-TEMĂ SEMINAR 3-

-ALGORITMI FUNDAMENTALI-

POPESCU PAULLO ROBERTTO KARLOSS
GRUPA 231

Exercițiul 8

Cerința: Fie un graf neorientat $G=(V,E)$, unde $V=\{1,2,\dots,n\}$ și E nu este inițial cunoscut. Graful poate fi descoperit prin interogări de tipul $Q(i,j)$ către un oracol, care va răspunde cu 1 dacă muchia neorientată (i,j) există în E și cu 0 altfel.

Demonstrați că, pentru orice algoritm A de determinare a componentelor conexe, există un graf G_A pentru care A trebuie să facă cel puțin $n(n-1)/2$ interogări.

Indiciu: algoritmi $\Leftrightarrow$ arbori de decizie.

Soluție: La fiecare pas excludem nodurile viz, $(x,y)=(y,x)$ Graf neorientat

-> Pentru nodul 1: $Q(1,2), Q(1,3), \dots, Q(1,n)$

Deci n interogări pentru nodul 1

-> Pentru nodul 2: $Q(2,3), Q(2,4), \dots, Q(2,n)$

Deci $(n-1)$ interogări pentru nodul 2

-> Pentru nodul 3: $Q(3,4), Q(3,5), \dots, Q(3,n)$

Deci $n-2$ interogări pentru nodul 3

⋮

→ Pentru nodul $(n-1)$: $Q(n-1, n)$ Deci 1 interogare pentru nodul $n-1$

Ne oprim la $(n-1)$! De ce? Pentru că nodul n va avea 0 interogări

$$\Rightarrow (n-1)+(n-2)+\ldots+1=\frac{(n-1)\cdot n}{2} \text{ interogări}$$

necesare

Problema: Fie n puncte de interes, reprezentate prin puncte în plan (x_i, y_i) $(1 \le i \le n)$. Un turist pornește din (x_1, y_1) și dorește să viziteze toate punctele întorcându-se de unde a plecat (ciclu hamiltonian, problema comis-voiajorului) parcurgând o distanță cât mai mică... După un timp realizează că nu poate obține minimul (e problema NP), dar se mulțumește cu o distanță de cel mult de două ori mai mare ca minimul. Descrieți un algoritm eficient pentru a rezolva problema.

Soluție: Știm că există muchie de la nodul i la nodul j, întrucât ne aflăm în plan (putem ajunge din (x_i, y_i) în (x_j, y_j)).

Știm și că $dist((x_i, y_i), (x_j, y_j)) \leq dist((x_i, y_i), (x_{i+1}, y_{i+1})) + dist((x_{i+2}, y_{i+2}), (x_{i+3}, y_{i+3})) + \ldots + dist((x_{j-1}, y_{j-1}), (x_j, y_j))$.

Trebuie să creăm arborele de cost minim pt. a afla drumul minim de la nodul de plecare la toate celelalte noduri. După facem un DF din nodul de plecare (punctul (x_1, y_1)) din care aflăm distanța totală (cu tot cu nodurile pe care le-am vizitat deja). Știm din cerință că este un ciclu Hamiltonian, iar dacă punem în evidență acest lucru asupra distanțelor noastre și înlocuim secvențele de noduri care apar de mai multe ori, cu scurtătura dintre acestea, dacă pt. a ajunge din p_i în p_j treb. să ne întoarcem la $p_{i+1}, \ldots, p_{j-1}$, mergem direct la p_i la p_j. Așadar dacă înlocuim dist. init cu una mai mică sau egală obținem un drum cu cost dublu față de cel inițial. Adăugăm nodul de plecare și revine la un ultimul pct dinspre acesta.

Cerința: Care dintre următorii algoritmi determină corect un arbore parțial de cost minim (justificați)? Pentru fiecare algoritm corect precizați ce complexitate are.

Alg 1: $T = G$

- cât timp T conține cicluri execută
- alege e o muchie de cost maxim care este conținută într-un ciclu din T
- T devine $T - e$

Alg 2: $T = G$

- cât timp conține cicluri execută
- alege C un ciclu oarecare din T, și fie e muchia de cost maxim din C
- $T = T - e$

Soluție: Observăm că ambii algoritmi sunt corecți în scopul determinării unui arbore parțial de cost minim (APM). De ce? Pentru că știm, că un APM trebuie să nu conțină cicluri! Ambii algoritmi elimină la fiecare pas din numărul total de cicluri conținute de graful nostru, un ciclu (un ciclu este eliminat atunci când eliminăm o muchie de la un nod la alt nod).

O soluție care nu este foarte eficientă, dar care funcționează în scopul eliminării unui ciclu, ar fi să facem un DF dintr-un nod $O(n+m)$. Dacă am găsit un ciclu căutăm muchia de cost maxim $O(m)$. După ce am găsit-o o eliminăm $O(1)$. Procedăm așa recursiv până când nu mai găsim niciun ciclu => AMP. Pentru cel de-al doilea alg.

Pentru primul algoritm ar trebui să sortăm la început muchiile după cost ($O(m \log m) = O(m \log n)$). După care am lua muchia de cost maxim, facem un DF din aceasta pt. a verifica dacă avem ciclu $O(n+m)$. Dacă avem ciclu trecem la următoarea muchie de cost maxim, întrucât pe aceasta am șters-o din graful nostru. Dacă nu închide un ciclu muchia rămâne în graf, trecem la următoarea muchie de cost maxim. Procedăm așa pentru toate muchiile $O(m) + O(n+m)$

Pentru al doilea alg. ar fi $O(\text{nr. cicluri} + n)$ n de la găsirea maximului.