Matematisk Modellering
Projekt C:
Planlægning og optimering

Anders "Bongo" Bjerg Pedersen
Lars Roholm
Martin Hvolby
Jesper Frank Christensen

29. januar 2006

Indhold

1 Indledning

Vi forestiller os at vi starter et firma kaldet FUP Transport a/s hvis opstartsvanskeligheder vi vil forsøge at løse. I den forbindelse har vi en række problemstillinger. Vi vil derfor i dette projekt se på forskellige logistiske udfordringer, firmaer bliver stillet overfor, når de skal etablere sig. For det frøste vil vi se på placeringsproblemer i forbindelse med f.eks. lagre, sygehuse, biblioteker og ligenende mhp. at minimere transportomkostningerne til udbringning/service. For det andet vil vi se på nogle forskellige muligheder for at planlægge denne transport mht. chauffører, arbejdstider og skemaplanlægning, så firmaets ressourcer bliver udnyttet bedst muligt. Specielt vil vi se nærmere på den såkaldte Tabu-søgning.

2 Lokalisering

2.1 Problemet

Et transportfirma, FUP Transport A/S, ønsker at etablere et lager i en større by for at servicere deres kunder (konsumenter) med varer af en bestemt type. Disse konsumenter har forskellig efterspørgsel efter varen, og problemet for firmaet er nu at placere lageret, så transporttid og -omkostninger bliver mindst mulige. Der er selvfølgelig mange variable at skrue på i et sådant setup ud over de sædvanlige; brændstofpriser, lønninger til chauffører, hviletidsbestemmelser, det faktum at vejen ikke altid kan bestemmes som fugleflugtslinie og andet, men vi vil her se på en mere generaliseret model, der kan give et godt hint om lagerets mest effektive placering. Vi vil med andre ord optimere lagerets placering matematisk.

2.2 Nogle termer

Vi indfører først nogle begreber, der kan være brugbare som generalisation. Lad os først tænke os, at vi kan betragte byen som et koordinatsystem i planen, og vi ønsker at finde den bedst mulige placering af lageret på koordinatsættet (x, y). Vi har n konsumenter i byen, der ligeledes ligger placeret som punkter (koordinatsæt) i koordinatsystemet $(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)$, og har afstandene (i fugleflugtslinie) $l_1, l_2, \ldots, l_n$ til lageret i (x, y). Desuden har konsumenterne en vis efterspørgsel efter firmaets vare, altså en bestemt mængde som konsumenten ønsker at aftage. Disse kalder vi $w_1, w_2, \ldots, w_n$.

Situationen er illustreres i Figur 1 på næste side.

2.3 Selve modellen

Vi ønsker altså at minimere udgifterne til transport ud til konsumenterne. d_i er afstanden mellem de to punkter i koordinatsystemet, og det er velkendt, at

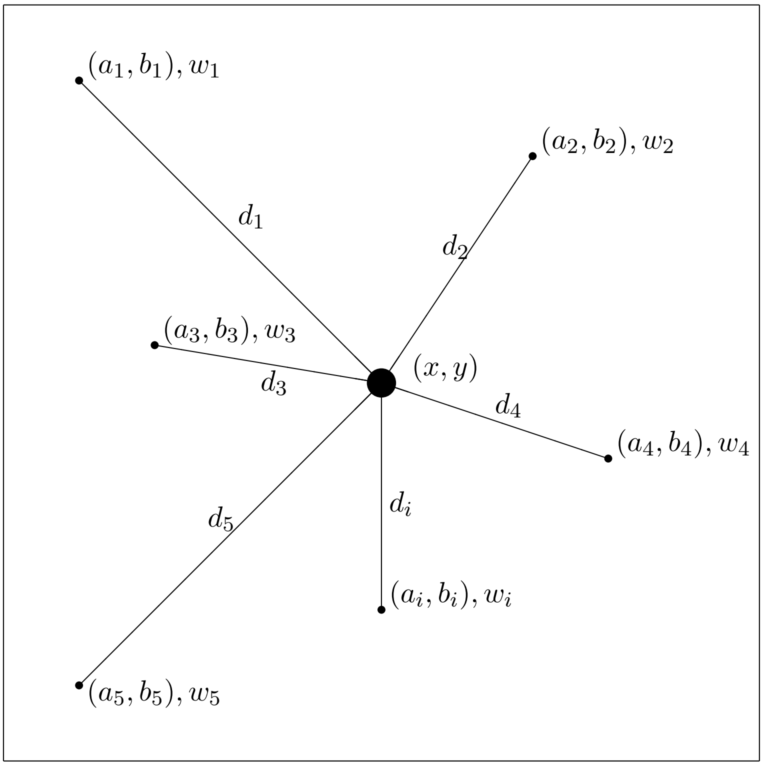
$$d_i(x, y) = \sqrt{(x - a_i)^2 + (y - b_i)^2}.$$

Vi kan nu generalisere en smule ved at sige, at det samlede transportarbejde er summen af disse afstande multipliceret med efterspørgslen for hver konsument, w_i. Dermed bliver det samlede transportarbejde:

$$\sum_{i=1}^{n} w_i \sqrt{(x - a_i)^2 + (y - b_i)^2},$$

som vi kan opstille som en funktion af lagerets placering:

$$f(x, y) = \sum_{i=1}^{n} w_i \sqrt{(x - a_i)^2 + (y - b_i)^2}.$$



Figur 1

Problemet for os bliver nu at minimere denne funktion af to variable, x, y, for at finde den optimale placering af lageret. Her er vi dog nødt til at antage nogle ting; for det første må vi antage, at transportomkostningerne er proportionale med w_i og d_i, og at man normalt arbejder med individuel forsyning af konsumenterne. Ligeledes kan et lager af praktiske årsager ikke altid placeres lige netop i det fundne punkt.. Men selvom disse antagelser ikke altid er opfyldt, har virkelighedens verden vist sig at passe meget pænt ind med modellen alligevel.

Jævnfør normal matematisk praksis vil vi nu differentiere vores udtryk og finde nulpunkter for den afledte. Vi differentiere først:

$$\frac{\partial f(x,y)}{\partial x} = \sum_{i=1}^{n} \frac{w_i(x-a_i)}{\sqrt{(x-a_i)^2+(y-b_i)^2}},$$

sætter dette udtryk lig 0 og trækker x uden for en parentes og omskriver:

$$x\left(\sum_{i=1}^{n} \frac{w_i}{d_i(x,y)}\right) - \sum_{i=1}^{n} \frac{w_i a_i}{d_i(x,y)} = 0.$$

Vi isolerer nu x:

$$x = \frac{\sum_{i=1}^{n} \frac{w_i a_i}{d_i(x,y)}}{\sum_{i=1}^{n} \frac{w_i}{d_i(x,y)}}.$$

Samme procedure kan selvfølgelig gentages for y-koordinaten, hvor vi får et lignende udtryk:

$$y = \frac{\sum_{i=1}^{n} \frac{w_i b_i}{d_i(x,y)}}{\sum_{i=1}^{n} \frac{w_i}{d_i(x,y)}}$$

Umiddelbart er der et åbenlyst problem i ovenstående udtryk for x og y, nemlig hvis man skulle vælge at placere lageret oveni en af konsumenterne. Her ville $d_i(x,y) = 0$, som vi jo ikke kan ha'! Dette løses normalt ved at tilføje en meget lille (til tider infinitesimalt lille) størrelse, ε, som lægges til afstanden, så den aldrig bliver 0. En sådan manøvre påvirker ikke resultatet nævneværdigt:

$$d_i(x,y) = \sqrt{(x-a_i)^2+(y-b_i)^2+\varepsilon}.$$

Men de ovenfor udledte metoder giver ikke umiddelbart noget egentligt resultat for placeringen, da der f.eks. i udtrykket for y også forekommer et y på højresiden. Dette problem løser men ved en iterativ metode, hvor man angiver et koordinatsæt (x_1, y_1), som man indsætter på højresiderne og regner på. Dette giver nu et nyt udtryk (x_2, y_2), som man igen indsætter på højresiden og regner på, og sådan fortsættes der j gange, så (x_{j+1}, y_{j+1}) ikke afviger nævneværdigt fra (x_j, y_j). Man har nu iterativt fundet den bedste placering af lageret.

2.4 Et eksempel på anvendelse

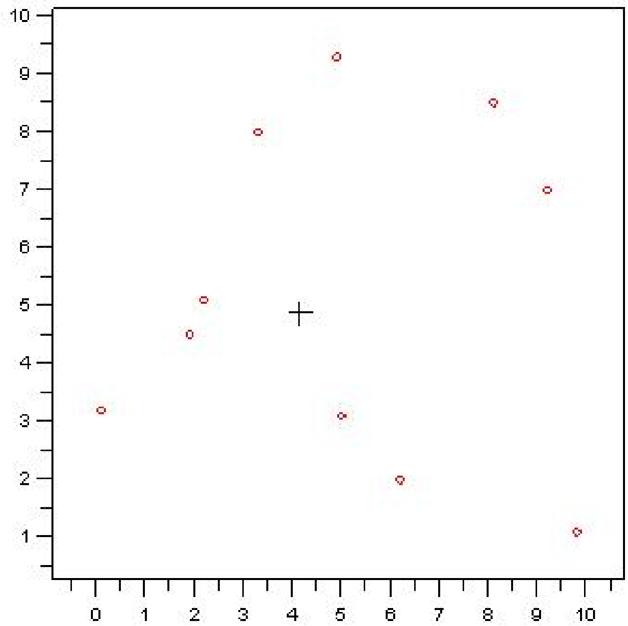
Herunder ses et eksempel på $n = 10$ konsumenters placering og deres efterspørgsel på FUP Transport A/S' vare:

Konsument nr. i	a_i	b_i	w_i
1	0.1	3.2	5
2	5.0	3.1	1
3	9.2	7.0	2
4	3.3	8.0	7
5	4.9	9.3	1
6	8.1	8.5	1
7	2.2	5.1	3
8	1.9	4.5	1
9	6.2	2.0	4
10	9.8	1.1	6

Vi vil i det følgende konstruere en Maple-procedure, der kan udregne den bedste placering for os i ovenstående eksempel og eventuelt vise os det grafisk i et koordinatskema. Proceduren tager som input tre lister, A,B og W, med hhv. konsumenternes x-koordinater, deres y-koordinater og deres efterspørgsler. Proceduren regner så iterativt den bedste placering ud ud fra ovenstående teoretiske udledninger og plotter konsumenter og lager i et koordinatsystem:

```
>restart;
>with(LinearAlgebra):with(plots):with(plottools):with(stats[statplots]):

>lager:=proc(A,B,W)

 local i,j,k,D::list,X0,X1,Y0,Y1;

 X0:=sum(A[i],i=1..nops(A))/nops(A);    #Initialgæt for placeringens x-koordinat
 Y0:=sum(B[i],i=1..nops(B))/nops(B);    #Initialgæt for placeringens y-koordinat
 print(X0,Y0);

 X1:=0.1;Y1:=0.1;                       #Sammenligningsbetingelser for placeringen

 #Følgende itererer, indtil koordinaterne ikke afviger for meget:

 while abs(X1-X0)/X1>0.00001 and abs(Y1-Y0)/Y1>0.00001 do

   X1:=X0;Y1:=Y0;

   #Følgende udregner de forskellige afstande til konsumenterne:

   for j from 1 to nops(A) do
     D[j]:=sqrt((X0-A[j])^2+(Y0-B[j])^2+10^(-10));
   end do;

   #Følgende udregner nye koordinatsæt og gemmer til sammenligning:

   X0:=sum(W[k]*A[k]/D[k],k=1..nops(A))/sum(W[k]/D[k],k=1..nops(A));
   Y0:=sum(W[k]*B[k]/D[k],k=1..nops(A))/sum(W[k]/D[k],k=1..nops(A));

   print(X0,Y0);

 end do;

 #Til sidst plottes konsumenterne (cirkler) og den fundne placering (krydset):

 plots[display]({scatterplot(A,B,axes=boxed,color=red),point([X1,Y1],color=black,symbol=cross,symbolsize=20)});

 end proc:

>A:=[0.1,5,9.2,3.3,4.9,8.1,2.2,1.9,6.2,9.8]:
>B:=[3.2,3.1,7,8,9.3,8.5,5.1,4.5,2,1.1]:
>W:=[5,1,2,7,1,1,3,1,4,6]:
>lager(A,B,W);
                    5.070000000, 5.180000000
                    4.561367761, 4.882432406
                    4.367041107, 4.814596187
                    4.279826841, 4.811785818
                    4.235089860, 4.823498566
                    4.209914395, 4.835507740
                    4.194919126, 4.844712963
                    4.185694394, 4.851117255
```

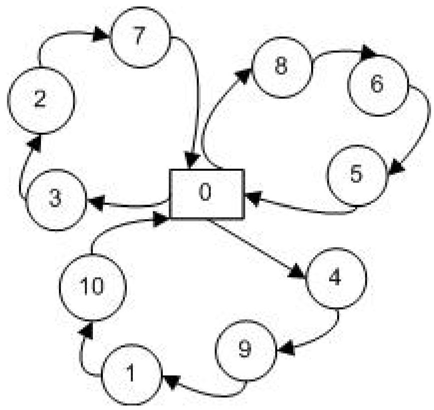
```
4.179918698, 4.855387830
4.176268319, 4.858177108
4.173949757, 4.859979504
4.172473324, 4.861137607
4.171531917, 4.861879475
4.170931262, 4.862353932
4.170547898, 4.862657102
4.170303188, 4.862850731
4.170146970, 4.862974366
4.170047242, 4.863053300
4.169983577, 4.863103694
4.169942937, 4.863135865
```



Proceduren giver os altså, at den bedste placering for lageret med den givne efterspørgsel vil være $(x, y) = (4.17, 4.83)$. Proceduren kan nemt justeres til mindre eller større nøjagtighed i while-linien.

3 Ruteplanlægning

For alle virksomheder i transportindustrien er ruteplanlægning en vigtig kilde i jagten på succes. Som nystartet i branchen er det derfor nødvendigt at tillægge dette en del opmærksomhed, da det bl.a. er her, at *FUP Transport A/S* kan spare tid og ikke mindst reducere transportomkostningerne.
Ruteplanlægningsproblemet handler grundlæggende om hvordan kunder og ordrer tilgodeses på den mest praktiske (læs: hurtigste og/eller billigste) måde. Problemet består i at vælge den bedste kombination af kunder, køretøjer og depoter. Men i tillæg til dette har hver virksomhed nogle forudsætninger og nogle retningslinjer for hvordan dette skal klares. Dette kan være antal depoter og køretøjernes kapacitet, samt regler om køre-hviletids bestemmelser.
Vi vil nu kigge på en af de simpleste modeller for ruteplanlægning.



Figur 1: Illustration af VRP

3.1 Vehicle Routing Problem

Vehicle Routing Problem VRP kan "kort"beskrives som en model for et *komplekst kombinatorisk optimerings problem.*[1] VRP er dog langgt fra den mest komplekse ruteplanlæninsmodel. Der findes utallige andre modeller af mere kompleks karakter. Disse indeholder langt flere variable og flere muligheder for de variable. Dette skyldes virksomheders individuelle behov og krav. Vi vil dog koncentrere os om VRP, da den bedst beskriver en lille virksomhed som FUP Transport A/S (med tiden må vi forhåbentligt søge til mere avancerede modeller).
Formålet med VRP-modellen beskrives ved følgende: Vi har givet et firma med ét enkelt depot og en tilhørende mængde køretøjer med samme kapacitet og dertil en række kunder. Ud fra dette ønskes det bestemt, hvilken ruteplanlægning, dvs. mængden af køretøjernes ture, som samlet set er billigst i driftomkostninger, samt tilgodeser alle kundernes behov. [Bemærk, at denne model kun indirekte berører tids-aspektet i form af sammenhængen mellem afstand og kørselstid].
Inden vi går videre vil vi lige indføre en række notationer.

- Punktmængden $V = \{v_o, v_1, v_2, ..., v_n\}$ indeholder alle punkterne/kunderne for ruteplanlægningen. Vi lader v_0 angive depotet og $v_1, ...v_n$ angive kunderne. Vi lader desuden $V' = V \backslash \{v_0\}$ angive mængden af kunder. Vi benytter i, j som index for både kunder og depfoter, således at v_i er placeringen af kunde i, hvor $i \in V$. Endvidere benyttes h til angivelse af en bestemt kunde (eller depot).

- $\mathcal{K}$ angiver mængden af køretøjer (med samme kapacitet). Vi benytter k for køretøjer.

- c_{ij} angiver afstanden (og deraf transportomkostningerne) mellem punkterne v_i og v_j.

[1] http://eden.dei.uc.pt/ jast/vrp/

- d_i angiver mængden af varer hos kunde v_i, $0 < i \leq n$.
- x_{ijk} er en binær beslutningsvariabel, som angiver om køretøj, k sørger for ruten mellem kunde i og j, dvs. mellem punkt v_i og v_j. Hvis dette er tilfældet, er $x_{ijk} = 1$ og ellers lig 0. Da x_{ijk} er afhængig af antallet af muligheder af de variable, kan vi definere dens domæne som $\mathbb{R}_+^{|V|^2 \times |\mathcal{K}|}$.

VRP modellen indeholder en række begrænsninger. Dem vil nu opstille.

1. $\sum_{kj} x_{ijk} = 1, \forall i : \exists v_i \in V'$
   Denne begrænsning sikrer, at hver kunde i besøges præcist én gang af præcist ét køretøj.
2. $\sum_{i,j} d_i x_{ijk} \leq q, \forall k \in \mathcal{K}$
   Denne begrænsning sikrer, at bilernes kapacitet, q overholdes.
3. $\sum_j x_{0jk} = 1, \forall k \in \mathcal{K}$
   Denne begrænsning sikrer, at alle køretøjer benyttes, dvs. forlader depotet, v_0
4. $\sum_i X_{ijk} - \sum_j x_{hjk} = 0, \forall h \in V, k \in \mathcal{K}$
   Denne begrænsning sikrer, at enhver bil, k der ankommer til punkt v_h (altså kunde eller depot h) også forlader det igen.
5. $\sum_i X_{i0k} = 1, \forall k \in \mathcal{K}$
   Denne begrænsning sikrer, at alle biler kommer tilbage til depotet v_0 (altså "kunde" 0).

Tilbage står nu kun at finde den bedste ruteplanlægning. Det vil sige, at vi kan koge vores problem ned til, at skulle minimere omkostningsfunktionen for ruterne.

$$\text{Min} \sum_{i,j,k} c_{ij} x_{ijk}$$

Dette er dog ikke helt så lige til. Vi nu kigge lidt nærmere på løsning af ruteplanlægnings-problemet. [2]

3.2 Løsning af VRP

Selv den simple udgave af ruteproblemet kan være praktisk talt umulig at løse. De mange kombinatoriske muligheder gør, at løsningen af algoritmen bliver såkaldt *"NP-hård"*, dvs. at den ikke kan løses på *polynomial tid.*[3] En nemmere måde er således at øge antallet af begrænsninger eller udelade tidsaspektet. Evt. må man indføre restriktioner for algoritmens handlingsmønster.[4] Disse kaldes "grådige algoritmer". Antagelsen bag

[2] Ottesen og Tranbjerg 2004, s. 20ff og s. 54ff
[3] http://neo.lcc.uma.es/radi-aeb/WebVRP/index.html?/Problem_Descriptions/VRPDesc.html
[4] Ottesen og Tranbjerg 2004, s. 47ff

disse er, at man ved gentagne valg af det lokale minimum opnår det globale minimum. Denne metode virker ved ruteplanlægning (Se bl.a. Djikstras algoritme), men ellers kan man generelt ikke være sikker på, at det er tilfældet.[5]. Denne type problemløsning tilhører heuristikken. Et eksempel herpå er “Tabu søgning”.

3.3 Tabu søgningens hovedtræk

Tabu søgning er en heuristik, som kan anvendes til optimering. Især inden for ruteplanlægning har den været anvendelig. Metoden kan opdeles i fem punkter.[6]

- Lokalisering eller konstruktion af en lovlig løsning, som benævnes den ‘*initielle løsning*. “Lovlig” indikerer, at de opstillede begrænsninger overholdes.
- Konstruktion af et *nabolag*. Dette er enkle modificeringer af den initielle løsning, eks. ombytning af ordrer. Den bedste af disse naboer vælges og et nyt nabolag kan konstrueres ud fra denne. Ved fortsættelse ender man i et lokalt minimum, som er bedre end alle løsninger i næste nabolag.
- Konstruktion af en *tabuliste*, som forhindrer algoritmen i at vende tilbage til allerede betragtede ombytninger. Derved forhindrer man, at algoritmen “hænger fast” i et lokalt minimum.
- Et *aspirationskriterium*, som skal forhindre at tabulsiten virker hæmmende, eksempelvis, at den hidtidigt bedste løsning vælges selvom den allerede står på tabulisten.
- Endelig sørger et *stopkriterium* for, at algoritmen stopper. Dette kan være bestemt ved tid eller antal ombytninger.

Den praktiske udførsel af tabu søgningen kan ses ved denne pseudo-kode[7]:

```
Find en initiel løsning s
Gentag indtil stopkriteriet mødes
    Generer omegn N(s)
    For alle løsninger s’ i omegnen N(s) beregnes målfunktionsværdien M(s’)
    Hvis M (bedste løsning fra N(s)) > M (hidtil bedste løsning)
        Så s = bedste løsning fra N(s)
        Ellers s= bedste tilladte løsning fra N(s) der ikke er på tabulisten
    Opdater tabulisten
```

[5] Rasmussen 2004, s. 1ff

[6] Rasmussen 2004; s. 13ff

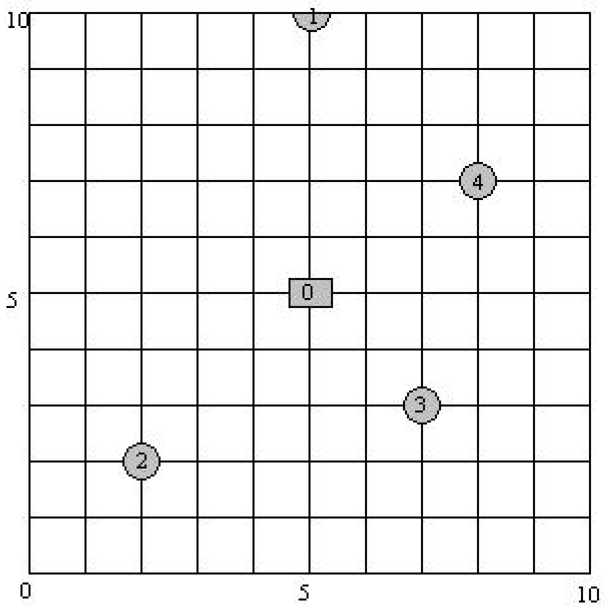
[7] Pesudo-koden er gengivet fra Rasmussen 2004, s. 14

3.4 Anvendelse af tabusøgning for FUP Transport A/S

For at illustrere anvendelsen af tabu søgning på et konkret tilfælde må vi først konstruere et sådanne. Til dette vender vi tilbage til vores nystartede firma, FUP Transport A/S. Vi antager samme setup som i VRP. Desuden lader vi kun to chauffører, k_1, k_2 være på arbejde med blot fire kunder $1, 2, 3, 4$ at servicere. Vi lader q angive køretøjernes kapacitet. og alle knuder (kunder og terminal) være givet som heltals-koordinater i et 10×10 kvadrat. Desuden lader vi blot den korteste afstand være repræsentant for den billigst og hurtigste rute. Nedenstående tabel viser de konkrete data for vores setup.

Index	type	Punkt	Koordinater	ordre-størrelse
0	Depot	v_0	$(5, 5)$	-
1	Kunde 1	v_1	$(5, 10)$	$0, 2\ q$
2	Kunde 2	v_2	$(2, 2)$	$0, 6\ q$
3	Kunde 3	v_3	$(7, 3)$	$0, 5\ q$
4	Kunde 4	v_4	$(8, 7)$	$0, 2\ q$

Grafisk ser placeringerne således ud:



Vi vil nu betragte de mulige ruteindelinger. Vi skelner i det følgende mellem ruter og ture, hhv. hele ruteplanlægningen og de enkelte køretøjers ture. Da begge køretøjer har samme kapacitet anses ruterne som ens selvom bilerne ombyttes. Ligeledes skelnes der ikke mellem køreretningerne. Dvs. turen $v_1 - v_2$ er lig $v_2 - v_1$ [Da køretøjerne altid skal starte og slutte ved depotet angiver vi ikke dette i beskrivelsen af turen]. For overskuelighedens skyld lader vi blot turene være givet ved kundernes numre. Eksempelvis $1 - 2 - 4$. Turens angivelse er den numeriske rækkefølge af kunderne. Ved en tur til ttre

kunder, kan chaufføren vælge flere ture. Vi betragter do kun den korteste af de mulie ture.
Begrænsningerne for VRP giver desuden, at alle biler skal benyttes. Altså har vi i dette konkrete tilfælde altid to ture for hver rute. Vi lader turene være angivet ved de to køretøjer og skelner som nævnt ikke mellem ombytning af disse.
Simpel kombinatorik giver, at der er 7 forskellige ruter. Vi vil nu betragte dem nærmere ved brug af tabu søgning.

Vi finder først en *initiel løsning*. Vi vælger den rute, som vi kalder a, hvor $k_1 = (2)$ og $k_2 = (1 - 3 - 4)$. Denne rute er lovlig, da kapacitetn overholdes. Vi finder nu et *nabolag* ved på skift at ombytte kunde 2 enkeltvist med hver af de andre. Da får vi:

Rute	k_1	k_2
a	2	1-3-4
b	1	2-3-4
c	3	1-2-4
d	4	1-2-3

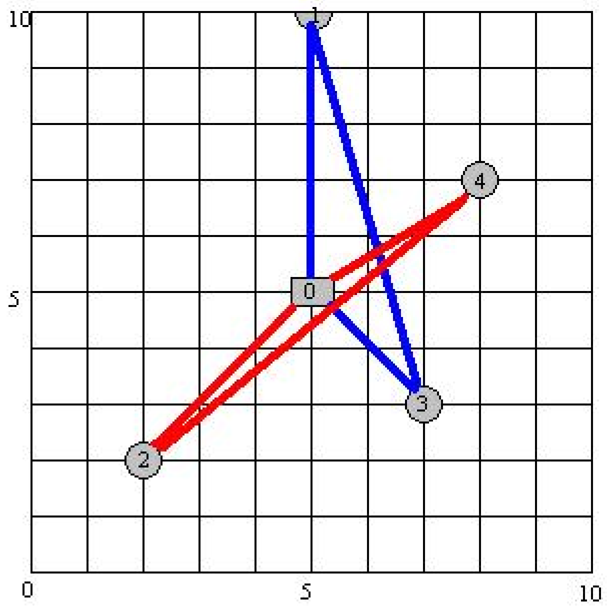
Vi vil nu betragte ruterns lovlighed. Det ses, at rute b og d ikke oveholder kapacitetsgrænsen. Dise tilføjes derfor tabulisten.
Vi sammenligner nu rute a og c mht. deres længde, som angives i tern/enheder. Vi antager, at fugleflugtslinjen altid er mulig!:

Rute	Afstand	Afstand udmålt
a	$2 \cdot c_{02} + c_{01} + c_{14} + c_{43} + c_{30}$	26,7
c	$2 \cdot c_{03} + c_{02} + c_{21} + c_{14} + c_{40}$	26,3

Altså har vi fundet en bedre løsning end den vi startede med. Rute a tilføjes da tabulisten, og c benyttes nu som udgangspunkt for det næste nabolag. Her flytter vi enkeltvis en af kunderne fra k_2's tur. Altså får vi følgende ruter:

Rute	k_1	k_2
c	3	1-2-4
e	1-3	2-4
f	2-3	1-4
g	3-4	1-2

Rute f oveholder ikke kapacitetskravet og tilføjes tabulisten. Vi kan nu udregne de to resterende ruters længde. Vi får at rute $e = 20,7$ og rute $g = 28,3$. Altså kan vi nu også tilføje g og c til tabulisten. Da der kun var syv mulige ruteinddelinger, har vi således, at rute e er den mest optimale. Grafisk ser den således ud:



4 Skemaplanlægning for de ansatte

Som virksomhedsleder stilles man overfor at skulle planlægge skema for de ansatte. Det må anses som en af de vigtigste opgaver for en leder at kunne tilrettelægge et 'retfærdigt' skema. Mange faktorer spiller ind. Lovgivning, fagforeningsregler og ikke mindst udbud af arbejde på forskellige tidspunkter har indflydelse på, hvorledes et skema kan planlægges optimalt.
Da arbejdsbyrden ikke er den samme på ugens forskellige arbejdsdage, nytter det ikke at give chaufførerne faste fridage, da det medfører uretfærdigt hårdt skema for nogle, og nemt for andre.
Vi vil i det følgende diskutere, hvorledes et optimalt skema kan opbygges.
Først defineres nogle begreber: Vi deler en given arbejdsperiode, typisk en uge, op i arbejdsskift, typisk dage. I et skift kan et arbejdshold enten arbejde eller have fri.
Det totale antal arbejdshold: N
Det antal arbejdshold, der er påsat det i'te arbejdsskift: n_i
Heraf får vi $N - n_i = r_i$, som er det antal arbejdshold vi kan give fri i det i'te skift.[8]
Forskellige faktorer som lovgivning og fagforeningregler afgør, hvad der kan accepteres som arrangement af arbejdsfrie skift for et arbejdshold, kaldet recreation clusters.[9] Eksempelvis kombinationen af dagene mandag-tirsdag, mandag-fredag eller lørdag-søndag.

[8] Beltrami, 1977
[9] Vi benytter her betegnelsen 'recreation cluster', da vi ikke kan finde noget bedre på dansk

Vi antager altså, at vi ved hvor mange hold, vi behøver på arbejde i hvert skift og skal se på, hvordan vi tilrettelægger et skema, der er optimalt for firmaet og for de ansatte.
Vi lader A være en matrix, hvis indgange a_{ij} er enten 0 eller 1. 1 indikerer at det i'te skift indgår i den j'te recreation cluster og 0 ellers. x_j er et non-negativt helt tal, der indikere, hvor mange gange den j'te recreation cluster skal benyttes hver uge. Det i'te skift skal optræde i en eller anden recreation cluster præcis i alt r_i gange på en uge for at der med sikkerhed findes n_j arbejdshold på arbejde på det skift. Dette kan vi formulere som

$$\sum_{j=1}^{m} a_{ij}x_j = r_i$$

eller mere abstract

$$\underline{\underline{A}}x = r$$

hvor x, r er henholdsvis m (antallet af recreations clusters) og s (antallet af skift) dimensionelle vektorer. Det skal her bemærkes, at der ikke altid vil findes heltalsløsninger til ovenstående ligning. I praksis må man derfor ofte sige at det i'te skift kan optræde i den j'te recreation cluster højst r_i gange, formuleret således

$$\sum_{j=1}^{m} a_{ij}x_j \leq r_i$$

Vi vil nu demonstrere ovenstående i praksis i forbindelse med skemaplanlægning for vores firma FUP transport A/S.

Eksempel 1 Vi inddeler en arbejdsuge, således et arbejdsskift er en dag og en recreation cluster er en kombination af to dage, en ansat kan have fri. Her består et arbejdshold af en chauffør med lastbil. Vi råder over 8 af disse, så $N = 8$. Vores firma bringer ikke varer ud om søndagen, så her har alle fri. Med ovennævnte notation kan arbejdskravet opstilles således

N=8	Man	Tir	Ons	Tor	Fre	Lør	Søn
n_i	7	6	7	7	8	6	0
r_i	1	2	1	1	0	2	8

Da hver chauffør arbejder gennemsnitligt 5 dage om ugen og der er 8 chauffører for vi total antal arbejdsdage: $5N = 40$, og total antal fridage: $2N = 16$. For at opstille et fair skema laver vi et 8-ugers rotationsskema, så alle chauffører over en 8 ugers periode udfører samme arbejdstider.
Vi kan nu opstille den førnævnte matrix med alle mulige kombinationer af fridage således

$$\underline{\underline{A}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

hvor rækker er dagene i ugen, og kolonerne er de forskellige recreations clusters (mandag-søndag, tirsdag-søndag osv.). Et 1-tal betyder den dagen ligger i den pågældende recreation cluster. 7. og 8. kolonne betegner en 3 dages weekend og 9. kolonne begner den situation, hvor kun søndag er fridag. Dette leder frem til følgende ligningssystemet

$$\begin{aligned} x_1 + x_8 &= r_1 = 1 \\ x_2 + x_8 &= r_2 = 2 \\ x_3 &= r_3 = 1 \\ x_4 &= r_4 = 2 \\ x_5 + x_7 &= r_5 = 0 \\ x_6 + x_7 &= r_6 = 2 \\ x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 &= r_7 = 8 \end{aligned}$$

Det bemærkes at

$$\sum_{i=1}^{6} r_i = 8 = r_7 \Rightarrow \sum_{i=1}^{6} r_i - r_7 = x_7 + x_8 - x_9 = 0 \Rightarrow x_7 + x_8 = x_9$$

dvs. det antal chaufører, der arbejder 4 dage er lig det antal der arbejder 6, hvilket øjensynligt må være tilfældet, hvis vi skal have et gennemsnit på 5 arbejdsdage.
Vi ønsker nu at optimere vores skema, således at vi giver så mange ansatte som muligt mulighed for 2-dage weekends, samtidig med vi overholder vores arbejdskrav.
Vi skal da optimere udtrykket $x_1 + x_6$, hvorfra vi får

$$x_1 + x_6 = r_1 + r_6 - x_7 - x_8$$

Da $r_1 + r_6$ er faste tal, ses det let at udtrykket maksimeres ved at sætte

$$x_7 = x_8 = 0$$

men da er også $x_9 = 0$
Da ses det fra ovenstående ligningsystem, at den optimale løsning må være, at $x_i = r_i$ for $i = 1, ..., 6$. Dvs. for hver 8 uger får vores ansatte

$$x_1 + x_6 = r_1 + r_6 = 1 + 2 = 3$$

2 dages weekends. Et passende skema kunne se således ud

Uge	Man	Tir	Ons	Tor	Fre	Lør	Søn
1		x					x
2	x						x
3				x			x
4			x				x
5						x	x
6				x			x
7						x	x
8		x					x
	$r_1 = 1$	$r_2 = 2$	$r_3 = 1$	$r_4 = 2$	$r_5 = 0$	$r_6 = 2$	$r_7 = 8$

hvor x betegner en fridag.

Eksempel 2 Vi forestiller os nu, at vores chauførerer kommer til ledelsen og kræver at have to sammenhængende fridage hver uge (vi kalder det en weekend). Hvordan sammensætter vi nu et optimalt skema? Vi kan øjensynligt ikke længere holde fast i vores princip om ingen udbringning på søndage. Vi laver derfor følgende nye arbejdsplan

N=8	Man	Tir	Ons	Tor	Fre	Lør	Søn
n_i	6	5	7	6	6	5	5
r_i	2	3	1	2	2	3	3

Igen opsætter vi matricen

$$\underline{\underline{A}}' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Igen er hver række udtryk for dagene (mandag, tirsdag...) og kolonnerne er udtryk for den pågældende recreation cluster (mandag-tirsdag, tirsdag-onsdag osv.)
Dette giver os følgende ligningssystem

$$\begin{aligned} x_1 + x_7 &= r_1 = 2 \\ x_2 + x_2 &= r_2 = 3 \\ x_2 + x_3 &= r_3 = 1 \\ x_3 + x_4 &= r_4 = 2 \\ x_4 + x_5 &= r_5 = 2 \\ x_5 + x_6 &= r_6 = 3 \\ x_6 + x_7 &= r_7 = 3 \end{aligned}$$

Som har en entydig bestemt løsning

$$x_1 = 2$$
$$x_2 = 1$$
$$x_3 = 0$$
$$x_4 = 2$$
$$x_5 = 0$$
$$x_6 = 3$$
$$x_7 = 0$$

hvilket eksempelvis kan give følgende skema

Uge	Man	Tir	Ons	Tor	Fre	Lør	Søn
1	x	x					
2						x	x
3		x	x				
4						x	x
5				x	x		
6	x	x					
7						x	x
8				x	x		
	$r_1 = 2$	$r_2 = 3$	$r_3 = 1$	$r_4 = 2$	$r_5 = 2$	$r_6 = 3$	$r_7 = 3$

hvor x igen betegner en fridag.
Det skal dog bemærkes, at man ikke altid finder non-negative løsninger til ligningssystemet ovenfor og desuden heller ikke altid heltalsløsninger. Dette komplicerer naturligvis skemaplanlægningen, da man da må indsætte halve arbejdsdage og tillægsfridage for at få det hele til at gå op. Dette problem gennemarbejdes dog ikke her, da det er temmelig opfattende.

5 Konklusion

Vi har i dette projekt blandt andet udledt en matematisk hjælpemetode til at placere f.eks. et lager strategisk rigtigt i forhold til et antal konsumenter eller kunder, så transportudgifterne blev mindst mulige. Modellen er dog stadig simpel, men kan give et godt fingerpeg om, hvor lokationen skal være. Næste problem der belyses er ruteplanlægningen hvor begrebet tabusøgning bla. kommer ind i billedet. Vi har ligeledes set på en matematisk model for opstilling af et praktisk arbejdsskema, der kan bruges til at udnytte virksomhedens ressourcer på bedste måde, samtidig med at man ikke lader f.eks. køretøjer være ubrugte.
Vi er af den overbevisning at et nystartet firma som f.eks. FUP Transport A/S vil have stor gavn af at gøre sig de samme overvejelser og betragte de samme modeller som vi i dette projekt har gjort. På den måde vil de kunne forbedre deres økonomi og effektivitet betragteligt.

6 Litteratur

- Jørgen Tind: *En model for lokalisering*, IMF, KU, 2005.
- Beltrami, E.: *Models for Public Systems Analysis*, London 1977.
- Rasmussen, A. H.: *Ruteplanlægning i praksis*, Lyngby 2004.
- Wilson, A. G.: *Urban and Regional Models in Geography and Planning*, Leeds 1974.
- Ottesen, B. og Tranberg, L.: *Planlægning og optimering af logistiske problemstillinger i et kulturelt divergent miljø*, Lyngby 2004, http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/3141/pdf/imm3141.pdf
- http://neo.lcc.uma.es/radi-aeb/WebVRP/index.html?/Problem_Descriptions/VRPDesc.html
- http://eden.dei.uc.pt/~jast/vrp/