**Najkrajše poti**

**7.1. Najkrajše poti iz danega izhodišča *s***

*Podatki:* usmerjen graf *G*, vozlišče *s* iz *V(G)*

*Rezultat:* aciklični graf *LG* vseh najkrajših poti iz vozlišča s do preostalih vozlišč grafa *G*

Graf *LG* konstruiramo s ***pregledovanjem v širino*** in z uporabo vrste.

Časovna zahtevnost tega postopka je *O*(*m*), kjer je *m* = |*E(G)|*.

**7.2. Najcenejše poti iz danega izhodišča *s***

*Podatki:* usmerjen graf *G*, vozlišče *s* iz *V(G)*, nenegativne cene *cij* za vse povezave *ij* iz *E(G)*

*Rezultat:* drevovseh najcenejših poti iz vozlišča s do preostalih vozlišč grafa *G*

Problem najkrajših poti v grafu brez uteži iz točke 7.1 je poseben primer problema najcenejših poti v grafu, v katerem so vse uteži enake 1.

Spoznali smo ***Dijkstrov algoritem***, kipoišče najcenejše poti od izbranega vozlišča *s* do vseh preostalih vozlišč grafa.

Utemeljili smo pravilnost Diskstrovega algoritma. Ugotovili smo, da je časovna zahtevnost Dijkstrovega algoritma O(*n*2), kjer je *n* število vozlišč grafa.

**7.3. Najcenejše poti med vsemi pari vozlišč**

*Podatki:* usmerjen graf *G*, realne cene *cij* za vse povezave *ij* iz *E(G)*

*Rezultat:* matrika cen najcenejših poti za vse pare vozlišč (*i, j*) grafa *G*

Ogledali smo si ***Floyd-Warshallov algoritem***, ki v primeru, ko graf ne vsebuje usmerjenih ciklov z negativno ceno, poišče najcenejše poti za *vse* pare vozlišč (*i, j*) iz množice *V(G)* x *V*(G).

Floyd-Warshallov algoritem smo dopolnili tako, da poleg tabele *d* dolžin najcenejših poti za vse pare vozlišč poišče tudi tabelo *oče*, katere (*i,j*)-ti element je kazalec na neposrednega predhodnika vozlišča *j* na najcenejši poti od *i* do *j.* Floyd-Warshallov algoritem deluje pravilno tudi v primeru, ko so cene posameznih povezav negativne, vendar graf ne vsebuje usmerjenega cikla z negativno vsoto cen povezav.

Ugotovili smo, da je njegova časovna zahtevnost O(*n*3).

Na primeru smo pogledali, kako deluje Floyd-Warshallov algoritem. Obravnavali smo tudi primer, ko ima graf cikle z negativno skupno ceno. V tem primeru Floyd-Warshallow algoritem ne deluje pravilno.

### 7.4. Kitajski problem poštarja

Poštar začne svoj vsakodnevni obhod na pošti. Prehoditi (ali prevoziti) mora vse ulice v svojem okolišu in se na koncu vrniti na pošto. Kako naj načrtuje obhod, da bo skupna prehojena (ali prevožena) razdalja čim manjša?

Matematični model: *Dan je neusmerjen povezan graf G s pozitivnimi cenami povezav. Poišči obhod, ki vsebuje vsako povezavo vsaj enkrat in ima najmanjšo skupno ceno!*

Najmanjšo ceno bi imel obhod, ki vsebuje vsako povezavo natanko enkrat, torej Eulerjev obhod. Ta pa obstaja natanko tedaj, ko so vsa vozlišča grafa *G* sode stopnje. V tem primeru rešimo problem tako, da poiščemo Eulerjev obhod v *G*.

Ostane še primer, ko *G* vsebuje vozlišča lihe stopnje. Teh je sodo mnogo, npr. 2*k*. V tem primeru ravnamo takole:

1. Naj bo *T* množica lihih vozlišč grafa *G*. S Floyd-Warshallovim algoritmom izračunamo cene *dij* najcenejših poti za vse pare vozlišč *i, j* iz *T*.

2. V polnem grafu na množici *T* z utežmi *dij* na povezavah poiščemo najcenejše popolno prirejanje. Tako dobimo pare lihih vozlišč (*ui, vi*) za *i* = 1, 2, ..., *k*.

3. Naj bo *Pi* najcenejša pot od *ui* do *vi* v grafu *G*, za *i* = 1, 2, ..., *k*. V grafu *G* podvojimo vse povezave, ki ležijo na kateri od poti *Pi*. Tako dobimo multigraf *G'*, v katerem so vsa vozlišča sode stopnje. V *G'* poiščemo Eulerjev obhod. To je tudi optimalni poštarjev obhod v *G*.