Algoritmer och komplexitet
Hösten 2009
Mästarprov 1: Algoritmer

Mästarprovet ska lösas **individuellt** och redovisas både skriftligt och muntligt. Inget samarbete är tillåtet, se vidare hederskodexen.

Skriftliga lösningar ska lämnas senast **måndag 16 nov klockan 17.00** i mitt postfack eller i kursens inlämningslåda på studentexpeditionen på Osquars backe 2 plan 2. Det är viktigt att du lämnar in i tid! Se till att spara en kopia av dina lösningar så att du kan läsa på inför den **muntliga redovisningen** som kommer att ske dagarna efter inlämningen.

Mästarprovet är ett obligatoriskt och betygsatt moment i kursen. Det består av fyra uppgifter med olika svårighetsgrad. För betyg E krävs rätt på två av uppgifterna. Sedan är betygssättningen i stort sådan att helt rätt på tre uppgifter ger betyg C och helt rätt på alla uppgifter ger betyg A.

Läs mer om betygskriterier och slutbetyg i kurs-PM eller på kursens webbsida.

1. Valutaväxling

Landet Brutopia har valutan brutos. I landet finns det fem typer av sedlar. De har valörerna 50, 40, 10, 5, 1 brutos. Sedlarna är gamla och regeringen i Brutopia vill därför att man skall använda så få sedlar som möjligt för att inte slita på dem i onödan. Därför skall man alltid betala med exakt rätt summa och göra det med så få sedlar som möjligt. Vi har därmed följande problem:

Givet ett heltal A vill vi skriva

$$A = k_{50}50 + k_{40}40 + k_{10}10 + k_5 5 + k_1$$

där $k_{50} + ... + k_1$ skall vara minimal.

Hjälp brutopierna genom att designa en algoritm som löser detta problem. Det skall vara en girig algoritm som har tidskomplexitet $O(\log A)$.

2. Kortaste vägar

Vi har sett exempel på flera algoritmer som beräkna kortaste avstånd mellan noder i viktade grafer. Vi har då också sett att en kortaste väg inte alltid är den väg som har minst antal kanter. Men låt oss nu arbeta med två avståndsbegrepp samtidigt. Vi sätter

$l(v)$ = längden av vägen v mätt som en summa av kantvikterna.

$l_{\#}(v)$ = längden av vägen v mätt som antalet kanter.

Vi tänker oss att vi har en graf där det kan finnas flera vägar av samma längd (i $l(v)$-mening) mellan två noder. Vi vill då välja en sådan väg som är minimal i $l_{\#}(v)$-mening bland dem. Mer exakt:

Vi söker en väg v^*, $i \to j$, så att

1. $l(v^*) \leq l(v)$ för alla vägar v: $i \to j$

2. $l_{\#}(v^*) \leq l_{\#}(v)$ för alla vägar v: $i \to j$ så att $l(v^*) = l(v)$

Bestäm en algoritm som hittar en sådan väg. För full poäng krävs att algoritmen fungerar i grafer med negativa kantvikter men som saknar negativa cykler.

3. Packa paket

Det miljövänliga budföretaget Godbud transporterar varor i lastbilar. Varorna är i form av olika paket $p_1, p_2, ..., p_n$. Dessa paket har vikter $w_1, w_2, ..., w_n$ kg. De skall lastas i lastbilar som klarar att ta precis M kg. Paketen skall lastas i lastbilarna i rätt ordning. Det betyder att paket $p_1, p_2, ..., p_{s_1}$ packas i lastbil 1 för något s_1. I bil två packas då paket $p_{s+1}, p_{s+2}, ..., p_{s_2}$ för något s_2 o.s.v. Då måste förstås $w_{s_i+1} + ... + w_{s_{i+1}} \leq M$ för alla i. Miljötrafiköversynsverket ställer nu ett ytterligare krav på bolaget. Godbud måste lasta lastbilarna så att spillet $d_i = M - (w_{s_i+1} + ... + w_{s_{i+1}})$ blir så litet som möjligt för alla i. Mer precist krävs det att $D = \sum_{i=1}^{n} d_i$ skall bli så litet som möjligt.

Konstruera en effektiv algoritm som löser detta problem d.v.s. bestämmer vad optimalt D är och hur $s_1, s_2, ...$ skall väljas.

4. Leta i kvadrat

Vi har sett att om vi vill avgör om ett visst element a finns i en lista $x_1, x_2, ..., x_n$ så krävs det $O(n)$ operationer om listan är osorterad. Om listan däremot är sorterad går det att avgöra om a finns i listan med $O(\log n)$ operationer.

Antag nu att vi har en $n \times n$-matris med n^2 tal x_{ij} som är sorterade i växande ordning längs rader och kolumner. Vi kan också anta att alla tal är olika. Det betyder alltså att $j < k \Rightarrow x_{ij} < x_{ik}$ och $x_{ji} < x_{ki}$ för alla i.

Vi vill nu avgöra om talet a finns bland x_{ij}-talen. Det är lätt att se att det går att göra med $O(n \log n)$ operationer. Men vi skall försöka göra bättre. Använd dekomposition. Går det att göra med $O(n)$ operationer eller bättre?