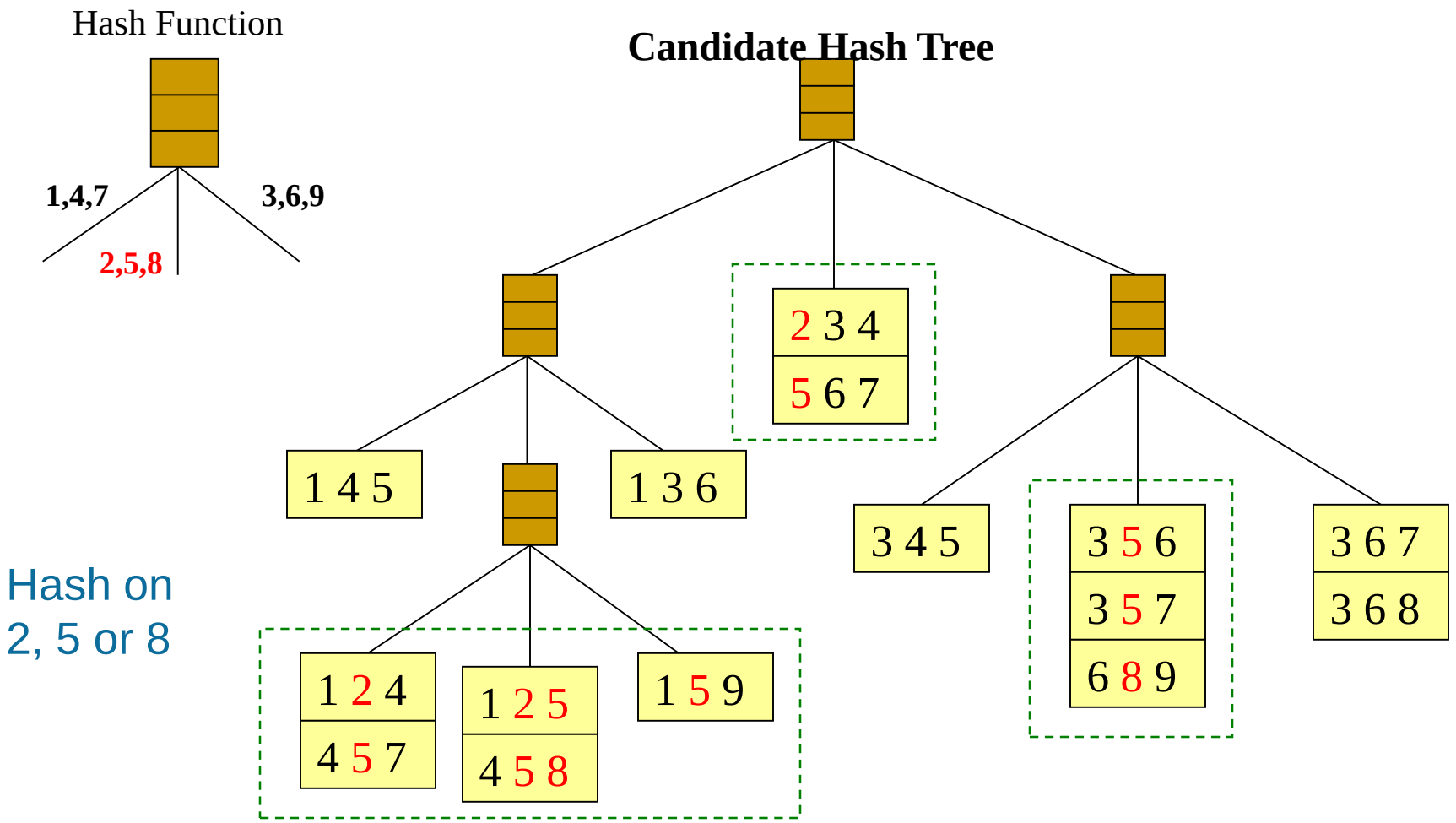




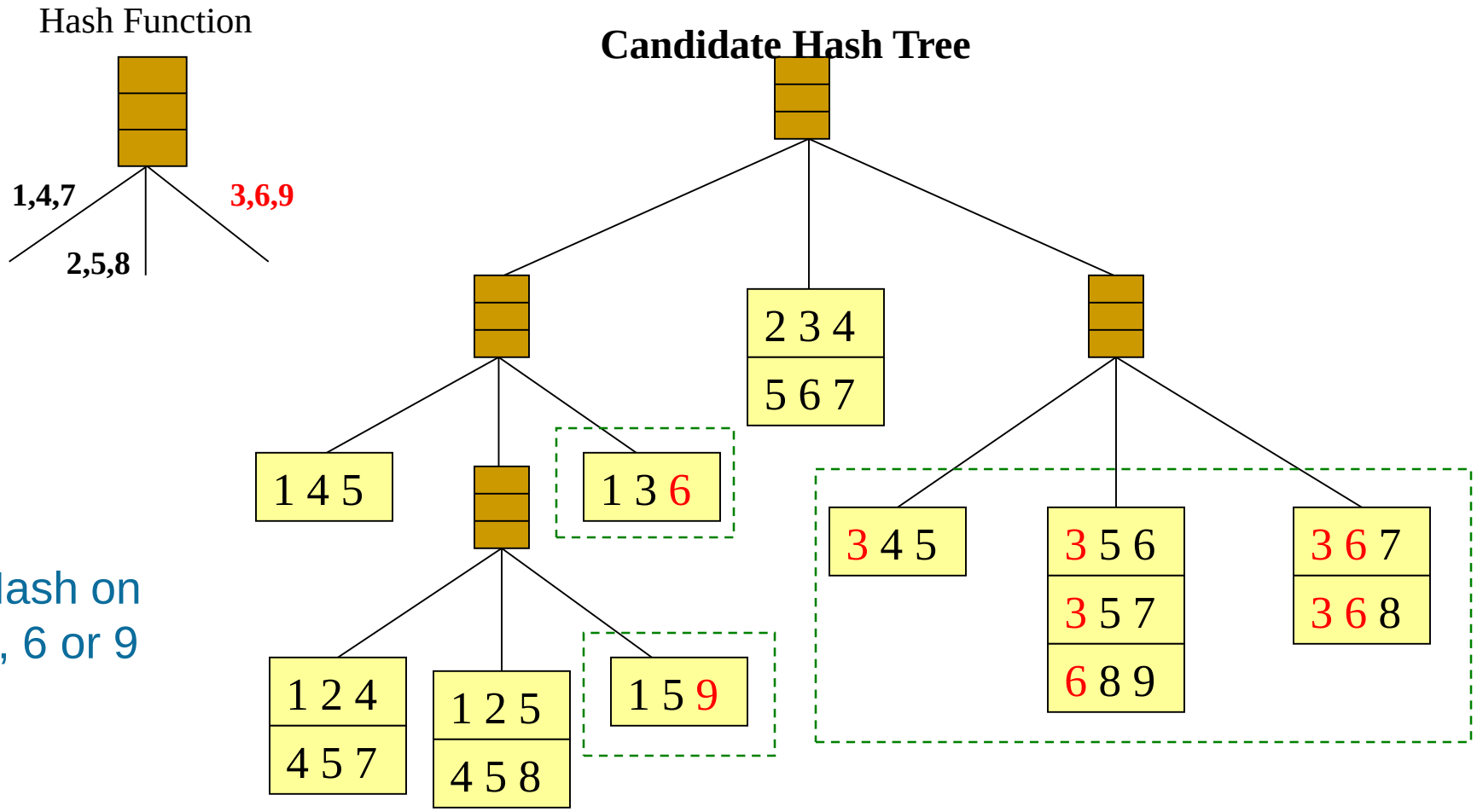
Kreiranje heš stabla 2



Kreiranje heš stabla 3

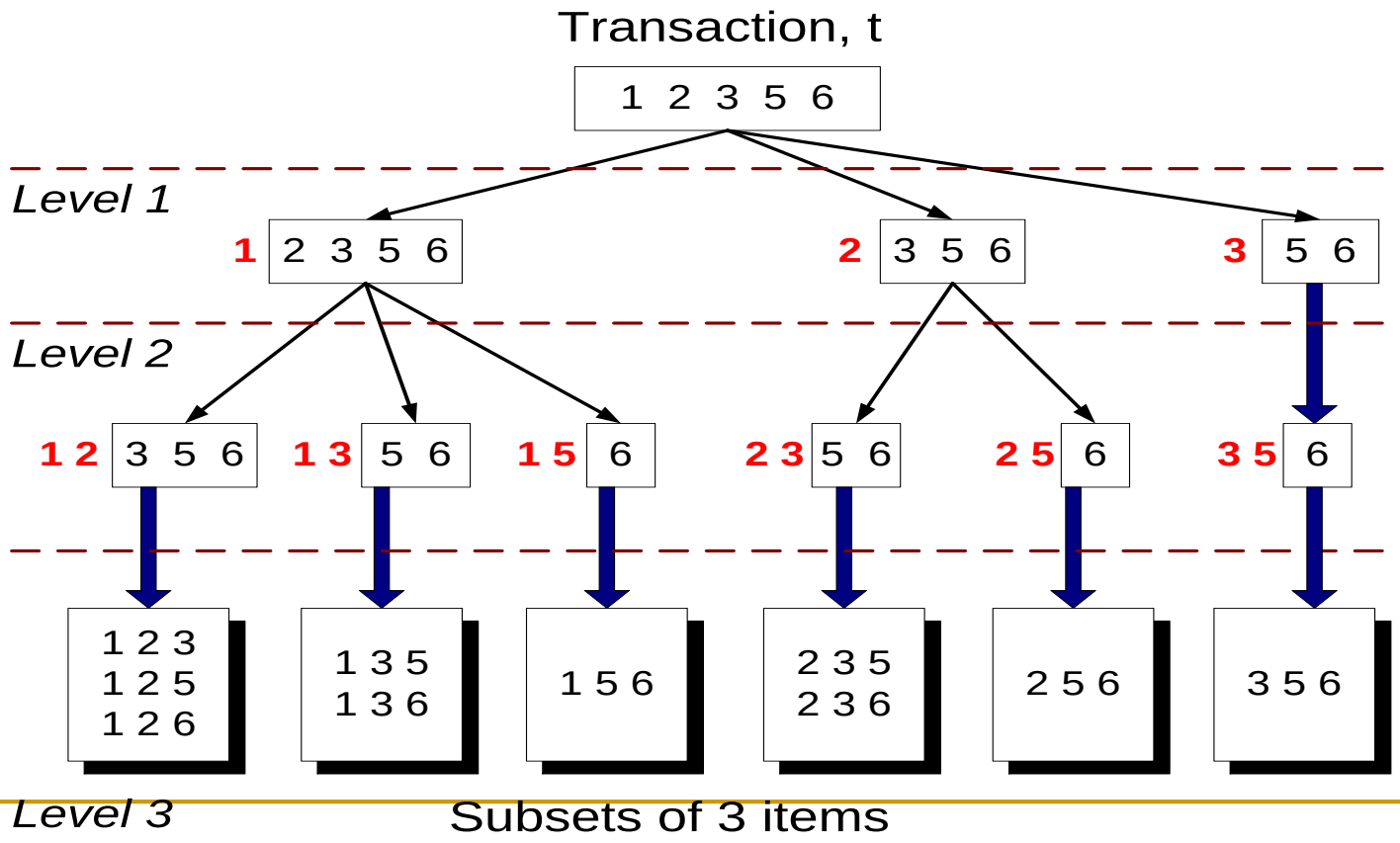


Kreiranje heš stabla 4

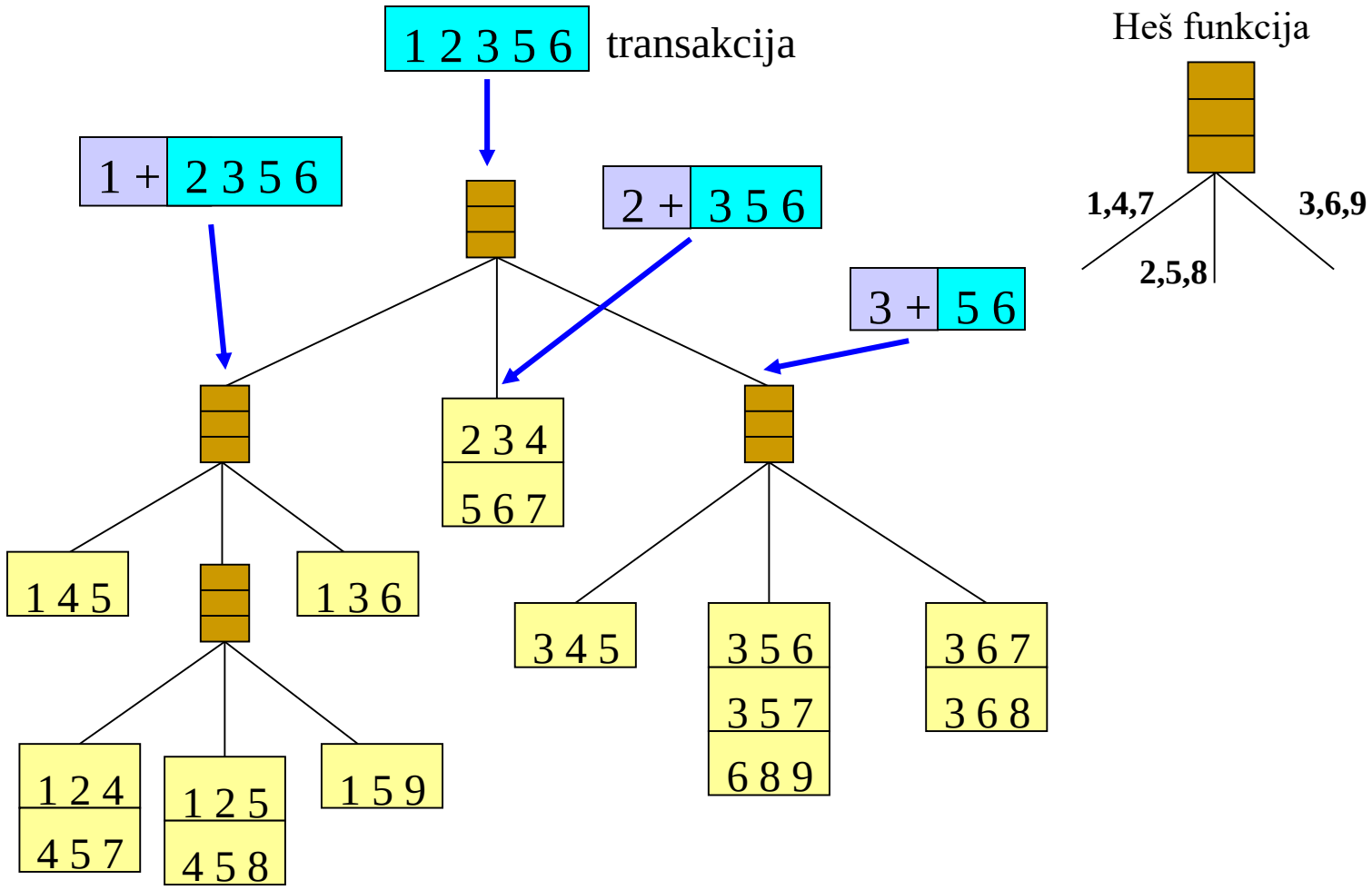


Mapiranje transakcija na heš stablo

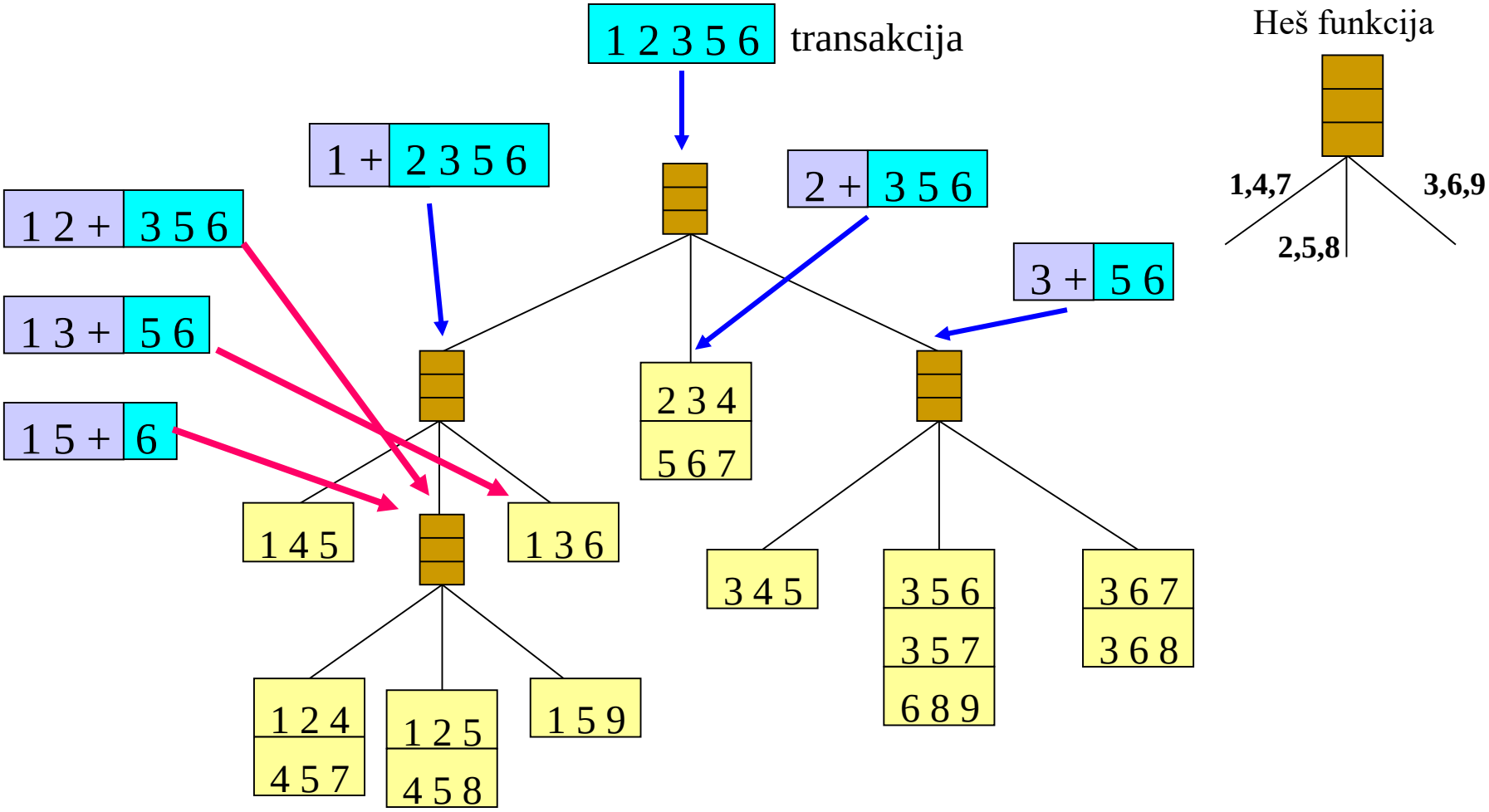
- Koji su 3-kandidati sadržani u transakciji t?



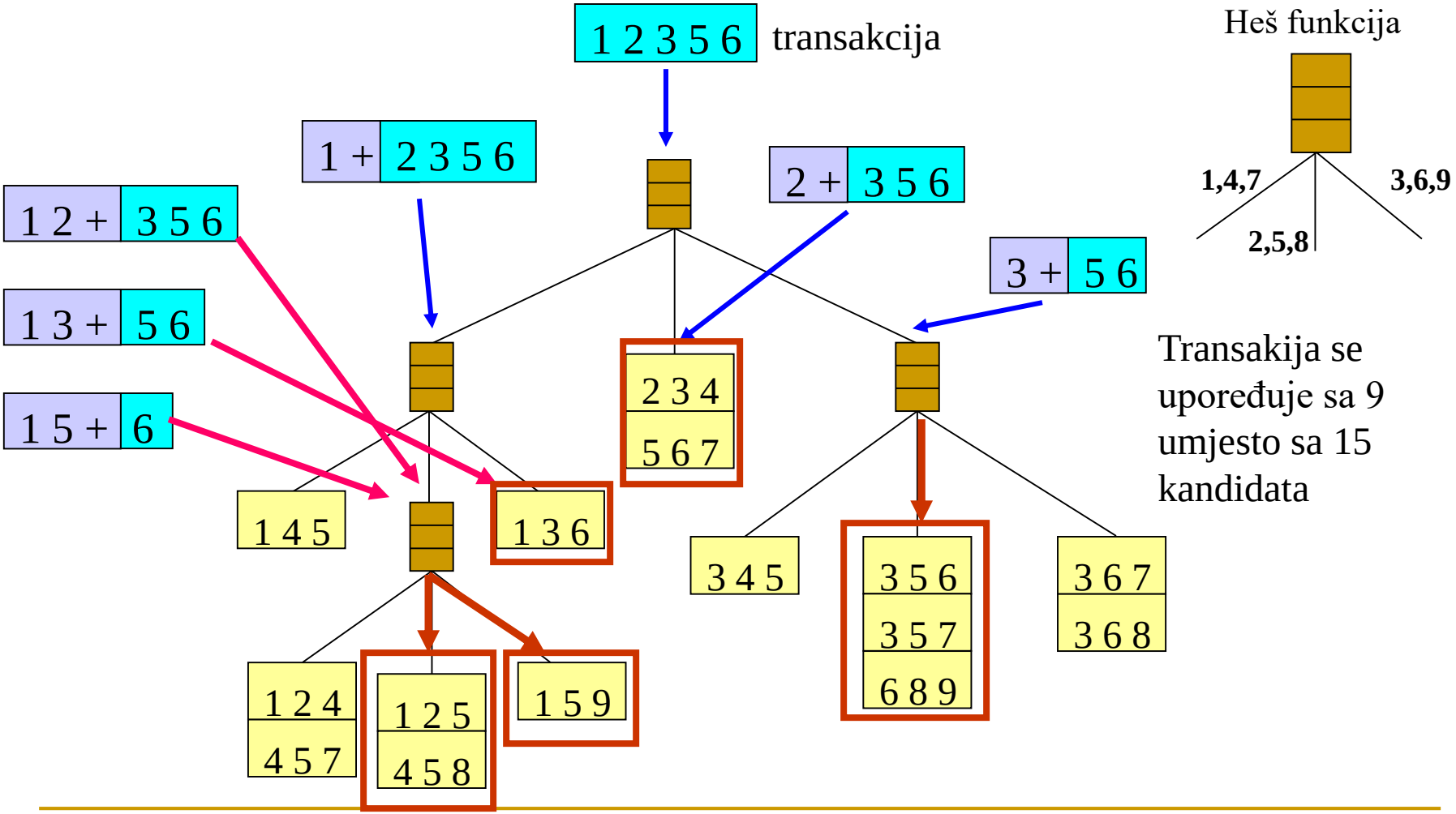
Mapiranje transakcija na heš stablo 1



Mapiranje transakcija na heš stablo 2



Mapiranje transakcija na heš stablo 3



Glavni faktori koji utiču na složenost

- Vrijednost za minsup parametar
  - Manja vrijednost za minsup utiče na generisanje većeg broja kandidata
- Broj itema (dimenzionalnost baze podataka)
- Veličina baze podataka
  - U jednoj iteraciji jedno čitanje baze podataka
- Prosječna dužina transakcija
  - Utiče na cijenu obilaska heš stabla

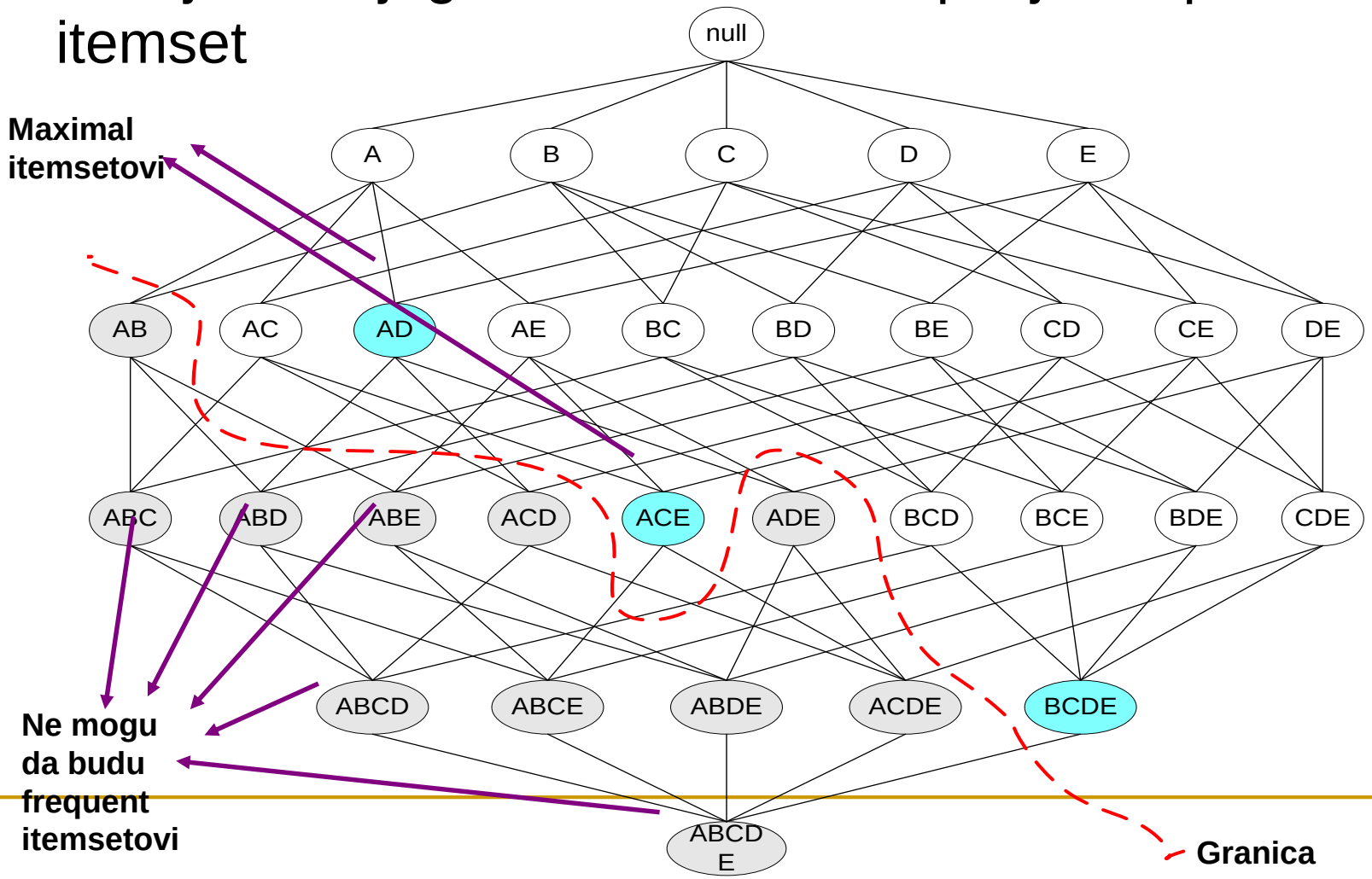
Poboljšanja

$$\text{Ukupan broj frequent itemsetova} = 3 \times \sum_{k=1}^{10} \binom{10}{k}$$

TID	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

Maximal frequent itemsets

- Maximal frequent itemset je frequent itemset takav da nijedan njegov direktni nadskup nije frequent itemset



Closed itemsetovi

- Itemset X je closed ako nijedan od njegovih direktnih nadksupova ima jednaku podršku kao X

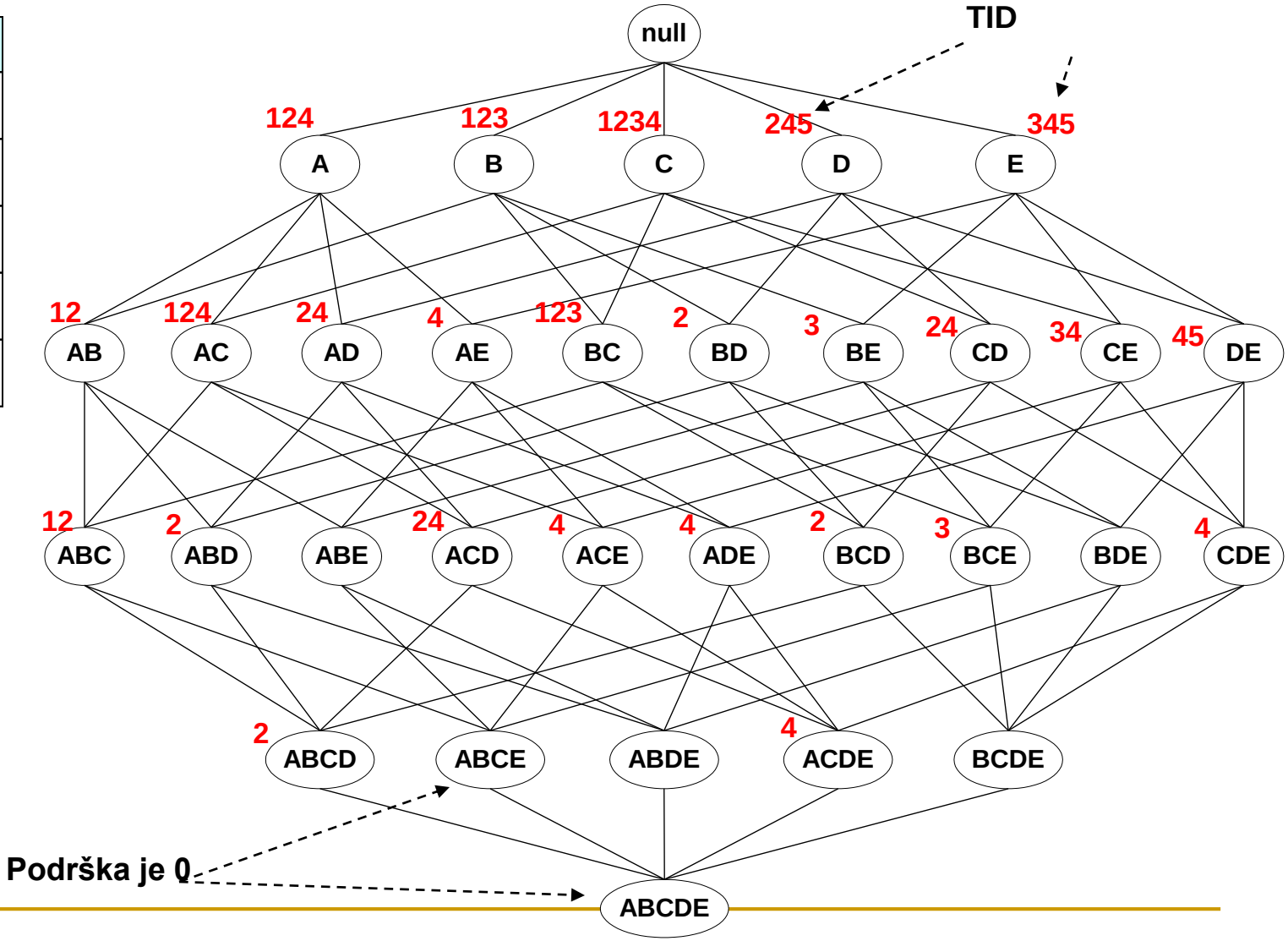
TID	Items
1	{A,B}
2	{B,C,D}
3	{A,B,C,D}
4	{A,B,D}
5	{A,B,C,D}

Itemset	Support
{A}	4
{B}	5
{C}	3
{D}	4
{A,B}	4
{A,C}	2
{A,D}	3
{B,C}	3
{B,D}	4
{C,D}	3

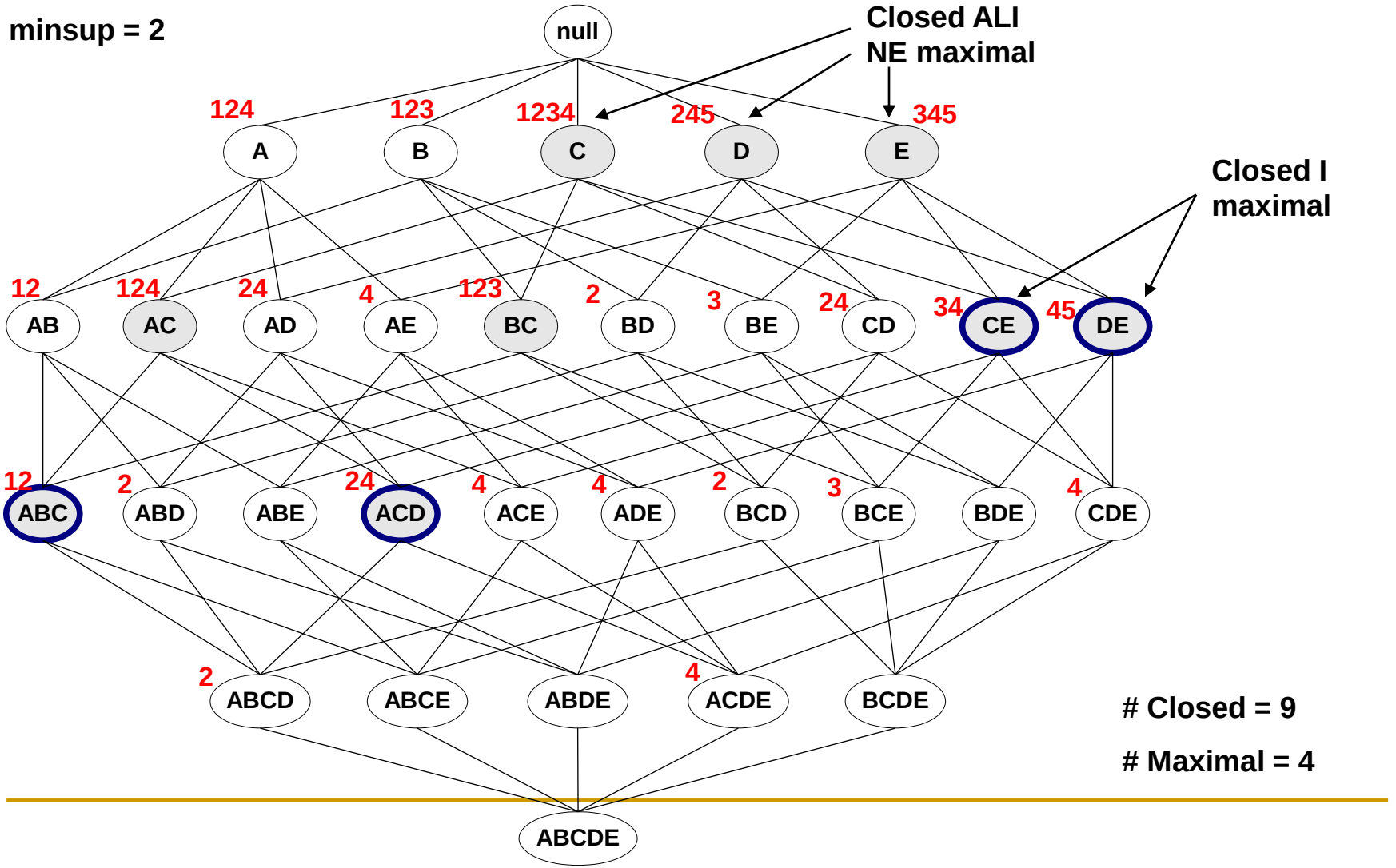
Itemset	Support
{A,B,C}	2
{A,B,D}	3
{A,C,D}	2
{B,C,D}	3
{A,B,C,D}	2

Maximal vs Closed itemsets

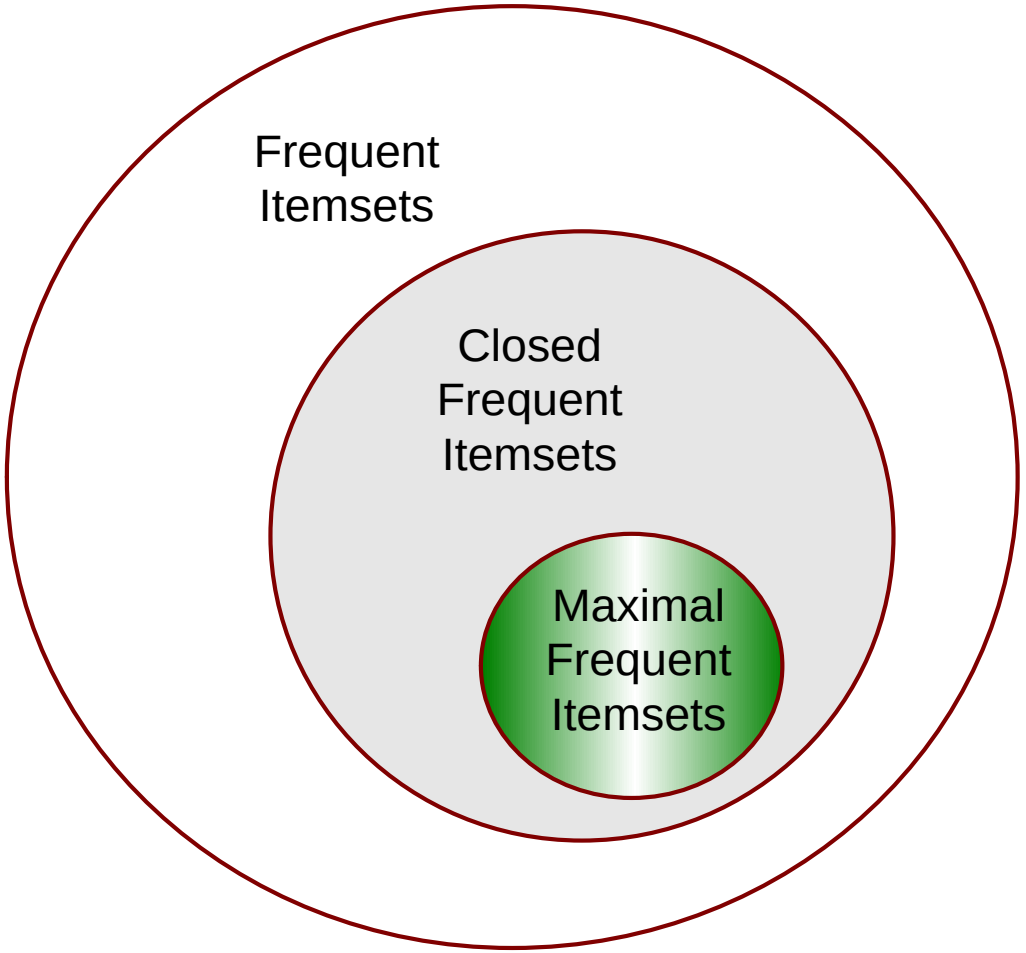
TID	Items
1	ABC
2	ABCD
3	BCE
4	ACDE
5	DE



Maximal vs Closed itemsets 2



Maximal vs Closed itemsets 3



Predstavljanje baze transakcija

Horizontal Data Layout

TID	Items
1	A,B,E
2	B,C,D
3	C,E
4	A,C,D
5	A,B,C,D
6	A,E
7	A,B
8	A,B,C
9	A,C,D
10	B

Vertical Data Layout

A	B	C	D	E
1	1	2	2	1
4	2	3	4	3
5	5	4	5	6
6	7	8	9	
7	8	9		
8	10			
9				

FP-Growth algoritam

- Koristi se specijalna struktura FP-stablo za predstavljanje baze transakcija u operativnoj memoriji
- FP-stablo se kreira sa dva čitanja baze transakcija
- Generisanje frequent itemsetova direktno iz FP stabla, bez višestrukih čitanja baze transakcija i bez generisanja kandidata

Kreiranje FP stabla

TID	*Itemi*	*poređani po podršci*
100	{*f, a, c, d, g, i, m, p*}	{*f, c, a, m, p*}
200	{*a, b, c, f, l, m, o*}	{*f, c, a, b, m*}
300	{*b, f, h, j, o*}	{*f, b*}
400	{*b, c, k, s, p*}	{*c, b, p*}
500	{*a, f, c, e, l, p, m, n*}	{*f, c, a, m, p*}

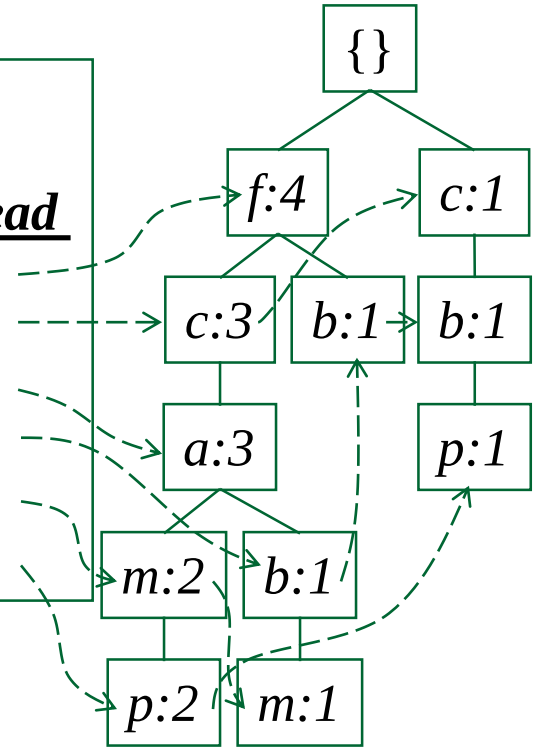
min_support = 0.5

Koraci:

1. Čitanje BP i pronalaženje velikih 1-itemsetova
2. Urediti velike 1-itemsetove u opadajućem poretku
3. Još jedno čitanje BP i kreiranje FP stabla

Heder tabela

Item	*Podrška*	*head*
f	*4*	
c	*4*	
a	*3*	
b	*3*	
m	*3*	
p	*3*	



Prednosti pattern growth pristupa

- Kompresija baze podataka i do 100 puta
- Opadajući redosled jer su itemi sa većom podrškom zajednički za mnoge transakcije
- U prvom koraku se eliminišu itemi koji nijesu veliki
- Algoritam je kompletan

Generisanje velikih itemsetova sa FP stablom

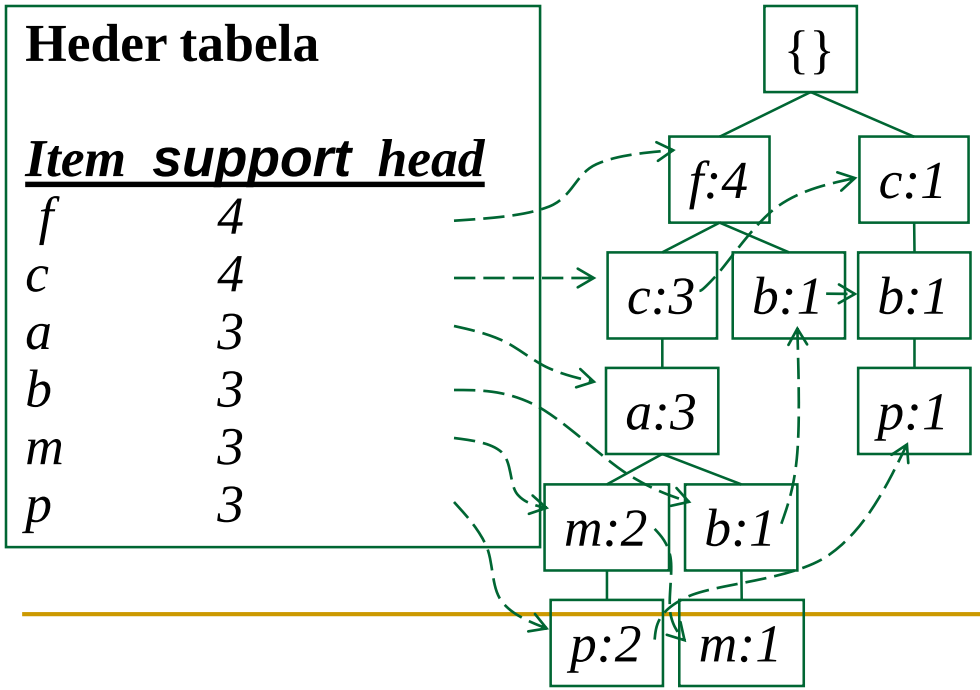
- Opšta ideja: rekurzivni obilazak stabla i generisanje velikih skupova pomoću puteva u stablu
- Algoritam
  - Za svaki item se posmatraju prefiks putevi i od njih se konstruiše uslovno FP stablo
  - Postupak se ponavlja za svako kreirano uslovno FP stablo
  - Postupak se završava kada FP stablo sadrži samo jednu putanju

Rudarenje FP stabla

- Za svaki frequent item se izdvoje prefiks putevi
- Od prefiks puteva se kreira uslovno FP stablo
- Rekurzivni poziv za uslovno stablo
  - Ako stablo sadrži samo jednu putanju dovoljno je navesti sve itemsetove koji su sadržani u toj putanji

Korak 1: prefiks putevi u FP stablu

- Heder tabela sadrži sve velike iteme sortirane u opadajućem poretku u odnosu na podršku
- Pokazivači iz heder tabele ukazuju na odgovarajuće čvorove u stablu i početna su tačka obilaska
- Prefiks putevi sa item i_j su svi putevi koji završavaju sa i_j



item	prefiks putevi
c	**f:3**
a	**fc:3**
b	**fca:1, f:1, c:1**
m	**fca:2, fcab:1**
p	**fcam:2, cb:1**

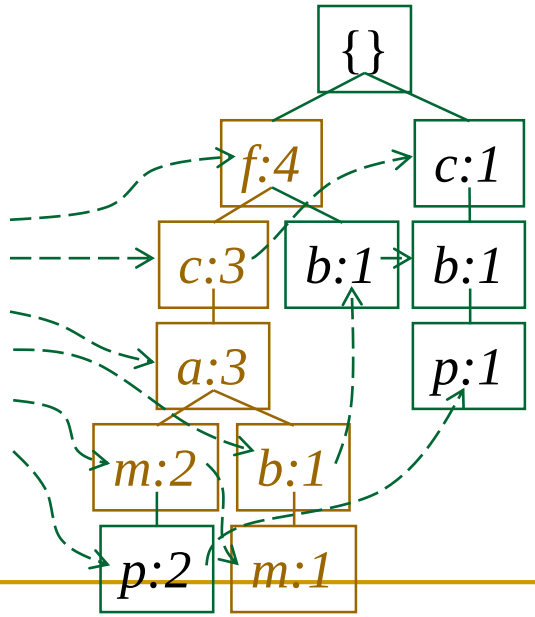
Svojstva prefiks puteva

- Za svaki item i_j, svi frequent itemsetovi koji sadrže i_j su samo oni do kojih se može doći preko pokazivača iz heder tabele
- Podrška prefiks puteva je podrška za i_j
- Uslovno FP stablo je strukturno isto kao i FP stablo, ali je dobijeno transformacijom prefiks puteva

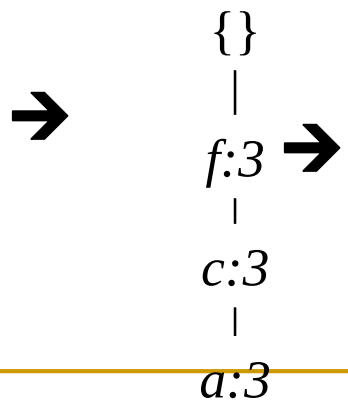
Korak 2: uslovno FP-stablo

- Za sve prefiks puteve
  - Izračunati support svakog itema u prefiks putu
  - Konstruiši FP stablo (uslovno)

Header Table	
Item	***support***
f	*4*
c	*4*
a	*3*
b	*3*
m	*3*
p	*3*



Prefiks putevi za m:
fca:2, fcab:1



Uslovno FP stablo za m

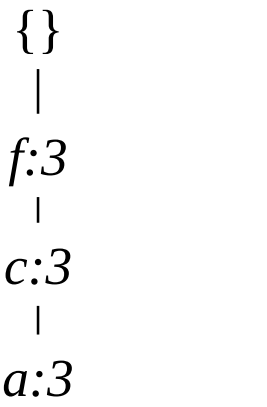
Frequent itemsetovi koji sadrže *m*

m,
fm, cm, am,
fcm, fam, cam,
fcam

Uslovno FP-stablo 2

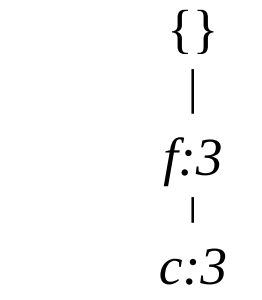
Item	Conditional pattern-base	Conditional FP-tree
p	{(fcam:2), (cb:1)}	{(c:3)}|p
m	{(fca:2), (fcab:1)}	{(f:3, c:3, a:3)}|m
b	{(fca:1), (f:1), (c:1)}	Empty
a	{(fc:3)}	{(f:3, c:3)}|a
c	{(f:3)}	{(f:3)}|c
f	Empty	Empty

Rekurzivno rudarenje uslovnog FP-stabla



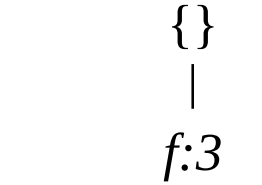
Uslovno FP-stablo za m

Prefiks putevi za “am”: (fc:3)



Uslovno FP-stablo za am

Prefiks putevi za “cm”: (f:3)



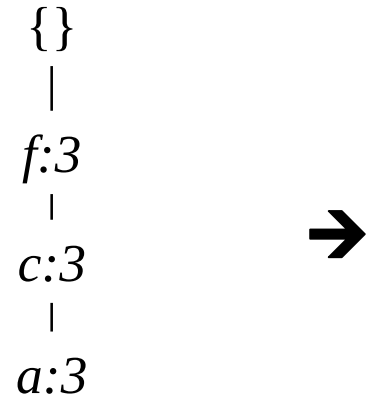
Uslovno FP-stablo za cm

Prefiks putevi za “cam”: (f:3)

Uslovno FP-stablo za cam

Rekurzivno rudarenje uslovnog FP-stabla 2

- Ako je uslovno FP-stablo sastavljeno od jedne putanje P, tada se svi frequent itemsetovi mogu dobiti kao podskupovi putanje P



➔

Svi frequent itemsetovi koji sadrže m

m,

fm, cm, am,

fcm, fam, cam,

fcam

Uslovno FP-stablo za M

Osnovni princip za FP-Growth algoritam

- Princip pattern growth
  - Neka je α frequent itemset, B su prefiks putevi za α, i neka je β itemset iz B. Tada $\alpha \cup \beta$ je frequent itemset akko je β frequent u B.
- "*abcdef* " je frequent itemset akko
  - "*abcde* " je frequent i
  - "*f* " je frequent u skupu transakcija koje sadrže "*abcde* "

Kreiranje FP stabla, sumarno

Algoritam za konstrukciju FP-stabla

Ulaz: $D = \{t_1, t_2, \ldots, t_n\}$ - baza transakcija; Min_Sup -minimalna podrška

Izlaz: T_{fp} - FP-stablo

Metod:

1. $L_1 = svi\_veliki\_1-skupovi(D, Min\_Sup)$
2. $SortL_1 = sortiraj\_desc(L_1)$

   /*sortiraju se veliki skupovi u opadajućem redosledu u odnosu na podršku*/
3. Kreiraj korijen za T_{fp} i označi na sa NULL
4. FOR EACH $t \in D$ DO

   [p|P]=sortiraj elemente iz t saglasno redosledu u $SortL_1$

   insert_tree([p|P], T_{fp})

   END FOR

FUNCTION insert_tree([p|P]:lista_objekata, T:čvor_FP-stabla)

IF $\exists N(čvor\_FP-stabla): N \in djeca(T) \wedge N.oznaka = p)$ THEN

$N.podrška++$

ELSE

$N = new(čvor\_FP-stabla)$

$N.podrška = 1 \wedge N.otac = T$

modifikuj heder tabelu zbog novog čvora N

END IF

IF $P \neq \varnothing$ THEN

insert_tree(P,N)

END IF

END FUNCTION

Rudarenje FP stabla, sumarno

FUNCTION *FP_Growth(T:FP-stablo, α :veliki_skup)*

IF *T sadrži samo jednu granu P* THEN

FOR EACH $\beta = kombinacija\_čvorova\_iz\_P$ DO

$novi\_veliki\_skup = \alpha \cup \beta$

$novi\_veliki\_skup.podrška = \min(c.podrška \wedge c \in \beta)$

END FOR

ELSE

FOR EACH $o_i \in heder\_tabela\_za\_drvo\_T$ DO

$novi\_veliki\_skup = \alpha \cup o_i$

$novi\_veliki\_skup.podrška = o_i.podrška$

$\beta = novi\_veliki\_skup$

pronađi prefiks puteve za β

T_β = *uslovno FP-stablo za* β

IF $T_\beta \neq \varnothing$ THEN

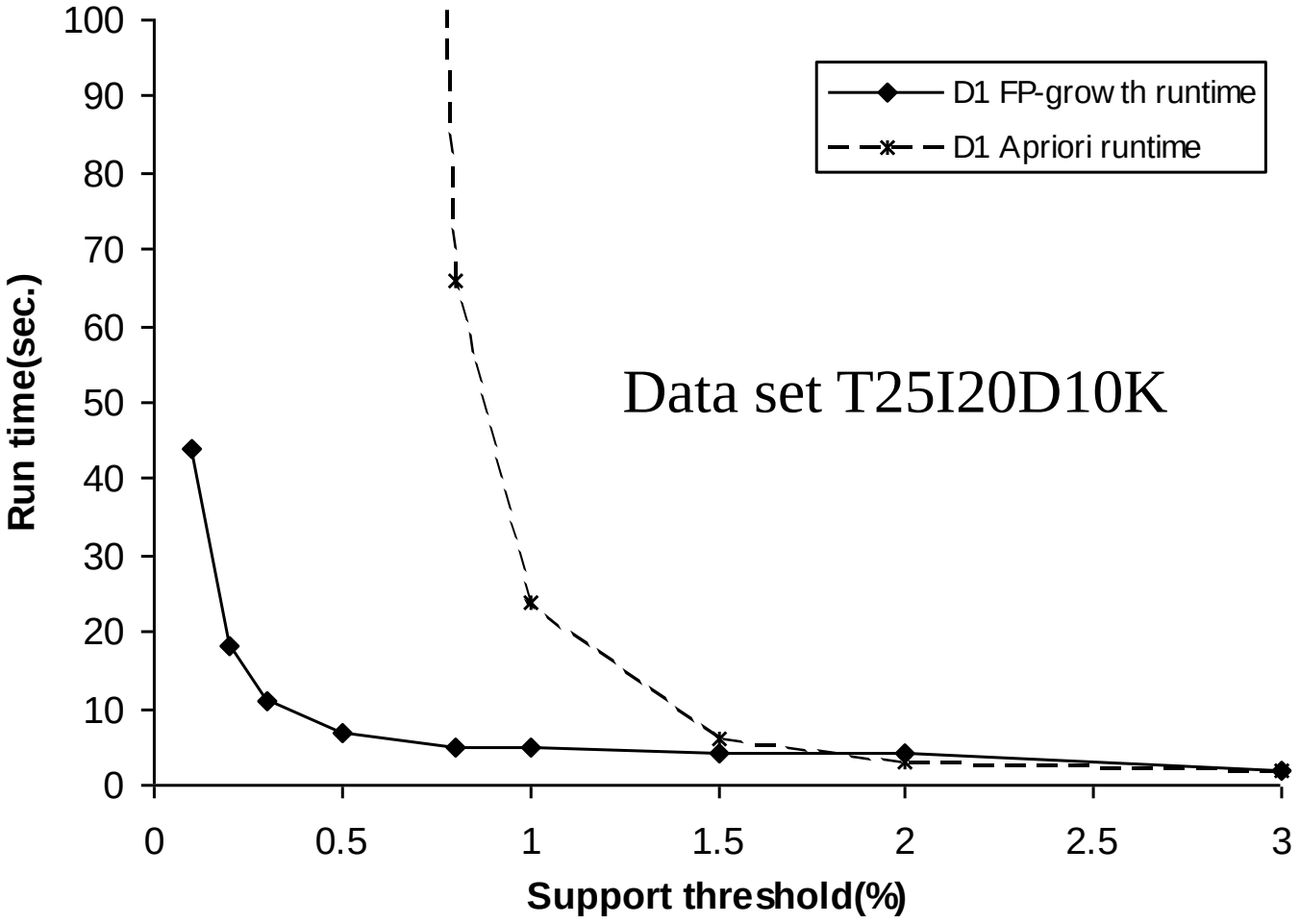
$FP\_Growth(T_\beta, \beta)$

END IF

END FOR

END IF

FP-Growth vs Apriori



Generisanje asocijativnih pravila

- Algoritam: za frequent itemset L generišu se svi neprazni podskupovi f $\subset$ L takvi da pravilo f $\rightarrow$ L – f zadovoljava kriterijum minconf
  - Neka je {A,B,C,D} frequent itemset, moguća pravila su:

ABC $\rightarrow$D,	ABD $\rightarrow$C,	ACD $\rightarrow$B,	BCD $\rightarrow$A,
A $\rightarrow$BCD,	B $\rightarrow$ACD,	C $\rightarrow$ABD,	D $\rightarrow$ABC
AB $\rightarrow$CD,	AC $\rightarrow$ BD,	AD $\rightarrow$ BC,	BC $\rightarrow$AD,
BD $\rightarrow$AC,	CD $\rightarrow$AB,		

- Ako je |L| = k, tada postoji 2^k – 2 mogućih asocijativnih pravila (ne računaju se L $\rightarrow \varnothing$ i $\varnothing \rightarrow$ L)

Contingency table

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

Pravilo: Tea $\rightarrow$ Coffee

Confidence= P(Coffee|Tea) = 0.75