Tema 2 Seminar

Dumitrescu Delia Ioana

January 2020

Exercitiul 1

Fluxul maxim pe reteaua data este 4, acesta fiind obtinut astfel:

- Pe $1 \to 3 \to 4 \to 5$ trimitem fluxul 2
- Pe $1 \to 2 \to 5$ trimitem fluxul 1
- Pe $1 \to 3 \to 2 \to 4 \to 5$ trimitem fluxul 1

Fluxul gasit este maxim, intrucat exista o taietura minima egala cu valoarea egala cu cea a fluxului. O astfel de taietura este cea care pastreaza in prima multime nodul 1, iar in cea de-a doua multime nodurile 2, 3, 4, 5 (taie muchiile $1 \to 3$ respectiv $1 \to 2$). Valoarea acestei taieturi este egala cu fluxul trimis pe $1 \to 3$(adica 3) plus fluxul trimis pe $1 \to 2$ (adica 1), deci in total 4. Nu exista nicio taietura cu valoarea mai mica ca 4.

Exercitiul 2

Adaptarea Edmonds-Karp a algoritmului de determinare a fluxului maxim are complexitate de timp ce se incadreaza in $\Omega(E^2)$ pe cazul nefavorabil. Vom lua un exemplu care sa evidentieze acest caz. Fie reteaua descrisa printr-un nod sursa s, un nod destinatie t si inca E noduri. Vom avea urmatoarele muchii:

- Cate o muchie de la s la fiecare dintre cele E noduri.
- Cate o muchie de la fiecare dintre cele E noduri la t.

In retea exista $\Omega(E)$ drumuri, iar parcurgerea *BFS* va satura cate 1 drum per iteratie. Astfel algoritmul va rula $\Omega(E)$ iteratii, iar complexitatea algoritmului va fi $\Omega(E^2)$.

Exercitiul 3

Intr-o retea, orice muchie care porneste din t sau care ajunge in s nu va fi relevanta pentru fluxul maxim. Acest lucru se intampla deoarece unitatile care creeaza fluxul maxim pornesc din s si ajung in t. Un flux care intra in s trebuie sa fi plecat initial tot din s, iar apoi sa fie transportat in t, acesta nu poate ramane in s. Un flux care pleaca din t nu poate influenta nici el fluxul maxim. In mod evident, daca un anume flux pleaca din t, inseamna ca acesta a ajuns acolo pornind din s(fluxul care intra trebuie sa si iasa), deci va trebui sa se intoarca inapoi in t la un moment dat. Aceste muchii sunt niste simple "canale", ele nu pot absorbi sau genera flux si pot fi sterse.

In cadrul algoritmilor de flux maxim cunoscuti, pe aceste muchii nu se va intoarce flux niciodata. In momentul rularii unui BFS, se porneste din s. Sa spunem ca la un moment dat vom vizita un nod x, care are muchia $x \to s$. Algoritmul nu o va lua pe astfel de muchie, deoarece s a fost deja vizitat. De asemenea, algoritmul se incheie in momentul in care ajungem in destinatie, deci orice muchie care pleaca din t va fi ignorata.

Exercitiul 4

Gasirea fluxului maxim intr-o retea in care atat muchiile, cat si nodurile au constrangeri de capacitate este, in esenta, acelasi lucru cu gasirea fluxului maxim intr-o alta retea in care doar muchiile au constrangeri.

Sa luam un exemplu: Fie nodul 1 cu capacitatea 4 sursa, nodul 2 cu capacitatea 3 destinatia si muchia $1 \to 2$ cu costul 5. In aceasta retea se va putea trimite fluxul 3, minimul dintre cele 3 capacitati, indiferent daca aceste capacitati provin din constrangeri ale muchiilor sau ale nodurilor. Putem vedea capacitatea unui nod drept o muchie imaginara de la el insusi la el insusi, avand capacitatea nodului.

Prin urmare, pentru a rezolva problema, pentru fiecare nod, vom adauga unul nou. Fie nodul x un nod oarecare pe care il prelucram si c capacitatea sa. Sunt necesare urmatoarele schimbari:

- Eliminam capacitatea nodului x. Aceasta va fi adaugata unei muchii.
- Cream un nou nod y. Adaugam muchia $x \to y$ cu capacitatea c.
- Toate muchiile care ieseau din x vor iesi acum din y. Muchiile care intrau in x raman neschimbate.

Facand aceste schimbari, ne asiguram ca printr-un nod nu pot trece mai multe unitati decat capacitatea sa initiala, dar si ca pastram constrangerile initiale ale muchiilor.

Numarul de noduri se va dubla si vom adauga *numarul initial de noduri* muchii, insa complexitatea va ramane aceeasi.

Exercitiul 5

Vom modela problema ca una de flux maxim. Vom porni de la o sursa si o destinatie, fie ele s si t. Vom avea cate un nod, fie el x_i, pentru fiecare nod din arbore. Fiecare nod inafara de radacina are un parinte, fie el p_i. Pentru fiecare oras, stim ca avem s_i oameni si d_i locuinte. Pentru inceput, vom adauga, pentru fiecare nod x_i, 2 muchii, anume:

- Muchia $s \to x_i$ cu capacitatea s_i
- Muchia $x_i \to t$ cu capacitatea d_i

In esenta, "trimitem" un numar de oameni(fluxul care pleaca din s) si "cazam" un numar de oameni(fluxul care intra in t). Pentru moment, putem gasi o locuinta pentru un om doar in orasul sau. Totusi, un om poate locui in orice oras "in sus pe arbore", pana la radacina. Astfel, este necesar sa mai adaugam:

- Muchia $x_i \to p_i$. Aceasta muchie nu va avea nicio capacitate maxima. Putem trimite oricati oameni dintr-un oras intr-un oras parinte. Mai mult, deoarece fiecare nod va avea o astfel de muchie, putem trimite oricati oameni dintr-un oras(nod) in alt oras din drumul spre capitala(radacina), inclusiv in aceasta.

In acest punct, putem afla daca exista o strategie de relocare prin care toti oamenii sa aiba o locuinta. Fluxul maxim in retea va reprezenta numarul de oameni care si-au gasit o locuinta. Daca la final, fluxul maxim va fi egal cu numarul de oameni din toate orasele, atunci toti si-au gasit locuinta. Altfel, exista cativa pe care i-am "trimis" sa isi caute locuinta si pe care nu i-am "cazat". Acestia au pornit din sursa dar nu au reusit sa ajunga in destinatie.

Mai departe, sa vedem o rezolvare eficienta. Este natural ca pentru fiecare oras sa cazam cat de multi locuitori putem inainte de a-i trimite in alt oras. Incepand din frunze, vom satura fiecare oras cu minimul dintre numarul de locuitori si numarul de locuinte. Mai exact, vom trimite cat de mult flux de locuitori putem pe muchia $x_i \to t$.

- Daca numarul de locuitori e mai mic ca numarul de locuinte, ne-am terminat treaba cu orasul curent. Il putem sterge direct si adaugam fluxul curent la fluxul maxim.
- Daca numarul de locuitori e mai mare decat capacitatea orasului de a-i caza, va trebui sa ii trimitem in orasul parinte. Adaugam fluxul pe care l-am putut trimite la fluxul maxim si continuam.

Dupa cum am zis, locuitorii care nu au fost cazati vor fi trimisi in orasul parinte. Asta inseamna ca fluxul ramas(diferenta dintre d_i si s_i) va fi trimis pe muchia $x_i \to p_i$. Putem sa stergem muchia $x_i \to p_i$ si sa trimitem fluxul(diferenta dintre d_i si s_i) direct din s in p_i. Asta inseamna ca adaugam la fluxul trimis din s in p_i valoarea $(d_i - s_i)$. Putem acum sa stergem nodul x_i.

Vom face acest lucru incepand cu frunzele si vom sterge, pe rand, cate un nod, manipuland fluxul asa cum am explicat mai sus. In final, vom ramane doar cu sursa si destinatia. Rezultatul va fi fluxul maxim dintre cele 2 noduri.

Din punct de vedere al complexitatii, stergerea unui nod se face in $O(1)$, si stergem n noduri. Calcularea fluxului maxim va fi realizat tot in $O(1)$, astfel ca vom avea o complexitate totala de $O(n)$.

Exercitiul 6

Vom modela problema drept una de flux maxim de cost minim pe un graf bipartit. Pentru inceput, vom lua 2 noduri, o sursa s si o destinatie t. Avem nevoie si de 2 multimi ce vor contine echipele, fie ele L Si R. Multimea L va contine nodurile $x_{1L}, x_{2L} ..., x_{nL}$, iar multimea R va contine nodurile $x_{1R}, x_{2R} ..., x_{nR}$. Nodurile x_{iL} si x_{IR} vor reprezenta echipa x_i.

In continuare, vom adauga urmatoarele muchii:

- Cate o muchie de la s la fiecare nod din L cu capacitatea 1 si costul 0. Aceste muchii vor semnifica faptul ca din fiecare echipe trebuie sa transferam fix 1 jucator. (Am folosit cuvantul trebuie deoarece vom considera cazul in care o echipa nu realizeaza niciun transfer echivalent cu un transfer de la o echipa la ea insasi).
- Cate o muchie de la fiecare nod din R la t. Aceste muchii semnifica faptul ca fiecare echipa trebuie sa primeasca 1 jucator.
- Cate o muchie de la x_{iL} la x_{iR} (pentru fiecare i intre 1 si n) cu capacitatea 1 si costul 0. Alegerea acestei muchii va insemna ca o echipa si-a transferat jucatorul ei insasi.

In acest moment, fiecare echipa trebuie sa trimita si sa primeasca fix 1 jucator. Mai departe, pentru fiecare transfer din cerinta (a_i, b_i, c_i) vom trage o muchie astfel: de la nodul din L care reprezinta echipa a_i la nodul din R care reprezinta echipa b_i cu costul $(-c_i)$. Adaugam costul cu - deoarece algoritmul calculeaza costul minim, iar noi vrem o suma cat mai mare la final.

In sfarsit, putem rezolva problema. Fluxul maxim va fi egal cu numarul de jucatori care trebuie transferati. (In cel mai rau caz, transferam fiecare jucator la propria echipa si nu obtinem castig). Suma preturilor transferurilor va fi egala cu (-suma), unde suma = costul fluxului maxim de cost minim din retea.

Exercitiul 7

Vom modela problema folosind o retea de flux maxim pe un graf bipartit. Pentru inceput, vom lua 2 noduri, o sursa s si o destinatie t. O pereche dintr-o runda care a dansat va fi reprezentata printr-o unitate de flux ce intra in t. Avem nevoie si de 2 multimi ce vor contine fetele si baietii, fie ele F Si B, fiecare avand cardinalul n.

In continuare, vom adauga urmatoarele muchii:

- Cate o muchie de la s la fiecare nod din F cu capacitatea k (exista k runde, deci fiecare fata trebuie sa danseze de k ori)
- Cate o muchie de la fiecare nod din B la t cu capacitatea k (exista k runde, deci fiecare baiat trebuie sa danseze de k ori)
- Cate o muchie de la o fata f_i la un baiat b_i cu capacitatea 1(deoarece perechea trebuie sa existe intr-o singura runda) daca si numai daca atat fata vrea sa danseze cu baiatul, cat si baiatul vrea sa danseze cu fata. Restul preferintelor vor fi ignorate. Acest lucru se intampla deoarece, pentru a respecta preferintele in coregrafia finala, nu putem cupla o fata cu un baiat care nu se afla in preferintele sale sau invers.

Coregrafia s-a putut realiza daca fluxul maxim transmis prin retea este $n * k$ (in fiecare dintre cele k runde au dansat cate n fete si n baieti). Aceasta poate fi aflata prin crearea unui graf bipartit unde tragem muchii doar intre perechile care au fost alese in graful anterior(adica muchiile saturate). Apoi cautam cuplajul maxim pe graf, afisam perechile specifice acestui cuplaj iar apoi eliminam aceste muchii si continuam pe aceeasi idee.