Arborele de cost minim

Este dat un graf G = (V,E) ale cărui muchii au fost "ponderate" cu valori nenegative. Se doreşte determinarea unui **arbore** generator H^* = (V,A) al cărui "cost" – dat de suma costurilor muchiilor alcătuitoare – să fie **minim**.

(Recomandăm cititorului să revadă noţinile de teoria grafurilor prezentate în unitatea de învăţare 4, secţiunea 4.1)

Următorul algoritm construieşte arborele "minimal" H^* "muchie cu muchie" şi ca urmare construcţia ar trebui să se termine în $n - 1$ paşi,dacă graful are n noduri (după cum este cunoscut, un arbore cu n noduri are exact $n - 1$ muchii…) Terminarea "mai devreme" a algoritmului semnalează faptul că graful original nu este conex şi în consecinţă nu are arbori generatori.

Algoritmul lui Kruskal (1956)

Start Muchiile grafului G vor fi cercetate în ordinea nedescrescătoare a costurilor asociate (muchiile cu acelaşi cost vor fi ordonate arbitrar).
Se selecteză o muchie e^1 cu cel mai mic cost posibil.

Pasul 1 Fie e^1 , e^2 ,…, e^k muchiile deja alese în etapele anterioare. Dacă $k = n - 1$, n fiind numărul nodurilor grafului, **Stop**: muchiile selectate sunt muchiile unui arbore (generator) de cost minim. Dacă $k < n - 1$ se trece la:

Pasul 2 Printre muchiile încă necercetate se caută o muchie, e^{k+1} , cu cel mai mic cost posibil şi care nu formează un ciclu cu (o parte din) muchiile e^1 , e^2 ,…, e^k. dacă o asemenea muchie nu există, **Stop**: graful G nu este conex şi prin urmare nu conţine arbori generatori. Dacă e^{k+1} există, ea se adaugă la muchiile deja selectate şi se revine la pasul 1.

Problema postasului chinez

Un poştaş ridică de la oficiu corespondenţa de distribuit, străbate străzile din zona repartizată lui şi împarte corespondenţa destinatarilor după care se întoarce la locul de plecare. Cum trebuie să se deplaseze poştaşul pentru ca distanşa totală parcursă să fie minimă?

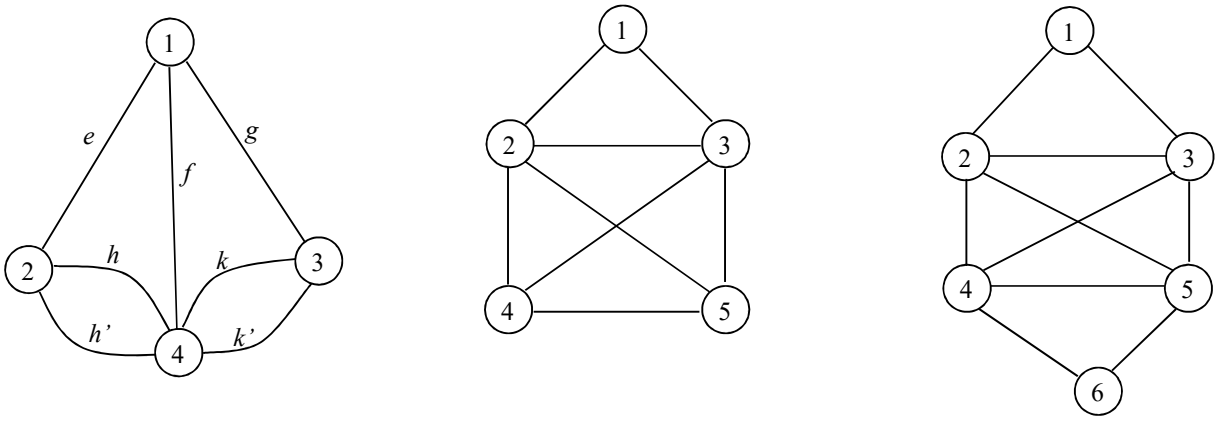
Chestiunea devine una din cele mai cunoscute şi studiate probleme de optimizare combinatorială în urma cercetărilor marelui combinatorist JACK EDMONDS care dă şi prima rezolvare eficientă în 1965.

Colectarea gunoiului menajer, deszăpezirea străzilor şi răspândirea de materiale antiderapante sau stabilirea traseelor patrulelor de poliţie sunt numai câteva din numeroasele aplicaţii practice ale problemei poştaşului chinez.

Deoarece o reţea stradală se reprezintă de obicei printr-un graf, CPP va fi modelată în termenii teoriei grafurilor. Sunt necesare câteva pregătiri (în completarea secţiunii 4.1 din unitatea de învăţare 4)

8.2 Grafuri euleriene

Fixăm un **graf** G = (V,E) în care V este mulţimea **nodurilor** şi E este mulţimea **muchiilor**. Pentru simplitate, vom presupune că G **nu este orientat**.
Un **ciclu eulerian** în graful G este un **ciclu** care conţine **toate muchiile** din G exact **o singură dată**.



a) b) c)

Figura 8.1

Reamintim că graful G se zice **conex** dacă oricare două noduri diferite sunt extremităţile unui lanţ de muchii (cel puţin).

Este clar că **dacă un graf are un ciclu eulerian el este conex**. **Reciproca nu este în general adevărată**; de exemplu grafurile din figurile 8.1a) şi b), deşi sunt conexe, nu au cicluri euleriene
(vom vedea de ce!). În schimb, graful din figura 8.1c) are ciclul eulerian.

$$1 — 2 — 5 — 4 — 3 — 2 — 4 — 6 — 5 — 3 — 1$$

care nu este şi singurul; succesiunea

$$1 — 2 — 3 — 5 — 4 — 6 — 5 — 2 — 4 — 3 — 1$$

este un alt ciclu eulerian în acelaşi graf.

Un graf se zice eulerian dacă posedă un ciclu eulerian.

Reamintind că **gradul** unui nod dintr-un graf este **numărul muchiilor** având o extremitate în acel nod, avem următoarea caracterizare a grafurilor euleriene:

Teorema 1 (EULER, HIERHOLZER) Un graf conex este eulerian dacă şi numai dacă toate nodurile sale au grad par.

Un ciclu al unui graf se zice **elementar** dacă toate nodurile prin care trece sunt diferite. Două cicluri se vor numi **disjuncte** dacă nu au muchii comune (pot avea însă noduri comune!)

Următoarea caracterizare alternativă a grafurilor euleriene ne va fi utilă.

Teorema 2 (VEBLEN) Un graf conex este eulerian dacă şi numai dacă este o reuniune de cicluri elementare disjuncte.

Grafurile conexe din figurile 8.1a) şi b) au noduri de grad impar şi în virtutea teoremei 1 nu sunt euleriene. Graful din figura 8.1c) este eulerian pentru că toate nodurile sale au gradul par. În baza teoremei 2 acest graf se poate descompune în mai multe cicluri elementare disjuncte.
Descompunerea nu este unică: figura 8.2 evidenţiază două descompuneri distincte ale grafului amintit.

Euristica: dublează muchiile unui arbore minimal

- În graful complet cu nodurile 0,1,...,*n* se determină un **arbore** H de valoare(lungime) minimă. Aceasta este o problemă de optimizare **uşoară** rezolvabilă cu algoritmul lui Kruskal – vezi unitatea de învăţare 5.

- În continuare fiecare muchie din H se înlocuieşte cu două muchii de aceeaşi lungime. Se obţine un multigraf H'.

Prin construcţie, H' este un **graf eulerian** (deoarece gradele tuturor nodurilor sunt pare!) şi ca urmare există posibilitatea traversării **tuturor** muchiilor, **o singură dată,** cu plecarea şi întoarcerea în nodul 0.

- Fie

$$0 \to x \to y \to \cdots\cdots \to u \to v \to 0 \qquad (*)$$

Succesiunea în care toate muchiile multigrafului H' au fost parcurse. În aceeastă secvenţă este posibil ca unul sau mai multe oraşe să apară de mai multe ori. Procedăm la următoarea operaţie de **scurtcircuitare**:

În succesiunea (*) se determină **prima** tripletă $i \to j \to k$ de noduri (oraşe) consecutive în care „mijlocul" j mai apare ulterior în succesiune. Înlocuim secvenţa $i \to j \to k$ cu arcul direct $i \to k$. Prin această operaţie:

- fiecare oraş continuă să fie vizitat măcar o dată;
- noul traseu are o lungime mai mică deoarece $c_{ij} + c_{jk} \geq c_{ik}$ (atenţie, avem în vedere în exclusivitate TSP euclidiene!!)

Scurtcircuitarea se repetă până când, în secvenţa (*) **actualizată,** fiecare oraş apare **o singură dată**, bineînţeles cu excepţia punctului de plecare şi întoarcere 0.