DOCUMENTAȚIE PROIECT GEOMETRIE

-

ACOPERIREA CONVEXA A UNUI POLIGON ARBITRAR

Studenți:

DLarisa

MVlad

IOana

FBianca

# Cerință problemă

Input: Un poligon P din.

Output: acoperirea convexă Conv(P) determinată în timp liniar.

# Descriere algoritm folosit

Se consideră nodurile poligonului în ordinea dată și se va folosi o abordare incrementală foarte similară cu faza a doua a algoritmului "Graham Scan".

Presupunem că deja avem acoperirea convexă { ,..., }. Cum ar trebui actualizat pentru a obține acoperirea convexă { ,..., , }, vom folosi o structură numită "deque" (double ended queue) notată cu pentru a reprezenta secvența de noduri ale acoperirii convexe.

Algoritm:

Input: un poligon simplu cu n vârfuri P[i].

Se adaugă primele 3 vârfuri într-o coadă D[] astfel încât:

* al treilea vârf P[2] este atât la începutul cât și la finalul lui D
* în D[] ele formează un triunghi CCW (counterclockwise)

Cât timp sunt mai multe vârfuri ale poligonului de procesat:

Se ia următorul vârf P[i]

/\*

* D[] este acum acoperirea convexă a vârfurilor deja procesate.
* D[b] = D[t] = ultimul vârf adăugat în D[]. //

\*/

// Testăm dacă P[i] este în interiorul lui poligonulu format de D

dacă P[i] este atât în stânga lui D[b]D[b+1] cât și în stânga lui D[t-1]D[t]

se ignoră P[i] și se continuă cu următorul vârf

// altfel P[i] extinde acoperirea și trebuie adăugat la D[]

// Se obține tangenta față de punctul inferior

cât timp P[i] este în dreapta lui D[b]D[b+1]

se șterge D[b] de la începutul lui D[]

Se inserează P[i] la începutul lui D[]

// Se obține tangenta față de punctul superior

cât timp P[i] este în dreapta lui D[t-1] D[t]

se șterge D[t] de la finalul lui D[]

Se inserează P[i] la finalul lui D[]

Output: D[] = acoperirea convexă CCW a lui

Complexitate: O(n)

# Demonstrație corectitudine

Pentru H acoperirea convexă și P poligonul simplu se cunosc următoarele date:

1. H convex;
2. P aparține conținutului lui H (se găsește în interiorul lui H);
3. Nodurile lui H sunt un subset al nodurilor lui P.

Algoritm:

unde:

b - bottom

t - top

nodurile acoperirii: ,.., ()

operații de bază: POPBOTTOM(), POPTOP, PUSHB, PUSHT

// acoperire convexă {}

pentru i 3, n execută

dacă este în afara atunci

cât timp isLeft of execută POPB()

cât timp isRight of execută POPT()

PushB(, )

PushT(, )

Pentru demonstrație se folosește inducția.

Cazul 1:

Știm că sunt conectate de un drum poligonal și că este legat de de un drum poligonal.

Din ipoteză, știm că cele două drumuri nu se intersectează, deci trebuie să fie în interiorul acoperirii convexe, deci va fi ignorat.

Cazul 2: (este în afara unghiului și a acoperirii convexe)

Algoritmul scoate vârfurile acoperirii convexe până ajunge la punctele finale ale tangentei de la la acoperirea convexă actuală.

Algoritm liniar: Dacă avem “n” noduri, face cel mult “2n” push-es si “2n-3” pops.

Operațiuni interzise de poligoane:

4 1 5 NU

2 nu avem voie “” 2 4

5

3 3

Exemplu: Pornim de la 3 puncte:

# 

Acum, vrem să adaugam al 4-lea punct. Daca e în interiorul triunghiului, apoi acoperirea convexa rămane aceiași. nu poate sa treaca peste linia și mai raman 2 cazuri, deci in total raman 4 cazuri.

caz 1:

Acoperirea convexă rămâne aceeași.

caz 2:

caz 3:

(la dreapta )

caz 4: