

7. Domača naloga pri predmetu Matematične metode v računalništvu

Slavko Žitnik
63060254

20. april 2011

Nalogo smo skupaj premislili z Neli, in Anžetom. Pri drugi je začetno idejo dobil Anže in smo skupaj tudi naredili en primer za $k=3$.

Naloga 1

Denimo, da ima graf G drevesno širino enako k in naj bo (T, \mathcal{V}) njegova drevesna dekompozicija širine k . Število točk grafa G naj bo enako n in $k \ll n$.

- a) Opiši postopek, ki bo s pomočjo (T, \mathcal{V}) konstruiral gladko drevesno dekompozicijo (T^*, \mathcal{V}^*) grafa G iste širine k . Drevesna dekompozicija širine k je gladka, če za vsako vrečo velja $|V_t^*| = k + 1$ in za vsaki sosednji vreči V_t^* in $V_{t'}^*$ velja $|V_t^* \cap V_{t'}^*| = k$

V algoritmu 1 smo napisali postopek¹, kako vsako vozlišče predelamo tako, da iz poljubne drevesne dekompozicije dobimo gladko. Prvega *if* stavka v našem primeru ne potrebujemo, ker predpostavljamo, da noben par vrečk ni enak ali vsebovan v sosednjemu.

Pravilnost:

Algoritem izvajamo s sprehodom čez vse vrečke in jih popravljamo, dokler niso ustrezne. Pokazati moramo, da v nobeni iteraciji ne pokvarimo dekompozicije.

Pri prvem pogoju (ki ga v našem primeru ne potrebujemo) je važno, da vidimo, da unija nove vrečke ne more biti večja kot $k + 1$. To velja zaradi tega, ker je že podana dekompozicija drevesne širine k .

V drugem pogoju vrečki, ki ne vsebuje dovolj vozlišč, dodamo vozlišče iz sosednje vrečke. Potem je lahko vrečka že vsebuje dovolj točk. Če jih ne, se postopek ponovi, če pa jih, pa je treba preveriti še tretji pogoj.

V tretjem pogoju ustvarimo novo vrečko, ki vsebuje k skupnih točk z vrečko V_i in eno skupno točko več z vozliščem V_j , kot jo je imelo z V_i . Če med novo vrečko in V_j , ni presek enak k , ponovimo postopek med njima.

- b) Določi časovno zahtevnost tvojega postopka.

Definiramo z m število vrečk in s k število točk v vrečki.

¹Algoritem pridobljen iz <http://www.cs.auckland.ac.nz/research/theses/2001/pengzhouthesis2001.pdf>

Data: G , drevesna dekompozicija (T, \mathcal{V}) širine k
Result: gladka drevesna dekompozicija (T, \mathcal{V}) širine k
begin

```

  while obstaja nepregledan par sosednjih vozlišč  $V_i, V_j \in T$ 
  do
    if  $V_i \subseteq V_j$  or  $V_j \subseteq V_i$  then
       $V_{new} \leftarrow V_i \cup V_j$ 
       $V_{new}.sosed_i = V_i.sosed_i \cup V_j.sosed_i$ 
      odstrani  $V_j$ 
      odstrani  $V_i$ 
    end
    if  $|V_i| < k + 1$  and  $V_j \not\subseteq V_i$  then
       $v \in V_j - V_i$ 
       $V_i = V_i + \{v\}$ 
    end
    if  $|V_i| = |V_j| = k + 1$  and  $|V_j \cap V_i| < k$  then
       $v \in V_j - V_i$ 
       $w \in V_i - V_j$ 
       $V_{new} = V_i - \{w\} \cup \{v\}$ 
      razdeli povezavo  $(i, j)$  na  $(i, new)$  in  $(new, j)$ , kjer se v  $new$ 
      nahaja novo vozlišče  $V_{new}$ 
    end
  end
end
return  $(T, \mathcal{V})$ 

```

Algorithm 1: Konstrukcija gladke drevesne dekompozicije

Pri pogoj za združevanje dveh vrečk porabi $O(k)$ časa.

Drugi pogoj $O(k)$ časa, vendar se lahko zgodi, da je ena vrečka velikosti 2, druga pa velikosti $k + 1$ in imata skupno le eno točko. V tem primeru moramo iti za popravek enega vozlišča $k - 1$ -krat. Torej je za drugi pogoj časovna zahtevnost enaka $O(k^2)$.

Podobno, kot v drugem pogoju, se lahko zgodi, da je tudi tu skupna točka v obeh vrečah le ena in potrebujemo skupaj $O(k^2)$ časa.

Celoten postopek potem potrebuje $O(k^2m) = O(m)$ - linearna časovna zahtevnost, ker je $k \ll m$.

Naloga 2

Naj bo G graf z n točkami z drevesno širino k in naj bo (T, \mathcal{V}) drevesna dekompozicija grafa G širine k . Opiši dinamični algoritem za konstrukcijo hamiltonovega cikla v času $O(f(k) \cdot n)$. Velja $k \ll n$.

Za rešitev te naloge smo dobili prosojnice, po katerih bi lahko delali: [http : //www.cs.bme.hu/ dmarx/papers/marx - warsaw - fpt2](http://www.cs.bme.hu/~dmarx/papers/marx-warsaw-fpt2), vendar smo premislili po svoje.

- a) Oцени $f(k)$. Kako strukturiraš karakteristike?

1. kaj je karakteristika:

Karakteristiko definiramo kot vse hamiltonove poti na podgrafu grafa G s točkami v trenutni vrečki.

2. konstanstnost števila karakteristik:

V vsaki vrečki imamo lahko $k + 1$ točk iz grafa G . Največ hamiltonovih ciklov bo obstajalo, če te točke v G tvorijo kliko. Vseh možnih karakteristik bi tedaj bilo $k!$.

3. algoritmi:

Start V listih imamo $k + 1$ točk. Med karakteristikami ohranimo tiste, ki predstavljajo hamiltonovo pot v grafu G . Poti označimo kot $a - b - c - d$.

Forget Odstranimo točko c . Od sinovih karakteristik ohranimo hamiltonove poti, kjer se c ne nahaja na krajiščih poti. Vsako takšno pot npr.: $a - b - c - d - e$ potem zapišemo kot $a - b = d - e$. S simbolom “=” označimo, da poznamo pot med b in d preko drugih vozlišč.

Introduce Dodamo točko f . K potem sinov dodamo točko f tako, da dobimo nove hamiltonove poti. Točko f lahko pridružimo kamorkoli, razen med točki b in d , med katerima je povezava tipa “=” . To bi pomenilo, da lahko pridemo od b do d na dva načina, s čimer pa nismo konstruirali poti.

Join Združujemo hamiltonove poti obeh sinov. Združene poti dobimo tako, da vse povezave tipa “=”, ki se pri drugem sinu pojavijo kot “-”, nadomestimo z “=”.

Če smo naredili *Join* v korenskem vozlišču moramo še preveriti, če v G obstaja neposredna povezava med točkama v krajiščih poti. Če obstaja, potem trdimo, da v G obstaja hamiltonov cikel.

4. dokaz pravilnosti: Po postopku lahko dobimo vse možne karakteristike, torej je algoritem pravilen.

Hamiltonov cikel po našem postopku dobimo tako, da se sprehodimo od korena proti listom. V korenu vzamemo znan hamiltonov cikel in ko se sprehodimo navzdol, dodajamo točke, ki smo jih brisali na tistih povezavah, ki jih imamo označene s simboli “=”.

Časovna zahtevnost: Najbolj časovno potratno je združevanje hamiltonovih poti pri *Join*, ki je reda $f(k) = O(k^k)$.