Práce s grafy

Programování 2 (NMIN102)

RNDr. Michal Žemlička, Ph.D.

Reprezentace grafů

Grafy můžeme reprezentovat mnoha různými způsoby.

Použitelnost a vhodnost jednotlivých reprezentací je dána vlastnostmi grafu a očekávaného použití dané reprezentace (jaké algoritmy ji budou používat).

Dijkstrův algoritmus

- hledá nejkratší cesty v grafu ze zvoleného vrcholu do vrcholů ostatních
- hledáme-li cestu do konkrétního vrcholu, můžeme algoritmus zastavit při dopočítání finální hodnoty pro daný vrchol
- pracuje s grafy, jejichž hrany jsou ohodnoceny nezápornými hodnotami

Dijkstrův algoritmus – co potřebujeme

- vrcholy
  - stav rozpracovanosti
  - vzdálenost od startu
  - vzdálenost od skupiny
  - odkud jsme se tam dostali
- hrany – cena

Dijkstrův algoritmus – co potřebujeme (2)

- rychlé nalezení prvku, co je k vyřešeným uzlům nejblíže
- rychlé nalezení sousedů daného uzlu
- rychlá aktualizace hodnot sousedů + rychlé dořešení důsledků

Dijkstrův algoritmus – co umíme

Rychlé nalezení prvku:

- halda – rychle najdeme minimum
- pole – rychle najdeme daný prvek

Dijkstrův algoritmus – co by mohlo jít hůře

- umíme efektivně udržovat množinu?
- umíme rychle najít sousedy?

Dijkstrův algoritmus – hledání sousedů

- je-li stupeň uzlu velmi vysoký, stačí matice s cenami hran
  (musíme ale pohlídat, který uzel je v jaké skupině)
- pro malý počet sousedních uzlů by byl výhodnější jejich seznam
  (i zde ale musíme pohlídat, který uzel je v jaké skupině)

Dijkstrův algoritmus – udržování struktur

- musíme udržet strukturu s rozpracovanými uzly (co umí rychle najít ten nejbližší)
  - musíme umět najít sousedy, změnit jejich vzdálenost a aktualizovat jejich pozici ve struktuře
  - potřebujeme udržet vazbu s dalšími informacemi o uzlech (zejména s kým sousedí, ale i v jaké je skupině, jak jsme se k němu dostali, ...)
  - musíme být schopni uzel ve struktuře rychle najít

Dijkstrův algoritmus – variace

- nalezení cesty jen pro jeden cílový vrchol
- hledání cest v síti VHD (hrany jsou k dispozici jen pro určité okamžiky)
  - v různých chvílích mohou být nejkratší různé cesty
  - můžeme sledovat více parametrů (dobu, cenu, přestupy,...)
- hledání minimální kostry (Jarník)

Floyd-Warshallův algoritmus

- máme graf nad n vrcholy
- hledáme nejkratší cesty z každého do každého
- předpokládáme, že ohodnocení hran netvoří záporné cykly
- matice – na diagonále nuly, mimo diagonálu hodnota hrany, nebo ∞

Floyd-Warshallův algoritmus – idea

- budeme postupně zvyšvat počty hran, které budeme brát v potaz pro výpočet délky nejkratší cesty
- nalezení cesty větší délky můžeme brát jako propojení vhodných cest kratších délek
- je-li matice D^m maticí kandidátních cest z i do j do délky nejvýše m, můžeme spočítat $D^m_{i,j}$ jako spojení vhodných kombinací cest z D^{m-1} a D^1.

Floyd-Warshallův algoritmus – jednoduše

Zavedeme operaci $\oplus$ pro vektory u, v dimenze n:

$u \oplus v = min_{i=1}^{n} u_i + v_i$

Podobně zavedeme operaci $\otimes$ nad maticemi (podobnou násobení matic), jako operaci nad jejich složkami: