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

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

Lucrarea prezintã modalitãti de indexare a informatiei din bazele de date pentru prelucrãri grafice. Sunt înfãtisate metode de indexare bazate pe arbori B, metode de indexare (ierarhicã sau neierarhicã) a informatiei din spatii multidimensionale.
***Cuvinte cheie**: indexare, arbore B, arbore B+, arbore k-d, index LSD, fisier BANG, fisier grilã.*

1. Introducere

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

Cele mai rãspândite structuri de indexare au la bazã arborii B si tehnicile de dispersie (hash). Procedeele din prima categorie permit realizarea unor algoritmi de cãutare logaritmici si implementarea eficientã atât a cererilor de regãsire a unei singure înregistrãri cât si a cererilor de determinare a tuturor înregistrãrilor care se încadreazã într-un anumit domeniu de cãutare (range queries). A doua categorie de metode de indexare necesitã un timp constant pentru regãsirea unei înregistrãri cu o cheie datã. Un dezavantaj al metodelor bazate pe dispersie este performanta scãzutã în cazul coliziunilor si al depãsirii capacitãtii paginilor de date.

O clasã de metode cu performante foarte bune în rezolvarea cererilor de cãutare a unor informatii în masive mari de date este formatã din metodele de indexare multidimensionale. In acest caz înregistrãrile bazei de date sunt asimilate cu puncte într-un spatiu de date multidimensional. Acest spatiu este divizat în mod repetat. Diviziunea spatialã poate fi efectuatã în raport cu valorile atributelor înregistrãrilor bazei de date sau, "hiperplanele de diviziune" ar putea fi alese independent de valorile atributelor. In acest din urmã caz implementarea este mai simplã dar structurile arborescente de indexare ar putea fi foarte dezechilibrate (In cazul unor înregistrãri 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 spatialã, multidimensionalã. Structurile de indexare multidimensionale pot fi utilizate în cazul unor volume de date mai mici (caz în care structura index este stocatã în întregime în memoria internã) sau în cazul unor volume de date foarte mari (de data aceasta structurile sunt stocate în întregime sau numai partial în memoria externã).

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

- inserarea unei înregistrãri în index
- stergerea unei înregistrãri
- cãutarea unei înregistrãri cu cheia datã
- determinarea unui set de înregistrãri ale cãror atribute sunt situate în intervale specificate

2. **Structuri de indexare bazate pe arbori B**

B-arborii (Bucket tree) [Corm 94] sunt arbori de cãutare echilibrati, proiectati pentru a permite algoritmi eficienti de regãsire a informatiei stocate pe medii de memorie externã cu acces direct. Proprietãtile care definesc un B-arbore de ordin m ($m \geq 2$) sunt:

- fiecare nod intern cu exceptia rãdãcinii are $k \in [\lfloor m/2 \rfloor, m]$ descendenti;
- rãdãcina are minimum doi fii (cu exceptia cazului când arborele are unul sau douã 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 crescãtoare si care sunt folosite pentru a determina intervalele de apartenentã ale valorilor datelor situate respectiv în fiecare dintre cei k subarbori ai lui (N). Aceste intervale sunt $(-\infty, c_1], (c_1, c_2], ..., (c_{k-1}, \infty)$.

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

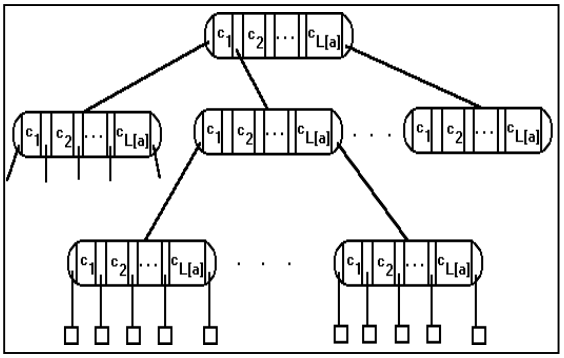

Fig. 1 – Arborele B

Cãutarea 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] repetã
        i ← i+1
□
dacã 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 structurã de indexare similarã, *arborele B+* este caracterizatã prin faptul cã valorile cheilor sunt stocate numai în nodurile frunzã ale arborelui. Nodurile interne servesc numai pentru localizarea valorilor cheilor si alcãtuiesc componenta de indexare a arborelui B+.

O metodã pentru indexarea datelor punctuale multidimensionale bazatã pe arbori B, este *arborele zkdB*. Dacã spatiul datelor are dimensiunea “d”, metoda constã în asocierea la fiecare punct de date a unui cod (numit z-cod) rezultat prin interclasarea bitilor din reprezentãrile binare ale celor “d” coordonate ale punctului si stocarea rezultatelor într-un arbore B. Dezavantajul acestei structuri constã în faptul cã z-codurile trebuie sã aibã lungime fixã. Valoarea acestei lungimi depinde de numãrul de biti pe care se reprezintã valoarea unei coordonate a punctului de date. Ea ar trebui predeterminatã în functie de numãrul total de puncte reprezentate. Acest numãr este necunoscut în cazul structurilor dinamice. Valorile lungi ale codurilor z afecteazã si eficienta algoritmilor de rezolvare a interogãrilor.

3. **Metode de indexare in spatii multidimensionale**

Arborii k-d. Aceste structuri de date sunt asemãnãtoare arborilor binari de cãutare dar

cheia de cãutare este dependentã de nivelul la care se aflã nodurile arborelui. Principiul de descompunere spatialã care stã la baza construirii arborilor k-d este preluat de numeroase metode de indexare a informatiei dintr-un spatiu multidimensional. Spre exemplu în cazul codificãrii printr-un arbore 2-d a unei multimi de puncte din spatiul bidimensional (k=2) dacã procesul de cãutare ajunge la un nod situat pe un nivel de rang par (rãdãcina arborelui se considerã pe nivelul 0) cheia arborelui de cãutare va fi coordonata x iar în cazul unui nod situat pe un nivel de rang impar cheia de cãutare 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-zisã a nodului (info), un altul indicã numãrul (numele) coordonatei în functie de care se face decizia de cãutare în acel nod, douã câmpuri contin referinte la fiii nodului curent iar celelalte k câmpuri reprezintã coordonatele punctului asociat nodului (N).

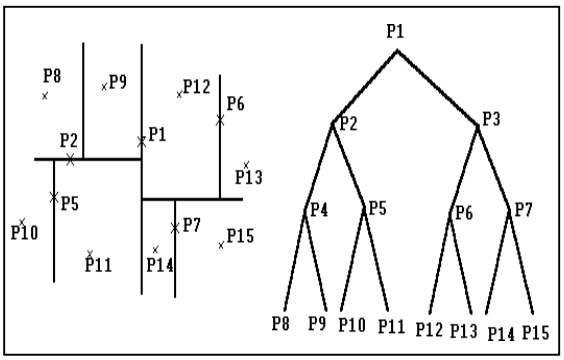

Fig. 2 – Arborele k-d (k=2)

Conventia de plasare a punctelor în nodurile unui arbore k-d (A) este cã în subarborele stâng al unui nod arbitrar (N) al lui (A) (având câmpul de decizie cea de-a j-a coordonatã a spatiului k-dimensional) sunt plasate toate punctele având valoarea celei de-a j-a coordonate strict mai micã decât aceeasi coordonatã a punctului de date asociat lui (N). In subarborele drept vor fi plasate câmpurile cu coordonata a j-a mai mare sau egalã decât coordonata corespunzãtoare a punctului de date al nodului (N).
In reprezentarea înfãtisatã mai sus se considerã cã punctele colectiei sunt disjuncte. Dacã se permite ca punctele de date sã nu fie unice, fiecare nod (N) al arborelui va trebui sã continã un câmp "listã de coliziuni" în care sã fie retinute informatiile referitoare la toate punctele multimii de date care au aceleasi coordonate cu punctul asociat lui (N).
Bentley a demonstrat cã 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 cãutare forma arborelui k-d depinde de ordinea în care sunt efectuate inserãrile de puncte în arbore.

O solutie performantã de indexare care prezintã caracteristici comune atât cu metodele ierarhice cât si cu cele bazate pe fisiere grilã este *arborele BD* cunoscut si sub numele de *fisier BANG* (Balanced and Nested Grid File) [Freest 87].
Arborele BD este un arbore binar asemãnãtor arborelui k-d, cãruia i se aplicã o tehnicã de comprimare a cãilor. El prezintã asemãnãri importante si cu tehnica de cãutare patricianã într-un arbore binar, fãrã a memora cheile în noduri. Tehnica *Patricia* (Practical Algorithm to Retrieve Information Coded in Alphanumeric) a fost elaboratã de D.E. Morisson si este descrisã pe larg în [Knuth 76].
Pentru a descrie modul de descompunere spatialã în cazul fisierului BANG se va considera un spatiu de date bidimensional de formã dreptunghiularã. Punctele de date (înregistrãrile bazei de date) sunt grupate în pagini. Fiecare paginã contine punctele aflate într-o regiune spatialã dreptunghiularã. Dacã prin inserarea unui punct capacitatea unei pagini (pag) va fi depãsitã, ea va fi divizatã în douã pagini. Divizarea se face prin împãrtirea regiunii spatiale asociate lui (pag) în douã pãrti egale, printr-o dreaptã paralelã cu una din axele de coordonate. Dacã prin aceastã divizare spatialã capacitatea unei pagini este

în continuare depãsitã, procesul de diviziune ca continua alegându-se altã axã. Procesul de împãrtire a spatiului datelor, prin aplicarea acestui algoritm poate fi reprezentat sub forma unui arbore binar numit *arbore k-d PR* (Fig. 2). Nodurile frunzã corespund unor pagini de date iar nodurile interne corespund unor valori dupã care se face diviziunea spatiului datelor.
In cazul unor colectii de puncte neuniform repartizate, existã posibilitatea obtinerii unor arbori puternic dezechilibrati (ordinea de inserare a punctelor influenteazã forma arborelui). Acesti arbori ar putea contine noduri interne cu un singur fiu nevid. Pentru eliminarea acestor noduri se va asocia fiecãrui nod al arborelui k-d PR un cod binar (numit DZE - Discriminator Zone Expression). Acesta constã din cifre 0 sau 1 care indicã 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 prezintã mai jos (Figura 3) o colectie de puncte 2D, arborele BD si arborele k-d PR corespunzãtor (Fig. 2) (am considerat cã dimensiunea paginii este 1 punct).

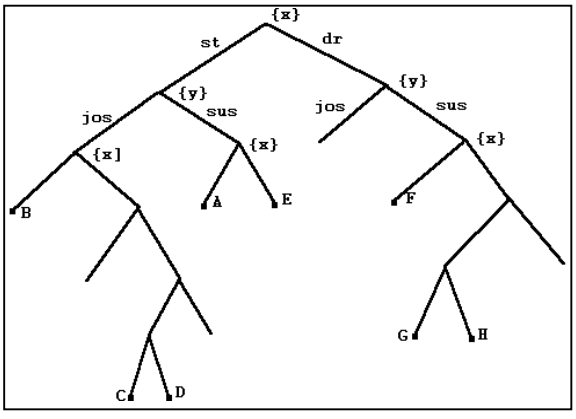

Fig. 2 – Arbore k-d PR

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 paginã de date contine mai multe puncte de date, fie c capacitatea ei. In acest caz se poate continua procesul de diviziune a spatiului pânã 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.

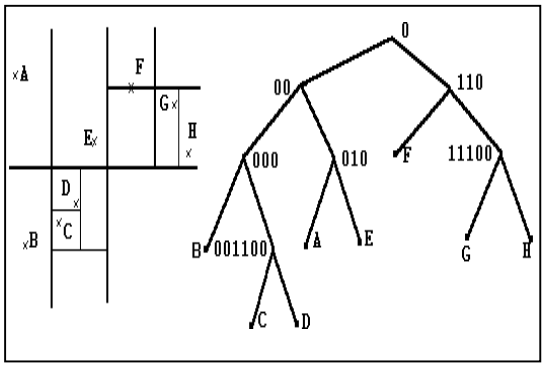

Fig. 3 – Arborele k-d PR dupã procesul de comprimare a cãilor

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

O variantã interesantã de indexare a datelor punctuale, asemãnãtoare cu arborii k-d sunt *arborii LSD* (Local Split Decision) [Henr 89]. Aceastã metodã de indexare partitioneazã spatiul datelor în celule (regiuni) de formã dreptunghiularã cãrora le sunt asociate pagini de date de dimensiuni fixe. Hiperplanele care despart spatiul datelor în regiuni pot fi situate în pozitii arbitrare (rãmânând normale pe hiperplanele de coordonate).
Catalogul LSD specificã un mod de a partitiona spatiul datelor asemãnãtor celui al arborilor k-d, cu deosebirea cã fiecare nod al arborelui are asociatã dimensiunea de-a lungul cãreia 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.
Sã considerãm cã b este capacitatea unei pagini de date iar h_p este capacitatea unei pagini din catalogul LSD (numãrul de

noduri ale arborelui LSD ce pot fi continute de o paginã de catalog). Catalogul LSD este împãrtit într-o portiune rezidentã în memoria internã si eventual portiuni (subarbori) rezidente în pagini de memorie externã. Algoritmul care decide modul în care nodurile arborelui LSD sunt grupate în pagini externe încearcã sã pãstreze urmãtoarea proprietate (*echilibrare externã*): Numãrul de pagini externe de catalog care sunt traversate pe oricare douã cãi de la rãdãcinã pânã la o paginã de date diferã cel mult prin 1.

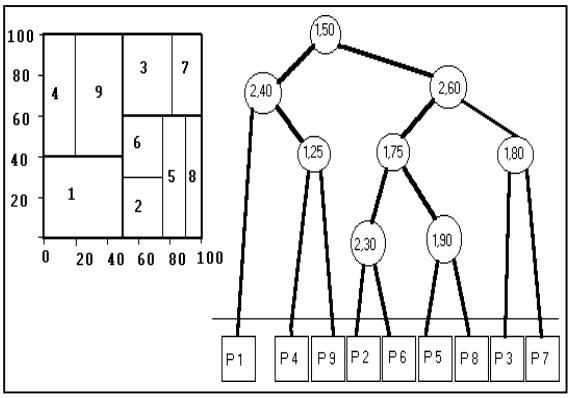

Fig. 4 – Metoda de descompunere spatialã caracteristicã arborelui LSD

Pe mãsura inserãrii unor noi puncte de date, catalogul LSD creste pânã în momentul când numãrul de noduri ale arborelui nu mai poate fi pãstrat în zona special alocatã din memoria internã. (Se va nota cu n_i numãrul de noduri ale arborelui LSD care poate fi retinut în memoria internã - capacitatea "arborelui prefix intern"). In acest moment algoritmul de paginare va determina un subarbore care va fi transferat în memoria externã a.î. sã fie pãstratã proprietatea de echilibrare externã. Prin inserarea unui nou punct de date acesta va fi plasat într-o paginã de date iar în momentul depãsirii capacitãtii 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 sã o refere (si o datã cu el un nou nod în arborele LSD).
Strategiile de divizare a datelor dintr-o paginã pot fi grupate în douã categorii: strategii dependente de pozitia punctelor stocate în regiunea asociatã paginii care trebuie divizatã (de exemplu se împarte spatiul dupã valoarea medianã a valorilor unei anumite coordonate ale acestor puncte) si strategii ce depind de modul de distributie spatialã (uniformã sau nu) a punctelor din paginã. De exemplu, regiunea asociatã paginii de divizat se împarte în douã 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 numãrul de noduri din catalogul arborescent LSD (notat în continuare cu T) depãseste numãrul maxim de noduri care pot fi pãstrate în memoria internã, un subarbore al lui T va fi evacuat într-o paginã de catalog din memoria secundarã. Informatia continutã într-o paginã de catalog este organizatã însiruind secvential nodurile sub forma unui heap având o înãltime maximã fixatã h_p; ea va fi divizatã în momentul când înãltimea 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)) constã în scrierea subarborilor stâng respectiv drept ai rãdãcinii lui (A) pe pagini separate de catalog si inserarea rãdãcinii lui (A) în catalogul T.
Este prezentat mai jos algoritmul de inserare a unui nod Q într-un arbore LSD (T) considerând cã portiunea internã (arborele prefix intern) T_i poate avea un numãr de maxim n_i-1 noduri:

dacã * tatãl P al nodului Q e un nod din memoria internã atunci
dacã * numãrul de noduri interne este < n_i-1 atunci
* inserează Q în T_i // T_i=arb. prefix intern
altfel
* inserează Q în T_i
* apelează algoritmul de paginare
□
altfel
/* tatãl P al noului nod Q este plasat într-un subarbore T_p al lui T stocat într-o paginã externã */
dacã * dupã inserarea lui Q h(T_p)<h_p atunci *sfârsit
altfel
* apelează algoritmul de divizare a unei pagini de catalog pentru T_p
□
□

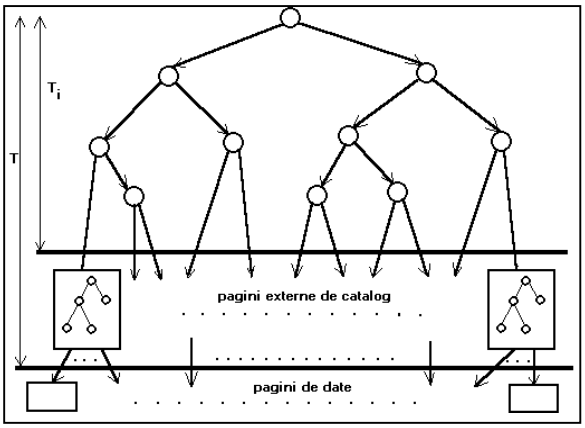

Fig. 5 – Structura indexului LSD

Algoritmul de paginare este apelat dacã prin inserarea unui nou nod se va depãsi capacitatea maximã a arborelui prefix intern T_i. In acest caz se încearcã determinarea unui subarbore T_s al lui T_i, pentru a fi evacuat pe mediul extern de stocare a informatiei. T_s trebuie sã satisfacã urmãtoarele proprietãti:

- sã aibã înãltimea $h(T_s)<h_p$ (*)
- parcurgerea oricãrei cãi de la nodul rãdãcinã al lui T_s si pânã la o paginã de date trebuie sã se facã cu un numãr minim posibil de accese la pagini externe de catalog. (**)

In cazul când existã în T_i mai multi subarbori T_s care sã îndeplineascã proprietãtile anterior mentionate va fi ales cel cu un numãr de noduri maxim. In cele ce urmeazã vom nota cu NMIN(v) numãrul minim de accese la pagini de catalog necesar pentru a parcurge o cale de la rãdãcina lui T, care trece prin nodul v si ajunge la o frunzã a arborelui LSD si cu NMAX(v) numãrul maxim analog. Fie de asemeni S(v) numãrul de noduri al unui arbore T_s maximal care satisface proprietãtile (*) si (**) si e dominat de nodul v. h(v) este înãltimea subarborelui din T_i care are rãdãcina în v. Algoritmul de paginare este descris prin urmãtoarea functie recursivã care întoarce nodul rãdãcinã al unui subarbore T_s al lui T_i.

```
functia paginare(w:arboreLSD) întoarce arboreLSD

dacã      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)
□
dacã 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 dreptunghiularã (Q), se traverseazã arborele LSD pentru a determina toate paginile ale cãror regiuni spatiale intersectează pe (Q). Fie w un nod oarecare al arborelui LSD si D(w) regiunea ce corespunde tuturor punctelor situate în pagini dominate de w. Dacã notãm cu w->fs, fiul stâng al nodului w, regiunea dreptunghiularã D(w->fs) se determinã cu usurintã pornind de la D(w) si informatia din nodul W care indicã dimensiunea de-a lungul cãreia s-a efectuat diviziunea spatialã precum si pozitia diviziunii. Este vorba de un algoritm de decupare a unui hiperdreptunghi de cãtre un hiperplan paralel cu unul din hiperplanele de coordonate. In continuare este prezentatã procedura pentru rezolvarea "cererii de domeniu dreptunghiular" în cazul unui index ierarhic de tip arbore LSD.

```
procedura orq(arboreLSD a, dreptunghi Q) este

dacã * a este frunzã atunci
        *verificã punctele din pagina asociatã
altfel
        dacã Q ∩ D(a->fd) = Φ atunci
                orq(a->fs, Q)
        altfel
                dacã Q ∩ D(a->fs) = Φ atunci
                        orq(a->fd, D)
                altfel
                        orq(a->fs, D)
                        orq(a->fd, D)
                □
        □
□
sfârsit
```

O metodã de indexare neierarhicã des utilizatã este *fisierul grilã* (grid file), ea a

fost prezentatã de Nievergelt si Hinterberger. Grila este o structurã de date analoagã unui tablou bi- (sau multi-) dimensional si este stocatã pe disc sub forma unui fisier catalog. Fiecare componentã a tabloului contine lista punctelor din interiorul unei singure celule a diviziunii spatiale. Este folosit un fisier catalog care este alcãtuit din codificări ale blocurilor de grilã; acestea corespund unor regiuni disjuncte de formã hiperdreptunghiularã care acoperã întregul spatiu al datelor. Toate punctele de date situate într-un anumit bloc al grilei vor fi stocate într-o aceeasi paginã de date. Este posibil ca mai multe blocuri de grilã (adiacente si care prin juxtapunere trebuie sã formeze un hiperdreptunghi) sã corespundã aceleiasi pagini de date. In interiorul unei pagini informatia poate fi eventual organizatã sub forma unei liste, arbore, etc.

Catalogul grilã are douã pãrti: prima (de obicei stocatã în memoria externã) este un tablou k-dimensional având câte o intrare asociatã fiecãrui bloc de date. Un element al acestui tablou este de fapt un pointer la pagina de date corespunzãtoare. A doua este o multime de k vectori unidimensionali, numiti scale liniare, care sunt retinuti în memoria internã. Acestia stabilesc o partitionare în subintervale a domeniului de valori ale fiecãrui atribut (coordonatã) al punctului de date.

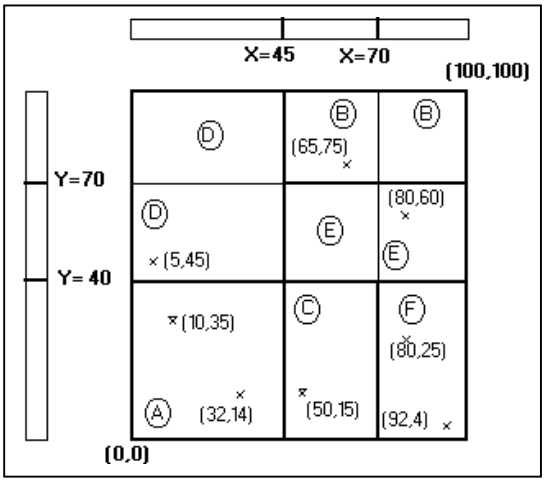

Fig. 6 – Descompunerea spatialã în cazul catalogului grilã

Folosind aceastã schemã orice punct poate fi regãsit prin douã accese la disc unul la fisierul catalog celãlalt la pagina de date. Timpul de rãspuns la o cerere de domeniu este redus desi uneori mãrimea blocurilor de grilã (a proiectiilor lor pe hiperaxele de coordonate) nu coincide cu domeniul specificat în cerere. Sã considerãm cazul unor inserări repetate de puncte de date într-o colectie memoratã folosind un fisier grilã. Se pot ivi douã situatii care sã necesite modificarea grilei sau alocarea de noi pagini de date. Prima apare dacã mai multe blocuri de grilã (pointeri de catalog) referã o aceeasi paginã de date, a cãrei capacitate a fost depãsitã. In acest caz se alocã o nouã paginã si se modificã corespondentele (bloc grilã - paginã), continute în catalogul fisierului grilã. A doua situatie apare dacã prin inserare s-a depãsit capacitatea unei pagini (P) ale cãrei puncte apartin în totalitate unui acelasi bloc (B) al grilei. In 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 recomandã ca politicã de divizare a spatiului, alegerea ciclicã a coordonatei de diviziune $1 \le i \le k$ _i efectuarea tãieturii de-a lungul medianei valorilor coordonatei x_i ce corespunde punctelor din pagina de divizat. O altã politicã de divizare ar favoriza unele coordonate fatã de altele, aceastã politicã ar putea fi folositã în cazul când cele mai multe din interogãrile care vor fi efectuate asupra fisierului grilã se referã la un anumit atribut. Scala respectivului atribut va avea o granularitate mai finã.

Dualul procesului de divizare este fuzionarea. Ea poate apare în cazul când o paginã de date devine vidã sau contine foarte putine elemente (eventual în urma unei operatii de stergere). Pot exista fuzionări de pagini de date (care corespund

unui acelasi bloc) sau de blocuri. Acest din urmã caz poate apare dacã se doreste comprimarea continutului fisierului grilã sau dacã se schimbã granularitatea de reprezentare a unor atribute din cauza modificãrii frecventei interogãrilor care se referã 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., San Francisco, 1996.

[Knuth 76] D. E. Knuth, Tratat de Programare a Calculatoarelor, Sortare si Cãutare, Editura Tehnicã, 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, martie 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 Algoritmicã, Editura Printech, 1998.