Algoritmica grafurilor - Tema 1

Bucătaru Andreea A2 Bulboacă Maria A2

09 noiembrie 2018

Problema 1

Definirea grafului: $G = (V, E)$ digraf conex, unde V este mulțimea nodurilor, un nod reprezentând intersecțiile dintre străzi, iar E este mulțimea muchiilor (sensurilor de stradă).

Întrebarea se rezumă la câte muchii trebuie eliminate astfel încât să nu existe circuite în graful dat. Observăm că blocarea unui sens al unei străzi cu dublu sens este același lucru cu inversarea acelui sens.

Pentru a nu exista circuite în digraful dat, nodurile ar trebui să poată fi sortate topologic.

Sortarea topologică realizează o aranjare liniară a nodurilor în funcție de muchiile dintre ele. Orientarea muchiilor corespunde unei relații de ordine de la nodul sursă către cel destinație. Astfel, dacă (u, v) este una dintre muchiile grafului, u trebuie să apară înaintea lui x în înșiruire. Dacă graful ar fi ciclic, nu ar putea exista o astfel de înșiruire (nu se poate stabili o ordine între nodurile care alcătuiesc un ciclu).

Altfel spus, pentru a obține un digraf aciclic putem elimina (inversa) p muchii (sensuri de străzi) astfel încât să obținem o sortare topologică, după următorul algoritm:

- cât timp mai există noduri, încercăm să le sortăm topologic
- dacă există un nod cu gradul interior 0, îl eliminăm din digraf și eliminăm toate muchiile incidente cu el
- dacă nu există niciun astfel de nod, atunci alegem unul cu gradul interior minim și îi inversăm sensurile (aceleași pe care le-am bloca) astfel încât gradul lui interior să devină 0. Apoi îl eliminăm pe el și toate muchiile incidente.

În acest fel, ne asigurăm că scapăm de circuite (nu putem merge în cerc) și că la fiecare pas blocăm/inversăm un număr minim de sensuri, deoarece alegem acel vârf cu gradul interior cel mai mic.

Mai jos prezentăm o soluție alternativă.

Fie mulțimea de circuite $C = \{C_1, ..., C_n\}$ înainte de blocarea vreunui sens. Procesăm fiecare circuit în următorul fel:

- În cazul în care circuitul curent nu mai există (a fost eliminat anterior), trecem la circuitul următor.
- În cazul circuitelor care conțin doar străzi cu dublu sens sau doar străzi cu sens unic, inversăm același sens pe care l-am fi blocat.
- În cazul circuitelor care conțin străzi cu dublu sens și străzi cu sens unic, alegem oricare stradă cu dublu sens și inversăm sensul care ajută la formarea circuitului.

Astfel, se poate îndepărta posibilitatea de a merge în cerc în oraș, inversând p sensuri de străzi.

Problema 2

Operația binară definită în enunț se referă la produsul cartezian (puțin modificat) între două grafuri, fiecare graf având măcar 2 noduri. Trebuie să arătăm că $G_1 \odot G_2$ este conex dacă și numai dacă G_1 și G_2 sunt conexe și unul dintre ele conține un circuit impar.

"$\Rightarrow$" Arătăm că G_1, G_2 conexe și unul dintre ele conține un circuit impar, dacă $G_1 \odot G_2$ conex.

Fie $u_1, v_1 \in V_1$ și $u_2 \in V_2$. Presupunem că există un drum de la (u_1, u_2) la (v_1, u_2) în $G_1 \odot G_2$, de forma $(u_1u_2, u_1x_2, ..., u_1t_2, u_1v_2, ..., x_1u_2, ..., v_1u_2, v_1v_2)$.

Din prima componentă extragem un drum din G_1, renunțând la copiile multiple ale aceluiași nod: $(u_1, t_1, w_1, ..., z_1, ..., v_1) \Rightarrow \exists$ în G_1 un drum de la u_1 la v_1, deci G_1 este conex.

Analog, extragem un drum din G_2, renunțând la copiile multiple ale aceluiași nod: $(u_2, x_2, t_2, ..., v_2) \Rightarrow \exists$ în G_2 un drum de la u_2 la v_2, deci G_2 este conex.

Pentru ca un graf să aibă un circuit impar, trebuie să nu fie bipartit.

"$\Leftarrow$" Arătăm că $G_1 \odot G_2$ conex, dacă G_1, G_2 conexe și unul dintre ele conține un circuit impar.

Fie $u_1, v_1 \in V_1$ și $u_2, v_2 \in V_2$. Presupunem că există un drum de la u_1 la v_1 din G_1 și un drum de la u_2 la v_2 din G_2.

Drumul din G_1 este de forma $(u_1, x_1, ..., t_1, v_1)$.

Drumul din G_2 este de forma $(u_2, x_2, ..., t_2, v_2)$.

Aplicând produsul cartezian, obținem: $(u_1u_2, u_1x_2, ..., u_1t_2, u_1v_2, ..., x_1u_2, ..., v_1u_2, v_1v_2)$.
$\Rightarrow \exists$ un drum de la u_1u_2 la v_1v_2 în $G_1 \odot G_2$, deci $G_1 \odot G_2$ este conex.

Problema 3

a) Arătăm că, dacă îndepărtăm din matricea de incidență a unui arbore o coloană corespunzătoare unui nod dat, se obține o matrice pătratică nesingulară.

Fie un arbore T. Notăm matricea de incidență a lui T cu M_T, iar matricea de incidență rezultată prin îndepărtarea unei coloane corespunzătoare unui nod dat, cu N_T.

Observații:

- O matrice pătratică A este nesingulară dacă $det(A) \neq 0$.
- O matrice pătratică $n \times n$ este nesingulară dacă $rang(A) = n$.
- Cum fiecare muchie este incidentă cu exact alte 2 muchii, matricea de incidență va avea pe fiecare coloană câte două valori de 1. Numărul de valori de 1 de pe fiecare linie este gradul nodului respectiv.
- Rangul matricei de incidență pentru un graf conex este $n - 1$. Rangul matricei de incidență pentru un graf neconex, cu k componente conexe, este $n - k$.

Un arbore T cu n noduri are $n - 1$ muchii și este conex. Deci N_T este o matrice pătratică de ordin $n - 1$ și rang $n - 1$.

Presupunem un graf G cu n noduri și $n - 1$ muchii care nu este un arbore. Deci G este neconex. Rangul unei matrice de incidență pentru un graf neconex, cu k componente conexe, este $n - k$. Deci rangul N_G este $< (n - 1)$. (1)

Matricea N_T este $(n - 1) \times (n - 1)$, deci rangul ei ar trebui să fie $n - 1$ pentru ca matricea să fie nesingulară. (2)

Din (1) și (2) rezultă contradicție. Matricea N_T este nesingulară dacă și numai dacă graful este un arbore.

b) Arătăm că dacă C este un circuit, atunci M_C este matrice nesingulară $\Leftrightarrow$ C este impar.

Fie C un circuit cu nodurile $1, \ldots, n$ cu $n \geq 3$ și M este matricea de incidență a lui C. Atunci determinatul lui M este 0 dacă n este par și 2 dacă n este impar.

Pentru ca un graf să aibă un circuit impar, trebuie să nu fie bipartit.

Fie G un graf conex cu n noduri și M matricea de incidență a lui G. Atunci rangul lui M este $n-1$ dacă G este bipartit și n dacă nu este bipartit.

Problema 4

a) Presupunem că suntem la iterația i a buclei *while*. Există un j din $V \setminus S$ și un drum p de la s la j, astfel încât $\min\limits_{x \in V \setminus S} u_x^F = a(p)$. $\min\limits_{x \in V \setminus S} u_x^F$ nu se schimbă cât timp j este din $V \setminus S$. În momentul în care j este ales ca j^*, j este adăugat mulțimii S, deci $\min\limits_{x \in V \setminus S} u_x^F \neq a(p)$. În acest moment, există un alt nod j' și un drum p' de la s la j', astfel încât $\min\limits_{x \in V \setminus S} u_x^F = a(p')$. Presupunem prin reducere la absurd că $a(p') < a(p)$ (minimul ar descrește). Dar dacă aceasta s-ar întâmpla, atunci j' ar fi fost ales în locul lui j la început. Deci $a(p') \geq a(p)$, deci $\min\limits_{j \in V \setminus S} u_j^F$ nu descrește.

După același raționament, $\min\limits_{l \in V \setminus T} u_l^B$ nu descrește.

b) Invariantul algoritmului Dijkstra ne spune că $u_j^F = a(P_{sj}), \forall j \in S$. Analog, $u_k^B = a(P_{kt}), \forall k \in T$.

Știm că P_{st} este un drum de cost minim între (s,t) format din nodurile $s, ..., j, j', l', l, ..., t$. Deducem că drumul $(i, ..., x, ..., j), \forall i, j \in V(P_{st})$, are costul minim. Deci $a(i, ..., j) = a(P_{ij})$. Presupunem prin reducere la absurd că ar exista un alt drum $(i, ..., x', ..., j)$ cu cost mai mic. Asta înseamnă că am putea alege această variantă de a ajunge din i în j (prin intermediul lui x') pentru a scădea costul drumului P_{st}. Dar știm deja că $a(P_{st})$ este minim, deci avem contradicție.

Folosind aceasta, demonstrăm $a(P_{st}) \geq u_j^F + a_{jj'} + a_{l'l} + u_l^B$. Putem scrie costul drumului ca fiind următorul: $a(P_{st}) = a(P_{sj}) + a_{jj'} + a(P_{j'l'}) + a_{l'l} + a(P_{l't})$. Deoarece $P_{sj}, P_{j'l'}$ și $P_{l't}$ au costuri minime, rezultă că $a(P_{sj}) = u_j^F$ și $a(P_{l't}) = u_{l'}^B$. Deci $a(P_{st}) = u_j^F + a_{jj'} + a(P_{j'l'}) + a_l'l + u_{l'}^B$. Lucrăm doar cu muchii cu costuri pozitive, deci $a(P_{j'l'}) \geq 0 \Rightarrow a(P_{st}) \geq u_j^F + a_{jj'} + a_{l'l} + u_l^B$. În cazul în care $j' = l'$, avem $a(P_{j'l'}) = 0$, deci $a(P_{st}) = u_j^F + a_{jj'} + a_{l'l} + u_l^B$.

Demonstrăm că $u_j^F + a_{jj'} + a_{l'l} + u_l^B \geq \min\limits_{h \in V \setminus S} u_h^F + \min\limits_{k \in V \setminus T} u_k^B$. Știm că drumul $(s, ..., j')$ are cost minim, deci $a(s, ..., j') = u_{j'}^F = u_j^F + a_{jj'}$, unde $j' \in V \setminus S$. Analog, $a(l', ..., t) = u_{l'}^B = u_l^B + a_{l'l}$, unde $l' \in V \setminus T$. Presupunem prin reducere la absurd că $\min\limits_{h \in V \setminus S} u_h^F > u_j^F + a_{jj'}$ și că $\min\limits_{k \in V \setminus T} u_k^B > u_l^B + a_{l'l} \Leftrightarrow \min\limits_{h \in V \setminus S} u_h^F > u_{j'}^F$ și $\min\limits_{k \in V \setminus T} u_k^B > u_{l'}^B$. Observăm că $h, j' \in V \setminus S$ și că $k, l' \in V \setminus T$. Dacă $\min\limits_{h \in V \setminus S} u_h^F$ ar fi mai mare decât $u_{j'}^F$, atunci $u_{j'}^F$ ar fi ales ca $\underset{h \in V \setminus S}{min} u_h^F$. Analog pentru $\underset{k \in V \setminus T}{min} u_k^B$. Contradicție $\Rightarrow \underset{h \in V \setminus S}{min} u_h^F \leq u_{j'}^F$ și $\underset{k \in V \setminus T}{min} u_k^B \leq u_{l'}^B \Rightarrow u_{j'}^F + u_{l'}^B \geq \underset{h \in V \setminus S}{min} u_h^F + \underset{k \in V \setminus T}{min} u_k^B \Rightarrow u_j^F + a_{jj'} + a_{l'l} + u_l^B \geq \underset{h \in V \setminus S}{min} u_h^F + \underset{k \in V \setminus T}{min} u_k^B$.

Avem $a(P_{st}) \geq u_j^F + a_{jj'} + a_{l'l} + u_l^B$ și $u_j^F + a_{jj'} + a_{l'l} + u_l^B \geq \underset{h \in V \setminus S}{min} u_h^F + \underset{k \in V \setminus T}{min} u_k^B$. Rezultă că $a(P_{st}) \geq u_j^F + a_{jj'} + a_{l'l} + u_l^B \geq u_j^F + a_{jj'} + a_{l'l} + u_l^B \geq \underset{h \in V \setminus S}{min} u_h^F + \underset{k \in V \setminus T}{min} u_k^B$.

c) Observăm că $after(x)$ și $before(y)$ întotdeauna sunt noduri din S, respectiv T. În funcție de *return*-ul pe care s-a intrat, avem două cazuri pentru drumul $p = s, ..., before(i), i, after(i), ..., t$:

1. $s, ..., before(i) \in S$ și $i, after(i), ..., t \in T$
2. $s, ..., before(i), i \in S$ și $after(i), ..., t \in T$