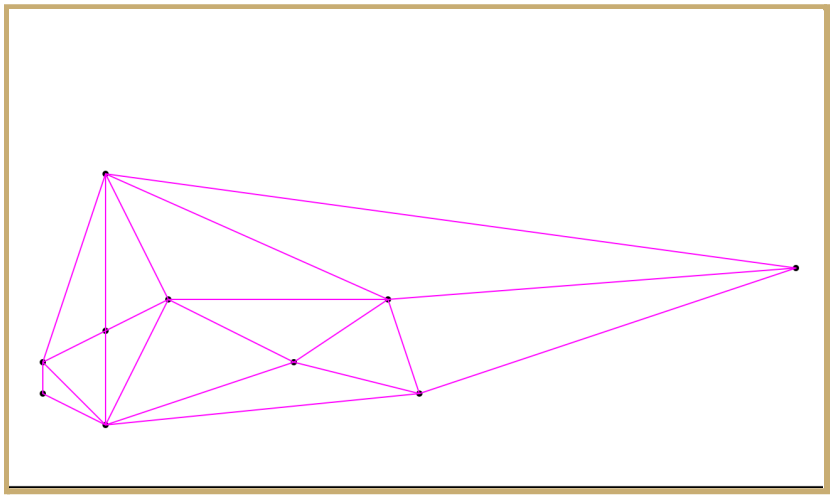
Geometrijski algoritmi @ MATF

Konstrukcija Deloneove triangulacije tehnikom razlaganja

Jovana Bošković



Opis problema

Triangulacija ima brojne primene u mnogim oblastima nauke i tehnike. Triangulacija skupa tačaka P je definisana kao maksimalna planarna podela. Podela je maksimalna ako se dodavanjem stranice koja spaja bilo koja dva temena skupa P narušava planarnost podele. Svaki element podele je trougao. Broj trouglova i ivica je isti u svakoj triangulaciji i zavisi od broja tačaka koje ulaze u konveksni omotač. Mnoge skupove tačaka moguće je triangulisati na vise načina. U ovom radu predstavljena je Delone triangulacija.

Delone triangulacija zadovoljava kriterijum da svaki opisan krug oko trougla sadrži samo temena trougla i nijednu drugu tačku skupa. Ova triangulacija je jedinstvena.

Najpoznatiji algoritmi za izračunavanje su inkrementalni algoritam i algoritam zasnovan na razlaganju. U nastavku biće opisan algoritam zasnovan na razglanju.

Naivno rešenje problema

Triangulacija je legalna ako ne sadrži nijednu nelegalnu stranicu.

Na osnovu ove teoreme može se kreirati algoritam koji vraća legalnu triangulaciju skupa tačaka.

Ulaz: Proizvoljna triangulacija T skupa P

Izlaz: Legalna triangulacija skupa P

- Dok god T sadrži nelegalnu stranicu P_iP
    - // obrni stranicu P_iP
    - Neka su $P_iP\ \ P_k$ i $P_iP\ \ P_l$ dva trougla sa stranicom P_iP
    - Izbaci P_iP iz T i u T dodaj stranicu P_kP_l
- Vrati T

Navedeni algoritam je korektan i daje legalnu triangulaciju, ali je suviše neefikasan za praktične primene. Složenost algoritma je $O(n^2)$.

U radu nije predstavljen naivni algoritam.

Optimalni algoritam

Optimalni algoritam implementiran je strategijom razlaganja i prati rad koji su prezentovali Guibas i Stolfi (link).

Strategija razlaganja zasniva se na podeli problema na manje podprobleme, koji su približno jednake veličine, a rešenje originalnog problema se dalje dobija povezivanjem rešenja podproblema.

Pre poziva algoritma potrebno je sortirati tačke rastuće u odnosu na x-koordinatu, a ako imaju istu x koordinatu redosled im se određuje na osnovu y-koordinate.

Opis algoritma:

Ulaz: Skup od n tacaka

Izlaz: Delone triangulacija

1. Ako je n = 2 return linijski segment
2. Ako je n = 3 return trougao

3. Inače, neka je m ceo deo od n/2. Podeliti skup tačaka P na $P_l = \{p_1, p_2, ..., p_m\}$ i $P_r = \{p_{m+1}, ..., p_n\}$
4. Konstruisati Delone triangulaciju skupa P_l i Delone triangulaciju skupa P_r rekurzivno
5. Povezati Delone triangulacije skupa P_l i skupa P_r u Delone triangulaciju skupa P algoritmom spajanja
6. Return Delone triangulaciju skupa P

Deljenjem tačaka dobija se binarno stablo sa listovima koji su skupovi od 2 ili 3 tačke. Listovi se trivijalno spajaju u linijski segment ili trougao. Kretanjem na gore, spajaju se parovi, poštujući kriterijum da ne postoji tačka u krugu opisanog oko trougla. U algoritmu su korišćene strukture podataka Edge i Quad Edge za lakše programiranje ove topološke osobine.

Opis strukture Edge (orijentisana stranica):

- Origin - tačka početka stranice
- Dest - tačka kraja stranice
- Next - označava smer suprotan kazaljki na satu (ccw)
- Rnext - sledeća stranica u desnoj oblasti, sa istom desnom oblašću
- Lnext - sledeća stranica u levoj oblasti, sa istom levom oblašću
- Onext - sledea stranica oko tačke početka, sa istom tačkom početka
- Dnext - sledeća stranica oko tačke kraja, sa istom tačkom kraja
- Prev - označava smer kazaljke na satu (cw)
- Rprev, Lprev, Oprev, Dprev su analogne Rnext, Lnext, Onext, Dnext, samo se odnose na stranicu koja prethodi trenutnoj.

Opis stukture QuadEdge:

- e - usmerena stranica
- Sym - suprotna stranica
- Rot - dualna stranica koja pokazuje levo od stranice e
- InvRot - dualna stranica koja pokazuje desno od stranice e

Ivice iz leve triangulacije se nazivaju LL-ivicama, dok se RR-ivicama nazivaju ivice desne triangulacije. Prilikom povezivanja leve i desne triangulacije nastaju nove ivice koje se nazivaju LR-ivicama. Prilikom procesa povezivanja LL i RR ivice mogu biti izbrisane, ali nove ne mogu biti kreirane.

Prvi korak povezivanje dve polovine je ubacivanje bazne LR-ivice, najniža LR-ivica, koja ne preseca nijednu LL i RR ivicu. Zatim se dodaje sledeća LR ivica, čija je krajnja tačka ili desna

ili leva krajnja tačka bazne ivice, a druga tačka će biti iz levog ili desnog podskupa tačaka. Izbor drugog kandidata se sužava na jednog kandidata iz levog podskupa i jednog kandidata iz desnog podskupa..

Gledano sa desne strane. Prvi potencijalni kandidat je tačka povezana sa desnom krajnjom tačkom bazne LR-ivice preko RR-ivice, koja obrazuje najmanji ugao u pravcu obrnutom od kazaljke na satu, počevši od bazne LR-ivice. Sledeći kandidat se bira po sledećem najmanjem uglu.

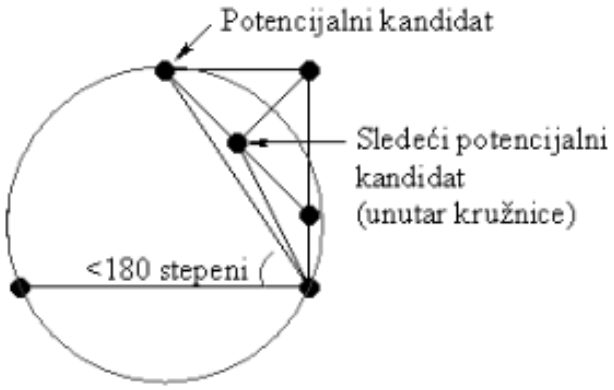
```
EdgeDQ* LeftCandidate(EdgeDQ* base_edge)

EdgeDQ* RightCandidate(EdgeDQ* base_edge)
```

Za potencijalne kandidate se proveravaju dva kriterijuma:

1. Ugao u pravcu obrnutom od kazaljke na satu, počevši od bazne LR-ivice mora biti manji od 180 stepeni
   ```
   Bool Valid(EdgeDQ* candidate, EdgeDQ* base_edge)
   ```
2. Kružnica kroz krajnje tačke bazne LR-ivice i potencijalnog kandidata ne sme sadržati sledećeg potencijalnog kandidata unutar

   ```
   Bool In_circle(QPointF baseDest, QPointF baseOrg, QPointF candidateDest, QPointF candidate.OprevDest )
   ```

Ako su oba kriterijuma ispunjena, kandidat postaje konačni desni kandidat. Ako prvi uslov nije ispunjen, kandidat sa desne strane nije izabran. Ako je prvi kriterijum ispunjen, a drugi nije, onda se ivica, koja vezuje potencijalnog kandidata i desnu krajnju tačku bazne LR-ivice briše. Proces se nastavlja sa sledećim potencijalnim kandidatom sve dok se ne izabere desni kandidat ili dok se ne dođe do zaključka da kandidat ne može biit izabran. Proces se potom ponavlja pri pronalaženju levog kandidata analogno.

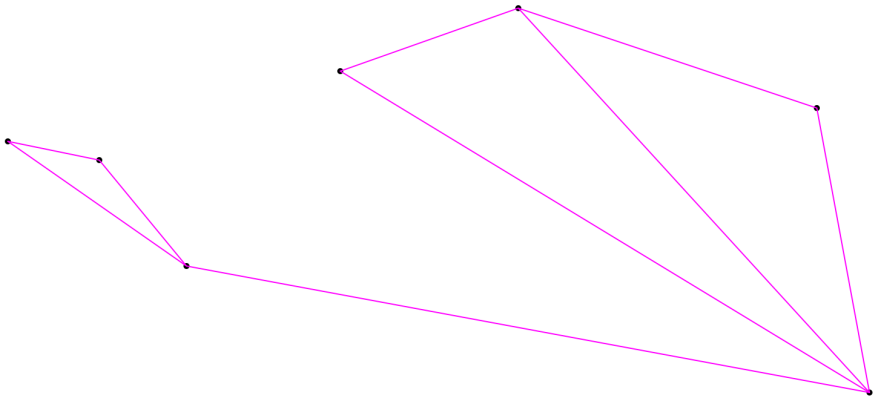


Kada ni levi ni desni kandidati ne mogu biti nađeni proces povezivanja je završen. Ako je pronađen samo jedan kandidat, njime je odredjena LR-ivica koju tražimo. U slučaju da su pronađena oba kandidata, nova LR-ivica se određuje testom: ako se desni kandidat ne nalazi unutar kružnice definisane dvema kranjim tačkama bazne LR-ivice i levim kandidatom, onda levi kandidat određuje LR-ivicu i obrnuto.

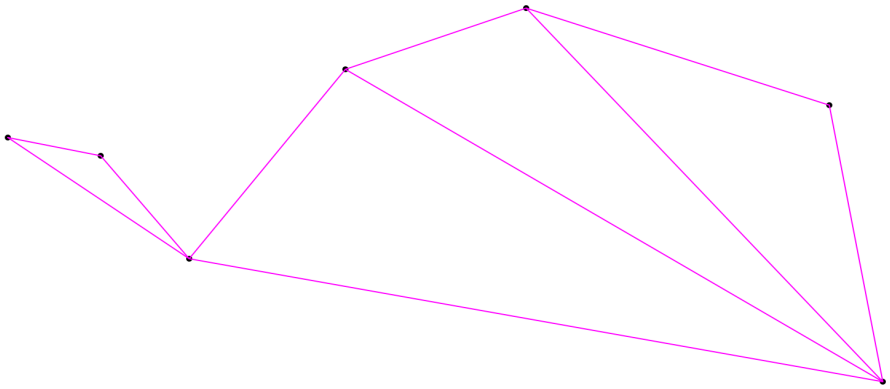
Nakon sto je pronađena nova LR-ivica proces se ponavlja sa novom LR-ivicom kao baznom LR-ivicom.

Sortiranje tačaka je vremenske složenosti O(n log n). Koraci povezivanja se izvršavaju u vremenu O(n), pa je onda složenost celog algoritma O(n log n).

Vizuelizacija algoritma



Povezivanje dve polovine počev od bazne ivice



Nalaženje nove bazne ivice

Testiranje ispravnosti algoritma

Naziv testa	Opis testa	Ulaz	Očekivani izlaz
WrongInput1	Zadavanje ulaza koji nije ispravan. Program ce ispisati poruku o gresci, a rezultujuci niz treba da bude prazan.	Niz dimenzije manje od 1	[]
TwoPoints	Zadavanje ulaza koji ima dve tacke. Algoritam poziva funkciju Line Primitive koja povezuje tacke u linijski segment.	Niz dimenzije 2	Poklapanje povratne vrednosti iz funkcije triangulation i LinePrimitive
ThreePoints	Zadavanje ulaza koji ima tri tacke. Algoritam poziva funkciju TrianglePrimitive koja povezuje tacke u trougao.	Niz dimenzije 3	Poklapanje povratne vresnoti iz funkcije triangulation i TrianglePrimitive
IsSorted	Proverava da li su tacke pre poziva algoritma za triangulaciju sortirane.	Niz dimenzije 100	Rastuce sortirane tacke u odnosu na x koordinatu
Input1 Input2 Input3	Zadavanje ulaza iz fajla. Racuna broj ivica i proverava da li je jednak izrazu 3n-3-h.	Niz tacaka iz fajla Input 1, Input 2, Input 3	Jednakost broja ivica sa izrazom 3n-3-h. Provera Quad strukture