Optimizarea frazelor de interogare

prof. dr. ing. Mircea Petrescu

Obiectivul optimizarii: creşterea vitezei de access la informaţia din baza de date (reducerea timpului de access). Evident de la început: timpul cel mai lung se consumă la prelucrarea interogărilor care conţin produse carteziene, respectiv joncţiuni.

Strategii generale folosite pentru optimizare:

1. Efectuarea selecţiilor cât mai devreme posibil;
2. Înainte de joncţiuni, se prelucrează preliminar fişierele (relaţiile) prin operaţiuni de sortare şi de indexare;
3. Căutarea de sub-expresii care se repetă în expresii mai mari;
4. Aranjarea în „cascadă" a selecţiilor şi proiecţiilor;
5. Combinarea proiecţiilor cu operaţiuni binare adiacente;
6. Combinarea unora din selecţii cu produse carteziene care eventual le preced, pentru a obţine joncţiuni.

Echivalenţa expresiilor

Folosită în procesul transformării expresiilor (algebrice relaţionale). În fapt, prelucrarea frazelor de interogare este iniţiată de un procesor de cereri (interogări), a cărui acţiune începe prin construirea unui arbore de derivare destinat evaluării unei expresii algebrice (conţinută implicit în fraza de interogare).

Reguli de echivalenţă a expresiilor:

1. Comutativitate: $E_1 \times E_2 \equiv E_2 \times E_1$; $E_1 \rhd\lhd E_2 \equiv E_2 \rhd\lhd E_1$.
2. Asociativitate: $(E_1 \times E_2) \times E_3 \equiv E_1 \times (E_2 \times E_3)$; $(E_1 \rhd\lhd E_2) \rhd\lhd E_3 \equiv E_1 \rhd\lhd (E_2 \rhd\lhd E_3)$.
3. Proiecţii în cascadă: $\Pi_{A_1 \ldots A_n}(\Pi_{B_1 \ldots B_m}(E)) \equiv \Pi_{A_1 \ldots A_n}(E)$, când fiecare $A_j \in \{B_i\}$.
4. Selecţii în cascadă: $\sigma_{F_1}(\sigma_{F_2}(E)) \equiv \sigma_{F_1 \wedge F_2}(E)$.
5. Comutativitatea selecţiilor cu proiecţiile: $\sigma_F(\Pi_{A_1 \ldots A_n}(E)) \equiv \Pi_{A_1 \ldots A_n}(\sigma_F(E))$.
6. Comutativitatea selecţiei cu produsul cartezian: $\sigma_F(E_1 \times E_2) \equiv \sigma_F(E_1) \times E_2$, dacă toate atributele din F fac parte din E_1. Dacă $F=F_1 \wedge F_2$, unde F_1 conţine numai atributele din E_1, iar F_2 conţine numai atribute din E_2, atunci: $\sigma_F(E_1 \times E_2) \equiv \sigma_{F_1}(E_1) \times \sigma_{F_2}(E_2)$. Dacă F_1 conţine numai atribute din E_1, iar F_2 conţine atât atribute din E_1, cât şi atribute din E_2: $\sigma_F(E_1 \times E_2) \equiv \sigma_{F_2}(\sigma_{F_1}(E_1) \times E_2)$.
7. Comutativitatea selecţie – reuniune, respectiv selecţie – diferenţă: $\sigma_F(E_1 \cup E_2) \equiv \sigma_F(E_1) \cup \sigma_F(E_2)$, $\sigma_F(E_1 - E_2) \equiv \sigma_F(E_1) - \sigma_F(E_2)$.
8. Comutativitatea proiecţie – produs cartezian: $\Pi_{A_1 \ldots A_n}(E_1 \times E_2) \equiv \Pi_{B_1 \ldots B_m}(E_1) \times \Pi_{C_1 \ldots C_k}(E_2)$, unde $\{B_j\}$ este mulţimea atributelor care $\in \{A_i\}$, iar B_j este prezent în E_1. Respectiv $\{C_j\}$ este mulţimea celorlalte atribute din $\{A_i\}$, fiecare C_j fiind prezent în E_2.
9. Comutativitatea proiecţie – reuniune: $\Pi_{A_1 \ldots A_n}(E_1 \cup E_2) \equiv \Pi_{A_1 \ldots A_n}(E_1) \cup \Pi_{A_1 \ldots A_n}(E_2)$.

Algoritm pentru optimizarea expresiilor relaţionale

Intrare: un arbore sintactic ce reprezintă o expresie relaţională.

Ieşire: program pentru evaluarea expresiei.

Metoda:

Pas 1. Fiecare selecție $\sigma_{F_1 \wedge \ldots \wedge F_n}(E)$ este transformată în secvența $\sigma_{F_1}(\ldots(\sigma_{F_n}(E)))$ (regula 4).

Pas 2. Fiecare selecție este deplasată cât mai jos posibil în arborele sintactic (regulile 4-7).

Pas 3. Fiecare proiecție este deplasată cât mai jos posibil în arborele sintactic (regulile 3, 8, 9, 5).

Pas 4. Secvențele de selecții şi proiecții sunt combinate în selecții sau proiecții unice, sau în selecții urmate de proiecții (regulile 3, 4, 5). Notă: în pasul 4, procesul recomandat poate încălca principiul potrivit căruia proiecțiile sunt efectuate cât mai curând posibil. Trebuie analizată eficiența!

Pas 5. Nodurile interne ale arborelui rezultate prin parcurgerea paşilor anteriori sunt grupate în „blocuri". Fiecare nod intern care corespunde unei operațiuni binare poate face parte din acelaşi bloc ca şi predecesorii săi imediați cu care sunt asociate operațiuni unare. Din bloc pot face parte şi orice lanț de noduri succesoare asociate cu operațiuni unare şi terminate cu o frunză. Ultima regulă nu se aplică atunci când operațiunea binară este un produs cartezian, iar acesta nu este urmat de o selecție care se combină cu produsul menționat, astfel încât să formeze o joncțiune cu Θ = semnul egalității (echi-joncțiune!).

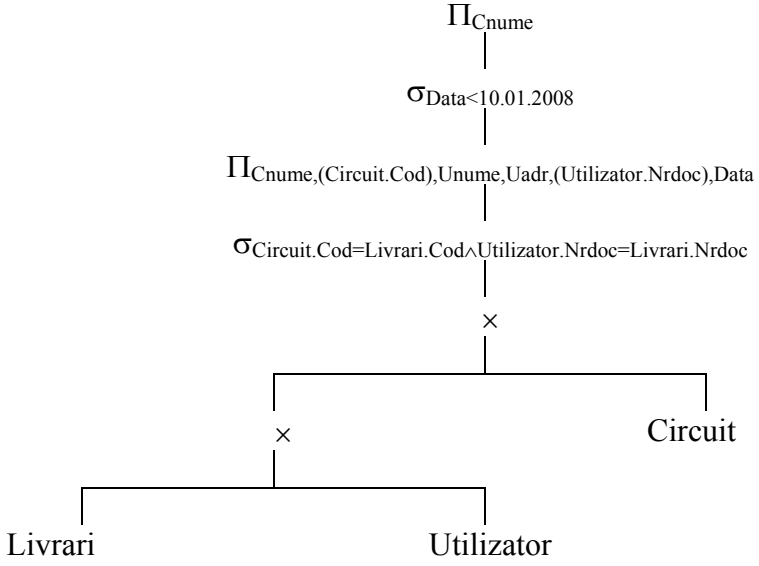
Pas 6. Se evaluează fiecare bloc, în orice ordine, astfel încât niciunul din blocuri nu este evaluat înaintea grupurilor sale succesoare. Ca rezultat al evaluării este produs un program.

Exemplu. Fie baza de date cu relațiile:
Circuit(Cnume, Fnume, Cod),
Furnizor(Fnume, Fadr),
Utilizator(Unume, Uadr, Nrdoc),
Livrari(Nrdoc, Cod, Data).

Presupunem că pentru a răspunde la anumite fraze de interogare privind livrările de circuite, este mai întâi construită o vedere care conține informații referitoare la circuitele livrate, cu ajutorul relațiilor Livrari, Utilizator, Circuit. Vederea va fi definită prin expresia relațională: $\Pi_V(\sigma_U(Livrari \times Utilizator \times Circuit))$, unde V = Cnume, Fnume, Cod, Unume, Uadr, Nrdoc, Data, iar U = Utilizator.Nrdoc = Livrari.Nrdoc $\wedge$ Circuit.Cod = Livrari.Cod.

Vom admite că fraza de interogare solicită lista numelor circuitelor livrate înainte de 10 ianuarie 2008. În termenii algebrei relaționale, această frază poate fi formulată prin expresia: $\Pi_{Cnume}(\sigma_{Data<10.01.2008}(\Pi_U(\sigma_V(Livrari \times Utilizator \times Circuit))))$.

Pentru evaluarea expresiei, este construit mai întâi arborele sintactic:



Ca urmare a aplicării algoritmului pentru optimizare, este obținut un nou arbore sintactic necesar evaluării frazei de interogare, cu forma: