

Videregående algoritmik

Projektudvælgelse

David Pisinger, *Projekt opgave 2*, blok 2, DIKU, 2006-07

Dette er den anden obligatoriske projektopgave på kurset "Videregående algoritmik". Opgaven stilles tirsdag 12. december 2006 og skal afleveres senest tirsdag 9. januar 2007 kl. 10.30 i informationen på DIKU.

For at blive godkendt skal der være gjort et reelt forsøg på at løse samtlige spørgsmål (undtagen ekstraopgaverne). Besvarelsen skal udarbejdes i grupper på to til tre deltagere. Grupper med én deltager kræver forhåndsaccept fra instruktoren senest 2/1.

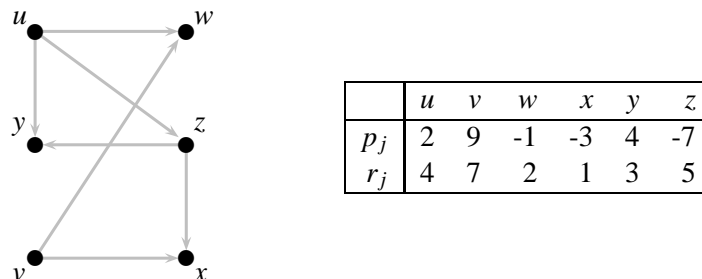
Læs venligst hele opgaveformuleringen igennem inden du går i gang. Hints til opgaverne kan fås ved øvelserne, hvor der er afsat tid til at arbejde med projektopgaven.

1 Problemet

Vi ønsker at udvælge et antal projekter således at vores fortjeneste bliver maximeret. Nogle projekter giver overskud, mens andre giver underskud. Endvidere kan der være bindinger mellem projekterne, således at valget af et projekt kræver at en mængde andre projekter også er valgt. F.eks. vil åbning af en internetforretning (et projekt som forventes at give overskud) kræve at man køber en stor server (et projekt som koster penge). Samtidig har vi for hvert projekt et ressourceforbrug (som f.eks. udtrykker hvor meget tid det tager at udføre projektet) samt en øvre grænse på den tilgængelige ressource.

Lidt mere formelt, har vi en mængde V af projekter. Hvert projekt $i \in V$ har en tilhørende profit $p_i \in \mathbb{Z}$ som kan være positiv eller negativ. Endvidere har hvert projekt et tilhørende ikke-negativt ressourceforbrug $r_i \in \mathbb{N}$ og vi har en øvre grænse (kapacitet) d på ressourceforbruget. Endvidere er der givet en orienteret graf $G = (V, E)$, kaldet *kravgrafen*, hvor en kant $(i, j) \in E$ angiver at hvis projekt i vælges så kræves det at projekt j også vælges.

Figur 1 viser en instans med kravgrafen $G = (V, E)$. Projekterne $\{w, x\}$ udgør en lovlig løsning, mens projekterne $\{v, x\}$ er en ulovlig løsning idet projekt v kræver projekt w .



Figur 1: En instans af PROJECT-SELECTION problemet med $V = \{u, v, w, x, y, z\}$. Kapaciteten af ressourcen er $d = 10$.

Hvis vi indfører en binær variabel for hvert projekt $i \in V$ hvor $x_i = 1$ hvis og kun hvis projekt i vælges, skal der gælde at

$$x_i = 1 \Rightarrow x_j = 1, \quad (i, j) \in E \quad (1)$$

hvilket kan omskrives til

$$x_i \leq x_j, \quad (i, j) \in E \quad (2)$$

Dermed kan problemet PROJECT-SELECTION formuleres på følgende vis

$$\text{maximer } \sum_{i \in V} p_i x_i \quad (3)$$

$$\text{hvor } \sum_{i \in V} r_i x_i \leq d \quad (4)$$

$$x_i \leq x_j, \quad (i, j) \in E \quad (5)$$

$$x_i \in \{0, 1\}, \quad i \in V \quad (6)$$

Begrænsning (4) udtrykker at ressourceforbruget af de valgte projekter ikke må overstige kapaciteten d . Begrænsning (5) sikrer at afhængighederne mellem projekterne er overholdt.

Opgave 1 Formuler PROJECT-SELECTION som et afgørlighedsproblem PROJECT-SELECTION-DECISION ■

Opgave 2 Vis at PROJECT-SELECTION-DECISION er NP-fuldstændig ved reduktion fra SUBSET-SUM ■

2 Grænseværdier og branch-and-bound

Vi sletter ressource-begrænsningen (4) og tilføjer differencen mellem ressourceforbruget og kapaciteten d i objektfunktionen. Hermed fremkommer følgende problem RELAXED-PROJECT-SELECTION

$$\text{maximer } \sum_{i \in V} p_i x_i - \lambda \left(\sum_{i \in V} r_i x_i - d \right) \quad (7)$$

$$\text{hvor } x_i \leq x_j, \quad (i, j) \in E \quad (8)$$

$$x_i \in \{0, 1\}, \quad i \in V \quad (9)$$

Opgave 3 Vis at RELAXED-PROJECT-SELECTION er en relaxering af PROJECT-SELECTION for $\lambda \geq 0$. ■

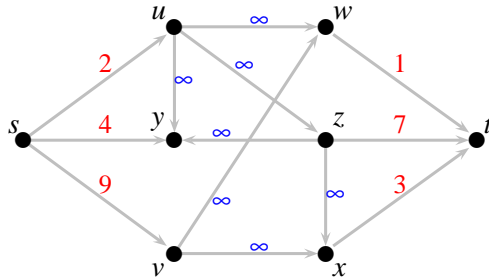
Hvis vi sætter $q_i = p_i - \lambda r_i$ i ovenstående model får vi problemet UNCAPACITATED-PROJECT-SELECTION givet ved

$$\text{maximer } \sum_{i \in V} q_i x_i + \text{konstant} \quad (10)$$

$$\text{hvor } x_i \leq x_j, \quad (i, j) \in E \quad (11)$$

$$x_i \in \{0, 1\}, \quad i \in V \quad (12)$$

Dette problem kan løses i polynomiel tid ved at løse et MAXIMUM-FLOW problem. Ved transformationen får alle kanter i E kapaciteten ∞ . Vi tilføjer to nye knuder s, t . Hvis en knude (projekt) $i \in V$ har profit $q_i > 0$ tilføjer vi en orienteret kant (s, i) med kapacitet q_i . Hvis en knude $i \in V$ har profit $q_i < 0$ tilføjer vi derimod en kant (i, t) med kapacitet $-q_i$. Figur 2 viser et eksempel på denne transformation.



Figur 2: Transformation af UNCAPACITATED-PROJECT-SELECTION til et MAXIMUM-FLOW problem. Koefficienterne svarer til instansen i figur 1 med $\lambda = 0$ benyttet i (10).

Opgave 4 Løs MAXIMUM-FLOW problemet for instansen i figur 2 og angiv det tilhørende minimale snit. ■

Betragt nu en generel instans af UNCAPACITATED-PROJECT-SELECTION. Antag at det tilhørende MAXIMUM-FLOW problem er blevet løst for den transformerede graf, og lad (S, T) være et minimalt snit. Betragt mængden af projekter A givet ved $A := S \setminus \{s\}$.

Opgave 5 Vis at hvis vi sætter $x_i = 1$ for $i \in A$ (og $x_i = 0$ ellers) vil dette være en lovlig løsning til UNCAPACITATED-PROJECT-SELECTION (Hint: vis at en kant $(i, j) \in E$ hvor $i \in A$ mens $j \notin A$ ikke kan ligge på snittet (S, T)) ■

Opgave 6 Vis at den optimale løsningsværdi z til UNCAPACITATED-PROJECT-SELECTION er givet ved

$$z = \sum_{v \in V: q_v > 0} q_v - c(S, T)$$

■

Opgave 7 For instansen fra figur 1 tegn løsningsværdien til RELAXED-PROJECT-SELECTION som funktion af λ i intervallet $0 \leq \lambda \leq 5$. ■

Opgave 8 Beskriv en algoritme som for enhver instans kan finde den værdi af λ i RELAXED-PROJECT-SELECTION der giver den strammeste grænseværdi. ■

Opgave 9 Implementer algoritmen til at finde den værdi af λ som giver den strammeste grænseværdi. Anvend algoritmen på de instanser som findes på kursets hjemmeside og rapporter for hver instans den bedste værdi af λ samt den tilhørende grænseværdi. ■

Vi vil nu løse PROJECT-SELECTION problemet med branch-and-bound. I hver branch-and-bound knude tildeles en beslutningsvariabel $x_i \in V$ en af de to binære værdier.

Opgave 10 Beskriv en fornuftig søgestrategi: Hvilken variabel er det mest hensigtsmæssigt at forgrene på? Hvilket delproblem undersøges først? Hvorledes kan det tilbageværende problem reduceres hvis kravgrafen skal være overholdt? ■

Opgave 11 Implementer en branch-and-bound algoritme som løser PROJECT-SELECTION problemet. Benyt den øvre grænseværdi fra opgave 9. Kør algoritmen for de instanser som findes på kursets hjemmeside og rapporter for hver instans: den optimale løsningsværdi, forskellen mellem den øvre grænseværdi i rodknuden og den optimale løsningsværdi (absolut værdi og afvigelse i procent), antal branch-and-bound knuder, samt forbrugt CPU-tid. ■

3 Dynamisk Programmering (ekstraopgave)

PROJECT-SELECTION kan også løses med dynamisk programmering hvis kravgraphen $G = (V, E)$ er et træ.

Opgave 12 (ekstraopgave) Beskriv en dynamisk programmerings algoritme som løser problem (13), og angiv dens kompleksitet. ■

Vi betragter nu en variant af projektudvælgelsesproblemet hvor kapacitetsbegrænsningen (4) alene kræver at der højst må vælges k projekter. Dermed kan LIMITED-PROJECT-SELECTION problemet formuleres som

$$\begin{aligned}
 &\text{maximer} && \sum_{i \in V} p_i x_i \\
 &\text{hvor} && \sum_{i \in V} x_i \leq k \\
 &&& x_j \leq x_i, \quad (i, j) \in E \\
 &&& x_i \in \{0, 1\}, \quad i \in V
 \end{aligned} \tag{13}$$

Opgave 13 (ekstraopgave) Vis at LIMITED-PROBLEM-SELECTION kan løses i polynomiel tid hvis kravgraphen $G = (V, E)$ er et træ. ■

Noter

I implementeringsdelen benyttes et rammeprogram skrevet i C. Rammeprogrammet kan indlæse instanser fra hjemmesiden og rummer en simpel implementering af FORD-FULKERSON algoritmen til løsning af MAXIMUM-FLOW problemet. Rammeprogrammet findes på kursets hjemmeside sammen med alle instanser. De benyttede instanser er:

filnavn	n	z^*
figur1.txt	6	5
randm10.txt	10	24
randm20.txt	20	40
randm30.txt	30	22
randm40.txt	40	68
randm60.txt	60	
randm80.txt	80	
randm100.txt	100	
randm120.txt	120	
randm140.txt	140	
randm160.txt	160	
randm180.txt	180	
randm200.txt	200	

De optimale løsningsværdier er angivet i søjlen z^* for de små instanser. Alle instanser bør kunne løses i løbet af 10 sekunder.

Litteratur

- [1] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, MIT Press, 2003.
- [2] D. Pisinger Branch-and-bound, Noter, Videregående Algoritmik, DIKU, 2006-07.