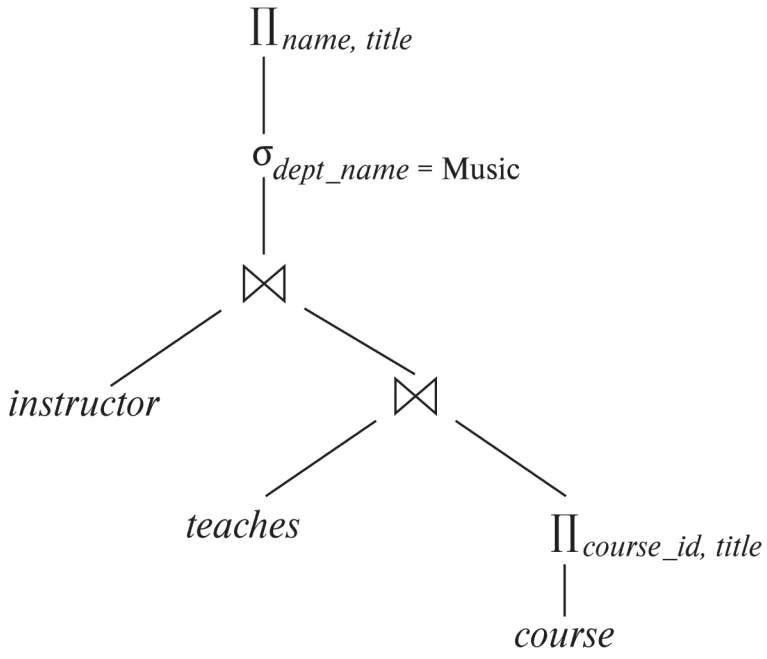
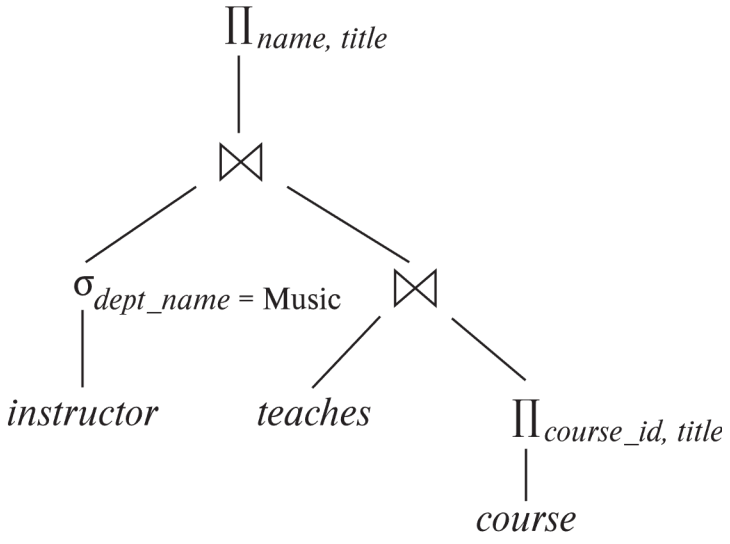
Optimizacija upita
Baze podataka 2

dr Miloš CVETANOVIĆ

Optimizacija upita

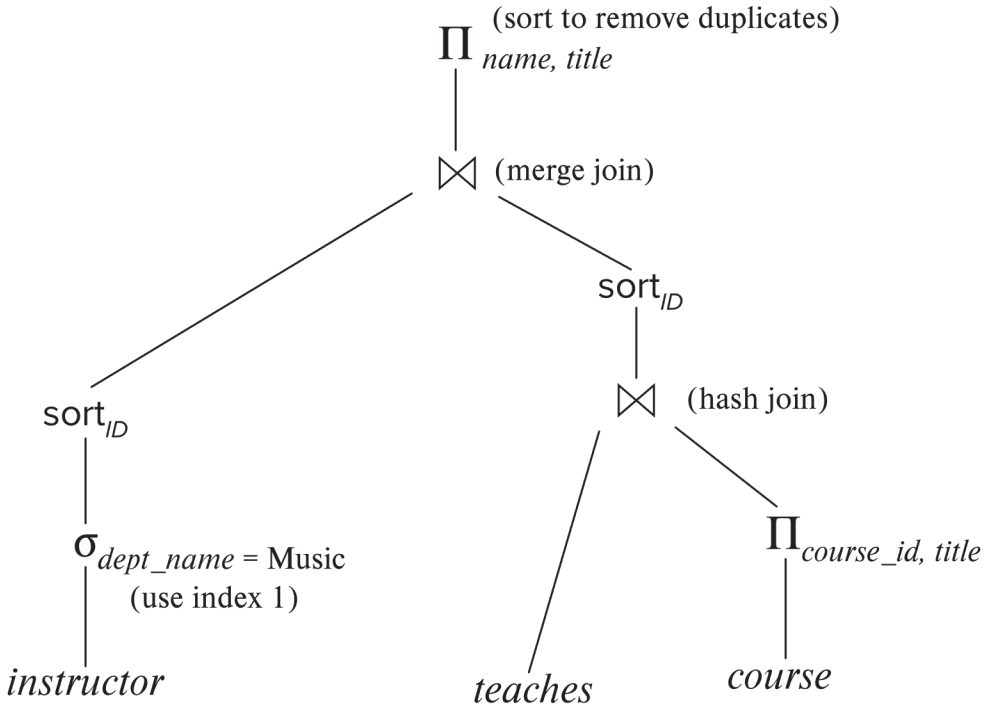
- Više načina za evaluaciju upita
  - Ekvivalentni iskazi
  - Različiti algoritmi za svaku od operacija





Optimizacija upita

- Plan evaluacije definiše tačno koji algoritam se koristi za koju operaciju, kao i kako se koordinišu operacije.



- Prikaz plana evaluacije / izvršavanja na DBMS – tekstualni ili grafički
  Npr. Oracle: explain plan for <query> by select * from table (dbms.xplan.display)
  SQL server: set showplan_text on
  PostgreSQL: explain analysis <query> -- prikazuje cenu za prvi red i za sve redove

Optimizacija upita

- Razlike u ceni između pojedinih planova evaluacije moju biti velike
  - Npr. U nekim slučajevima sekunde naspram dana
- Koraci u optimizaciji zasnovanoj na ceni
  1. Generisanje logički ekvivalentnih iskaza koristeći pravila ekvivalencije
  2. Anotiranje odabranog iskaza kako bi se došlo do alternativnih planova
  3. Odabir najjeftinijeg plana na osnovu procene cene
- Procena cene plana zasnovana na:
  - Statističkim informacijama o relacijama
    (npr. broj redova, broj različitih vrednosti za neke od atributa)
  - Statističkih procena za među-rezultate, kako bi se sračunala cena kompleksnih iskaza
  - Upotreba formula sa cenama za algoritme, izračunate koristeći statistike

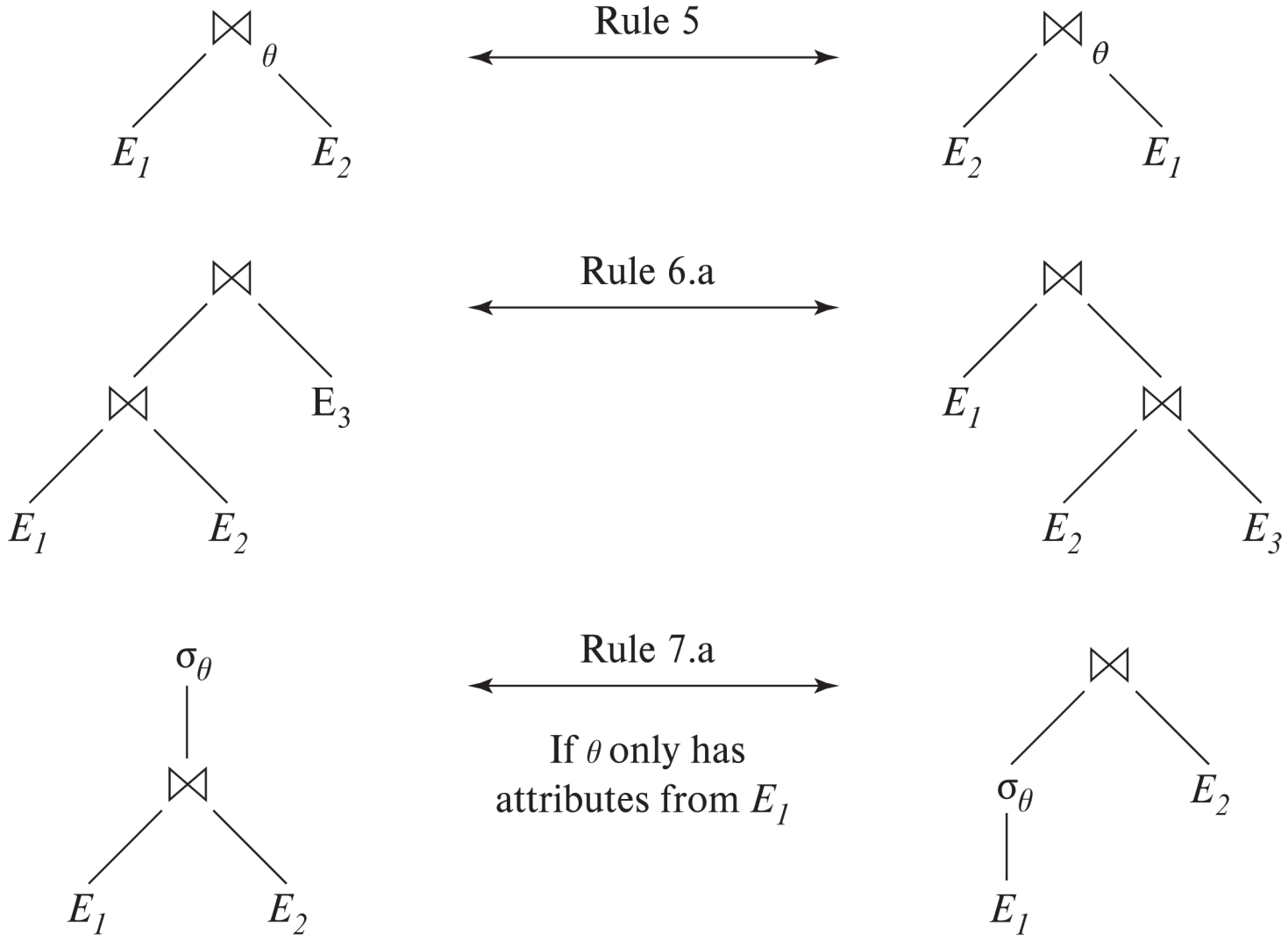
Transformacija iskaza relacione algebre

- Dva iskaza relacione algebre su ekvivalentna ako generišu isti skup redova za svaku legalnu instancu baze (instancu koja ne narušava nijedno ograničenje)
  - Redosled redova nije bitan
  - Nije važno ako generišu različit rezultat na bazama koji narušavaju neko od ograničenja
- Kod SQL ulaz i izlaz su multiskupovi (**multiset**, **bag sementacs**) redova
  - Operacije relacione algebre treba modifikovati
- Pravilo ekvivalencije (**equivalence rule**) govori da su dve forme iskaza ekvivalentni, te se jedan može zameniti drugim, i obratno

Pravila ekvivalencije

- 1. Konjukcija operacija selekcije se može dekonstruisati u sekvencu individualnih selekcija
  $\sigma_{\theta_1 \wedge \theta_2}(E) \equiv \sigma_{\theta_1}(\sigma_{\theta_2}(E))$
- 2. Operacije selekcija su komutativne
  $\sigma_{\theta_1}(\sigma_{\theta_2}(E)) \equiv \sigma_{\theta_2}(\sigma_{\theta_1}(E))$
- 3. Samo poslednja u nizu operacija projekcija je potrebna, a ostale se mogu izostaviti
  $\prod_{L_1}(\prod_{L_2}(\ldots(\prod_{L_n}(E))\ldots)) \equiv \prod_{L_1}(E)$ gde je $L_1 \subseteq L_2 \ldots \subseteq L_n$
- 4. Selekcije mogu biti kombinovane sa Dekartovim proizvodom i teta spajanjem
  a. $\sigma_{\theta}(E_1 \times E_2) \equiv E_1 \bowtie_{\theta} E_2$
  b. $\sigma_{\theta_1}(E_1 \bowtie_{\theta_2} E_2) \equiv E_1 \bowtie_{\theta_1 \wedge \theta_2} E_2$
- 5. Operacije teta spajanja (i prirodnog spajanja) su komutativne
  $E_1 \bowtie E_2 \equiv E_2 \bowtie E_1$
- 6.
  a. Operacije prirodnog spajanja su asocijativne
  $(E_1 \bowtie E_2) \bowtie E_3 \equiv E_1 \bowtie (E_2 \bowtie E_3)$
  b. Operacije teta spajanja su asocijativne u sledećem obliku
  $(E_1 \bowtie_{\theta_1} E_2) \bowtie_{\theta_2 \wedge \theta_3} E_3 \equiv E_1 \bowtie_{\theta_1 \wedge \theta_3} (E_2 \bowtie_{\theta_2} E_3)$
  gde θ_2 uključuje atribute samo iz E_2 i E_3

Pravila ekvivalencije



Pravila ekvivalencije

- 7. Operacija selekcije se distribuira po teta spajanju pod određena dva uslova
  a. Kada svi atributi u θ_0 uključuju samo atribute jednog od izraza (E_1) koji učestvuje u spajanju

$$\sigma_{\theta_0}(E_1 \bowtie_\theta E_2) \quad \equiv \quad (\sigma_{\theta_0}(E_1)) \bowtie_\theta E_2$$

  b. Kada θ_1 uključuje samo atribute iz E_1 i θ_2 uključuje samo atribute iz E_2

$$\sigma_{\theta_1 \wedge \theta_2}(E_1 \bowtie_\theta E_2) \quad \equiv \quad (\sigma_{\theta_1}(E_1)) \bowtie_\theta (\sigma_{\theta_2}(E_2))$$

- 8. Operacija projekcije se distribuira po teta spajanju na sledeći način
  a. Ako θ uključuje samo atribute iz $L_1 \cup L_2$

$$\prod_{L_1 \cup L_2}(E_1 \bowtie_\theta E_2) \quad \equiv \quad \prod_{L_1}(E_1) \bowtie_\theta \prod_{L_2}(E_2)$$

  b. U opštem slučaju, razmatrati spajanje $E_1 \bowtie_\theta E_2$
  - Neka su L_1 i L_2 skupovi atributa iz E_1 i E_2, respektivno
  - Neka su L_3 atributi iz E_1 koji su uključeni u uslov spajanja θ, ali nisu u $L_1 \cup L_2$, i
  - Neka su L_4 atributi iz E_2 koji su uključeni u uslov spajanja θ, ali nisu u $L_1 \cup L_2$, i

$$\prod_{L_1 \cup L_2}(E_1 \bowtie_\theta E_2) \quad \equiv \quad \prod_{L_1 \cup L_2}(\prod_{L_1 \cup L_3}(E_1) \bowtie_\theta \prod_{L_2 \cup L_4}(E_2))$$

Slična pravila ekvivalencije važe i za spoljna spajanja: ⟕, ⟖, i ⟗

Pravila ekvivalencije

- 9. Operacije unije i preseka su komutativni (razlika skupova nije)

  $E_1 \cup E_2 \equiv E_2 \cup E_1$

  $E_1 \cap E_2 \equiv E_2 \cap E_1$

- 10. Unija i presek skupova su asocijativni

  $(E_1 \cup E_2) \cup E_3 \equiv E_1 \cup (E_2 \cup E_3)$

  $(E_1 \cap E_2) \cap E_3 \equiv E_1 \cap (E_2 \cap E_3)$

- 11. Operacija selekcije se distribuira po uniji, preseku i razlici

  a. $\sigma_\theta (E_1 \cup E_2) \equiv \sigma_\theta (E_1) \cup \sigma_\theta(E_2)$

  b. $\sigma_\theta (E_1 \cap E_2) \equiv \sigma_\theta (E_1) \cap \sigma_\theta(E_2)$

  c. $\sigma_\theta (E_1 - E_2) \equiv \sigma_\theta (E_1) - \sigma_\theta(E_2)$

  d. $\sigma_\theta (E_1 \cap E_2) \equiv \sigma_\theta(E_1) \cap E_2$

  e. $\sigma_\theta (E_1 - E_2) \equiv \sigma_\theta(E_1) - E_2$

  prethodno pravilo ne važi za uniju

- 12. Operacija projekcije se distribuira po uniji

  $\Pi_L(E_1 \cup E_2) \equiv (\Pi_L(E_1)) \cup (\Pi_L(E_2))$

Pravila ekvivalencije

- 13. Operacije selekcije se se distribuira po operaciji agregacije
  $\sigma_\theta({}_G\gamma_A(E)) \equiv {}_G\gamma_A(\sigma_\theta(E))$ ukoliko θ uključuje samo atribute iz G
- 14.
  a. Puno spoljno spajanje je komutativno
  $E_1 \mathbin{⟗} E_2 \equiv E_2 \mathbin{⟗} E_1$
  b. Levo i desno spoljno spajanje nije komutativno, ali
  $E_1 \mathbin{⟕} E_2 \equiv E_2 \mathbin{⟖} E_1$
- 15. Operacija selekcije se distribuira po levom i desno spoljnom spajanju, ako θ_1 uključuje samo atribute iz E_1
  a. $\sigma_{\theta_1}(E_1 \mathbin{⟕}_\theta E_2) \equiv (\sigma_{\theta_1}(E_1)) \mathbin{⟕}_\theta E_2$
  b. $\sigma_{\theta_1}(E_1 \mathbin{⟖}_\theta E_2) \equiv E_2 \mathbin{⟕}_\theta (\sigma_{\theta_1}(E_1))$
- 16. Spoljna spajanja mogu biti zamenjena unutrašnjim spajanjem pod određenim uslovima
  a. $\sigma_{\theta_1}(E_1 \mathbin{⟕}_\theta E_2) \equiv \sigma_{\theta_1}(E_1 \bowtie_\theta E_2)$
  b. $\sigma_{\theta_1}(E_1 \mathbin{⟖}_\theta E_1) \equiv \sigma_{\theta_1}(E_1 \bowtie_\theta E_2)$
  ukoliko θ_1 nije (**null-rejecting**) po E_2

  za uslov se kaže da je null-rejecting ukoliko se izračunava na false ili unknown za svaki red koji je dopunjen sa null poljima prilikom spoljnog spajanja

Pravila ekvivalencije

- Primetiti da nekoliko pravila ekvivalencije koja važe za spajanja, ne važe za spoljna spajanja
  - Npr. $\sigma_{year=2017}(instructor \bowtie teaches) \not\equiv \sigma_{year=2017}(instructor \bowtie teaches)$
  - Spoljna spajanja nisu asocijativna

    $(r \bowtie s) \bowtie t \quad \not\equiv \quad r \bowtie (s \bowtie t)$

    npr. za primer relacija

    r(A,B) = {(1,1)},
    s(B,C) = {(1,1)},
    t(A,C) = { }

Pravila ekvivalencije – primer transformacije

- Upit: Vraća imena svih instruktora koji su na Music departmanu, kao i nazive svih kurseva koji ti instruktori predaju

  $\Pi_{name,\ title}(\sigma_{dept\_name=\text{'Music'}}(instructor \bowtie (teaches \bowtie \Pi_{course\_id,\ title}(course))))$

- Transformacija nakon primene pravila 7a.

  $\Pi_{name,\ title}((\sigma_{dept\_name=\text{'Music'}}(instructor)) \bowtie (teaches \bowtie \Pi_{course\_id,\ title}(course)))$

- Sprovođenje selekcije što ranije dovodi do smanjena relacija koje se spajaju
  isto važi i za sprovođenje projekcije što ranije

Pravila ekvivalencije – primer više transformacija

- Upit: Vraća imena svih instruktora koji su na Music departmanu koji su držali bar jedan kurs u 2017. godini, kao i nazive tih kurseva

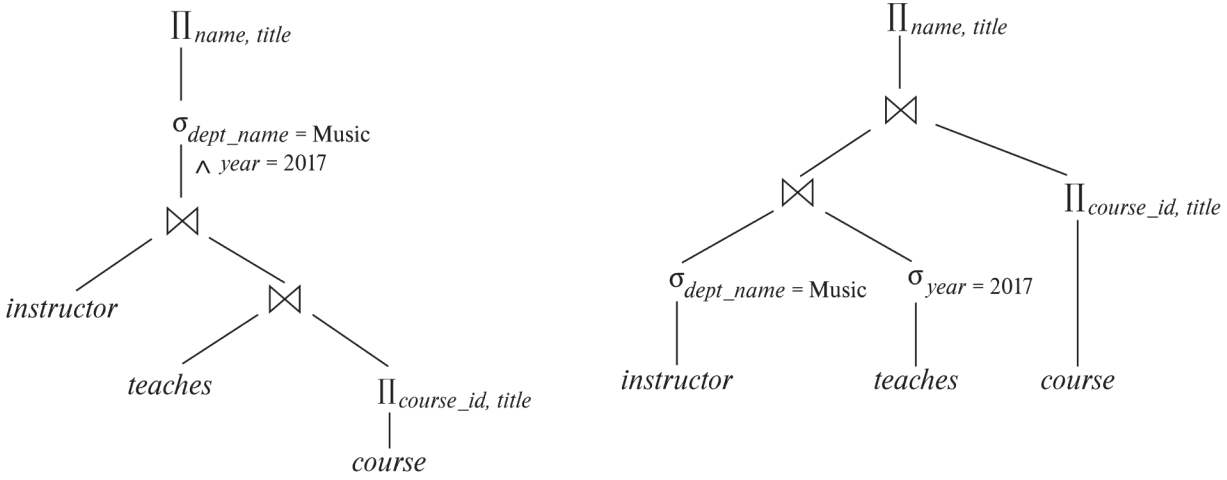
  $\Pi_{name,\ title}(\sigma_{dept\_name=\text{"Music"} \wedge year = 2017}\ (instructor \bowtie (teaches \bowtie \Pi_{course\_id,\ title}\ (course))))$

- Transformacija nakon primene pravila asocijacije 6a.

  $\Pi_{name,\ title}(\sigma_{dept\_name=\text{“Music”} \wedge year = 2017}\ ((instructor \bowtie teaches) \bowtie \Pi_{course\_id,\ title}\ (course)))$

- Postoji prilika da se selekcija sprovede što ranije, a time dobija pod izraz

  $\sigma_{dept\_name = \text{“Music”}}\ (instructor) \bowtie \sigma_{\ year = 2017}\ (teaches)$



Enumeracija ekvivalentnih izraza

- Optimizator upita koristi pravila ekvivalencije da sistematično generiše izraze koji su ekvivalentni datom izrazu
- Može da generiše sve ekvivalentne izraze na sledeći način:

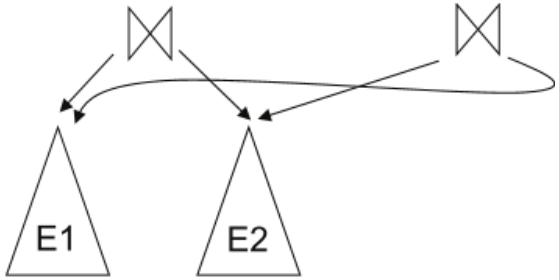
  Repeat

  - Primeni sva pravila ekvivalencije koja se mogu primeniti na svaki pod-izraz svakog ekvivalentnog izraza pronađenog do tog trenutka
  - Dodati novo generisane izraze u skup ekvivalentnih izraza

  Until (nije generisan nijedan novi izraz)
- Prethodno navedeni algoritam enumeracije je i prostorno i vremenski složen
  - Optimizovano generisati planove na osnovu pravila transformacije
  - Specijalni pristupi za upite koji sadrže samo selekciju, projekciju i spajanje

Enumeracija ekvivalentnih izraza – pravila transformacije

- Prostorna složenost se smanjuje ukoliko se koriste deljeni pod-izrazi
  - Kada je E1 generisan na osnovu E2 primenom nekog pravila ekvivalencije, obično je samo vršni nivo različit, dok su podstabla ista i mogu biti deljana
    npr. nakon primene pravila komutativnosti



  - Takođe, identični pod-izrazi mogu biti generisani više puta te je moguće obrisati duplikate i čuvati samo jednu kopiju
- Vremenska složenost se smanjuje ukoliko se ne generišu svi mogući izrazi
  - Dinamičko programiranje (**dynamic programming**)
    npr. Specijalni slučaj dinamičkog programiranja za optimizaciju redosleda spajanja

Redosled spajanja

- Za sve relacije r_1, r_2, and r_3 važi pravilo asocijativnosti $(r_1 \bowtie r_2) \bowtie r_3 = r_1 \bowtie (r_2 \bowtie r_3)$

  Ukoliko je među-rezultat spajanja $r_2 \bowtie r_3$ veliki, a spajanja $r_1 \bowtie r_2$ mali, onda je najbolje odabrati $(r_1 \bowtie r_2) \bowtie r_3$ tako da se čuvaju manji međurezultati

- Na primer, $\Pi_{name,\ title}(\sigma_{dept\_name=\ \text{“Music”}}$ (*instructor)* $\bowtie$ *teaches*) $\bowtie$ $\Pi_{course\_id,\ title}$ (*course*))))

  može se najpre izračunati: *teaches* $\bowtie$ $\Pi_{course\_id,\ title}$ (*course*)
  a potom spojiti sa: $\sigma_{dept\_name=\ \text{“Music”}}$ (*instructor)*
  međutim rezultat prvog spajanja može biti veliki

  Imajući u vidu da verovatno mali broj profesora predaje na Music departmanu, bolje je najpre izračunati $\sigma_{dept\_name=\ \text{“Music”}}$ (*instructor)* $\bowtie$ *teaches*

Optimizacija po ceni

- Cena svake od operacija relacione algebre (najbolje pojedinačne cene ili ukupna cena)
  - Npr. merge-join dobar kada treba sortiranost, ali nested-loop dobar za pipelining
- Pronalaženje najboljeg redosleda spajanja za $r_1 \bowtie r_2 \bowtie \ldots \bowtie r_n$
- Postoji $(2(n-1))!/(n-1)!$ različitih redosleda spajanja za n relacija
  za n=7, broj je 665280, dok je za n=10 to 176 milijardi.
- Nema potrebe generisati sve redoslede spajanja
  Koristeći dinamičko programiranje pronaći najjefitiniji redosled spajanja
  za bilo koji podskup relacija $\{r_1, r_2, \ldots r_n\}$ se odredi jednom i čuva za buduću upotrebu
- Pronalaženje najboljeg stable spajanja za skup od n relacija:
  - Razmotriti sve moguće planove u formi: $S_1 \bowtie (S - S_1)$
    gde je S_1 bilo koji neprezan podskup od S
  - Rekurzivno izračunati cenu za spajanje podskupova od S
    kako bi se odredila cena svakog plana.
    Odabrati najjeftiniji od 2^n-2 alternativa
  - Osnovni slučaj za rekurziju: plan pristupa za jednu relaciju
    Primeniti sve selekcije na R_i koristeći najbolji izbor indeksa nad R_i
  - Kada se plan za bilo koji podskup relacija izračuna, to se čuva i ponovo koristi
    umesto da se ponovo izračunava (dinamičko programiranje)

Redosled spajanja – optimizacija dinamičkim programiranjem

procedure findbestplan(*S*)
if (*bestplan*[*S*].*cost* $\neq \infty$)
    return *bestplan*[*S*]
// else *bestplan*[*S*] has not been computed earlier, compute it now
if (*S* contains only 1 relation)
  set *bestplan*[*S*].*plan* and *bestplan*[*S*].*cost* based on the best way
  of accessing *S* using selections on *S* and indices (if any) on S
else for each non-empty subset *S1* of *S* such that *S*1 $\neq$ *S*
    P1= findbestplan(*S*1)
    P2= findbestplan(*S* - *S*1)
    for each algorithm A for joining results of *P*1 and *P*2
  … compute plan and cost of using A (*…pogledati narednu stranu*) …
      if *cost* < *bestplan*[*S*].*cost*
          bestplan[*S*].*cost* = cost
          bestplan[*S*].*plan* = *plan*;
return *bestplan*[*S*]

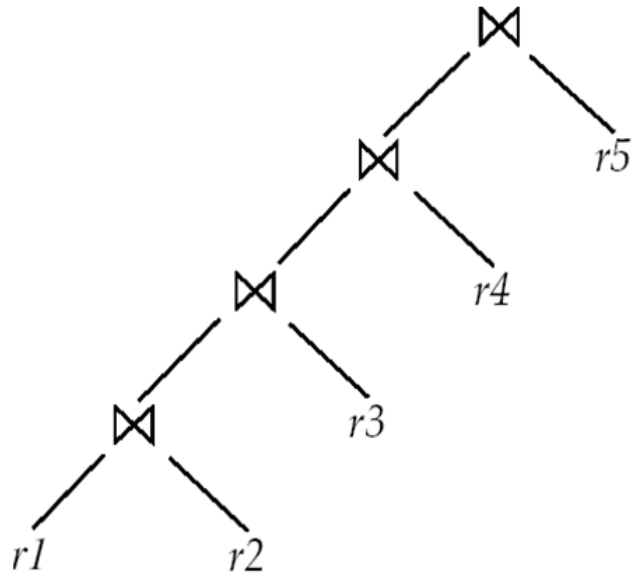
Redosled spajanja – optimizacija dinamičkim programiranjem

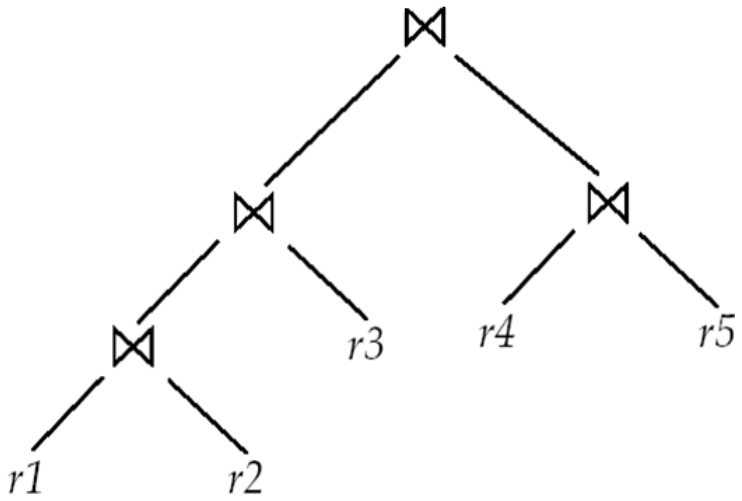
for each algorithm A for joining results of *P*1 and *P*2
// For indexed-nested loops join, the outer could be *P1* or *P2*
// Similarly for hash-join, the build relation could be *P1* or *P2*
// We assume the alternatives are considered as separate algorithms
if algorithm *A* is indexed nested loops
Let P_i and P_o denote inner and outer inputs
if P_i has a single relation r_i and r_i has an index on the join attribute
plan = "execute P_o.*plan*; join results of P_o and r_i using *A*",
with any selection conditions on P_i performed as part of the join condition
cost = P_o.*cost* + cost of *A*
else *cost* = ∞; /* cannot use indexed nested loops join */
else
plan = "execute *P1.plan*; execute *P2.plan*;
join results of *P1* and *P2* using *A*;"
cost = *P1.cost* + *P2.cost* + cost of *A*

…pogledati prethodnu stranu

Levo duboko stablo spajanja (left deep join tree)

- Kod levog dubokog stabla spajanja, desna strana ulaza svakog spajanja je relacija, a ne među-rezultat nekog drugog spajanja





Optimizacija po ceni

- Sa dinamičkim programiranjem vremenska kompleksnost sa proizvoljnim stablima je $O(3^n)$
  - Za n=10, broj je 59000 umesto 176 milijardi
- Prostorna kompleksnost je $O(2^n)$
- Da bi se odredilo najbolje levo duboko stablo spajanja za skup od n relacija:
  - Razmotriti n alternativa sa jednom relacijom na desnoj strain ulaza spajanja i ostalim relacijama na levoj strain ulaza spajanja
  - Modifikovati algoritam za optimizaciju redosleda spajanja
    zameniti: "**else for each** non-empty subset *S1* of *S* such that $S1 \neq S$"
    sa: "**for each** relation r in S
    let S1 = S – r . "
- Ako se samo leva duboka stabla spajanja razmatraju, vremenska kompleksnost za pronalaženje najboljeg redosleda spajanja je $O(n\ 2^n)$, dok je prostorna i dalje $O(2^n)$
- Optimizacija po ceni je skupa, ali vredi za upite nad velikom količinom podataka (većina upita ima malo n, generalno < 10)
- Intersantan sortirani redosled (**interesting sort order**) je određeni sortirani redosled redova koji može učiniti kasniju operaciju (spajanje, grupisanje, uređenje) jeftinijom
  - Nije dovoljno naći najbolji redosled spajanja,
    već najbolji redosled spajanja sa interesantnim sortiranim redosledom

Struktura optimizatora

- Optimizacija po ceni je skupa čak i sa dinamičkim programiranjem,
  pa se zbog toga koriste heuristika (koje često, ali ne uvek, pomažu)
  npr. selekcija što pre, projekcija što pre, što restriktivnija selekcija i spajanje što pre…
  - Npr. upotrebiti heuristike za jeftine upite,
    a sprovesti iscrpnu enumeraciju za skupe upite
- Budžet cene optimizacije (**optimization cost budget**) – zaustaviti optimizaciju rano ukoliko je cena plana manja od cene optimizacije
- Keširanje plana (**plan cashing**) – ponovo upotrebiti prethodni određen plan, ukoliko se upit ponovi, čak i sa različitim konstantama

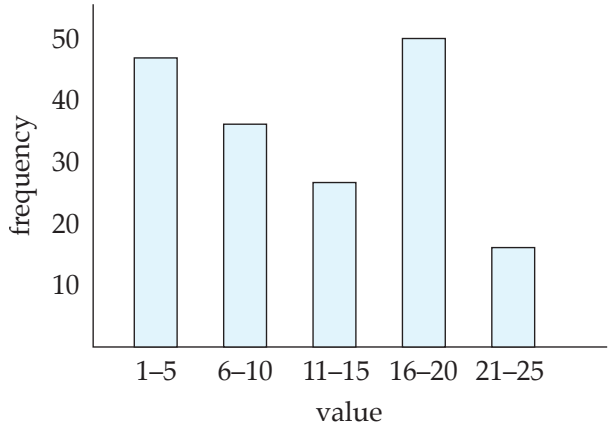
Statističke informacije za procenu cena

- n_r – broj redova relacije *r*
- b_r – broj blokova koje zauzima relacija *r*
- l_r – veličina jednog reda relacije *r*
- f_r – faktor blokiranja relacije *r*, tj. broj redova koji staje u jedan blok
- *V(A, r)* – broj različitih vrednosti koje se u relaciji *r* javljaju za atribut *A*
- Ukoliko su redovi relacije *r* sačuvani fizički zajedno u datoteci, onda

$$b_r = \left\lceil \frac{n_r}{f_r} \right\rceil$$

Histogrami

- Npr. Histogram za atribut godine u relaciji osoba



- Histogram sa podjednakim širinama (**equi-width**)
- Histogram sa podjednakim dubinama – opseg je podeljen na pod-opsege tako svaki pod-opseg ima (približno) isti broj redova (npr. 4, 8. 14, 19)
- Mnogi DBMS takođe čuvaju *n* najčešćih vrednosti i njihove brojače
  - U tom slučaju se histogrami prave na osnovu preostalih vrednosti
- Histogrami i druge statistike se obično izračunavaju na osnovu slučajnog uzorka
- Statistika može da „obajati“
  - Potrebno je pokrenuti ponovnu analizu da bi se statistika ažurirala
  - Moguće je automatski održavati statistiku ažurnom (kao kada bi se koristio okidač)

Procena veličine – selekcija

- Selekcija po uslovu jednakosti $\sigma_{A=v}(r)$
  - n_r / *V(A,r)* broj zapisa koji zadovoljavaju uslov selekcije
  - Ukoliko je uslov jednakosti po ključnom atributu: procena veličine = 1

- Selekcija po uslovu nejednakosti $\sigma_{A \leq v}(r)$ (odnosno, $\sigma_{A \geq v}(r)$ je simetričan slučaj)
  - Neka je *c* procenjen broj redova koji zadovoljavaju uslov
  - Ako su min(A, r) i max(A, r) dostupni u katalogu
    c=0 ako je v < min(A, r)

    $$c = n_r . \frac{v - \min(A,r)}{\max(A,r) - \min(A,r)}$$

  - Ako su histogrami dostupni, onda se procena može uraditi preciznije
  - U nedostatku statističkih informacija, pretspostavlja se da je $c=n_r/2$

Procena veličine – kompleksna selekcija

- Selektivnost (**selectivity**) uslova θ_i je verovatnoća da će red relacije *r* zadovoljiti uslov θ_i
  - Ako je s_i broj redova koji zadovoljava uslov, onda je selektivnost uslova θ_i data sa s_i/n_r

- Konjukcija: $\sigma_{\theta 1 \wedge \theta 2 \wedge \ldots \wedge \theta n}(r)$ procena veličine (tj. broja redova) je:

$$n_r * \frac{s_1 * s_2 * \ldots * s_n}{n_r^n}$$

- Disjunkcija: $\sigma_{\theta 1 \vee \theta 2 \vee \ldots \vee \theta n}(r)$ procena veličine (tj. broja redova) je:

$$n_r * \left(1 - (1 - \frac{s_1}{n_r}) * (1 - \frac{s_2}{n_r}) * \ldots * (1 - \frac{s_n}{n_r})\right)$$

- Negacija: $\sigma_{\neg\theta}(r)$ procena veličine (tj. broja redova) je:

$$n_r - size(\sigma_\theta(r))$$

Procena veličine – spajanje

- Dekartov proizvod $r \times s$ sadrži $n_r \cdot n_s$ redova, a svaki red zauzima $l_r + l_s$ bajtova
- Ako je $R \cap S = \varnothing$, onda je $r \bowtie s$ isto što i $r \times s$
- Ako je $R \cap S$ ključ relacije R, onda će se red relacije s spojiti sa najviše jednim redom iz r
  - Zbog toga broj redova u $r \bowtie s$ nije veći od broja redova u relaciji s
- Ako je $R \cap S$ u S jeste strani ključ koji referiše R, onda je broj redova u $r \bowtie s$ tačno jednak broju redova u relaciji s
  - Slučaj kada je $R \cap S$ strani ključ koji referiše S, je simetričan
- Ako $R \cap S = \{A\}$ nije ključ ni za R ni za S
  Ako se pretpostavi da svaki red t u relaciji r proizvodi redove u $r \bowtie s$, onda je broj redova

$$\frac{n_r * n_s}{V(A,s)}$$

  Ako je obrnuto tačno, onda je procena broja redova $\frac{n_r * n_s}{V(A,r)}$

  Manja od ova dva broja je verovatno tačnija procena

- Prethodne procene mogu biti popravljene ukoliko su histogrami dostupni

Procena veličine – ostale operacije

- Projekcija: procena veličine je $\Pi_A(r)$ = $V(A,r)$
- Agregacija: procena veličine je ${}_G\gamma_A(r)$ = $V(G,r)$
- Operacije nad skupovima
  - Za uniju/presek selekcija nad istom relacijom: koristiti procemu veličine za selekciju
    npr. $\sigma_{\theta 1}(r) \cup \sigma_{\theta 2}(r)$ može biti zapisano i kao $\sigma_{\theta 1 \vee \theta 2}(r)$
  - Za operacije nad različitim relacijama:
    procena veličine za $r \cup s$ = size of r + size of s
    procena veličine za $r \cap s$ = minimum size of r and size of s
    procena veličine za $r - s$ = r

    sve tri procene mogu biti poprilično neprecizne, ali daju gornju granicu veličine
- Spoljno spajanje:
  procena veličine za $r \mathbin{⟕} s$ = *size of* $r \bowtie s$ + *size of* r
  procena veličine za $r \mathbin{⟗} s$ = *size of* $r \bowtie s$ + size of r + size of s

Procena broja različitih vrednosti – selekcija

- Selekcija $\sigma_\theta(r)$
  - Ako θ forsira da A ima određenu vrednost: $V(A, \sigma_\theta(r))=1$
  - Ako θ forsira da A ima jednu određenu vrednost iz skupa vrednosti:
    $V(A, \sigma_\theta(r))$=*broj mogućih vrednosti u skupu*
  - Ako je uslov θ u formi *A op r* onda je procena:
    $V(A, \sigma_\theta(r)) = V(A, r) \cdot s$ gde je *s* selektivost selekcije
  - U svim ostalim slučajevima: koristiti približnu procenu $min(V(A, r), n_{\sigma_\theta(r)})$

Procena broja različitih vrednosti – spajanje

- Spajanje $r \bowtie s$
  - Ako su svi atributi u A iz relacije r
    onda je procena $V(A, r \bowtie s) = \min(V(A, r), n_{r \bowtie s})$

  - Ako A sadrži atribute A1 iz relacije r, kao i atribute A2 iz relacije s
    onda je procena $V(A, r \bowtie s) = \min(V(A1,r) * V(A2 - A1,s), V(A1 - A2,r) * V(A2,s), n_{r \bowtie s})$

Procena broja različitih vrednosti – ostale operacije

- Projekcija: procena je ista kao i $\prod_{A\,(r)}$
- Ista procena važi i za atribute po kojima se vrši grupisanje u agregaciji
- Procena za agregirane vrednosti:
  - Za min(A) i max(A) procena je
    min(V(A, r), V(G, r)) gde G označava atribute po kojima vrši grupisanje
  - Za sve ostale agregatne funkcije, pretpostaviti da su sve vrednosti različite dakle, koristiti *V(G, r)*

Određivanje cene upita – primer

- Neka je data relaciona šema:
  Katedra (SifD, Naziv)
  Predmet (SifP, Naziv)
  Profesor (SifP, Ime, SifD)
  Predaje (SifPro, SifPre, Semestar)

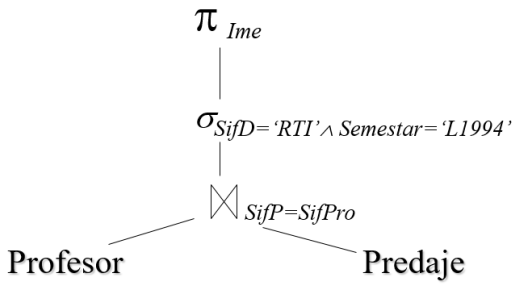
- Neka je dat upit:
  SELECT P.Ime
  FROM Profesor P, Predaje T
  WHERE P.SifP = T.SifPro AND P.SifD='RTI' AND T.Semestar='L1994';

- Neka su poznate sledeće informacije:
  $n_{Profesor}$=1000, $b_{Profesor}$= 200, Hash (*SifP*), $B^{+}_{depth=2}$(*SifD*)
  $n_{Katedra}$=50
  $n_{Predaje}$=10000, $b_{Predaje}$= 1000, *V(Semestar, Profesor)*=4, $B^{+}_{depth=2}$(*Semestar*), Hash(*SifPro*)

- Neka je poznato da je veličina bafera u memoriji M=52 stranica

Određivanje cene upita – primer

- Ekvivalentan iskaz relacione algebra (1)
  $\prod_{Ime}(\sigma_{SifD=\text{"RTI"} \wedge Semestar=\text{"L1994"}}(Profesor \bowtie Predaje))$

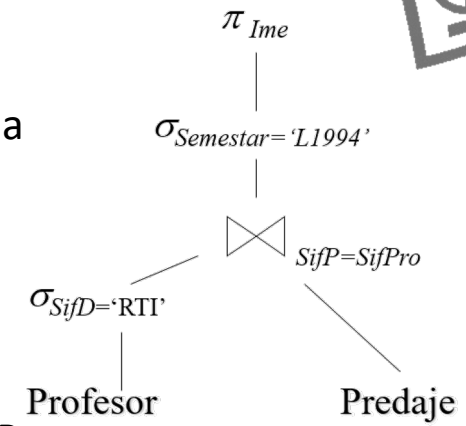


Plan evaluacije 1:

- Spajanje: blok ugnježdena petlja – Profesor spoljna, Predaje unutrašnja petlja
- Selekcija i priojekcija: skeniranje redova dobijenih u međurezultatu nakon spajanja

- Cena spajanja: $b_{Profesor} + \lceil b_{Profesor} / (M\text{-}2) \rceil \cdot b_{Predaje} = 200 + 4 \cdot 1000 = 4200$ stranica
- Cena selekcije i projekcija: Pipelining, te je cena 0.
- Ukupna cena: 4200

Određivanje cene upita – primer



- Ekvivalentan iskaz relacione algebra (2) - parcijalno potisnuta selekcija $\prod_{Ime}(\sigma_{Semestar="L1994"}(\sigma_{SifD="RTI"}(Profesor)) \bowtie Predaje))$

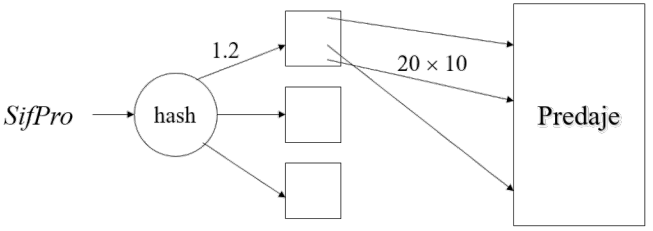
Plan evaluacije 2 :

- Selekcija 1: Access path upotreba indeksa B^+ stabla nad SifD
- Spajanje: Indeksirana ugnježdena petlja koristeći indeks Hash nad SifPro
- Selekcija 2: skeniranje redova dobijenih u međurezultatu nakon spajanja

- Cena selekcije: h=2, pretpostavka da su unutrašnji čvorovi u memoriji, onda je h=1
  V(SifD, $\sigma_{SifD="RTI"}(Profesor)$)= broj mogućih vrednosti (1000 profesora, 50 katedri) = 20
  $f_{Profesor} = n_{Profesor}/b_{Profesor}$ = 5, dakle za 20 je potrebno dohvatiti 4 stranice
  za selekciju ukupno: 4+2=6
- Cena spajanja: Spoljna petlja po selektovanim profesorima, unutrašnja pristup indeksom svih 4 stranica spoljne petlje mogu biti u memoriji

  $1,2 \cdot V(SifD, \sigma_{SifD="RTI"}(Profesor) + V(SifD, \sigma_{SifD="RTI"}(Profesor)) \cdot n_{Predaje}/ V(SifPro, (Predaje))$
  gde je $n_{Predaje}/V(SifPro, (Predaje)$ = 10000 / 1000 = 10
  odnosno, svakom profesoru odgovara 10 redova u *Predaje*)

  dakle, dobija se $1,2 \cdot 20 + 20 \cdot 10 = 224$
  (1,2 je const za hash, da bi se došlo do ulaza u indeks)



- Cena selekcije: Pipelining, te je cena 0.
- Ukupna cena: 6+224+0=230

Određivanje cene upita – primer

- Plan evalucije 1 (bez potiskivanja selekcije i projekcije): 4200
- Plan evaluacije 2 (delimično potisnuta selekcija): 230
- Plan evaluacije 3 (potpuno potisnuta selekcija): ?