Tema 3 – Structuri de date (seria CB)
Arbori de partiționare

Responsabili tema:	Emanuela Haller, Irina Mocanu
Data publicarii:	**7.05.2015**
Termenul de predare:	**21.05.2015 ora 23:55** Se accepta teme trimise cu penalizare 10 puncte /zi (din max 100 puncte) până la data 24.05.2015 ora 23:55.

1. Introducere

Un arbore de partiționare binară a spațiului reprezintă o structură de date obținută prin divizarea recursivă a spațiului în semispații, folosind hiperplane de poziție și orientare cunoscute. Arborele reprezintă o subdivizare ierarhică a unui spațiu n dimensional în semispații convexe. Nodurile interne ale arborelui au asociate hiperplane de partiționare, iar cei doi fii corespund celor două semispații. Într-un spațiu bidimensional, hiperplanul este reprezentat de o dreaptă, iar într-un spațiu tridimensional, hiperplanul este reprezentat de un plan.

În literatura de specialitate, acest tip de arbore este referit ca arbore BSP (binary space partitioning) [1]. Arborii BSP au fost dezvoltați pentru optimizarea operațiilor în grafica 3D. Structura unui arbore BSP permite accesarea in timp liniar a informațiilor legate de poziționarea obiectelor în scenă, cum ar fi ordonarea acestora din perspectiva unui observator.

În cadrul temei curente, ne vom limita interesul asupra spațiilor bidimensionale, considerând o variantă simplificată a arborilor BSP. În acest context, hiperplanele de separație vor fi reprezentate de drepte, care vor fi definite utilizând ecuația dreptei:

$$ax + by + c = 0$$

O dreaptă $d: ax + by + c = 0$ va diviza planul $\mathcal{P}$ în două semiplane deschise:

$$\begin{cases} \mathcal{P}_1: \{P(x,y) |\ ax + by + c < 0\} \\ \mathcal{P}_2: \{P(x,y) |\ ax + by + c > 0\} \end{cases}$$

Considerând trei drepte în plan, $d_1: a_1x + b_1y + c_1 = 0$, $d_2: a_2x + b_2y + c_2 = 0$ și $d_3: a_3x + b_3y + c_3 = 0$, acestea pot diviza un plan în șapte regiuni $\mathcal{P}_i\ i = \overline{1,7}$. Punctele marcate în Figura 1 ($A_i, i = \overline{1,7}$), sunt exemple de puncte din regiunile determinate de cele trei drepte. Fiind date cele trei drepte, cele șapte puncte pot fi considerate ca identificatori ai regiunilor formate.

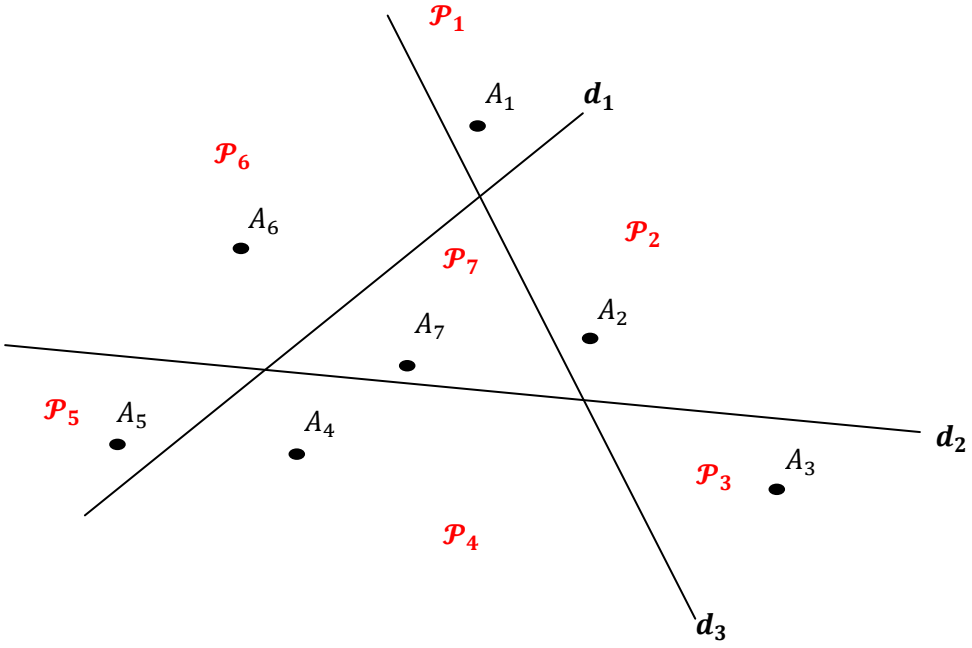

Figura 1. Divizarea planului pe baza dreptelor d_1, d_2, d_3

În continuare vom considera N drepte în planul XOY, care definesc M regiuni (definite ca semiplane deschise, deci punctele dreptelor de separație nu fac parte din regiuni). De asemenea, pentru fiecare din cele M regiuni se va furniza

un punct în plan, care aparține respectivei regiuni. Astfel, se va construi un arbore de partiționare binară a spațiului. Fiecare nod intern al arborelui va avea asociată una din cele N drepte și va partiționa planul asociat lui în două semiplane deschise definite de dreapta asociată. Fiecare nod frunză va avea asociată o regiuni definită de restricțiile impuse de strămoșii respectivului nod.

Considerând dreapta $d: ax + by + c = 0$, aceasta va partiționa planul $\mathcal{P}$ asociat nodului curent, în două semiplane:

$$\begin{cases} \mathcal{P}_1: \{P(x,y) |\ ax + by + c < 0\} \\ \mathcal{P}_2: \{P(x,y) |\ ax + by + c > 0\} \end{cases}$$

Fiul stâng va avea asociat semiplanul $\mathcal{P}_1$, iar fiul drept va avea asociat semiplanul $\mathcal{P}_2$. Este necesar să fie respectată această convenție pentru implementarea temei.

Pentru a putea determina cărei regiuni îi aparține un punct P(x,y), va trebui să traversăm arborele, respectând condițiile impuse de fiecare nod, până ajungem într-o frunză. Regiunea asociată respectivei frunze va fi regiunea căruia îi aparține punctul P.

Vom considera o ordine inițială a celor N drepte (ordine aleatoare). Nodul rădăcină al arborelui va avea asociată prima dreaptă, și va avea doi fii, conform convenției de mai sus. La un anumit moment, vom avea un arbore generat de primele i drepte considerate și va trebui să introducem dreapta j în arbore. Fiecare din nodurile frunză ale arborelui curent are asociată o regiune, deci trebuie să decidem care din aceste regiuni trebuiesc partiționate considerând dreapta j. Pentru aceasta, vom traversa arborele cu dreapta j și frunzele în care ajungem vor fi partiționate utilizând această dreaptă.

Presupunem că intenționăm să traversăm arborele creat cu o dreaptă d, pentru a determina care din regiunile aflate în frunze pot fi partiționate utilizând această dreaptă. Considerăm nodul curent r, care are asociată o dreaptă d'. Dacă $d \parallel d'$, atunci vom determina de care parte a dreptei d' se află dreapta d și vom continua parcurgerea doar cu fiul respectiv. Dacă $d \nparallel d'$, atunci vom continua parcurgerea cu ambii fii ai nodului curent.

Există situații în care o regiune nu poate fi divizată de o anume dreaptă, fără a avea relații de paralelism între dreptele suport, dar nu vom considera aceste situații în cadrul implementării curente. Considerând această limitare, soluția finală nu va fi una optimă. De asemenea, vom avea frunze care nu vor avea asociată nici o regiune (regiuni create artificial).

2. Cerință

Fie N drepte în planul XOY, ce definesc M regiuni și M puncte în plan, fiecare aparținând uneia din cele M regiuni. Aceste M puncte sunt utilizate pentru a defini identitatea celor M semiplane (dacă punctul cu indicele i va aparține unui plan, acest plan va primi la rândul său indicele i).

Se cere să se determine pentru Q puncte, regiunea în care sunt incluse, utilizând arbori de partiționare binară a spațiului.

Se vor considera două moduri de funcționare

1) Arborele de partiționare binară a spațiului este furnizat
2) Arborele de partiționare binară a spațiului trebuie determinat

Pentru simplificare considerăm că dreptele nu sunt paralele cu axele sistemului de coordonate.

Se garantează că cele M puncte utilizate pentru identificarea regiunilor nu aparțin nici uneia dintre cele N drepte. Se garantează aceeași proprietate și pentru cele Q puncte de interogare.

Se garantează că nu vor exista două puncte (in cadrul celor M puncte de identificare), care să facă parte din aceeași regiune.

3. Implementare

3.1. Rulare

Programul va fi rulat:

./tema3 tip date.in date.out

Unde:

tip – indicele modului de funcționare (1/2)
date.in – fișierul cu date de intrare
date.out – fișierul cu date de ieșire

3.2. Formatul datelor de intrare/ieșire

Formatul fișierului de intrare depinde de modul de funcționare.

1) Se vor furniza N drepte, M puncte de identificare, arborele BSP și Q puncte de interogare.
   Arborele va fi furnizat specificându-se organizarea nodurile utile (care au asociată o dreaptă de partiționare). Se presupune o parcurgere în preordine, iar pentru fiecare nod este menționat indexul dreptei asociate cât și 2 valori binare, pentru a specifica existența fiilor stâng și respectiv drept.

2) Se vor furniza N drepte, M puncte de identificare și Q puncte de interogare

Formatul fișierului de ieșire nu depinde de modul de funcționare:

Se va furniza arborele BSP, utilizând o parcurgere în postordine. Pentru fiecare nod intern se specifică ecuația dreptei și pentru frunze se specifică coordonatele punctului de identificare asociat regiunii respective, cât și indexul regiunii. Dacă o frunză nu are asociată o regiune, atunci se va afișa șirul de caractere 'null'. Se vor furniza regiunile din care fac parte cele Q puncte, precizând indexul lor.

Atât parametrii ecuațiilor cât și coordonatele punctelor sunt numere întregi.
Indexarea dreptelor și a regiunilor de face de la 0.

date.in

1)

N
$a_1\ b_1\ c_1$
$a_2\ b_2\ c_2$
...
$a_N\ b_N\ c_N$
M
$x_1\ y_1$
$x_2\ y_2$
...
$x_M\ y_M$
$nod_1\ existaFiuStâng\ existaFiuDrept$
$nod_2\ existaFiuStâng\ existaFiuDrept$
...
Q
$x_1\ y_1$
$x_2\ y_2$
...
$x_Q\ y_Q$

2)

N
$a_1\ b_1\ c_1$
$a_2\ b_2\ c_2$
...
$a_N\ b_N\ c_N$
M
$x_1\ y_1$
$x_2\ y_2$
...
$x_M\ y_M$
Q
$x_1\ y_1$
$x_2\ y_2$
...
$x_Q\ y_Q$

date.out

$(a_1,b_1,c_1)\ (a_2,b_2,c_2)...((x_1,y_1)\text{- } index_1)...$
$indexReg_1$
$indexReg_2$
...
$indexReg_Q$

3.3 Exemplu

Considerăm exemplul din Figura 2.

$d_0: x - y = 0$
$d_1: x + y - 50 = 0$

Cele două drepte vor defini 4 regiuni, numerotate 0, 1, 2, 3 (ce corespund regiunilor $\mathcal{P}_0$, $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$) din Figura 2.

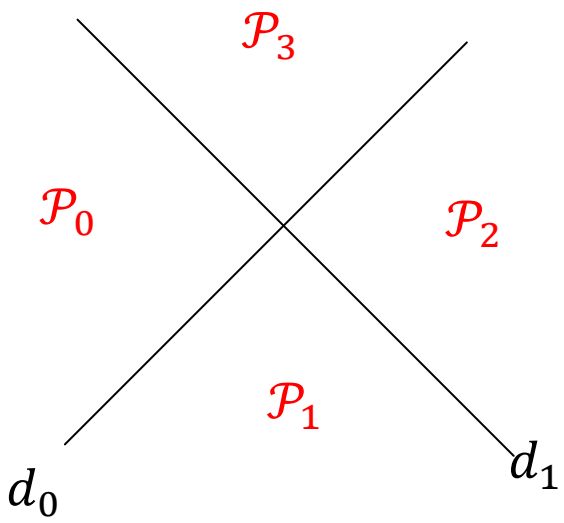

Figura 2: Divizare plan pe baza dreptelor d_0, d_1

Arborele BSP rezultat prin divizarea planului specificată în Figura 2 este redat în Figura 3.

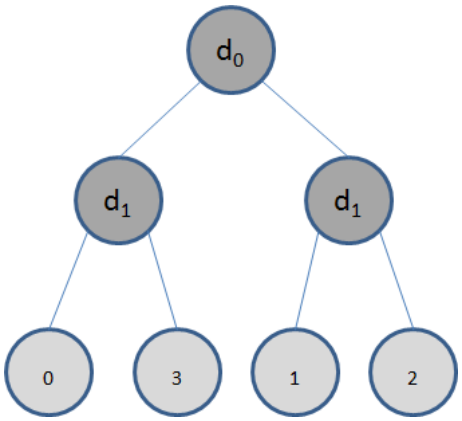

Figura 3. Arborele BSP obținut pentru divizarea spațiului din Figura 2

date.in

1)

```
2
1 -1 0
1 1 -50
4
0 25
25 0
50 25
25 50
0 1 1
1 0 0
1 0 0
8
35 5
5 20
40 25
25 40
45 19
3 24
10 2
25 35
```

2)

```
2
1 -1 0
1 1 -50
4
0 25
25 0
50 25
25 50
8
35 5
5 20
40 25
25 40
45 19
3 24
10 2
25 35
```

date.out

```
((0,25)-0)((25,50)-3)(1,1,-50)((25,0)-1)((50,25)-2)(1,1,-50)(1,-1,0)
1
0
2
3
2
0
1
3
```

4. Notare

- 80 de puncte obținute pe testele de pe vmchecker
  - 40 de puncte pentru modul 1 de funcționare
  - 40 de puncte pentru modul 2 de funcționare
- 10 puncte: coding style, codul trebuie să fie comentat, consistent și ușor de citit (a se vedea [2]). De exemplu, tema nu trebuie să conțină:
  - warninguri la compilare
  - linii mai lungi de 80 de caractere
  - tab-uri amestecate cu spații, denumire neadecvată a funcțiilor sau a variabilelor
- 10 puncte: README + comentarii
- **Bonus 20 de puncte** pentru soluțiile ce nu au memory leak-uri (bonusul se va calcula proporțional cu numărul de teste trecute)
- **O tema care nu va compila se va nota automat cu 0**

5. Reguli de trimitere a temelor

A se vedea și Regulile generale de trimitere și punctare a temelor [3]

Temele vor fi încărcate atât pe vmchecker (în secțiunea Structuri de Date (CB): SD-CB), cât și pe cs.curs.pub.ro, în secțiunea aferentă temei 3.

Arhiva cu rezolvarea temei trebuie să fie .zip și să conțină:

- fișiere sursă; fiecare fișier sursă creat sau modificat va trebui să înceapă cu un comentariu de forma:
  /* NUME Prenume – grupa */
- fișier README care să conțină detalii despre implementarea temei
- fișier Makefile; fișierul pentru make trebui denumit obligatoriu Makefile și trebuie să conțină următoarele reguli:
  - build – care va compila sursele și va obține executabilul, cu numele tema3
  - clean – care va șterge executabilele generate

Arhiva nu trebuie să conțină fișiere executabile sau obiect.

Dacă arhiva nu respectă specificațiile de mai sus nu va fi acceptată la upload și tema nu va fi luată în considerare.

Referințe

[1] http://en.wikipedia.org/wiki/Binary_space_partitioning
[2] http://ocw.cs.pub.ro/courses/programare/coding-style
[3] http://cs.curs.pub.ro/2014/mod/page/view.php?id=4309