

Introduktion til optimering og operationsanalyse

Asymmetric Traveling Salesman Problem

David Pisinger, Efterår 2003

Dette er den anden obligatoriske projektopgave på kurset “Introduktion til optimering og operationsanalyse”. Opgaven skal afleveres senest 5. december 2003 kl. 12.00 i DIKU’s 1. del-administration. Besvarelsen skal udarbejdes i grupper på en til tre deltagere. Læs venligst hele opgaveformuleringen igennem inden du går igang.

Opgaverne vil blive rettet senest 8/12 således at en eventuel genaflevering kan ske 12/12. Dermed burde alle besvarelser være rettet inden eksamen.

Indledning

Et IP-problem kan ofte formuleres på mange måder. Selv om de matematisk set er ækvivalente, kan en “stærk” formulering være at foretrække, idet den er tættere på det konvekse hylster indeholdende alle IP-løsninger til problemet [7].

Traveling Salesman Problemet (TSP) er et klassisk, svært optimeringsproblem som vi kun kan løse takket være stærke formuleringer af problemet [1, 2, 3, 4]. Opgavens formål er at eksperimentere med forskellige formuleringer af TSP samt at kombinere to formuleringer i et “cutting plane” system.

Traveling Salesman Problemet

Lad der være givet en komplet graf $G = (V, E)$ hvor hver kant $(i, j) \in E$ har en tilhørende omkostning c_{ij} . Vi antager at problemet ikke nødvendigvis er symmetrisk, dvs. der kan være situationer hvor $c_{ij} \neq c_{ji}$. Traveling Salesman Problemet har til opgave at finde den korteste Hamilton kreds i grafen, dvs. en kreds som besøger alle knuder netop een gang, og som minimerer den tilhørende omkostning af kanterne.

Lad n betegne antallet af knuder i V . For at formulere problemet som et IP problem, kan man indføre beslutningsvariablene

$$x_{ij} = \begin{cases} 1 & \text{hvis kant } (i, j) \text{ indgår i kredsen} \\ 0 & \text{ellers} \end{cases}$$

for $i, j = 1, \dots, n$. For at gøre skrivemåden nemmere vil vi i det følgende antage at $x_{ii} = 0$ for $i = 1, \dots, n$.

Idet man skal ankomme til hver knude netop een gang, og skal forlade hver knude een gang, vil en naiv formulering af TSP være

$$\begin{aligned}
 & \text{minimize} && \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij} \\
 & \text{subject to} && \sum_{i=1}^n x_{ij} = 1, && j = 1, \dots, n \\
 & && \sum_{j=1}^n x_{ij} = 1, && i = 1, \dots, n \\
 & && x_{ij} \in \{0, 1\}, && i, j = 1, \dots, n
 \end{aligned} \tag{1}$$

Ovenstående begrænsninger kaldes “assignment” begrænsninger.

Opgave 1 Vis at selv om man løser LP-relaxeringen af ovenstående problem, vil man altid finde en heltallig løsning. ■

Desværre er formuleringen (1) ikke tilstrækkelig til at løse TSP, idet en optimal løsning kan indeholde delture. For at forhindre disse kan man tilføje følgende deltur begrænsninger

$$\sum_{i,j \in S} x_{ij} \leq |S| - 1 \tag{2}$$

for alle $S \subset V$ hvor $2 \leq |S| \leq n - 1$.

Opgave 2 Angiv hvor mange uligheder af formen (2) der vil være. ■

Miller, Tucker og Zemlin [5] foreslog en formulering som forhindrer delture, men som kun er polynomielt stor. Ideen er at indføre nogle nye variable u_i for hver knude, som angiver rækkefølgen af knuden i turen. Knude 1 har altid $u_1 = 1$ og hvis $u_i = k$ betyder det at knude i er den k 'te besøgte knude på turen. I sagens natur er $u_i \leq n$ for alle knuder $i \in V$. Endvidere skal vi kræve at

$$\text{hvis kant } (i, j) \text{ benyttes i turen, så skal } u_j \geq u_i + 1 \tag{3}$$

for alle $i = 1, \dots, n$ og $j = 2, \dots, n$ hvor $i \neq j$. Disse begrænsninger kaldes “MTZ” begrænsninger.

Opgave 3 Formuler MTZ begrænsningerne som IP-model. Angiv hvor mange uligheder der vil være. ■

Opgave 4 Løs problemet burma14 med begge formuleringer, ved brug af rammeprogrammet beskrevet sidst i opgaven. Angiv modellens størrelse (i bytes), samt løsnings tid for CPLEX. ■

Trods MTZ formuleringens polynomielle størrelse, er der i litteraturen ikke rapporteret anvendelser af denne model som kunne løse TSP problemer med mere end 50 knuder.

Opgave 5 Lad C være en kreds i grafen. Summer begrænsningerne (3) formuleret som LP-model over alle kanter $(i, j) \in C$. Ved summationen vil u_j leddene gå ud mod hinanden. Sammenlign den resulterende ulighed med uligheden

$$\sum_{(i,j) \in C} x_{ij} \leq |C| - 1 \quad (4)$$

fra deltur formuleringen. Hvilken af ulighederne er stærkest, og hvilken af formuleringerne ville man derfor foretrække? ■

Vi vil nu udvikle en simpel “cutting plane” algoritme til at løse TSP.

- 1 Start med en simpel formulering, der kun indeholder “assignment” begrænsningerne
- 2 for $i = 1$ to M
- 3 Løs modellen til IP-optimalitet med CPLEX
- 4 Såfremt den returnerede løsning ikke indeholder deltur, standser algoritmen
- 5 Generer et antal uligheder som bryder de nuværende deltur
- 6 Tilføj de genererede uligheder til modellen
- 7 end for
- 8 Tilføj MTZ begrænsningerne til modellen
- 9 Løs modellen til IP-optimalitet med CPLEX

Opgave 6 Beskriv og implementer en algoritme som kan generere et antal deltur uligheder på formen (2) der bryder de nuværende deltur i linie 5. ■

Opgave 7 Eksperimenter med hvor mange uligheder der skal tilføjes i hver iteration af linie 5, samt med antallet af iterationer M . Giv en begrundet beskrivelse af dit endelige valg. ■

Opgave 8 Beskriv og implementer en heuristik som giver en løsning i skridt 3, der indeholder deltur, konstruerer en sammenhængende Hamilton kreds. Anvend denne efter skridt 4 i “cutting plane” algoritmen. Såfremt den heuristiske løsning svarer til den fundne græseværdi i skridt 3, kan algoritmen standse. ■

Opgave 9 Løs så store problemer af typen $ftvXX$ som muligt med den udviklede algoritme. Rapportér køretid, antal iterationer i “cutting plane” algoritmen, samt antal genererede cuts. ■

Opgave 10 (ekstraopgave) Tilføj flere typer af lovlige uligheder til modellen. En god beskrivelse af de mest kendte lovlige uligheder for TSP findes i [9]. Overvej specielt hvordan ulighederne kan separeres effektivt. Rapportér køretid, antal iterationer i “cutting plane” algoritmen, samt antal genererede cuts. ■

Instanser

Følgende instanser er (med få undtagelser) hentet fra TSPLIB hjemmesiden [8] og konverteret til et format der er nemt at indlæse. Alle instanser findes på kursets hjemmeside.

n	instans	beskrivelse
5	rand5	tilfældigt genererede kantvægte
10	rand10	tilfældigt genererede kantvægte
8	bornholm	afstande mellem otte byer på Bornholm
14	burma14	14 byer i Burma, geografisk afstand
17	gr17	17 byer i Tyskland
21	gr21	21 byer i Tyskland
24	gr24	24 byer i Tyskland
48	gr48	48 byer i Tyskland
120	gr120	120 byer i Tyskland
29	bays29	29 byer i Bayern (street distance)
29	bayg29	29 byer i Bayern (geographic distance)
42	swiss42	42 byer i Schweiz (Fricker)
17	br17	asymmetrisk TSP (Repetto)
34	ftv33	asymmetrisk TSP (Fischetti)
36	ftv35	asymmetrisk TSP (Fischetti)
39	ftv38	asymmetrisk TSP (Fischetti)
45	ftv44	asymmetrisk TSP (Fischetti)
47	ftv48	asymmetrisk TSP (Fischetti)
56	ftv55	asymmetrisk TSP (Fischetti)
71	ftv70	asymmetrisk TSP (Fischetti)
171	ftv170	asymmetrisk TSP (Fischetti)
48	ry48p	asymmetrisk TSP (Fischetti)
323	rbg323	Stacker crane application (Ascheuer)
358	rbg358	Stacker crane application (Ascheuer)
403	rbg403	Stacker crane application (Ascheuer)
443	rbg443	Stacker crane application (Ascheuer)
535	si535	TSP (M. Hofmeister)
1032	si1032	TSP (M. Hofmeister)

Den optimale løsningsværdi er angivet i hovedet af de fleste instanser. De nederste instanser kræver at programmets tabeller udvides.

Noter

Til opgaven benyttes et rammeprogram `tsp.c` som er skrevet i C og som varetager kommunikationen med CPLEX. Da der kun er nogle få CPLEX-licenser til rådighed på DIKU, vil ram-

meprogrammet højst bruge CPLEX i 60 sekunder, hvorpå licensen frigives. CPLEX licenser er tilgængelige på Linux pc'er samt SUN maskiner.

For at benytte CPLEX er det nødvendigt at tilføje følgende linie i sin `.tcshrc` fil.

```
setenv ILOG_LICENSE_FILE /usr/local/stow/etc/cplex.ilm
```

Rammeprogrammet oversættes med kommandoen

```
gcc -O5 -o tsp tsp.c -I/usr/local/stow/include/ilcplex -L/usr/local/stow/lib/ -lcplex -lm -lpthread
```

Der benyttes et meget simpelt interface til CPLEX: IP-modellen skrives til en fil `/tmp/infile.lp`, hvorpå rammeprogrammet kalder CPLEX med den fil som inddata. Det simple interface gør det nemt at finde fejl i IP-modellen, idet man kan anvende CPLEX interaktivt med den genererede input fil: Skriv `cplex` i kommandolinien, og indlæs datafilen med `read /tmp/infile.lp`. Såfremt der er syntaxfejl i IP-modellen vil CPLEX rapportere disse. Ellers kaldes `optimize` og CPLEX vil rapportere om modellen er ubegrænset (“unbounded”), har et tomt løsningsrum (“infeasible”), eller lignende.

Litteratur

- [1] <http://www.caam.rice.edu/~bico/>, home page of Bill Cook at Rice University.
- [2] <http://www.keck.caam.rice.edu/concorde.html>.
- [3] E. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, eds., *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, Wiley, Chichester, UK, 1985
- [4] A. Langevin, F. Soumis, and J. Desrosiers, Classification of travelling salesman formulations, *Oper. Res. Lett.*, 9 (1990), pp. 127-132.
- [5] C. E. Miller, A. W. Tucker, and R. A. Zemlin, Integer programming formulations and traveling salesman problems, *J. ACM*, 7 (1960), pp. 326-329.
- [6] M. Padberg, and T.-Y. Sung, An analytical comparison of different formulations of the travelling salesman problem, *Math. Programming*, 52 (1991), pp. 315-357.
- [7] L. A. Wolsey, *Integer Programming*, Wiley, Chichester, UK, 1999.
- [8] <http://www.iwr.uni-heidelberg.de/iwr/comopt/software/TSPLIB95/>
- [9] <http://rodin.wustl.edu/~kevin/dissert/node11.html>