



Databázové technologie

Provádění dotazů II

Petr Krajča



Katedra informatiky
Univerzita Palackého v Olomouci



- vnořené smyčky
- není jednorůchodový (jen částečně pro jeden operand)
- pracuje se jednotlivými řádky
- ideální realizace formou iterátoru

```
for each s in S:  
    for each r in R:  
        if joinable(r, s):  
            output join(r, s)
```

- problém s vnitřní smyčkou, počet vyvolání je $T(S)$
- lze vylepšit např. použitím indexu nebo prací s bloky

Nested-loop join (pracující s bloky, nested block join)



- obě relace jsou organizovány do bloků (redukce I/O)
- do paměti je načtena co největší část S (snížení počtu prochodů R)

for each chunk of $M-1$ blocks of S:

 read blocks to memory

 organize in search structure

 search key is the common attrs. of R and S

 for each block b of R:

 read b into memory

 for each (tuple) t in b:

 find in search structure tuples joinable with t

 output the join of t with these tuples

- (zdánlivě) tři smyčky
- dvě vnitřní smyčky realizují průchod R po blocích



- počet iterací vnější smyčky

$$\frac{B(S)}{M - 1}$$

- načtení dat z S

$$M - 1$$

- vnitřní smyčka (načtení R)

$$B(R)$$

- to jest:

$$\begin{aligned} & \frac{B(S)}{M - 1} (M - 1 + B(R)) \\ & = B(S) + \frac{B(S)B(R)}{M - 1} \end{aligned}$$

- postaveny na stejném principu jako dvoufázový mergesort
- fáze 1: řazení
 - 1 Do paměti je načteno M bloků tabulky R .
 - 2 Tato část je seřazena vhodným algoritmem a je zapsána na disk.
- fáze 2: operace se seřazenými částmi
- druhá fáze určuje chování operátoru

Eliminace duplicitních řádků

- 1 pro každou samostatně seřazenou část se načte do paměti jeden blok
 - 2 je vybrán nejmenší dosud neuvažovaný řádek
 - 3 ten je zapsán na první volné místo výstupního bloku (bufferu)
 - 4 ostatní identické řádky jsou ignorovány a odstraněny z příslušných bufferů
- náročnost operace $3 \cdot B(R)$
 - v druhé fázi nesmí být počet částí větší než M (počet bufferů), tj. $B(R) \leq M^2$
 - tj. algoritmus vyžaduje $\sqrt{B(R)}$ paměti

- 1 do paměti je načteno M bloků tabulky R, tato část je seřazena podle atributů, dle kterých dochází k seskupení, a je zapsána na disk
 - 2 pro každou samostatně seřazenou část se načte do paměti jeden blok.
 - 3 opakuj dokud existuje nezpracovaný řádek
 - (a) ze seřazených částí je vybrán nejmenší řádek (vzhledem k hodnotám atributů, dle kterých dochází k seskupení), představuje novou skupinu
 - (b) inicializuje se výpočet agregačních funkcí
 - (c) pro každý řádek, jenž má stejné hodnoty atributů dle, kterých dochází k seskupení, jsou upraveny agregované hodnoty
 - (d) pokud je buffer prázdný, načti další blok dané části
 - (e) pokud nejsou k dispozici další řádky se shodnými hodnotami atributů, dle kterých dochází ke slučování, zapiš výsledek i s agregovanými hodnotami na výstup
- náročnost na I/O i na paměť je stejná jako v případě eliminace duplicitních řádků



- narozdíl od dosud uvažovaných operací binární operace
- v první fázi seřazeny části obou tabulek
- v druhé fázi pro každou seřazenou část obou operandů načten jeden blok do paměti

množinové sjednocení

- 1 do paměti je načteno M bloků tabulky R , tam jsou seříděny a zapsány na disk
 - 2 dtto pro tabulku S
 - 3 pro každou samostatně seřazenou část (obou tabulek) se načte do paměti jeden blok
 - 4 opakuje se, dokud existují nezpracované řádky
 - (a) najde se nejmenší řádek t a je zapsán na výstup
 - (b) jsou odstraněny všechny řádky t z bufferu (jsou nejvýše dva)
 - (c) pokud je vstupní buffer prázdný načte se další blok dané části
- náročnost I/O: $3 \cdot (B(R) + B(S))$ ($2 \times$ čtení a $1 \times$ zápis obou tabulek)
 - pamět: $B(R) + B(S) \leq M^2$



- další operace analogicky sjednocení
- pro každý nejmenší řádek t je spočítán počet výskytů v tabulce R a S
- **množinový průnik** – řádek t je zapsán na výstup, pokud je v obou tabulkách
- **multimnožinový průnik** – řádek je zapsán na výstup tolikrát, kolik je nejmenší počet výskytů
- **množinový rozdíl** ($R - S$) – řádek je zapsán na výstup, pokud je v tabulce R a není v S
- **multimnožinový rozdíl** ($R - S$) – řádek je zapsán na výstup tolikrát, kolik je počet výskytů v tabulce R minus počet výskytů v tabulce S (pokud je 0 nebo méně, není na výstup zapsán)
- **multimnožinové sjednocení** – není potřeba
- náročnost, viz množinové sjednocení



- vstupem jsou tabulky $R(XY)$ a $S(YZ)$, kde Y jsou společné atribut
- 1 tabulky R a S jsou seřazeny dvoufázovým mergesortem dle Y
- 2 spojení R a S (používají se jen dva buffery, jeden pro R a druhý pro S)
 - (a) najdi nejmenší hodnotu y společných atributů (je na aktuálním začátku některého z bufferů)
 - (b) pokud v druhé tabulce není řádek s y , odstraň řádek s y
 - (c) jinak načti do paměti všechny řádky obou tabulek s hodnotami společných atributů y
 - (d) zapiš na výstup všechny spojené řádky z obou tabulek R a S , které mají hodnoty společných atributů y
 - (e) pokud je některý z bufferů prázdný, načti další blok
- náročnost: seřazení $4 \cdot (B(R) + B(S))$ (je zde zápis navíc) + spojení $B(R) + B(S)$, celkem $5 \cdot (B(R) + B(S))$
- cf. nested-loop join (kvadratická složitost) vs. sort-based join (lineární)



- při spojení může počet řádků, které mají stejné hodnoty atributů, být tak velký, že se nevejdou do paměti, viz krok (c)
- extrémní případ: právě jedna hodnota, na které dochází ke spojení (de facto kartézský součin)

řešení

- pokud pro jednu tabulka platí, že se řádky s hodnotami atributů y vejdou do $M - 1$ bufferů
- jsou načteny do paměti a druhá tabulka je postupně načítána do zbývajících bufferů a z něj brány jednotlivé řádky ke spojení
- de facto se jedná o jednopřechodový algoritmus spojení (ale pro jednu hodnotu)
- pokud není splněna počáteční podmínka, je nutné provést nested-loop join (pro hodnotu y)
- celkově paměťové nároky tedy dány operací řazení, tj. $B(R) \leq M^2$ a $B(S) \leq M^2$



- pokud není problém s velkým množstvím řádků se shodnými atributy, lze sloučit fázi řazení a spojení
- tím se ušetří 2 I/O operace na blok
- tabulky R a S jsou rozděleny na části o velikosti M a seřazeny podle Y (společných atributů)
- pro každou seřazenou část je načten jeden blok do paměti
- dokud existuje nezpracovaný řádek
 - (a) najdi nejmenší hodnotu y společných atributů
 - (b) najdi řádky obou relací mající hodnotu společných atributů y a ulož do volných bufferů
 - (c) zapiš navýstup spojené řádky
 - (d) pokud je některý buffer prázdný načti další blok



- náročnost I/O: $3 \cdot (B(R) + B(S))$
- paměť: $B(R) + B(S) \leq M^2$ (v druhá fázi musí být jednotlivé části v paměti)

omezená paměť pro spojení řádků?

- pokud je hodnota y klíčem (např. tabulky R) postupně projdeme jen řádky z tabulky S mající hodnoty y (případně opačně)
- pokud $B(R) + B(S) \ll M^2$, máme k dispozici velké množství bufferů
- lze použít nested-join za cenu více I/O získáme korektní výsledek



Všechny obrázky:

Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom. Database System Implementation.