Metode de acces la informatie în bazele de date pentru prelucrari grafice

Sef lucr.dr.ing. Marius Dorian ZAHARIA
Catedra de Calculatoare, Universitatea POLITEHNICA Bucuresti

Lucrarea prezinta modalitati de indexare a informatiei din bazele de date pentru prelucrari grafice. Sunt înfatisate metode de indexare bazate pe arbori B, metode de indexare (ierarhica sau neierarhica) a informatiei din spatii multidimensionale.
Cuvinte cheie: *indexare, arbore B, arbore B+, arbore k-d, index LSD, fisier BANG, fisier grila.*

1. Introducere

Realizarea accesului la colectii masive de date gestionate prin intermediul SGBD-urilor relationale, sau al celor orientate spre obiecte este facuta cu ajutorul mecanismelor de indexare a informatiei. Administratorii bazelor de date pot crea structuri de indexare peste colectii de date cu o structura similara. Pentru a îmbunatati eficienta executiei cererilor într-un sistem de gestiune a bazelor de date pentru prelucrari grafice, au fost dezvoltate metode avansate de indexare si memorare a datelor. Aceste tehnici sunt utilizate în optimizoarele ce intra în componenta sistemelor de interogare ale SGBD-urilor mai sus mentionate.

Cele mai raspândite structuri de indexare au la baza arborii B si tehnicile de dispersie (hash). Procedeele din prima categorie permit realizarea unor algoritmi de cautare logaritmici si implementarea eficienta atât a cererilor de regasire a unei singure înregistrari, cât si a cererilor de determinare a tuturor înregistrarilor care se încadreaza într-un anumit domeniu de cautare (range queries). A doua categorie de metode de indexare necesita un timp constant pentru regasirea unei înregistrari cu o cheie data. Un dezavantaj al metodelor bazate pe dispersie este performanta scazuta în cazul coliziunilor si al depasirii capacitatii paginilor de date.

O clasa de metode cu performante foarte bune în rezolvarea cererilor de cautare a unor informatii în masive mari de date este formata din metodele de indexare multidimensionale. În acest caz înregistrarile bazei de date sunt asimilate cu puncte într-un spatiu de date multidimensional. Acesta este divizat în mod repetat. Diviziunea spatiala poate fi efectuata în raport cu valorile atributelor înregistrarilor bazei de date sau, "hiperplanele de diviziune" ar putea fi alese independent de valorile atributelor. În acest din urma caz, implementarea este mai simpla, dar structurile arborescente de indexare ar putea fi foarte dezechilibrate (în cazul unor înregistrari neuniform distribuite în spatiul datelor).

Similitudinea dintre tuplele unei relatii si punctele unui spatiu de date multidimensional poate sta la baza multor metode de indexare spatiala, multidimensionala. Structurile de indexare multidimensionale pot fi utilizate în cazul unor volume de date mai mici (caz în care structura index este stocata în întregime în memoria interna) sau în cazul unor volume de date foarte mari (de data aceasta structurile sunt stocate în întregime sau numai partial în memoria externa).

Articolul descrie metode de indexare utilizabile în cazul unor volume de date foarte mari prezentând si aspecte referitoare la operatiile de baza asociate structurilor index:

- inserarea unei înregistrari în index;
- stergerea unei înregistrari;
- cautarea unei înregistrari cu cheia data;
- determinarea unui set de înregistrari ale caror atribute sunt situate în intervale specificate.

2. Structuri de indexare bazate pe arbori B

B-arborii (Bucket tree) [Corm 94] sunt arbori de cautare echilibrati, proiectati pentru a permite algoritmi eficienti de regasire a informatiei stocate pe medii de memorie externa cu

acces direct. Proprietatile care definesc un B-arbore de ordin m ($m \geq 2$) sunt:

- fiecare nod intern cu exceptia radacinii are $k \in [\lfloor m/2 \rfloor, m]$ descendenti;
- radacina are minimum doi fii (cu exceptia cazului când arborele are unul sau doua noduri);
- toate frunzele sunt situate pe acelasi nivel, un nod intern (N) cu k fii va contine k-1 chei c_1, c_2, ..., c_{k-1} având valori crescatoare si care sunt folosite pentru a determina intervalele de apartenenta ale valorilor datelor situate respectiv în fiecare dintre cei k subarbori ai lui (N). Aceste intervale sunt $(-\infty, c_1]$,$(c_1, c_2]$, ..., (c_{k-1}, ∞).

Înaltimea unui B-arbore (care ar putea sa indice numarul maxim de accese la disc necesare pentru regasirea unei anumite informatii) este $\log_{\lfloor m/2 \rfloor}((n+1)/2)$ unde n este numarul total de chei ale arborelui. Fiecare frunza a B-arborelui ar corespunde unei colectii de k pagini de date.

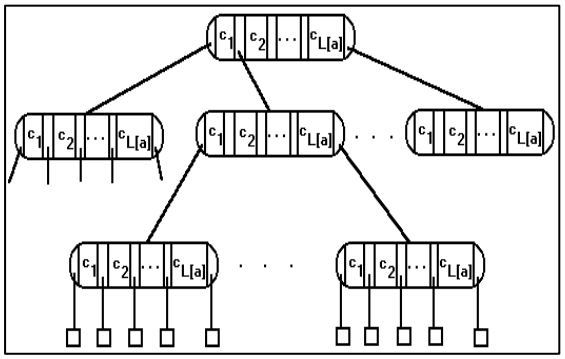

Fig. 1. Arborele B

Cautarea într-un arbore B a unei valori d se face cu:

```
procedura cauta_B(arbore_B a, data d) este
cât timp i≤L[a] si d>c[i] repeta
        i ← i+1
daca i≤L[a] si d=c[i] atunci
        rezultate a, i
daca *a este frunza atunci
        * valoarea d nu exista în arbore
altfel
* citeste pagina de la adresa c[i] (i. e. un nou nod al B_arborelui). Fie PNT(c[i]) adresa acestui nod.
        cauta_B(PNT(c[i]), d)
sfârsit
```

O structura de indexare similara, *arborele B+* este caracterizata prin faptul ca valorile cheilor sunt stocate numai în nodurile frunza ale arborelui. Nodurile interne servesc numai pentru localizarea valorilor cheilor si alcatuiesc componenta de indexare a arborelui B+.

O metoda pentru indexarea datelor punctuale multidimensionale bazata pe arbori B, este *arborele zkdB*. Daca spatiul datelor are dimensiunea "d", metoda consta în asocierea la fiecare punct de date a unui cod (numit z-cod) rezultat prin interclasarea bitilor din reprezentarile binare ale celor "d" coordonate ale punctului si stocarea rezultatelor într-un arbore B. Dezavantajul acestei structuri consta în faptul ca z-codurile trebuie sa aiba lungime fixa. Valoarea acestei lungimi depinde de numarul de biti pe care se reprezinta valoarea unei coordonate a punctului de date. Ea ar trebui predeterminata în functie de numarul total de puncte reprezentate. Acest numar este necunoscut în cazul structurilor dinamice. Valorile lungi ale codurilor z afecteaza si eficienta algoritmilor de rezolvare a interogarilor.

3. Metode de indexare în spatii multi-dimensionale

Arborii k-d. Aceste structuri de date sunt asemanatoare arborilor binari de cautare dar cheia de cautare este dependenta de nivelul la care se afla nodurile arborelui. Principiul de descompunere spatiala care sta la baza construirii arborilor k-d este preluat de numeroase metode de indexare a informatiei dintr-un spatiu multidimensional. Spre exemplu în cazul codificarii printr-un arbore 2-d a unei multimi de puncte din spatiul bidimensional (k=2) daca procesul de cautare ajunge la un nod situat pe un nivel de rang par (radacina arborelui se considera pe nivelul 0) cheia arborelui de cautare va fi coordonata x iar în cazul unui nod situat pe un nivel de rang impar cheia de cautare în arbore va fi coordonata y.

Un nod (N) al unui arbore k-d va fi reprezentat printr-o înregistrare cu k+4 câmpuri. Un câmp contine informatia propriu-zisa a nodului

(info), un altul indica numarul (numele) coordonatei în functie de care se face decizia de cautare în acel nod, doua câmpuri contin referinte la fiii nodului curent iar celelalte k câmpuri reprezinta coordonatele punctului asociat nodului (N).

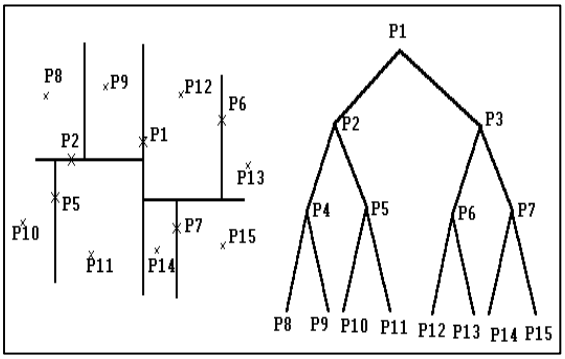

Fig. 2. Arborele k-d (k=2)

Conventia de plasare a punctelor în nodurile unui arbore k-d (A) este ca în subarborele stâng al unui nod arbitrar (N) al lui (A) (având câmpul de decizie cea de-a j-a coordonata a spatiului k-dimensional) sunt plasate toate punctele având valoarea celei de-a j-a coordonate strict mai mica decât aceeasi coordonata a punctului de date asociat lui (N). In subarborele drept vor fi plasate câmpurile cu coordonata a j-a mai mare sau egala decât coordonata corespunzatoare a punctului de date al nodului (N).

În reprezentarea înfatisata mai sus se considera ca punctele colectiei sunt disjuncte. Daca se permite ca punctele de date sa nu fie unice, fiecare nod (N) al arborelui va trebui sa contina un câmp "lista de coliziuni" în care sa fie retinute informatiile referitoare la toate punctele multimii de date care au aceleasi coordonate cu punctul asociat lui (N).

Bentley a demonstrat ca în cazul insertiei unui punct într-un arbore k-d cu N noduri sunt efectuate $O(\log_2 N)$ operatii elementare (comparatii) [Bent 75]. Ca si în cazul arborilor binari de cautare forma arborelui k-d depinde de ordinea în care sunt efectuate inserarile de puncte în arbore.

O solutie performanta de indexare care prezinta caracteristici comune atât cu metodele ierarhice cât si cu cele bazate pe fisiere grila este *arborele BD* cunoscut si sub numele de *fisier BANG* (Balanced and Nested Grid File) [Freest 87].

Arborele BD este un arbore binar asemanator arborelui k-d, caruia i se aplica o tehnica de comprimare a cailor. El prezinta asemanari importante si cu tehnica de cautare patriciana într-un arbore binar, fara a memora cheile în noduri. Tehnica *Patricia* (Practical Algorithm to Retrieve Information Coded in Alphanumeric) a fost elaborata de D.E. Morisson si este descrisa pe larg în [Knuth 76].

Pentru a descrie modul de descompunere spatiala în cazul fisierului BANG se va considera un spatiu de date bidimensional de forma dreptunghiulara. Punctele de date (înregistrarile bazei de date) sunt grupate în pagini. Fiecare pagina contine punctele aflate într-o regiune spatiala dreptunghiulara. Daca prin inserarea unui punct capacitatea unei pagini (pag) va fi depasita, ea va fi divizata în doua pagini. Divizarea se face prin împartirea regiunii spatiale asociate lui (pag) în doua parti egale, printr-o dreapta paralela cu una din axele de coordonate. Daca prin aceasta divizare spatiala capacitatea unei pagini este în continuare depasita, procesul de diviziune ca continua alegându-se alta axa. Procesul de împartire a spatiului datelor, prin aplicarea acestui algoritm poate fi reprezentat sub forma unui arbore binar numit *arbore k-d PR* (figura 3). Nodurile frunza corespund unor pagini de date iar nodurile interne corespund unor valori dupa care se face diviziunea spatiului datelor.

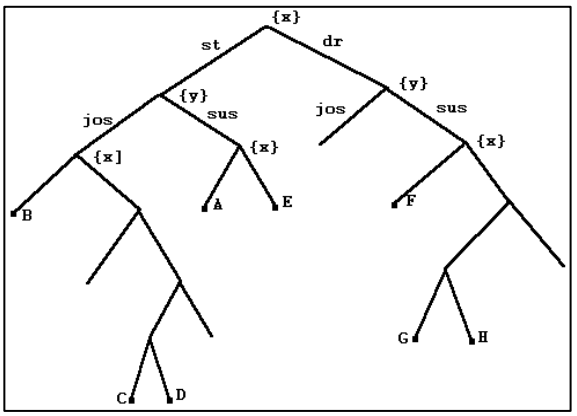

Fig. 3. Arbore k-d PR

În cazul unor colectii de puncte neuniform repartizate, exista posibilitatea obtinerii unor arbori puternic dezechilibrati (ordinea de inserare a punctelor influenteaza forma arborelui). Acesti arbori ar putea contine noduri interne cu un singur fiu nevid. Pentru eliminarea acestor noduri se va asocia fiecarui nod al arborelui k-d PR un cod binar (numit DZE - Discriminator Zone Expression). Acesta consta din cifre 0 sau 1 care indica rezultatele testelor necesare pentru a determina regiunea din spatiul de date în care este amplasat fiul stâng al lui (N). Fisierul BANG va retine numai paginile de date si codurile DZE asociate. Pentru claritate se prezinta mai jos (Figura 3) o colectie de puncte 2D, arborele BD si arborele k-d PR corespunzator (figura 2) (am considerat ca dimensiunea paginii este 1 punct).

Ordinea de inserare a punctelor în arborele BD este A, B, C, D, E, F, G, H. Arborele BD nu este unic, i se pot aplica operatori de rotatie analogi celor folositi pentru echilibrarea arborilor AVL. De obicei o pagina de date contine mai multe puncte de date, fie c capacitatea ei. În acest caz se poate continua procesul de diviziune a spatiului pâna când toate regiunile obtinute vor avea mai putin de x*c (0<x<1) puncte. Factorul de umplere x este ales de obicei 2/3.

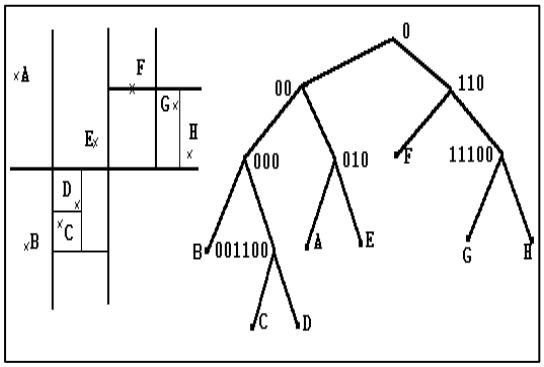

Fig. 4. Arborele k-d PR dupa procesul de comprimare a cailor

Fisierul BANG poate fi utilizat si pentru reprezentarea de date cu numar arbitrar de dimensiuni.

O varianta interesanta de indexare a datelor punctuale, asemanatoare cu arborii k-d sunt *arborii LSD* (Local Split Decision) [Henr 89]. Aceasta metoda de indexare partitioneaza spatiul datelor în celule (regiuni) de forma dreptunghiulara carora le sunt asociate pagini de date de dimensiuni fixe. Hiperplanele care despart spatiul datelor în regiuni pot fi situate în pozitii arbitrare (ramânând normale pe hiperplanele de coordonate).

Catalogul LSD specifica un mod de a partitiona spatiul datelor asemanator celui al arborilor k-d, cu deosebirea ca fiecare nod al arborelui are asociata dimensiunea de-a lungul careia se va face diviziunea spatiului precum si pozitia diviziunii. Deci modul de divizare a spatiului dintr-un nod oarecare al arborelui LSD nu este influentat de deciziile de divizare luate anterior. Henrich, Six si Widmayer descriu un mod de a pagina atât datele punctuale cât si continutul catalogului LSD.

Sa consideram ca b este capacitatea unei pagini de date iar h_p este capacitatea unei pagini din catalogul LSD (numarul de noduri ale arborelui LSD ce pot fi continute de o pagina de catalog). Catalogul LSD este împartit într-o portiune rezidenta în memoria interna si eventual portiuni (subarbori) rezidente în pagini de memorie externa. Algoritmul care decide modul în care nodurile arborelui LSD sunt grupate în pagini externe încearca sa pastreze urmatoarea proprietate (*echilibrare externa*): Numarul de pagini externe de catalog care sunt traversate pe oricare doua cai de la radacina pâna la o pagina de date difera cel mult prin 1.

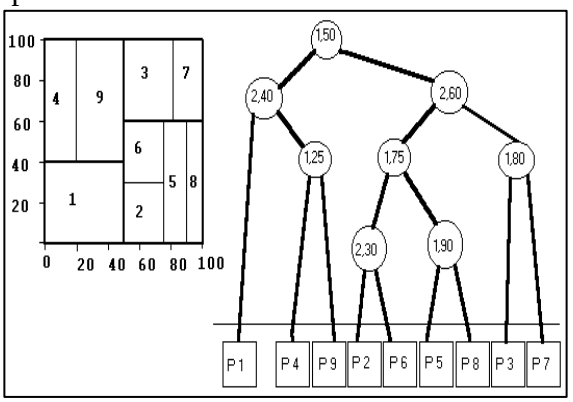

Fig. 5. Metoda de descompunere spatiala caracteristica arborelui LSD

Pe masura inserarii unor noi puncte de date, catalogul LSD creste pâna în momentul când numarul de noduri ale arborelui nu mai poate fi pastrat în zona special alocata din memoria interna. (Se va nota cu n_i numarul de noduri ale arborelui LSD care poate fi retinut în memoria interna - capacitatea "arborelui prefix intern"). In acest moment algoritmul de paginare va determina un subarbore care va fi transferat în memoria externa a.î. sa fie pastrata proprietatea de echilibrare externa. Prin inserarea unui nou punct de date acesta va fi plasat într-o pagina de date iar în momentul depasirii capacitatii acestei pagini (b) va fi aplicat un algoritm de diviziune a paginilor de date. Evident, prin aparitia unei noi pagini de date, va rezulta în catalogul LSD un pointer care sa o refere (si o data cu el un nou nod în arborele LSD).

Strategiile de divizare a datelor dintr-o pagina pot fi grupate în doua categorii: strategii dependente de pozitia punctelor stocate în regiunea asociata paginii care trebuie divizata (de exemplu se împarte spatiul dupa valoarea mediana a valorilor unei anumite coordonate ale acestor puncte) si strategii ce depind de modul de distributie spatiala (uniforma sau nu) a punctelor din pagina. De exemplu, regiunea asociata paginii de divizat se împarte în doua subregiuni de arii egale. In urma procesului de diviziune punctele de date din fiecare dintre subregiuni vor fi plasate în pagini separate.

In cazul când numarul de noduri din catalogul arborescent LSD (notat în continuare cu T) depaseste numarul maxim de noduri care pot fi pastrate în memoria interna, un subarbore al lui T va fi evacuat într-o pagina de catalog din memoria secundara. Informatia continuta într-o pagina de catalog este organizata însiruind secvential nodurile sub forma unui heap având o înaltime maxima fixata h_p; ea va fi divizata în momentul când înaltimea subarborelui asociat va deveni mai mare decât h_p. Algoritmul de divizare a unei pagini de catalog ce contine un subarbore LSD (fie el (A)) consta în scrierea subarborilor stâng respectiv drept ai radacinii lui (A) pe pagini separate de catalog si inserarea radacinii lui (A) în catalogul T.

Este prezentat mai jos algoritmul de inserare a unui nod Q într-un arbore LSD (T) considerând ca portiunea interna (arborele prefix intern) T_i poate avea un numar de maxim n_i-1 noduri:

daca * tatal P al nodului Q e un nod din memoria interna atunci
daca * numarul de noduri interne este < n_i-1 atunci
    * insereaza Q în T_i // T_i=arb. prefix intern
altfel
    * insereaza Q în T_i
    * apeleaza algoritmul de paginare
altfel
/* tatal P al noului nod Q este plasat într-un subarbore T_p al lui T stocat într-o pagina externa */
daca * dupa inserarea lui Q $h(T_p)<h_p$ atunci *sfârsit
altfel
    * apeleaza algoritmul de divizare a unei pagini de catalog pentru T_p

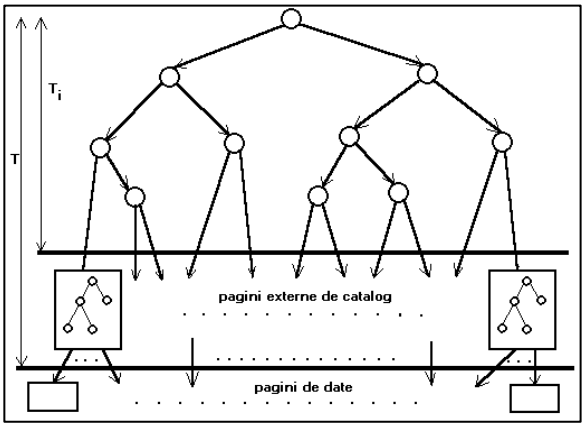

Fig. 6. Structura indexului LSD

Algoritmul de paginare este apelat daca prin inserarea unui nou nod se va depasi capacitatea maxima a arborelui prefix intern T_i. În acest caz se încearca determinarea unui subarbore T_s al lui T_i, pentru a fi evacuat pe mediul extern de stocare a informatiei. T_s trebuie sa satisfaca urmatoarele proprietati:

- sa aiba înaltimea $h(T_s)<h_p$ (*)
- parcurgerea oricarei cai de la nodul radacina al lui T_s si pâna la o pagina de date trebuie sa se faca cu un numar minim posibil de accese la pagini externe de catalog. (**)

În cazul când exista în T_i mai multi subarbori T_s care sa îndeplineasca proprietatile anterior mentionate va fi ales cel cu un numar de noduri maxim. In cele ce urmeaza vom nota cu

NMIN(v) numarul minim de accese la pagini de catalog necesar pentru a parcurge o cale de la radacina lui T, care trece prin nodul v si ajunge la o frunza a arborelui LSD si cu NMAX(v) numarul maxim analog. Fie de asemeni S(v) numarul de noduri al unui arbore T_s maximal care satisface proprietatile (*) si (**) si e dominat de nodul v. h(v) este înaltimea subarborelui din T_i care are radacina în v. Algoritmul de paginare este descris prin urmatoarea functie recursiva care întoarce nodul radacina al unui subarbore T_s al lui T_i.

```
functia paginare(w:arboreLSD) întoarce arboreLSD
daca NMIN(FiuStâng(w))≠NMIN(FiuDrept(w)) atunci
        r ← * fiul lui w cu cel mai mic NMIN
altfel
        r ← * fiul lui w cu cel mai mare S(w)
daca h(r)≤hp si NMIN(r)=NMAX(r) atunci
        * întoarce r
altfel
        * întoarce paginare(r)
sfârsit
```

Pentru a rezolva o cerere de determinare a tuturor punctelor aflate într-o regiune dreptunghiulara (Q), se traverseaza arborele LSD pentru a determina toate paginile ale caror regiuni spatiale intersecteaza pe (Q). Fie w un nod oarecare al arborelui LSD si D(w) regiunea ce corespunde tuturor punctelor situate în pagini dominate de w. Daca notam cu w->fs, fiul stâng al nodului w, regiunea dreptunghiulara D(w->fs) se determina cu usurinta pornind de la D(w) si informatia din nodul W care indica dimensiunea de-a lungul careia s-a efectuat diviziunea spatiala precum si pozitia diviziunii. Este vorba de un algoritm de decupare a unui hiperdreptunghi de catre un hiperplan paralel cu unul din hiperplanele de coordonate. În continuare este prezentata procedura pentru rezolvarea "cererii de domeniu dreptunghiular" în cazul unui index ierarhic de tip arbore LSD.

```
procedura orq(arboreLSD a, dreptunghi Q) este
daca * a este frunza atunci
        *verifica punctele din pagina asociata
altfel
        daca Q ∩ D(a->fd) = Φ atunci
                orq(a->fs, Q)
        altfel
                daca Q ∩ D(a->fs) = Φ atunci
                        orq(a->fd, D)
                altfel
                        orq(a->fs, D)
                        orq(a->fd, D)
sfârsit
```

O metoda de indexare neierarhica des utilizata este *fisierul grila* (grid file), ea a fost prezentata de Nievergelt si Hinterberger. Grila este o structura de date analoaga unui tablou bi- (sau multi-) dimensional si este stocata pe disc sub forma unui fisier catalog. Fiecare componenta a tabloului contine lista punctelor din interiorul unei singure celule a diviziunii spatiale. Este folosit un fisier catalog care este alcatuit din codificari ale blocurilor de grila; acestea corespund unor regiuni disjuncte de forma hiperdreptunghiulara care acopera întregul spatiu al datelor. Toate punctele de date situate într-un anumit bloc al grilei vor fi stocate într-o aceeasi pagina de date. Este posibil ca mai multe blocuri de grila (adiacente si care prin juxtapunere trebuie sa formeze un hiperdreptunghi) sa corespunda aceleiasi pagini de date. În interiorul unei pagini informatia poate fi eventual organizata sub forma unei liste, arbore etc.

Catalogul grila are doua parti: prima (de obicei stocata în memoria externa) este un tablou k-dimensional având câte o intrare asociata fiecarui bloc de date. Un element al acestui tablou este de fapt un pointer la pagina de date corespunzatoare. A doua este o multime de k vectori unidimensionali, numiti scale liniare, care sunt retinuti în memoria interna. Acestia stabilesc o partitionare în subintervale a domeniului de valori ale fiecarui atribut (coordonata) al punctului de date.

Folosind aceasta schema orice punct poate fi regasit prin doua accese la disc unul la fisierul catalog celalalt la pagina de date. Timpul de raspuns la o cerere de domeniu este redus desi uneori marimea blocurilor de grila (a proiectiilor lor pe hiperaxele de coordonate) nu coincide cu domeniul specificat în cerere. Sa consideram cazul unor inserari repetate de puncte de date într-o colectie memorata folosind un fisier grila. Se pot ivi doua situatii care sa necesite modificarea grilei sau alocarea de noi pagini de date. Prima apare daca mai multe blocuri de grila (pointeri de catalog) re-

fera o aceeasi pagina de date, a carei capacitate a fost depasita.

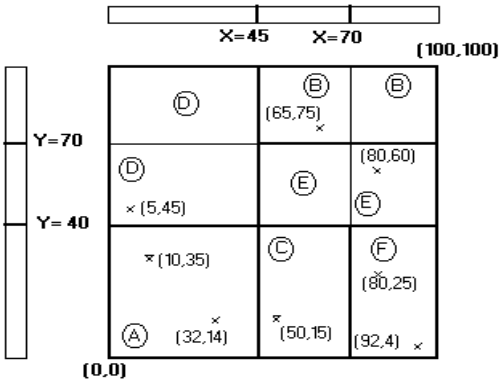

Fig. 7. Descompunerea spatiala în cazul catalogului grila

În acest caz se aloca o noua pagina si se modifica corespondentele (bloc grila - pagina), continute în catalogul fisierului grila. A doua situatie apare daca prin inserare s-a depasit capacitatea unei pagini (P) ale carei puncte apartin în totalitate unui acelasi bloc (B) al grilei. În acest caz respectivul bloc va trebui divizat (printr-un hiperplan normal unei axe de coordonate) în raport cu una din cele k dimensiuni ale spatiului datelor. Evident se va face si o alocare a unei noi pagini de date (P1) si redistribuirea punctelor din (P) între (P) si (P1) în raport cu pozitia lor în regiunile ce corespund blocurilor rezultate prin diviziunea lui (B).

Nievergelt recomanda ca politica de divizare a spatiului, alegerea ciclica a coordonatei de diviziune $1 \le i \le k$ _i efectuarea taieturii de-a lungul medianei valorilor coordonatei x_i ce corespunde punctelor din pagina de divizat. O alta politica de divizare ar favoriza unele coordonate fata de altele, aceasta politica ar putea fi folosita în cazul când cele mai multe din interogarile care vor fi efectuate asupra fisierului grila se refera la un anumit atribut. Scala respectivului atribut va avea o granularitate mai fina.

Dualul procesului de divizare este fuzionarea. Ea poate apare în cazul când o pagina de date devine vida sau contine foarte putine elemente (eventual în urma unei operatii de stergere). Pot exista fuzionari de pagini de date (care corespund unui acelasi bloc) sau de blocuri. Acest din urma caz poate apare daca se doreste comprimarea continutului fisierului grila sau daca se schimba granularitatea de reprezentare a unor atribute din cauza modificarii frecventei interogarilor care se refera la acele atribute.

Bibliografie

[Bent 75] J. L. Bentley, Multidimensional binary search trees used for associative searching, Communications of the ACM, septembrie, 1975.

[Catt 94] R. G. Cattell, Object Data Management, Addison Wesley Publishing Company, 1994.

[Corm 94] T. Cormen, Ch. Leiserson, D. Rivest, Introduction a l'Algorithmique, Ed. Dunod, Paris, 1994

[Freest 87] M. Freeston, The BANG file: A New Kind of Grid File, Proceedings of SIGMOD Conference, San Francisco, Mai 1987.

[Henr 89] A, Henrich, H. W: Six, P. Widmayer, The LSD tree: Spatial Access to Multidimensional Point and Nonpoint Objects, Proceedings of the Fifteenth International Conference on Very Large Databases, 1989.

[Khos 96] S. Khoshafian, A. B. Baker, Multimedia and Imaging Databases, Morgan Kaufmann Publishers Inc., 1996.

[Knuth 76] D. E. Knuth, Tratat de Programare a Calculatoarelor, Sortare si Cautare, Editura Tehnica, Bucuresti, 1976.

[Niev 84] J. Nievergelt, H. Hinterberger, K. C. Sevcik, The grid file: an adaptable symmetric multikey file structure, ACM Transactions on Database Systems, 1984.

[Over 82] M. H. Overmars, J. van Leuwen, Dynamic multidimensional data structures based on quad and k-d trees, Acta Informatica, Nr. 3, 1982.

[Zah 97] M. Zaharia, Aspects Concerning the Implementation of a Multidimensional Index Structure – the BANG file, The 11-th Conference on Control Systems and Computer Science, Bucharest, 1997.

[Zah 98] M. Zaharia, Elemente de Geometrie Algoritmica, Editura Printech, 1998.