TEKST NR 67 1983

ELIPSOIDE METODEN - EN NY METODE TIL LINEÆR PROGRAMMERING ?

LONE BIILMANN

LARS BOYE

VEJLEDER:

MOGENS BRUN HEEFELT

TEKSTER fra

IMFUFA ROSKILDE UNIVERSITETSCENTER
INSTITUT FOR STUDIET AF MATEMATIK OG FYSIK SAMT DERES FUNKTIONER I UNDERVISNING, FORSKNING OG ANVENDELSER

IMFUFA, Roskilde Universitetscenter, Postbox 260, 4000 Roskilde.

ELIPSOIDE METODEN - EN NY METODE TIL LINEÆR PROGRAMMERING ?

Af
Lone Biilmann,
Lars Boye.

Vejleder
Mogens Brun Heefelt.

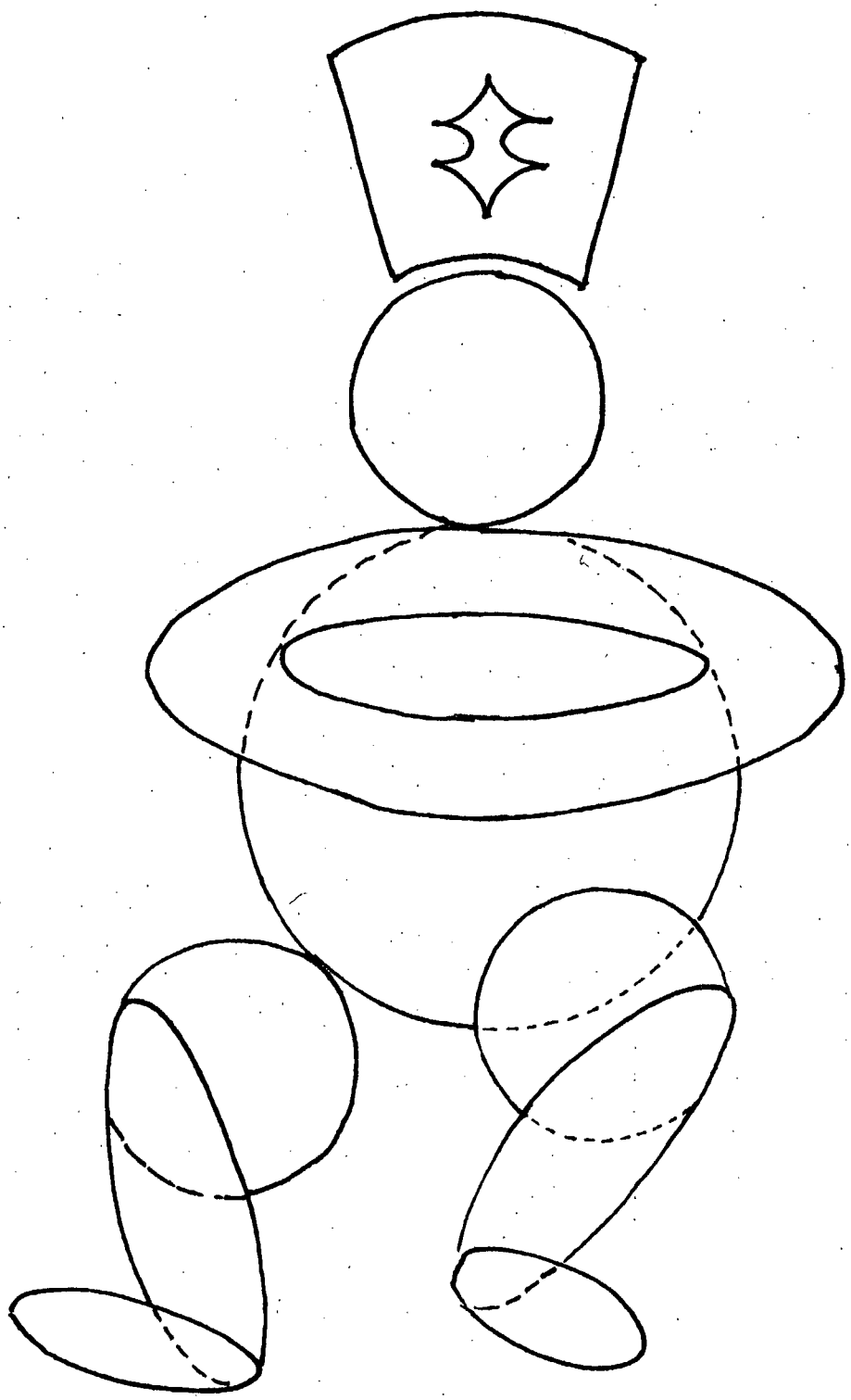
IMFUFA tekst nr. 67/83, RUC. 63 sider. ISSN 0106-6242

---

0. Abstract.

Denne rapport omhandler fremkomsten af den såkaldte elipsoidemetode til løsning af lineære uligheder og metodens betydning for lineær programmering og for teorien om algoritmers komplexitet.

De grundlæggende begreber ved NP-teorien og lineær programmering, herunder den mest brugte løsningsalgoritme Simplex, gennemgås. Derefter beskrives elipsoidemetoden og der føres bevis for hvorfor den, i modsætning til Simplex-algoritmen, er polynomial. Der gives et eksempel på en implementation af elipsoidemetoden og på de dermed forbundende problemer. Tilsidst diskuteres de teoretiske og praktiske konsekvenser af elipsoidemetodens fremkomst.



MATEMATIK MODUL II

Indholdsfortegnelse.

1. Introduktion.

Da vi skulle starte på dette projekt, blev vi af en datalogilærer gjort opmærksomme på en artikel i Byte(Berresford 1980). Artiklen beskriver en algoritme til løsning af lineære programmerings-problemer(L.P.) i polynomiel tid. Endvidere behandler artiklen det mediemæssige røre, algoritmens ophavsmands artikel(Khachiyan 1979) afstedkom.

Khachiyan's algoritme blev omtalt i de store aviser(Lawler 1980) som en opdagelse, der ville få betydning langt ud over fagkredse. Motivationen for en sådan og andre lignende udtalelser vil her kort blive opsummeret.

Når man vurderer en algoritme, er det oftest afgørende, hvor hurtig den er. Hvor hurtig en algoritme er afhænger naturligvis af, hvor stort et tilfælde af et givet slags problem, der skal løses. Denne størelse betegnes n. Man kan så finde en funktion af n, der betegner, hvor lang tid det højst kan tage at løse et tilfælde af størelsen n. Denne funktion kaldes komplexiteten. Hvis komplexiteten kan udtrykkes som et polynomium af n, kaldes algoritmen polynomiel. Alle andre komplexiteter kaldes eksponentielle. Den nedenstående figur viser nogle eksempler på forskellige algoritmers komplexitet samt deres tidsforbrug ved forskellige n.

kom- n / plexitet	10	20	30	40	50	60
n	$1\cdot10^{-5}$s	$2\cdot10^{-5}$s	$3\cdot10^{-5}$s	$4\cdot10^{-5}$s	$5\cdot10^{-5}$s	$6\cdot10^{-5}$s
n^2	$1\cdot10^{-4}$s	$4\cdot10^{-4}$s	$9\cdot10^{-4}$s	$16\cdot10^{-4}$s	$25\cdot10^{-4}$s	$36\cdot10^{-4}$s
n^3	$1\cdot10^{-3}$s	$8\cdot10^{-3}$s	$27\cdot10^{-3}$s	$64\cdot10^{-3}$s	$125\cdot10^{-3}$s	$216\cdot10^{-3}$s
n^5	1s	3.2s	24.3s	1.7m	5.2m	13m
2^n	$1\cdot10^{-3}$s	1s	17.9m	12.2d	35.7år	$4\cdot10^{4}$år
3^n	$5\cdot10^{-2}$s	58m	6.5år	$4\cdot10^{50}$år	$2\cdot10^{10}$år	$1\cdot10^{15}$år

For store værdier af n er det indlysende, at en polynomiel-tids-algoritme er langt at foretrække fremfor en exponentiel-tids-algoritme. Hvis man prøver at tænke på fremtidens datamater, bliver de måske 100 eller 1000 gange hurtigere. Nedenstående figur viser, hvor meget tid der vindes for de samme komplexiteter, som foregående figur. N_x betegner den største størelse af et problem, der på en nudags-datamaskine kan løses på en time.

komplexitet	nu-dags-data-mat	100 gange huttigere	1000 gange hurtigere
n	N_1	$100N_1$	$1000N_1$
n^2	N_2	$10N_2$	$31.6N_2$
n^3	N_3	$4.64N_3$	$10N_3$
n^5	N_4	$2.5N_4$	$3.98N_4$
2^n	N_5	$N_5+6.64$	$N_5+9.97$
3^n	N_6	$N_6+4.19$	$N_6+6.29$

Det ses, at også i fremtiden vil polynomielle algoritmer langt bedre kunne udnytte en forbedring af teknologien.

L.P.-problemer er oftest store problemer, d.v.s. n er stor. Man har siden krigen brugt Simplex-algoritmen til at løse L.P.-problemer. Denne algoritme vokser exponentielt med størrelsen af problemet. Derfor er det teoretisk meget opsigtsvækkende, at der nu eksisterer en polynomiel-tids-algoritme til løsning af L.P.-problemer. Spørgsmålet er så, hvilken praktisk betydning algoritmen vil få nu og på længere sigt.

Formålet med dette projekt er at give en introduktion til emnet og derigennem prøve at vurdere denne nye algoritmes betydning teoretisk såvel som praktisk.

2. Lineær programmering.(L.P.)

2.1. Operationsanalyse.

Når der skal tages en beslutning i en eller anden situation, er der ofte et utal af mulighedder at vælge imellem. Man stræber naturligvis efter at udpege den bedst mulige løsning blandt de mange mulige. Det er ofte tilfældet, at det er svært umiddelbart at vurdere de mange mulige løsninger m.h.p. den bedste. Der er derfor udviklet hjælpemidler til den slags situationer. Dette kaldes operationsanalyse.

Operationsanalyse bliver brugt i de fleste komplicerede situationer militært såvel som civilt. Eksempler på situationer, der er oplagte at anvende hjælpemidler til at analysere m.h.p. en beslutning, er produktionsplanlægning og investering. I begge tilfælde er det interessant at maximere afkastet på grundlag af de forudsætninger, man arbejder på. Disse forudsætninger giver sig udslag i en begrænsning i handlefriheden. Det kunne i de ovenstående eksempler være tidligere indgåede kontrakter eller skatteloven. Det kunne også være minimering, der var målet. F.eks., hvilke råstoffer skal anvendes til en given produktion for, at tungmetaludledningen bliver mindst mulig.

En meget anvendt del af operationsanalysen er L.P.. I det følgende vil den katagori af problemer, der kan analyseres v.h.a. L.P., blive beskrevet formelt.

2.2. Notation.

Et lineært programmerings-problem består af:

a) objektfunktionen, den funktion der skal optimeres
b) begrænsninger
c) ikke-negativitetsbetingelsen på de variable

a) Objektfunktionen er af formen:

$c_1x_1+c_2x_2+c_3x_3+.....+c_nx_n=z$, hvor

z er den værdi funktionen antager. Det er z, der enten skal minimeres(z(min)) eller maximeres(z(max)), altså z skal optimeres.
c'erne er konstanter, der kan betragtes som omkostningsfaktorer.
x'erne er de variable, hvis størelse ønskes bestemt, så z minimeres eller maximeres.
Objektfunktionen skrives ofte på følgende korte måde:
cx=z ,hvor
c er en n-række vektor og x en n-søjle vektor.
Iøvrigt minimeres funktionen, når maximum af -cx udregnes (og omvendt).

b) Når z skal optimeres, er de variable x ofte underlagt nogle begrænsninger. Disse opstilles som m relationer:

$$a_{11}x_1+a_{12}x_2+.....+a_{1n}x_n \{\leq,=,\geq\} b_1$$
$$a_{21}x_2+a_{22}x_2+.....+a_{2n}x_n \{\leq,=,\geq\} b_2$$
$$\vdots \qquad\qquad\qquad \vdots \qquad \vdots$$
$$a_{m1}x_1+a_{m2}x_2+.....+a_{mn}x_n \{\leq,=,\geq\} b_m$$

Disse relationer kan imidlertid omskrives til ligninger ved indførelse af skyggevariable. Hvis der indføres m skyggevariable $y_1,y_2,....,y_m$ kan begrænsningerne omskrives. Dette er illustreret ved de omringede relationer:

$$a_{11}x_1+a_{12}x_2+.....+a_{1n}x_n+y_1=b_1$$
$$a_{21}x_1+a_{22}x_2+.....+a_{2n}x_n+0y_2=b_2$$
$$\vdots \qquad\qquad\qquad \vdots \qquad \vdots$$
$$a_{m1}x_1+a_{m2}x_2+.....+a_{mn}x_n-y_m=b_m$$

Dette system af ligninger kan kortere skrives, som

$Ax+y=b$,hvor

A er en m×n-matrix,x en n-søjle vektor, y en m-søjle vektor og b en m-søjle vektor.

c) De n variable $x_1,x_2,....,x_n$, der skal tillægges værdi ved en optimering, skal tillægges ikke-negative værdier. Man kan f.eks. ikke producere et negativt antal af en bestemt vare. Ikke-negativitetsbetingelsen skrives kort:

$x \geq 0$, hvor

0 er en n-søjle nulvektor.

Det ses af ovenstående, at de problemer, der kan optimeres v.h.a. L.P., skal kunne opskrives i lineære udtryk. F.eks. kan funktioner, hvor potenser af et eller flere x'er indgår ikke optimeres v.h.a. L.P..

Når en løsningsmetode til L.P.-problemer beskrives, benyttes som udgangspunkt et L.P.-problem på standardform. I matrix-notation ser standardformen ud, som følger:

$cx=z$
$Ax=b$
$x \geq 0$

Her er x-vektoren tilføjet skyggevariablene. Disse er altså også underlagt ikke-negativitetsbetingelsen. Til betegnelse af matricernes størrelser anvendes de samme bogstaver som tidligere, dog betegner n nu summen af det oprindelige n(nu n') og m. D.v.s., matricernes størrelse er:

$c(1\times n): (c_1\ c_2\ \dots\ c_n)$

$x(n\times 1): \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n'} \\ y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}$ $b(m\times 1): \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}$

$$A(m\times n): \begin{pmatrix} a_{11} & a_{12} \cdots\cdots a_{1n} & 1 & 0 \cdots 0 \\ a_{21} & a_{22} \cdots\cdots a_{2n} & 0 & 1 \cdots 0 \\ \cdot & \cdot \qquad\quad \cdot & \cdot & \qquad \cdot \\ \cdot & \cdot \qquad\quad \cdot & \cdot & \qquad \cdot \\ a_{m1} & a_{m2} \qquad a_{mn} & 0 & 0 \qquad 1 \end{pmatrix}$$

Fordi skyggevariablene optræder netop en gang i hver sit bånd, kan de indeholdes i A-matricen, ved tilføjelse af en m×n enheds-matrix.

2.3. Dualitet.

Når man maximerer et L.P.-problem af formen:

$$cx=z(max)$$
$$Ax \leq b$$
$$x \geq 0$$

minimeres samtidig L.p.-problemet på formen:

$$by=w(min)$$
$$A^t y \geq c$$
$$y \geq 0$$

Maximeringsproblemet siges at være det primale problem, mens minimeringsproblemet siges at udgøre det duale problem.

Hvis relationen i det primale L.P.-problem er $Ax \leq b$, ganges igennem men -1, så relationen ændres til $\geq$. Hvis det z, der skal udregnes i det primale L.P.-problem er minimum af objektfunktionen, udregnes maximum w af det duale problems objektfunktion. D.v.s., hvis et L.P.-problem på standardform løses, er der i denne løsning også indeholdt løsningen på det duale problem(hvis man har brug for den?). Dette kan illustreres i det såkaldte Trucker-diagram:

		PRIMAL		
	variable	$x_1 \geq 0 \ldots\ldots\ldots x_n \geq 0$	realationer	konstanter
D U A L	$y_1 \geq 0$	$a_{11} \ldots\ldots\ldots a_{1n}$	$\geq$	b_1
	.	. .	.	.
	.	. .	.	.
	.	. .	.	.
	$y_m \geq 0$	$a_{m1} \ldots\ldots\ldots a_{mn}$	$\geq$	b_m
	realationer	$\leq \ldots\ldots\ldots \leq$		$\leq w(max)$
	konstanter	$c_1 \ldots\ldots\ldots c_n$	$\geq z(min)$	

Diagrammet viser,at:
Objektfunktionen skal minimeres i den primale form og maximeres i den duale form.
Ulighedstegnene i minimeringsproblemet overalt er $\geq$ og $\leq$ i maximeringsproblemet.
Koeficienterne i den ene forms objektfunktion udgør højresiden af begrænsningerne i den anden.
A-matricen transponeres ved overgang fra den ene form til den anden.
Antallet af variable i den ene form er lig antallet af begrænsninger i den anden.
Alle variable er underlagt ikke-negativitets-betingelsen.

3. Klasserne P og NP.

3.1. Algoritme analyse.

Et vigtigt område inden for teoretisk datalogi er analysen af algoritmer og beregningen af deres effektivitet. En del af de algoritmer, man kender til løsning af kombinatoriske problemer, kan kun anvendes til løsning af mindre tilfælde. Dette skyldes, at antallet af operationer, der skal udføres af algoritmen, vokser eksponentielt med størelsen af tilfældet. Et eksempel herpå er problemet med "Den Rejsende Sælger"(DRS), hvor man skal finde den hurtigste vej, som besøger alle byer i et givet område(Garey 1979).

For at forstå disse problemers sammenhæng er der udviklet en teori, som kaldes teorien om NP-komplethed. Det er denne teori, som dette kapitel omhandler. I afsnit 3.2. forklares begrebet algoritmers komplexitet og NP-teorien gennemgås i afsnit 3.5..

3.2. Algoritmers komplexitet.

Når man vurderer, hvor "god" en algoritme er, måler man oftest dette i kørselstid. Kørselstiden er selvføgelig ikke ens på forskellige datamater. Når man har en algoritme, afhænger kørselstiden også af, hvor stort et tilfælde af problemet man ønsker beregnet. For at kunne generalisere målet for en algoritmes effektivitet maskinuafhængigt har man indført begrebet: en algoritmes komplexitet.

En algoritmes komplexitet er en funktion, der beskriver tiden for en algoritmes udførelse afhængig af tilfældets størelse. Tilfældets størelse betegnes n, og er størrelsen af det input, algoritmen kræver for at løse tilfældet. n er altså fysisk en tegnstreng. Komplexiteten betegner så antallet af elementære instruktioner, der skal udføres for at løse problemet. Disse instruktioner, forudsættes at have samme udførselstid.

Komplexiteten udtrykker altid den størst mulige udførselstid for et givet n. Man kan som oftest finde tilfælde med samme n, der kræver et langt mindre antal elementære instruktioner. Hvis man beregner komplexiteten for en algoritme, kan man altså teoretisk (maskinuafhængigt) sammenligne den med andre algoritmer. Man kan

også (hvis det har interesse) udfra komplexiteten lave overslag over kørselstiden på en bestemt datamat udfra manualernes oplysninger om udførselstider.

3.3. Eksponentielle og polynomielle algoritmer.

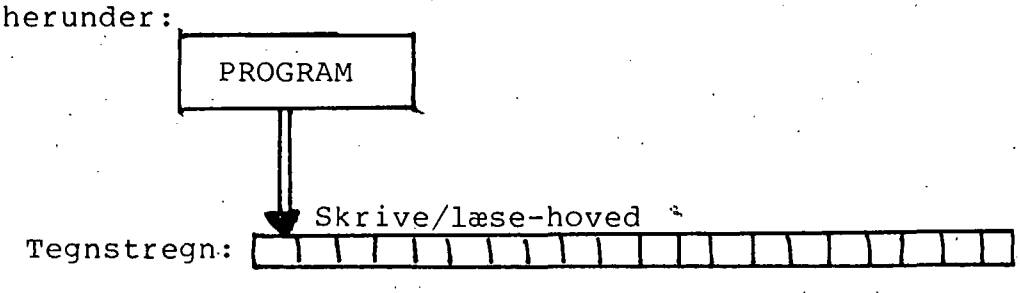
I kapitel 1 er der vist to figurer, som viser, hvordan algoritmer med forskellig komplexitet vil opføre sig. Algoritmer, hvis komplexitet kan udtrykkes som et polynomie, p(n), i størrelsen n, kalder man for polynomielle. Man siger også, at der eksisterer en polynomialtids algoritme til løsning af problemet. Algoritmer, hvis komplexitet vokser eksponentielt med størrelsen af n, siges at være eksponentielle. De to figurer viser, at både nu og i fremtiden er det idealet at konstruere polynomielle algoritmer.

Dog er der konstrueret polynomial-tids algoritmer, der først syntes meget tidskrævende. Det er en tommelfingerregel, at forbedringer plejer med tiden at føre til hurtige algoritmer, hvis bare udgangsalgoritmen er en polynomial-tids algoritme. Mange eksponentiel-tids algoritmer virker i dag fint i praksis, fordi de tilfælde man beregner ikke kræver udførelse af det teoretiske største antal instruktioner.

Som det også ses af første figur i kapitel 1 kan en eksponentiel-tids algoritme være hurtigere for små n (her 10) end en polynomial-tids algoritme. Man kan finde flere ekstremer, hvor det praktisk viser sig fordelagtigt at benytte en eksponentiel-tids algoritme. Der er dog alligevel enighed om, at de største fremskridt opnås ved konstruktion af polynomial-tids algoritmer.

3.4. Turing Maskinen.

Når man analyserer en algoritme må man have en model på hvilken algoritmen kan udføres. Her vil blive brugt en meget primitiv model, som kaldes en Turing Maskine (T.M.). Modellen er skitseret herunder:



Instruktionerne i programmet er af formen:

1:IF σ THEN (σ';0;1'), som har følgende betydning:

1,1' : er etiketter i programmet
σ,σ' : er symboler fra et endeligt alfabet
0 : er et af numrene 1,0,-1.

Makskinen virker altså, som følger: Hvis det sidst læste symbol er σ, så slet det og skriv i stedet σ'. Flyt derefter skrive/læse-hovedet til symbolet, der står 0 pladser væk fra den nuværende position. Hvis det nuværende symbol ikke er σ, så fortsæt med næste instruktion.

Selvom "instruktionssættet" synes meget begrænset, kan man godt udføre mere komplicerede instruktioner ved små ændringer i modellen.

Ved at omskrive en algoritme, så den udføres på en T.M., kan man få et mål for algoritmens komplexitet og for, hvor "svært" det bagvedliggende problem er at løse.

3.5. NP-teorien.

NP-teorien bygger på en opdeling af datalogiske problemer i klasser, for hvilke man definerer nogle overordnede begreber.

Der er defineret en klasse P, der indeholder alle de problemer, som kan løses af en polynomial-tids algoritme. Disse er "gode" problemer. Eksempler herpå er følgende hentet fra (Papadimitriou 1982):

I. givet en graf G, er kanterne i G forbundet?
II. givet en orienteret graf D og to delmængder S,T af D's knuder. Er der en vej fra en knude i S til en knude i T?

Klassen P kan defineres på flere måder, f.eks. som de problemer, der kan løses på en T.M..

Som tidligere nævnt, er det langt fra alle problemer, der kan løses af en polynomial-tids algoritme. Men de fleste datalogiske problemer kan omskrives til en genkendelses form, hvor der bare skal svares ja eller nej i polynomiel tid. D.v.s., at man har en løsning til et givet tilfælde af et problem, og så spørges: "Er

denne løsning rigtig?" Hertil skal svares ja eller nej. Det viser sig (Papadimitriou 1982), at de fleste problemer kan omskrives til genkendelsesform fra hvilken svaret (ja eller nej) beregnes i polynomiel tid. Disse problemer udgør klassen NP.

For at kunne udtrykke dette mere formelt indføres nogle symboler. Σ er et givet sæt af tegn, et alfabet. x er en løsning til det omskrevne problem A bestående af tegn fra Σ. $|x|$ angiver antallet af tegn i x. c(x) er det kriterie, skrevet i Σ, som x skal opfylde. $|c(x)|$ må højst være polynomiel i $|x|$. \$ er et skilletegn fra Σ.

DEFINITION: A tilhører NP, hvis der eksiterer en algoritme α og et polynomie p(n), således at hvis α får x\$c(x) som inddata, hvor $|c(x)| \leq p(|x|)$, så når α frem til svaret ja efter højst $p(|x|)$ skridt.

Eksempel: Som et eksempel tages L.P.. Hvis man har m uligheder med n ubekendte, og kender max af cx og $x \geq 0$, tager det n(2m+1)+1 operationer at verificere at x er en løsning. Inddata-strengen består f.eks. af $x\$a_{11}x_1+..+a_{n1}x_n \leq b_1,..,a_{m1}x_1+..+a_{mn}x_n \leq b_m, x_1 \geq 0,..,x_n \geq 0, c_1x_1+..+c_nx_n=max$. Da $|c(x)| \leq max_c(m(3n+1)+3n-1)$, hvor max_c er længden af det største tal i c(x), tilhører L.P. altså NP.

Det fremhæves i (Papadimitriou 1982), at det ikke er nødvendigt at vise, at c(x) kan beregnes effektivt udfra x. Det er nok at vise, at der bare eksisterer en tegnstreng c(x). Det viser sig, at klassen P er en delmængde af NP, fordi, det, at løse et problem fra P, kan siges at svare til at afgøre, om x opfylder c(x).

Når man skal løse et datalogisk problem, kan det være en hjælp at kunne omskrive det til et problem, for hvilket man allerede kender en brugbar løsningsalgoritme.

DEFINITION: Et genkendelsesproblem A_1 siges at kunne transformeres polynomielt til et andet genkendelsesproblem A_2, hvis der eksisterer en polynomial-tids algoritme, der omformer en løsningsstreng for A_1 til en løsningsstreng for A_2 (Papadimitriou 1982).

Det er en meget praktisk definition. Den medfører bl.a., at hvis A_1 kan transformeres polynomielt til A_2, og der eksisterer en polynomial-tids algoritme for A_2, så eksisterer der også en polynomial-tids algoritme for A_1. Definitionen bruges også til at definere en klasse af NP-komplette problemer.

DEFINITION: Et problem siges at være NP-komplet, hvis alle andre problemer i NP kan transformeres polynomielt til det (Papadimitriou 1982).

For at kunne opfylde denne definition må klassen af NP-komplette problemer have følgende to egenskaber:

I : Intet NP-komplet problem kan løses v.h.a. en plynomial-tids algoritme.

II : Hvis der findes en polynomial-tids algoritme for et enkelt NP-komplet problem, så findes der polynomial-tids algoritmer for alle Np-komplette problemer.

Disse to egenskaber ved NP-komplette problemer gør NP-kompletheden til et meget effektfuldt begreb. NP-komplette problemer er f.eks. problemer som D.S.R. og heltals L.P.. Begge problemer som det indtil nu har vist sig umuligt at løse i polynomiel-tid.

Hvis man nu for et givet problem kan vise, at det er NP-komplet, så ved man, at der ikke er nogen polynomial-tids algoritme til løsning af det.

For at bevise at et problem er NP-komplet, skal det opfylde følgende to betingelser:

I : Problemet skal tilhøre NP.

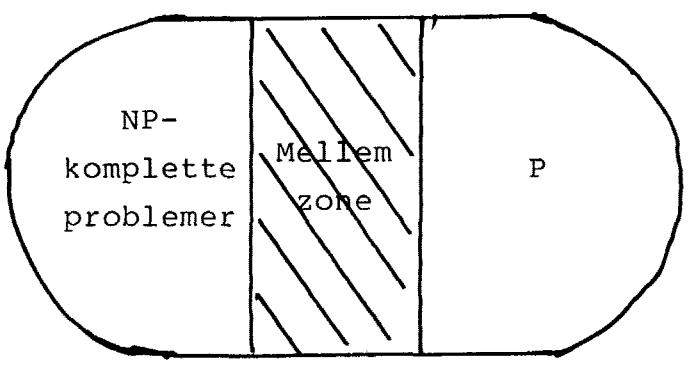
II : Alle andre problemer i NP kan transformeres polynomielt til dette problem.

I praksis vil det være et tilstrækkeligt bevis for II, at et af de kendte NP-komplette problemer kan transformeres til det pågældende problem.

Teorien om NP-komplette problemer medfører også, at hvis man først finder en polynomial-tids algoritme, som løser et NP-komplet problem, så kan man også finde polynomial-tids algoritmer til alle de andre NP-komplette problemer. Det vil da med andre ord sige, at P=NP.

Spørgsmålet, om P=NP eller om P≠NP, er et meget diskuteret spørgsmål. Problemet er også, at der er problemer, som synes hverken at være NP-komplette eller tilhøre P.

NP-klassens topografi synes at kunne anskueliggøres på følgende måde:



4. Simplex.

4.1. Indledning.

Siden krigen har der stort set kun været anvendt en algoritme til numerisk løsning af L.P.-problemer. Denne algoritme hedder Simplex-algoritmen. Det er Simplex-algoritmen (og variationer af samme), der i dag er det datalogiske værktøj til løsning af L.P.-problemer.

Som forudsætning for at forstå og beskrive Simplex-algoritmen introduceres først nogle egenskaber ved L.P.-problemers løsningsstruktur. På grundlag af disse egenskaber beskrives Simplex-algoritmen, og afsnittet afsluttes med en diskussion af algoritmens effektivitet.

4.2. Matematiske knæbøjninger.

Formålet med dette afsnit er at beskrive L.P.-problemers løsningsegenskaber. Disse vil derfor ikke blive bevist, men beviserne kan læses i (Krarup 1974). De løsningsegenskaber, der nu beskrives, skal anvendes som grundlag for forståelsen af Simplex-algoritmen. Denne skal senere udsættes for en nærmere effektivitetsanalyse.

Givet et L.P.-problem på standardform ($cx=z$(max), $Ax=b$ og $x \geq 0$) vil følgende tre begreber blive diskuteret:

a) løsningsmængden
b) ekstrempunkter
c) optimale løsning(er)

a) Løsningsmængden M til et L.P.-problem på standardform udgør en lukket konvex mængde. D.v.s., hvis vektorerne $x^1, x^2, .., x^r$ er mulige løsninger til et L.P.-problem, så er enhver vektor x, som er en konvex kombination af $x^1, .., x^r$ også en løsning.

Eksempler:

a1) To af et L.P.-problems begrænsninger ser ud, som følger:

$$x_1 \leq 7$$
$$x_1 \geq 9$$

Her vil M være tom. Det betyder, at L.P.-problemet ikke har nogen løsning, men M er alligevel en lukket konvex mængde.

a2) Hvis båndene ser ud som følger:

$$x_1 \geq 3$$
$$x_2 \geq 2$$
$$(x_1, x_2) \geq 0$$

er M ubegrænset og Simplex-algoritmen giver besked i et sådant tilfælde. Derimod har et minimeringsproblem med følgende bånd:

$$x_1 \geq 1$$
$$x_1 \geq 0$$

løsningen 1, selvom M er ubegrænset.

a3) Når M består af mere end et punkt, så indeholder M uendelig mange punkter.

b) En optimal løsning til et L.P.-problem findes i et ekstremt punkt af løsningsmængden M, d.v.s. i et punkt som tilhører M, men som ikke kan dannes som en konvex kombination af andre punkter i M. Såfremt en optimal løsning findes i flere ekstreme punkter, så er enhver konvex kombination af disse også en optimal løsning. Med andre ord kan optimale løsninger også findes ved ikke-eksteme punkter af løsningsmængden i tilfælde, hvor der er flere og derfor uendeligt mange løsninger.
Eksempel:
b1) Hvis et optimum findes i et ikke-ekstemt punkt, da må dette punkt kunne skrives som en konvex kombination af mindst to ekstreme punkter, der svarer til optimale løsninger.

c) Til enhver ekstrempunktsløsning er knyttet en base af m lineært uafhængige søjle-vektorer fra A-matricen. De basis-vektorer a^j, der svarer til positive c_j i objektfunktionen, angiver værdierne af x_j. De x_i, hvor søjle-vektoren a^i ikke indgår i basen, indgår i beregningen af objektfunktionen med værdien 0. At finde en optimal løsning svarer til at finde de(n) base(r) af vektorer for ekstrempunkt(er), hvor cx antager sit max eller min.

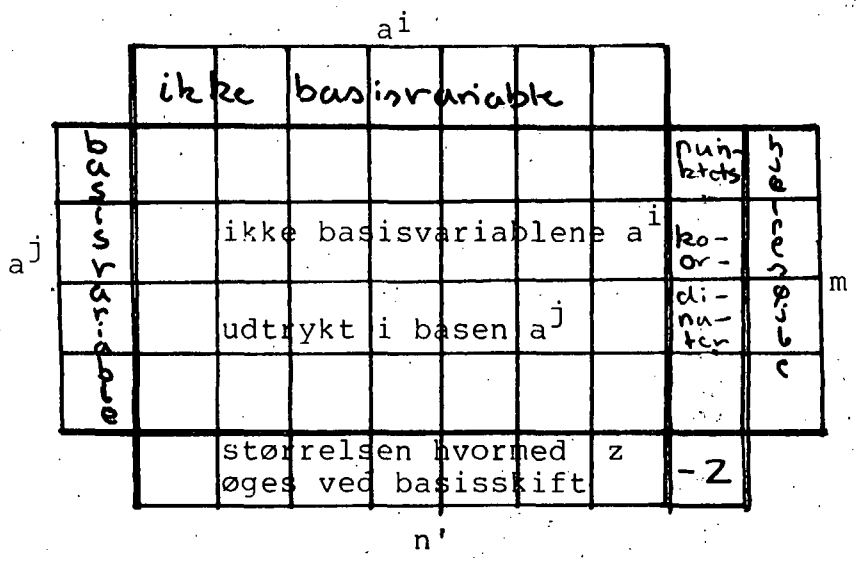
4.3. Simplex-algoritmen.

Når man vil finde en optimal løsning til et L.P.-problem, skal denne søges i de ekstreme punkter i løsningsmængden M. Der vil dog være $\binom{n}{m}$ ekstreme punkter, så selv for et beskedent L.P.-problem vil det være en fordel at systematisere eftersøgningen af en eller

flere optimale ekstreme punkter i M.

Givet et L.P.-problem på standardform behandler Simplex-algoritmen dette i to faser. Den første fase finder en mulig basisløsning. Fase to forbedrer iterativt v.h.a. basisskift ekstrempunktsløsningen til en optimal foreligger. Det skal her tilføjes at Simplex-algoritmen finder cx(max) ved at finde -cx(min).

Til beregningerne i Simplex-algoritmen anvendes en tabel. Den ser skematisk ud som følger:



I første fase vælges en startbasis for et ekstremt punkt. Et naturligt valg er her en base bestående af de m søjle-vektorer for skygge - variablene. Koordinaterne for ekstempunktet udgøres da af b-vektoren. Søjle-vektorene for skyggevariablene udgør, som tidligere nævnt, en mxm-enhedsmatrisse. Derfor kan de n'ikke-basisvektorer skrives som den gamle mxn'-matrisse A i den valgte base. Den værdi, som z øges med (mindskes med) ved udskiftning af en basisvedktor a^j med en ikke basisvektor a^i, er for den første tabel værdien c_i. Da alle skyggevariablene indgår i objektfunktionen med vægten 0, bliver z initialiseret til 0. Den første tabel ser ud, som følger:

	x_1	x_2	.	.	.	.	$x_{n'}$		
y_1	a_{11}	a_{12}	.	.	.	.	$a_{1n'}$	b_1	
.	.	.	.	.	.	.	.	.	
y_m	a_{m1}	a_{m2}	.	.	.	.	$a_{mn'}$	b_m	
	c_1	c_2	.	.	.	.	$c_{n'}$	0	

Fase to indledes med at undersøge, om værdien af z kan forøges ved at lave et basisskift. Hvis ingen af ikke-basisvektorene kan bidrage med en positiv værdi, stopper algoritmen. Ellers vælges en vektor a^i blandt ikke-basis-variablene. Man kan f.eks. vælge den a^i, som bidrager mest til optimering af z.

Nu findes en basisvektor a^j, som skal give plads for a^i. At foretage dette basisskift svarer til at bevæge sig fra et ekstremt punkt over imod et andet i retning a^i. Punktet er nået, når en af basisvektorene bliver 0. Et mål for, hvor langt man kan gå før dette sker, får man ved at dividere de positive elementer i a^i-vektoren op i de positive koordinater for det gamle ekstem punkt. Dette gøres række for række, når begge operander er positive, og resultatet indføres i hjælpesøjlen. Hvis alle koordinater for ekstrem punktet er ikke-positive, stopper algoritmen. Som a^j vælges den basisvektor, som svarer til den mindste positive værdi i hjælpesøjlen.

Basisskiftet transformerer Simplex-tabellen. Transformeringen sker ved en pivotering. Som pivoteringselement vælges elementet a_{ji}. Hvis tabellen betragtes som en (m+1)x(n'+1)-matrix med elementerne d_{kl} (k=1,..,m+1 og l=1,..,n'+1), kan reglerne for denne pivotering om d_{ji} opstilles, som følger:

a) d_{ji} bliver ændret til $1/d_{ji}$

b) d_{jl} " " " d_{il}/d_{ji} $l \neq i$

c) d_{ki} " " " $-d_{ki}/d_{ji}$

d) d_{kl} " " " $d_{kl}-(d_{ki}d_{jl}/d_{ji})$ $l \neq i$ } $k \neq j$

Den nye Simplex-tabel ser nu ud, som følger:

	x_1	.	y_j	.	$x_{n'}$	$n'+1$	
y_1	$d_{1i}-\frac{d_{1i}d_{j1}}{d_{ji}}$	.	$-\frac{d_{1i}}{d_{ji}}$	.	.	.	
.	.	.	.	.	.	.	
x_i	$\frac{d_{j1}}{d_{ji}}$	.	$\frac{1}{d_{ji}}$	.	.	$\frac{d_{jn'+1}}{d_{ji}}$	
.	.	.	.	.	.	.	
y_m	.	.	.	.	.	.	
m+1	.	.	$-\frac{d_{m+1,i}}{d_{ji}}$	.	.	.	

Når algoritmen stopper, kan det have tre årsager:

a) Alle værdierne i række m+1 er negative.

b) Alle værdierne i række m+1 er negative eller nul.

c) Alle værdierne i søjle j er negative eller nul.

a) Hvis alle værdierne i række m+1 er negative, før man påbegynder en ny iteration, betyder der, at objektfunktionen ikke kan forøges ved basisskift. Det ekstrem punkt, man befinder sig i, er den optimale løsning. Løsningen kan aflæses i søjle n'+1. Løsningsvektoren x består af koordinaterne til de basisvektorer a^j som indgår i objektfunktionen med positive c_j. De x_i, som svarer til ikke-basisvektorerne a^i, sættes til nul.

b) Hvis nogle af værdierne i række m+1 er nul istedet for negative, betyder det, at man kan foretage et basisskift med vektorer a^i, uden at værdien af z ændres. Den optimale løsning eksisterer i flere ekstrem punkter, og der er derfor uendeligt mange løsninger. Løsningsrummets dimension svarer til antalllet af søjler med nul i række m+1, og dette rum er fastlagt af ekstem punkterne.

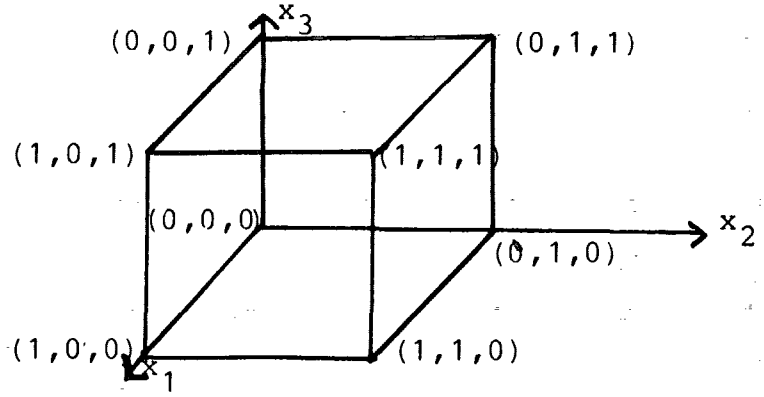
c) Hvis alle værdierne i søjle j bliver ikke-positive, betyder det, at der ikke er nogen løsning til problemet.

4.4. Simplex-algoritmens komplexitet.

Hvis Simplex-algoritmen skal placeres i klassen P skal det vises, at Simplex-algoritmen i et polynomielt antal skridt finder den optimale løsning til et L.P.-problem. Hvis det derimod antages at Simplex-algoritmen tilhører klassen NP, skal det vises, at Simplex-algoritmen bruger et eksponentielt antal skridt. Det kan så let verificeres, at det er en optimal løsning, der er beregnet.

Målet for en algoritmes komplexitet angiver altid et "værste tilfælde". Derfor er det nok at vise, at ét L.P.-problem giver anledning til en eksponentiel komplexitet. Mere formelt skal det vises, at der eksisterer en mængde ekstrempunkter $x^1, x^2, .., x^k$, som parvis er forbundet af en kant i løsningsområdet, og som tilfredsstiller betingelsen $cx^{i+1} < cx^i$, når i=1,2,..,k-1. Denne mængde skal vokse eksponentielt med størrelsen af L.P.-problemet. Det følgende vil vise, at Simplex er ikke-polynomiel. Beviset er konstrueret af (Klee 1972).

Betragt først denne kube:



Kuben tilfredsstiller begrænsningerne

$$0 \leq x_j \leq 1, \quad j=1,2,3.$$

Sådan en 3-dimensional kube har 6 sider og 8 knuder (ekstrempunkter til ovenstående uligheder). Generelt har d-dimensionale kuber defineret ved ulighedderne

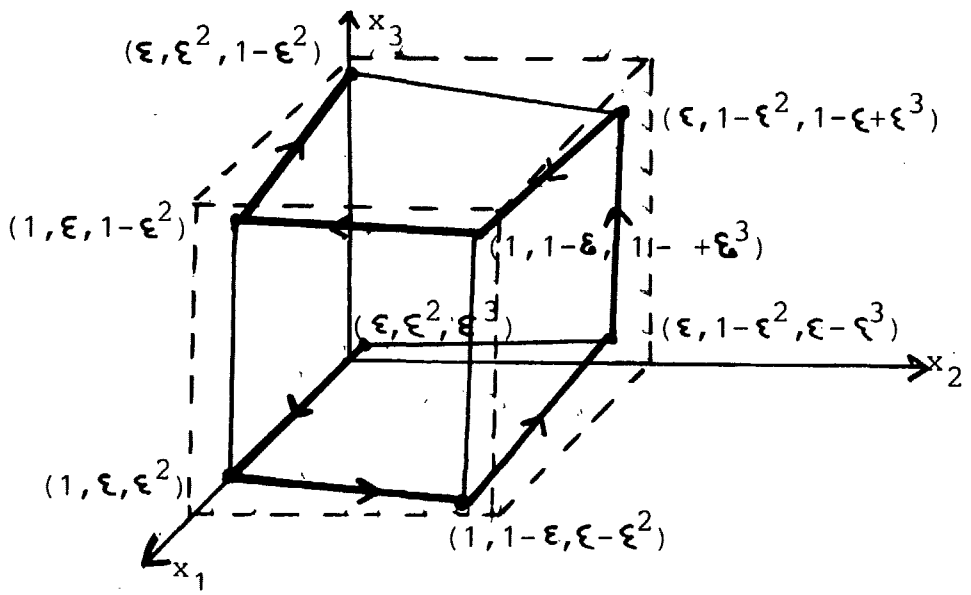
$$0 \leq x_j \leq 1, \quad j=1,2,..,d$$

2d sider (en for hver ulighed) og 2^d knuder (en permutation af d,0 og 1). Nu defineres en polyeder v.h.a. et ε i intervallet $0<\varepsilon<\frac{1}{2}$ ved ulighedderne

$$\varepsilon \leq x_1 \leq 1$$

$$\varepsilon x_{j-1} \leq x_j \leq 1-\varepsilon x_{j-1}, \quad j=2,3,..,d.$$

Polyederen er en vridning af en d-kube og kan se ud som her:



Det specielle ved dette polyeder er, at der eksisterer en vej (den aftegnede) igennem polyederens knuder (ekstrempunkter) således, at objektfunktionen mindskes hver gang man bevæger sig til den næste knude. Når alle knuderne er gennemløbet er funktionen minimeret. Vejens længde, d.v.s. antallet af pivoteringer, svarer til antallet af knuder i polyederen. Antallet af knuder findes ved at lade ε gå imod nul, hvor ved d-kuben bliver grænsen for løsningsområdet.

D.v.s. at mængden af ekstrempunkter som skal gennemløbes er 2^d-1. Resten af afsnittet uddyber dette bevis matematisk.

På ovenstående figur er indikeret en række af knuder, der minimerer objektfunktionen. De ovenstående uligheder omskrives nu til standardform ved at tilføje 2d skyggevariable

$$-cx=z(\min) \qquad (z \text{ maximeres})$$

$$\left.\begin{aligned} x_j-\varepsilon x_{j-1}-r_j&=0\\ x_j+\varepsilon x_{j-1}+s_j&=1 \end{aligned}\right\} \quad j=2,3,..,d$$

$$x_j,r_j,s_j \geq 0 \quad j=1,2,..,d$$

Mængden af ekstrempunktsløsninger til ovenstående L.P.-problem er alle delmængder af $\{x_1,..,x_d,r_1,..,r_d,s_1,..,s_d\}$, bestående af alle x'erne og nøjagtigt et af $\{r_j,s_j\}$ for hvert j, j=1,2,..,d.

BEVIS: Fordi $x_1 \geq \varepsilon$ og $x_{j+1} \geq \varepsilon x_j$, j=1,..,d-1, er $x_j \geq \varepsilon^j > 0$ i enhver mulig løsning. Så alle mulige ekstrempunkter må indeholde alle d søjler svarende til x'erne. Forudsæt nu, at $r_j=s_j=0$ for et vilkårligt j. Dette j undersøges nu for

j=1: $\varepsilon=x_1=1$, som er falsk fordi $0<\varepsilon<\frac{1}{2}$.

j>1: $x_j=\varepsilon x_{j-1}$ og $x_j+\varepsilon x_{j-1}=1$ hvilket giver
$\varepsilon x_{j-1}=\frac{1}{2}$, som er falsk fordi $0<\varepsilon<\frac{1}{2}$ og $x_{j-1} \leq 1$.

Nu kan det konkluderes, at ethvert ekstrempunkt må indeholde en af søjlerne svarende til s_j og r_j for hvert j. Da, der er 2d skyggevariable, svarer det til 2d ekstrempunkter.

Et ekstrempunkt betegnes som x^S, hvor S er den delmængde af $\{1,2,..,d\}$, der svarer til de r i x^S hvor $r \neq 0$. Værdien af x_j i x^S vil blive betegnet x_j^S. Forudsæt, at $d \in S$ men at $d \notin S$, så gælder $x_d^S > x_d^{S'}$. Endvidere, hvis $S'=S-\{d\}$ så er $x_d^{S'}=1-x_d^S$.

BEVIS: Når $d \in S$ så er $s_d=0$ og den fjerde begrænsning i standardformen bliver

$$x_d^S=1-\varepsilon x_{d-1}^S. \quad x_{d-1}^S \leq 1 \text{ og } \varepsilon<\tfrac{1}{2} \text{ så } x_d^S>\tfrac{1}{2}.$$

Når $d \notin S'$ så er $r_d=0$ og den tredie begrænsning i standardformen bliver $x_d^{S'}=\varepsilon x_{d-1}^{S'}<\frac{1}{2}$. Altså er $x_d^S > x_d^{S'}$. Hvis $S'=S-\{d\}$ så er $S=S' \cup \{d\}$. D.v.s. $x_{d-1}^{S'}=x_{d-1}^S$ så er $x_d^{S'}=\varepsilon x_{d-1}^{S'}=1-(1-\varepsilon x_{d-s}^S)=1-x_d^S$.

Hvis man opstiller delmængderne af $\{1,2,..,d\}$ sådan at

$$x_d^{S_1} \leq x_d^{S_2} \leq \ldots \leq x_d^{S_{2^d}},$$

så vil uligheddene være strikte og ekstrempunkterne x^{S_j} og $x^{S_{j+1}}$ vil være tilstødende for $j=1,2,..2^d-1$.

BEVIS: Beviset gennemføres v.h.a. induktion. Det ovenstående gælder for d=1, her er to ekstrempunkter: $(x_1,r_1,s_1)=(\varepsilon,0,1-\varepsilon)$ og $(x_1,r_1,s_1)=(1,1-\varepsilon,0)$. Punkterne har forskellige x_1'er og støder op til hinanden. Induktionsskridtet udføres ved at antage at forholdet gælder for en d-kube hvor $S_1,..,S_{(2^d)}$ er ordnede. $S_1,..,S_{(2^d)}$ er også delmængder af $\{1,2,..,d+1\}$ og $x_{d+1}^{(S_j)} = \varepsilon x_d^{(S_j)}$.

Det gælder derfor at

$$x_{d+1}^{(S_1)} < x_{d+1}^{(S_2)} < \ldots < x_{d+1}^{(S_{2^d})}.$$

Betragt de resterende delmængder af $\{1,2,..,d+1\}$,

$$S_j' = S_j \cup \{d+1\}, \quad j=1,2,..,2^d.$$

Det er tidligere vist at

$$x_{d+1}^{(S_j')} > x_{d+1}^{(S_{2^d})} \quad \text{og} \quad x_{d+1}^{(S_j')} = 1-x_{d+1}^{S_j},$$

derfor gælder det nu at

$$x_{d+1}^{S_j} < \ldots < x_{d+1}^{(S_{2^d})} < x_{d+1}^{(S_2'd)} < \ldots < x_{d+1}^{(S_1')}.$$

Ifølge induktionen er x^{S_j} og $x^{S_{j+1}}$ tilstødende ligesom $x^{(S_j')}$ og $x^{(S_j')}$ er det. Det gælder nu, at $x^{(S_{2^d})}$ og $x^{(S_2'd)}$ er tilstødende fordi $x^{(S_2'd)}$ fås fra $x^{(S_{2^d})}$ ved at udskifte s_{d+1} med r_{d+1}.

Det kan så vises at der for et hvert d>1 er et L.P.-problem med 2d uligheder, 3d variable og med heltallige koeficienter, hvis absolutte værdi er afgrænset af 4, hvorom det gælder, at det tager 2^d-1 iterationer for Simplex, at finde den optimale værdi.

BEVIS: Det kan ses ved at vælge $\varepsilon=1/4$ og multiplicerer alle uligheddene i standardformen med 4, så alle koeficienterne bliver heltallige. Da x_d i standardformen skal maximeres har den eksponentielt lange række af ekstrempunkter som det lige er vist kan dannes faldende omkostninger.

5. Elipsemetoden(E.M.).

5.1. Indledning.

Indtil nu er L.P.-problemet blevet beskrevet og den indtil nu bedst kendte løsningsalgoritme, Simplex-algoritmen, beskrevet og analyseret. Da man længe har haft svært ved at placere L.P.-problemet i NP-teorien, er det af stor teoretisk betydning, at man for nylig(Khachiyan 1979) har fået kendskab til en polynomialtids algoritme til løsning af L.P.-problemet. Som grundlag for algoritmen ligger elipsemetoden(E.M),der er grundlæggende forskellig fra Simplex-metoden. I stedet for at undersøge ekstrempunkter i et løsningsområde, konstrueres en række af elipsoider, som snævrer sig sammen om den optimale løsning.

E.M. vil blive teoretisk behandlet i dette afsnit. Først vil metoden blive introduceret for i det følgende at blive opskrevet formelt. Derefter følger et ret omfattende bevis for metoden, som også er et bevis for, at E.M. er en polynomial-tids algoritme.

I dette kapitel vil de teoretiske landvindinger, E.M. er, blive omtalt, og først i de følgende afsnit vil de praktiske problemer blive beskrevet og diskuteret.

5.2. Beskrivelse af E.M..

E.M. kan, givet et sæt af lineære uligheder, finde en vektor, der tilfredsstiller ulighedssystemet, eller i et endeligt antal skrit afgøre, om løsningsmængden er tom. Først beregnes en elipsoide, som er stor nok til altid at indeholde en løsning til ulighedssystemet. Herefter beregnes iterativt en række elipsoider, som er mindre og mindre, men som altid indeholder en løsning. Hver iteration starter med at beregne, om den sidst beregnede elipsoides centrum er en løsning til ulighedssystemet. Hvis dette er tilfældet returneres denne løsningsvektor. Ellers beregnes en ny mindre elipsoide. I det følgende vil E.M. blive beskrevet begrebsligt og illustrativt.

E.M. kan altså, givet et sæt af lineære uligheder:

$$A^t x \leq b \qquad \text{,hvor}$$

A er en n×m matrix
x er en n-søjle vektor
b er en m-søjle vektor

bestemme, om der findes løsning(er) til systemet, og beregne en (af dem). Ulighedssystemet kan omskrives til

$$a_i^t x \leq b_i \qquad (i=1,.....,m)$$

a er en kolonne i A(en n-søjle vektor), og det forudsættes i det følgende, at $n>1$.

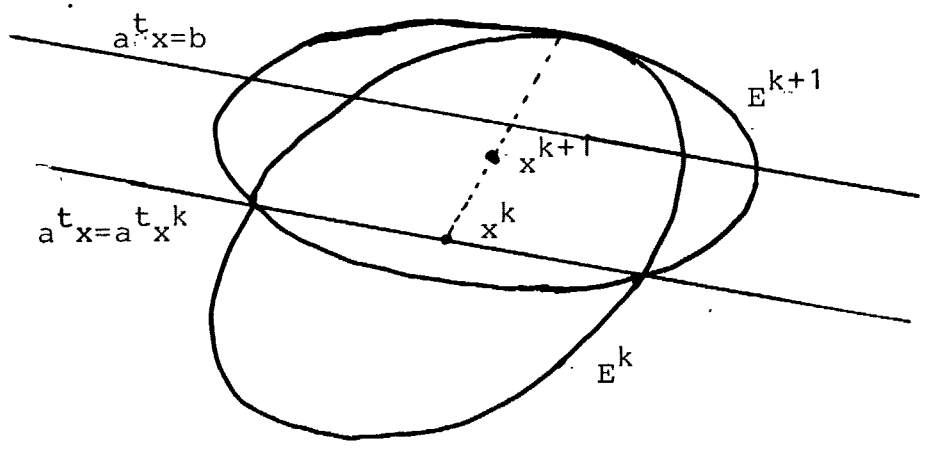
E.M. konstruerer en serie af elipsoider $E^0, E^1, ..., E^k, ...$ svarende til det samme antal iterationer. Iteration nr. k+1 undersøger først, om E^k's center x^k er en løsning til ulighedssystemet. Hvis dette ikke er tilfældet, betyder det, at mindst et af båndene er overtrådt. Et tilfældigt bånd kan skrives, som

$$a^t x \leq b$$

Dette bånd er altså overtrådt af x^k, og der konstrueres en ny elipsoide E^{k+1}, som skal indeholde følgende mængde:

$$\{x \in E^k \mid a^t x = a^t x^k\}$$

En iteration kan illustreres, som følger:



For at beregne en ny elipsoide E^{k+1} udfra E^kskal der opskrives nogle formler. En elipsoide kan repræsenteres, som følger:

$$E^k=\{x\in\mathbb{R}^n \mid (x-x^k)^t(B^k)^{-1}(x-x^k)\leq 1\} \quad \text{,hvor}$$

x^k er elipsoidens center
B^ker en positiv definit symmetrisk matrix.

Først beregnes centret for den nye elipsoide:

$$x^{k+1}=x^k-\tau(B^k a/\sqrt{a^t B^k a})$$

Herefter beregnes den nye B-matrix:

$$B^{k+1}=\delta(B^k-\sigma(B^k a(B^k a)^t/(a^t B^k a)))$$

De græske bogstaver betegner følgende konstanter:

$\tau=1/(n+1)$
$\sigma=2/(n+1)$
$\delta=n^2/(n^2-1)$

Ved at beregne en række af elipsoider efter de ovennævnte to formler kan man i et endeligt antal skridt bestemme, om et system af lineære uligheder har løsning eller ej.

Geometrisk kan E.M. illustreres ved et lille to-dimmensionalt eksempel(d.v.s. n=2). Uligheddерne, der skal behandles ser ud, som følger:

$$-x_1+0x_2\leq -1$$
$$0x_2- x_2\leq -1$$

og i matrix-notation:

$$\begin{pmatrix}-1 & 0\\ 0 & -1\end{pmatrix}\begin{pmatrix}x_1\\ x_2\end{pmatrix}\leq\begin{pmatrix}-1\\ -1\end{pmatrix}$$

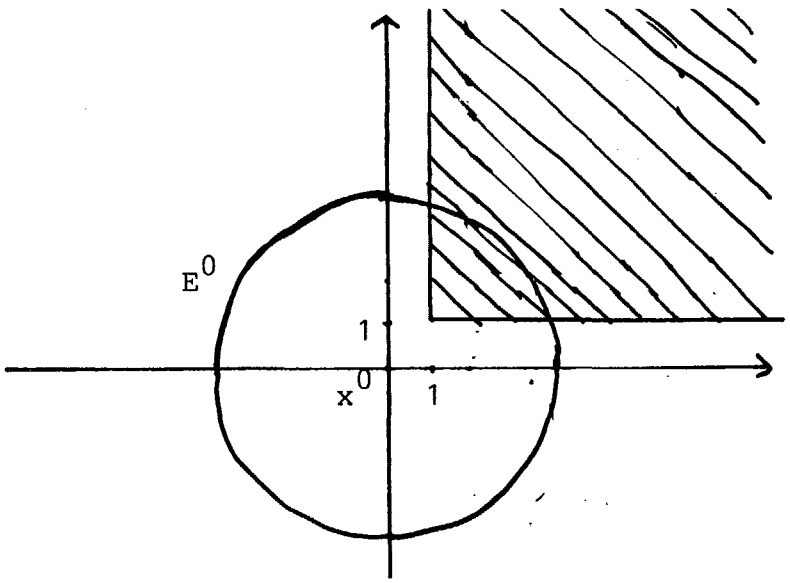
Givet en foregående elipsoide E^k konstrueres den næste således:

(I) Tegn en korde gennem x^k paralelt med det overtrådte bånd. Denne korde skærer elipsoiden E^k i to punkter p_1 og p_2. Den ene del af elipsoiden indeholder nu kun punkter, der tilfredsstiller det overtrådte bånd og kaldes løsningssiden.

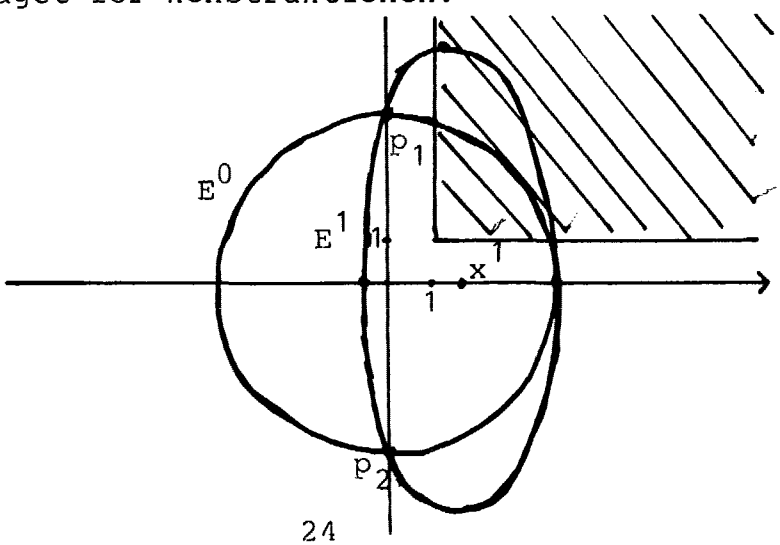
(II) Den næste elipsoide E^{k+1} skal indeholde hele løsningssiden af E^k, d.v.s. den skal gå igennem p_1 og p_2 og et tredie punkt.

(III) E^{k+1} er den elipsoide med det mindste areal, der opfylder de ovenstående betingelser.

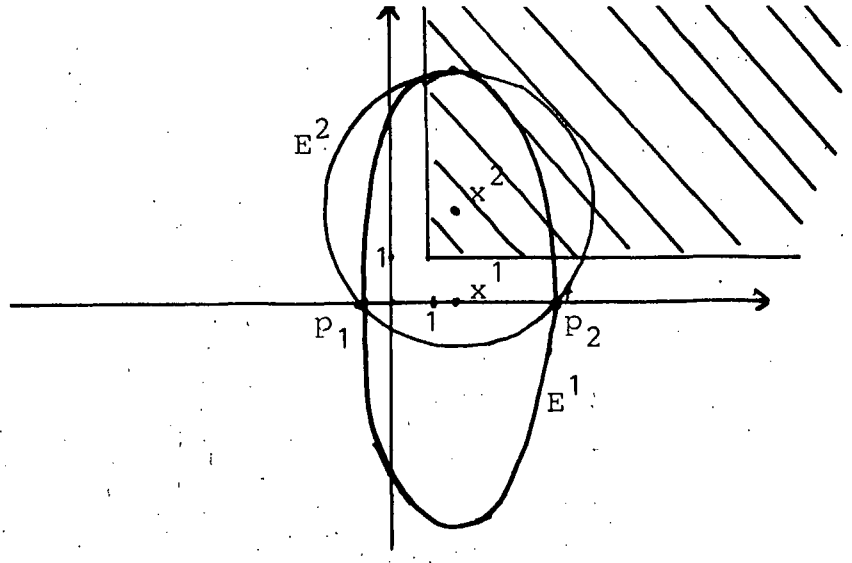
E^0 er en elipsoide, der garanteret indeholder punkter fra løsningsmængden. Her konstrueres E^0 som en cirkel med radius 4, og løsningsområdet markeres med skravering:



Da x_0 ikke ligger i løsningsområdet konstrueres en ny elipsoide E^1 efter de før beskrevne regler, idet det er det andet bånd, der er grundlaget for konstruktionen:



x_1 ligger heller ikke i løsningsområdet, så der konstrueres endnu en elipsoide, hvor første bånd er grundlag for konstruktionen:



x_2 er et punkt i løsningsområdet, så det er nu vist, at uligheds-systemet er løsbart, og et punkt i løsnignsområdet er konstrueret.

5.3. L.P.-problemer kan løses v.h.a. E.M..

Et L.P.-problem på standardform og input til E.M. er ikke det samme. E.M. behandler kun et sæt af uligheder og finder en tilfældig vektor i løsningsområdet. For at E.M. skal kunne løse et L.P.-problem må problemet omskrives og udvides. Det er tidligere vist, at et L.P.-problem på formen:

$$c^t x = \max$$
$$A^t x \leq b$$
$$x \geq 0$$

har et tilhørende dualt problem på formen:

$$b^t y = \min$$
$$Ay \geq c$$
$$y \geq 0$$

Hvis dette samles til et stort ulighedssystem, kommer det til at se ud, som følger:

$$
\begin{aligned}
A^t x &\leq y \\
-x &\leq 0 \\
-Ay &\leq -c \\
-y &\leq 0 \\
-c^t x + b^t y &\leq 0 \quad \text{(Idet det modsatte altid er opfyldt)}
\end{aligned}
$$

Hvis dette system har løsning, er det også den(de) optimal(e). Derfor kan E.M. altså også bruges til løsning af L.P.-problemer. Der er dog store numeriske problemer forbundet med numerisk at løse et L.P.-problem v.h.a. E.M.(det vil blive diskuteret i implementeringsafsnittet), men teoretisk kan E.M. opskrives til en algoritme, der kan løse L.P.-problemer.

5.4. Indledning til beviset for E.M.

Givet et sæt af ulighedder:

$$(a^i)^t x \; b_i \qquad (i=1,.....,m,\; a \in Z^n,\; b_i \in Z)$$

skal det vises, at E.M. kan opskrives som en polynomial-tids algoritme. Indkodningslængden betegnes L og er størelsen:

$$L=\sum_{i,j} \log(|a_{ij}|+1)+\sum_{j} \log(|b_i|+1) \log\, nm+1$$

Det største antal iterationer algoritmen kan foretage er $6n^2L$, og komplexiteten er dermed polynomiel i n.

Beregningerne i algoritmen er baseret på, at en elipsoide E er repræsenteret, som beskrevet i forrige afsnit:

$$E=\{x \mid (x-x_0)^t A^{-1} (x-x_0)=1\}$$

Endvidere antages det, at $|x^k| \leq 2^{L+k}$, $||A^k|| \leq 2^{L+k}$ og $||(A^k)^{-1}|| \leq 2^{-2L+2k}$, hvor k betegner den iteration, hvor elipsoiden (x^k, A^k) er beregnet.

Når $a \in R^n$ og $a \neq 0$ beregnes den næste elipsoide $E^a(x^{0'}, A')$ ved følgende:

$$x^{0'} = x^0 - \frac{1}{n^2-1} \cdot A \cdot \frac{a}{\sqrt{a^t A a}}$$

$$A' = \frac{n^2}{n^2-1}\left(A - \frac{2}{n+1}\cdot\frac{(Aa)(Aa)^t}{a^tAa}\right)$$

Ideen i det formelle bevis, for at algoritmen stopper inden det $6n^2L$ skridt, er meget simpel. Det antages, at algoritmen har gennemløbet $6n^2L$ iterationer uden at finde en løsning endnu. Det vises så, at den sidst beregnede elipsoides volumen er mindre end det mindste løsningsområde.

5.5. Bevis for E.M.

I dette afsnit skal der gennemføres et bevis for, at E.M. er polynomiel. Beviset bygger på Gacs og Lovacz(Gacs 1981) og kræver brug af en del hjælpesætninger. Hjælpesætningerne vil kun blive nævnt her, mens beviserne kan læses i bilag I.

Påstanden, som skal vises, er:

PÅSTAND: Hvis E.M. stopper, er den fremkomne x-vektor en løsning til problemet $Ax<b$. Hvis E.M. ikke er stoppet efter $6n^2L$ skridt, så er der ingen løsning til problemet.

At bevise påstandens første del er let nok. Det følger af metoden selv. For at bevise påstandens anden del, bruges følgende hjælpesætninger:

SÆTNING 1: Ethvert ekstrempunkt i polyederet defineret ved:

$$a^i x \leq b \qquad (i=1,.....,m)$$
$$x \geq 0$$

opfylder $|v| < 2^L$, og dets koordinater er rationale tal, hvis nævnerer højst antager værdien 2^L.

SÆTNING 2: Hvis der er en løsning til problemet, så gælder det om løsningsrummet, som ligger indenfor en kube defineret af $|X_i| \leq 2^L$, at dets volumen er mindst $2^{-(n+1)L}$.

SÆTNING 3: Antag at der eksisterer en løsning til systemet:

$$(a^i)^t x < b_i + 2^L \qquad (i=1,.....,m)$$

så eksisterer der også en løsning for systemet

$$(a^i)^t x \leq b_i \qquad (i=1,.....,m)$$

SÆTNING 4: Hvis elipsoiden givet ved centret x^k og matricen B^k kaldes for E^k og halvelipsoiden $E^k \cap (x:(x-x^0)^t a=0)$ kaldes for $\frac{1}{2}E^k$, så gælder det, at

$$\tfrac{1}{2}E^k \subset E^{k+1}.$$

SÆTNING 5: Betegn volumnet af E^k med $V(E^k)$. Det gælder da, at:

$$V(E^{k+1}) = c(n) \cdot V(E^k) \qquad \text{,hvor}$$

$$c(n) = \left(\frac{n^2}{n^2-1}\right)^{(n-1)/2} \frac{n}{n+1} < e^{-\left(\frac{1}{2(n+1)}\right)}$$

Påstandens anden del kan nu vises. Antag at systemet har en løsning, men den er ikke fundet efter $k=6n^2L$ skridt. Ifølge sætning 2 gælder det for løsningsrummet P indeholdt i E^0, at $V(P)=2^{-(n+1)L}$. Løsningsrummet er ifølge sætning 4 indeholdt i E^k, men ifølge sætning 5 er

$$V(E^k) < e^{-(k/2(n+1))} V(E^0) < e^{-(k/2(n+1))} 2^{2L(n+1)} < 2^{-nL}.$$

D.v.s. $V(P) > V(E^k)$.

Hvis man ønsker at afgøre, om uligheddene

$$(a^i)^t x \leq b \qquad (i=1,.....,n)$$

har en løsning, kan man i stedet betragte

$$2^{L_{a_i}} x < 2^{L_{b_i}+1} \qquad (i=1,.....,n)$$

Dette ulighedssystem har i følge sætning 3 en løsning, netop når det forrige system har en løsning.

6. Implementering.

6.1. Indledning.

I dette kapitel beskrives en inplementering af E.M. på en datamaskine. Implementeringen foregår i programmeringssproget SIMULA, som er det mest anvendte programmeringssprog på RUC. SIMULA har bl.a. den fordel, at det er velegnet til nye datastrukturer, f.eks. regning med multipel præcision, hvilket evt. kan bruges til at løse numeriske problemer.

Det store problem ved en implementation af Khachiyan's algoritme er, at den kræver uendelig præcision. Khachiyan kommer ud over dette ved at begrænse nøjagtigheden til 23L bit før kommaet og 38nL bit efter kommaet. Tilgengæld runder han de fremkomne resultater op. I de kilder,vi har brugt, er der ikke givet nogen endelig løsning. Men selv de størrelser, som Khachiyan og andre arbejder med, er så store, at det ikke umiddelbart er muligt for os, at regne med dem. Da vores implementation primært er beregnet som et eksempel, har vi valgt at se bort fra de store nøjagtighedskrav og kun bruge SIMULA's dobbelt præcision(datatypen LONG REAL).

Ved implementation af E.M skal man vælge en repræsentation af elipsoiden E^k. Der er flere mulighedder, f.eks. kan man repræsentere elipsoiden ved dens center x^k og en ikke-singulær matrix J^k, som transformerer enhedskuglen om til E^k med center i origo. Vi har valgt at bruge samme metode som bl.a. (Bland 1981) og (Gacs 1981). Elipsoiden E^k repræsenteres som dens center x^k og en positiv definit matrix B^k, således at

$$E^k=\{ x\in R^n \mid (x-x^k)^t (B^k)^{-1}(x-x^k)=1\}$$

Denne repræsentation giver nogle meget simple opdateringsformler for x^{k+1} og B^{k+1}.

Som baggrund for implementeringen er valgt den version af Khachyan's algoritme, som opstilles af (Aspvall 1979). Denne algoritme er yderligere forbedret v.h.a. "dybe" snit i valget af a^i, som beskrevet i (Bland 1981). Der bliver ikke ført bevis for de dybe snits gyldighed, men metoden og de omskrivnin-

ger, som den medfører, forklares.

I det følgende forudsættes det, at alle vektorene optræder som søjlevektorer.

6.2. Beskrivelse af algoritmen.

6.2.1. Grundalgoritmen.

Algoritmen afgør, om et sæt af lineære ulighedder

$$Ax \leq b \ , \qquad A \in Z^{m \times n} \quad \text{og} \quad B \in Z^{m}$$

kan løses og finder i så fald en mulig løsning. Problemet, der skal undersøges, er givet med heltallige koeficienter på formen:

$$a_{i1}x_1 + \ldots\ldots\ldots a_{in}x_n \leq b_i \qquad i=1,\ldots\ldots,m;\ m \geq 2;\ n \geq 2$$

En anden form er:

$$a^{it}x \leq b_i \qquad a^{i} \in Z^{n} \quad i=1,..,m;\ m \geq 2;\ n \geq 2$$

I det værste tilfælde kræver algoritmen $o(n^3(m+n)L)$ antal operationer, hvor o er et positivt heltal, og L er længden af den binære indkodning af problemet. L findes/beregnes, som:

$$L=\sum_{i,j} \lceil \log_2(|a_{ij}|+1) \rceil + \sum_{i} \lceil \log_2 |b_i| + 1 \rceil + \lceil \log_2 mn \rceil + 1$$

Algoritmen består af tre skridt:

I) initialisering
II) Afgør, om algoritmen stopper
III) Find et nyt center og en ny elipsoide

1.SKRIDT: Initialeser x-vektoren som 0-vektor, B-matricen som en n×n diagonal matrix med diagonalværdierne 2^{2L} og sæt k=0.
2.SKRIDT: Hvis x^k er en løsning til $Ax \leq b$ så stop, og returner x^k. Hvis $k < 4(n+1)^2L$, så tæl k op og gå videre med 3. skridt, ellers så stop og svar, at der ingen løsning er.

3. skridt: Vælg en af de uligheder, som ikke tilfredsstilles af x^k, f.eks. $a^{it}x \geq b_i$ og brug a^i-vektoren til at finde x^{k+1} og B^{k+1}. Det nye center x^{k+1} beregnes:

$$x^{k+1}=x^k - \tau \frac{B^k a_i}{\sqrt{a_i^t B^k a_i}}$$

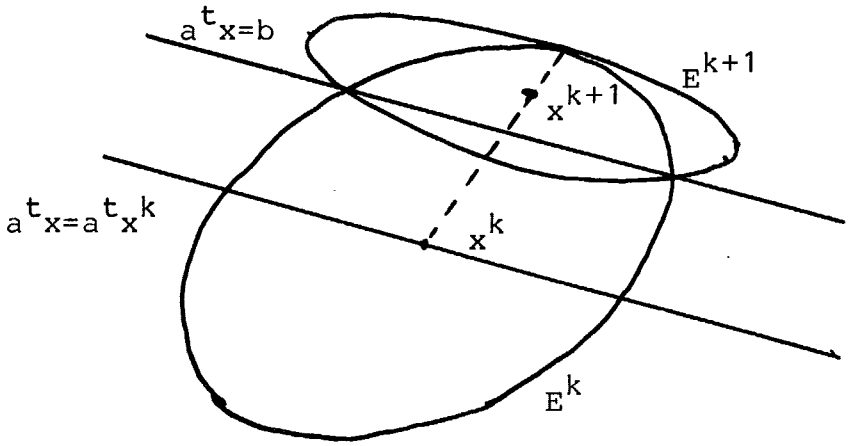
Den nye elipsoide B^{k+1} beregnes:

$$B^{k+1}=\delta\left(B^k - \sigma \frac{(B^k a_i)(B^k a_i)^t}{a_i^t B^k a_i}\right)$$

Gå derefter til 2.skridt.

6.2.2. De "dybe" snit.

E.M. bygger på, at man i det k+1'ste skridt danner en elipsoide E^{k+1}, som indeholder den halvelipsoide $\frac{1}{2}E^k$, hvori det overtrådte bånd kan opfyldes, d.v.s. $\frac{1}{2}E^k=\{x \in E^k \mid a^t x \leq a^t x^k\}$. Da det faktisk kun er den delmængde af E^k, hvor båndet bliver opfyldt, der er interessant,kan man nøjes med at indkludere denne delmængde, d.v.s. $\{x \in E^k \mid a^t x \leq b\}$. Dette kan illustreres, som:



Der skal nu findes det dybest mulige snit blandt båndene, d.v.s. det snit, hvor der skæres mest af E^k væk. For hvert bånd beregnes afstanden fra centret x^k til halvplanet $H=\{x \in R^n \mid a_i^t x \leq b_i\}$ i den metrik, som er givet af matricen B^k. Den største værdi vælges, som

$$\alpha = \max\{\alpha_i \in R \mid \alpha_i=(a_i^t x^k - b_i)/\sqrt{a_i^t B^k a_i}\ ,\ i=1,\ldots,m\ .\}$$

Det bånd, som giver den største værdi af α, bliver nu brugt til at danne den nye elipsoide med.

Det nye center x^{k+1} og den nye B^{k+1} benyttes på sædvanlig måde, men parametrene τ, σ og δ sættes nu til:

$$\tau = \frac{1+n\alpha}{n+1}$$

$$\sigma = \frac{2(1+n\alpha)}{(n+1)(1+\alpha)}$$

$$\delta = \frac{n^2(1-\alpha^2)}{n^2-1}$$

Metoden kan ikke "bruges" for alle værdier af α. α er givet i B-matricens metrik, d.v.s. $\alpha=1$, når x^{k+1} ligger på randen af E^k og $\alpha=0$, når $H=\frac{1}{2}E^k$. Hvis $\alpha>1$, betyder det, at H ikke skærer E^k d.v.s. at mindst et af båndene ikke kan opfyldes, og der er derfor ingen løsning. Hvis $\alpha \leq 0$ betyder det, at x^k er en løsning til systemet, da alle båndene er opfyldt.

6.3. Den endelige algoritme.

På baggrund af de dybe snit kan den færdige algoritme nu opstilles. Da $\sqrt{a^t B^k a}$ indgår som led i flere af beregningerne, kaldes denne størrelse M og beregnes for sig.

1.SKRIDT: Indlæs A og b.

beregn:

$$L=\sum_{i,j}\lceil \log_2(|a_{ij}|+1)\rceil + \sum_i \lceil \log_2|b_i|+1\rceil + \lceil \log_2 mn\rceil + 1$$

sæt $x^0=0$

$B^0=2^{2L}\cdot I^n$

$k=0$

$maxit=4(n+1)^2L$

2.SKRIDT: beregn:

$M_i=\sqrt{a^{it}B^k a^i}$ $\quad i=1,\ldots,m$

$\alpha=\max\{\alpha_i=(a^{it}x^k-m_i/m_i\}$, $\quad i=1,\ldots,m$

Hvis $\alpha \leq 0$ så stop og giv x^k som svar.

Hvis $\alpha \leq 1$ og $k <$ maxit, så tæl k op og gå til 3. skridt, ellers stop og skriv at der ingen løsninger er.

3.SKRIDT: beregn:

$$x^{k+1} = x^k - \frac{1+na}{n+1} \cdot \frac{B^k a^i}{m_i}$$

$$B^{k+1} = \frac{n^2(1-\alpha^2)}{n^2-1}\left(B^k - \frac{2(1+na)}{(n+1)(1+\alpha)} \cdot \frac{B^k a^i}{m_i} \cdot \frac{(B^k a^i)^t}{m_i} \right)$$

6.4. Gennemgang af programmet.

I dette afsnit beskrives et SIMULA-program, som v.h.a. elipsoidemetoden afgør, om et system af uligheder har en løsning. Det forudsættes, at læseren har et vist kendskab til programmering.

Selve algoritmen udføres af en procedure LI, som kaldes i programmet. LI's skelet ser således ud:

```
procedure LI(N,M);
integer N,M;
begin
  --erklæring af tabeller og globale variable--
  procedure INDLÆS;.....;
  procedure INITALLISER;.....;
  procedure BEREGNM;.....;
  procedure FINDALFA;.....;
  procedure STOP;.....;
  procedure NYTCENTER;.....;
  procedure NYELIPSOIDE;.....;
  procedure UDSKRIV;.....;
  INDLÆES;
  INITIALISER;
  BEREGNM;
  FINDALFA;
  while not STOP do
  begin
```

```
        NYTCENTER;
        NYELIPSOIDE;
        BEREGNM;
        FINDALFA;
      end;
      UDSKRIV;
    end%%%LI%%%;
```

LI forudsætter, at inddata består af matricen A række for række og vektoren b. Programmet kalder LI, og n og m gives som inddata før A og b. Det ser ud, som:

```
begin
  procedure LI(N,M);
  integer N,M;.....;
  LI(inint,inint);
end;
```

Når LI har fundet en mulig løsning eller er nået frem til, at der ingen er, udskrives følgende: L, det maksimale antal iterationer, A-matricen, b-vektoren, antallet af iterationer samt enten en løsningsvektor eller beskeden "Der er ingen løsning".

Hvis man ønsker at løse et L.P.-problem, må man opstille det duale problem, som tidligere beskrevet, og bruge det som inddata.

Procedure LI og dens procedurer vil nu blive beskrevet nærmere. Selve program-teksten findes i bilag II.

I procedure LI er der erklæret en del et og to dimmensionelle tabeller samt nogle alm. variable. De bruges af procedurene i LI. Erklæringerne ser ud som:

```
integer array A(1:M,1:N);
long real array E(1:N,1:N),X(1:N),B,MALFA(1:N);
integer L,MAXIT,K,ALFABAAND;
long real ALFAMAX,MI;
```

Tallene N og M er overført som parametre til procedurekaldet. De betegner hhv. antallet af ubekendte og antallet af bånd. LI starter med at kalde proceduren INDLAES, som indlæser matricen A række for række og b-vektoren. Derefter kaldes proceduren INITIALISER. Her beregnes L som

$$L=\sum_{i,j}\lceil \log_{10}(|A_{ij}|+1)/\log_{10}(2)\rceil+\sum_{i}\lceil \log_{10}(|b_i|+1)/\log_{10}(2)\rceil$$
$$+\lceil \log_{10}(M\cdot N)/\log_{10}(2)\rceil+1$$

$\lceil x\rceil$-funktionen opnås, fordi L er en heltalsvariabel, og resultatet derfor trunkeres (afskæres). MAXIT, som er det maksimale antal iterationer, findes og B-matricen, som her kaldes E, initialiseres til diag($2^{2L},\ldots,2^{2L}$). Initialiseringen af x-vektoren og k til nul er foretaget ved erklæringen af dem, hvor de automatisk får værdien nul. Hermed er 1. skridt afsluttet.

Algoritmens 2.skridt foretages i programmet af procedurene BEREGNM, FINDALFA og STOP. Proceduren BEREGNM finder m_i for alle i udfra formlen:

$$m_i=\sqrt{\sum_{j=1}^{n}\left(\sum_{h=1}^{n} a_{ih}\cdot e_{hj}\right)\cdot a_{ij}} \qquad (i=1,\ldots,m)$$

d.v.s.:

$$m_i=\sqrt{\sum_{j=1}^{n} a^{it}\, e^{j}\, a_{ij}} \qquad (i=1,\ldots,m)$$

Proceduren FINDALFA beregner α_i på formen:

$$\alpha_i=\left(\left(\sum_{j=1}^{n} a_{ij}\cdot x_j\right)-b_i\right)/m_i \qquad (i=1,\ldots,m)$$

FINDALFA starter med at sætte ALFAMAX til α_1 og ALFABAAND til 1. Nu findes i en løkke α_i, i=2,.....,m, og sammenlignes med ALFAMAX. Hvis $\alpha_i >$ ALFAMAX, så sættes ALFAMAX til α_i og ALFA-BAAND til i. Når løkken er færdig, gælder det , at:

$$ALFAMAX=\alpha_{MALFABAAND} \geq \alpha_i \qquad (i=1,.....,m)$$

Til sidst sættes MI til $M_{ALFABAAND}$.

Proceduren STOP er en boolsk procedure, som kaldes for at se, om algoritmen stopper. STOP kaldes før hver ny iteration. STOP tæller K op, så K angiver nummeret på den nye iteration. Hvis K bliver større end MAXIT, eller hvis ALFAMAX=0 eller ALFAMAX>1, så bliver STOP sand, og algoritmen stopper, ellers bliver STOP falsk. Sålænge STOP er falsk, så beregnes det nye center og den nye E-matrix. Inden STOP kaldes næste gang, findes den nye ALFAMAX med BEREGNM og FINDALFA.

Proceduren NYTCENTER beregner den nye X-vektor, som

$$X_i=X_i-\left(\sum_{J=1}^{N} e_{ij}\ a_{alfabaand,j}\right)\cdot SP/MI \qquad i=(1,.....,m)$$

hvor

$$SP=\frac{1+n\cdot ALFAMAX}{n+1}$$

Når man bruger dybe snit, må konstanterne beregnes i hver iteration.

Procedure NYELIPSOIDE beregner den nye E-matrix, E^{k+1}. Først beregnes en hjælpe-vektor $V=E^k\ A_{ALFABAAND}$. V er en n-vektor og beregnes, som:

$$V_i=\sum_{J=1}^{N} a_{ALFABAAND\ j}\ e_{ij} \qquad i=(1,.....m)$$

Vektoren V bruges derefter til at finde E^{k+1} udfra formen

$$e_{ij}=EP\cdot\left(e_{ij}-DP\cdot\frac{V_i}{m_i}\cdot\frac{V_j}{m_i}\right)$$

hvor

$$EP=\frac{n^2(1-ALFAMAX^2)}{n^2-1}\qquad \text{, og}$$

$$DP=\frac{2(1+n\cdot ALFAMAX)}{(n+1)(1+ALFAMAX)}$$

Når E^{k+1} er beregnet gennemløbes den i en dobbeltløkke, som sikrer at den er symetrisk. Det gøres sådan:

$$\left.\begin{array}{l} |e_{ij}| > |e_{ji}| \\ |e_{ij}| < |e_{ji}| \end{array}\right\} \Rightarrow \begin{array}{l} e_{ji} := e_{ij} \\ e_{ij} := e_{ji} \end{array} \qquad i=(1,.....n-1), j=(i+1,.....,n)$$

Begrundelsen for dette kommer senere.

Når STOP bliver sand, slutter iterationen og procedure UDSKRIV kaldes inden programmet slutter. Hvis der ingen løsning var, udskrives "Der er ingen løsning." i stedet for x-vektoren.

6.5. Kørselsresultater.

Programmet er kørt med både LI-problemer og L.P.-problemr på dual form. De opnåede resultater kan ikke siges, at være særligt tilfredsstellende.

Det simpleste LI-problem var: afgør om systemet givet ved:

$$2x_1+x_2 \leq 4 \wedge x_1+2x_2 \leq 3 \qquad\qquad x_1, x_2 \geq 0$$

har en løsning. Det kan opskrives på matrixformen $Ax \leq b$, $x \geq 0$, hvor

$$A=\begin{pmatrix} 2 & 1 \\ 1 & 2 \\ -1 & 0 \\ 0 & -1 \end{pmatrix} \quad \text{og} \quad b=\begin{pmatrix} 4 \\ 3 \\ 0 \\ 0 \end{pmatrix}$$

Som resultat kom udskriften:

```
L:        16;    MAXIT:       576
INITIALISERING AF TABELLERNE:

A-MATRISSEN:
     2     1
     1     2
    -1     0
     0    -1

B-VEKTOREN:
     4     3     0     0

PROGRAMMET STOPPER EFTER 0 ITERATIONER.

X-VEKTOREN:
      0.0000000&&+000
      0.0000000&&+000
```

Båndene på x blev nu ændret til $x_1 \geq 0.00001$, $x_2 \geq 0$. Resultatet blev:

```
L:      16;   MAXIT:      576
INITIALISERING AF TABELLERNE:

A-MATRISSEN:
     2     1
     1     2
    -1     0
     0    -1

B-VEKTOREN:
     4     3    -0.00001     0

PROGRAMMET STOPPER EFTER 22 ITERATIONER.

X-VEKTOREN:
     5.91383238&-002
     2.74223758&-001
```

Det andet LI=problem var at afgøre, om systemet givet på formen:

$$\begin{pmatrix} 1 & 1 & 2 \\ 0 & 2 & 1 \\ -1 & -3 & -2 \\ 2 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \leq \begin{pmatrix} 55 \\ 50 \\ -95 \\ 50 \end{pmatrix}$$

har en løsning. Dette problem gav følgende resultat:

```
L:      45;   MAXIT:     2880
INITIALISERING AF TABELLERNE:

A-MATRISSEN:
     1     1     2
     0     2     1
    -1    -3    -2
     2     1     0

B-VEKTOREN:
    55    50   -95    50

PROGRAMMET STOPPER EFTER 294 ITERATIONER.
DER ER INGEN LOESNING.
```

Det er ikke korrekt. Løsningen er et enkelt punkt.(15,20,10). Den ophobede fejl bliver for stor til, at punktet kan bestemmes. Hvis man derimod udvider løsningsområdet ved at sætte det tredie bånd til $x_1+3x_2+2x_3 \geq 90$, giver det følgende resultat:

```
B-VEKTOREN:
    55    50   -90    50

PROGRAMMET STOPPER EFTER 136 ITERATIONER.

X-VEKTOREN:
     1.48890008&+001
     1.97903658&+001
     8.94095828&+000
```

L.P.-problemerne er forsøgt løst v.h.a. det duale problem. Et eks. er: find maksimum af $c^t x$ givet, som:

$$(2\ \ 3)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

under båndene $Ax \leq b$, givet som:

$$\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 4 \\ 3 \end{pmatrix} \qquad \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq 0$$

Til dette svarer det duale problem: find minimum af $b^t y$, givet som:

$$(4\ \ 3) \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$$

under båndene $-A^t x \leq -c$, givet som:

$$-\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \leq -\begin{pmatrix} 2 \\ 3 \end{pmatrix} \qquad \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \geq 0$$

Maksimum og minimum er fundet, når $b^t y - c^t x = 0$, hvilket også kan skrives, som:

$$(4\ \ 3)\ \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} - (2\ \ 3)\ \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq 0$$

$$\left(\lambda\ (2\ \ 3)\ \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - (4\ \ 3)\ \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq 0\right)$$

Der bruges kun den første ulighed, da den anden altid er sand. Det LI-problem, som opstilles, ser ud som:

$$\begin{pmatrix} 2 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -2 & -1 \\ 0 & 0 & -1 & -2 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ -2 & -3 & 4 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ y_1 \\ y_2 \end{pmatrix} \leq \begin{pmatrix} 4 \\ 3 \\ 0 \\ 0 \\ -2 \\ -3 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Problemet ved denne opstilling er det sidste bånd: $b^t y - c^t x \leq 0$. Det tilfredsstilles kun ved $b^t y = c^t x$. For at tage højde for usikkerheden på beregningerne er det nødvendigt at udvidde løsningsområdet ved at indføje et $\varepsilon > 0$. Nu kan båndet udtrykkes som $b^t y - c^t x \leq \varepsilon$. Med et epsilon på 10^{-5} giver kørslen følgende resultat:

```
L:      46;    MAXIT:     4600
INITIALISERING AF TABELLERNE:
A-MATRISSEN:
     2     1     0     0
     1     2     0     0
    -1     0     0     0
     0    -1     0     0
     0     0    -2    -1
     0     0    -1    -2
     0     0    -1     0
     0     0     0    -1
    -2    -3     4     3

B-VEKTOREN:
     4     3     0     0    -2    -3     0     0    0.00001   0.00001
PROGRAMMET STOPPER EFTER 376 ITERATIONER.
X-VEKTOREN:
      1.0666601&&+000
      6.0666882&&-001
      3.3333227&&-001
      1.3333355&&+000
```

Den fremkomne x-vektor skal være(5/3,2/3,1/3,4/3).

Problemet maximer $3x_1+3x_2+4x_3$ under båndene:

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 1 \\ 1 & 3 & 2 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \leq \begin{pmatrix} 16 \\ 13 \\ 17 \end{pmatrix} \qquad x \geq 0$$

giver med $\varepsilon = 10^{-5}$ følgende resultat:

```
L:    77:   MAXI:   15092
INITIALISERING AF TABELLERNE:

A-MATRISSEN:
     1     2     3     0     0     0
     2     1     1     0     0     0
     1     3     2     0     0     0
    -1     0     0     0     0     0
     0    -1     0     0     0     0
     0     0    -1     0     0     0
     0     0     0    -1    -2    -1
     0     0     0    -2    -1    -3
     0     0     0    -3    -1    -2
     0     0     0    -1     0     0
     0     0     0     0    -1     0
     0     0     0     0     0    -1
    -3    -3    -4    16    13    17

B-VEKTOREN:
    16    13    17     0     0     0    -3    -3    -4     0     0     0    0.00001
PROGRAMMET STOPPER EFTER 1153 ITERATIONER.

X-VEKTOREN:
      4.4416274&&+000
      7.9186199&&-001
      3.3248824&&+000
      1.0000002&&+000
      9.9999996&&-001
      2.6695192&&-007
```

Maximum beregnes til 28.999.., det er 29.

De hidtil omtalte kørsler er alle gået godt. Men så snart der skal løses større systemer, går det galt. Et eksempel herpå er: Find maximum af $4x_1+3x_2+4x_3+5x_4$ under båndene:

$$\begin{pmatrix} 3 & 2 & 2 & 3 \\ 1 & 1 & 2 & 2 \\ 2 & 2 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \leq \begin{pmatrix} 59 \\ 37 \\ 32 \\ 18 \end{pmatrix} \qquad x \geq 0$$

Selv med epsilon så stor som 0.1, opstår der problemer, som gør, at programmet går i fejl. Opstillingen af matricen ser således ud:

```
L:      120;    MAXIT:    38880
INITIALISERING AF TABELLERNE:

A-MATRISSEN:
     3     2     2     3     0     0     0     0
     1     1     2     2     0     0     0     0
     2     2     1     1     0     0     0     0
     0     1     1     1     0     0     0     0
    -1     0     0     0     0     0     0     0
     0    -1     0     0     0     0     0     0
     0     0    -1     0     0     0     0     0
     0     0     0    -1     0     0     0     0
     0     0     0     0    -3    -1    -2     0
     0     0     0     0    -2    -1    -2    -1
     0     0     0     0    -2    -2    -1    -1
     0     0     0     0    -3    -2    -1    -1
     0     0     0     0    -1     0     0     0
     0     0     0     0     0    -1     0     0
     0     0     0     0     0     0    -1     0
     0     0     0     0     0     0     0    -1
    -4    -3    -4    -5    59    37    32    18

B-VEKTOREN:
    59    37    32    18     0     0     0     0    -4    -3    -4    -5     0     0     0     0     0.1
```

Og fejlmeddelelsen således:

```
INVALID PARAMETER TO STANDARD FUNCTION
IN PROCEDURE BEREGNM DECLARED IN PROCEDURE LI AND ENTERED AT LINE 161
IN PROCEDURE LI DECLARED IN BLOCK 1 AND ENTERED AT LINE 166
IN BLOCK 1
```

I næste afsnit vil problemerne ved programmet blive gennemgået.

6.6. Numeriske overvejelser.

Brugen af Khachiyan's metode til løsning af L.I. giver nummeriske problemer. Khachiyan forudsætter en nøjagtighed på 23L bit før kommaet og 38nL bit efter kommaet. Ved et simpelt 4x2 system kan det give et behov for ca. 600 bit. I vores implementering af algoritmen har vi ikke taget højde for disse krav på anden måde end, at vi bruger datatypen LONG REAL, som giver 60 bits nøjagtighed. Som man kan forvente, giver den manglende nøjagtighed problemer. Problemerne er specielt knyttet til beregningen af den nye E-matrix (B-matrix) i proceduren NYELIPSOIDE og matricens anvendelse til beregning af m_i i proceduren BEREGNM. P.g.a. den manglende nøjagtighed falder præcisionen af beregningen af E-matricen. Det gør, at den efter et vist antal iterationer, ophører med at være symmetrisk. For at sikre at den fremkomne E-matrix bliver symmetrisk gennemløbes alle talparrene (e_{ij},e_{ji}), i=1,..,n-1 j=i+1,..,n. For hvert talpar sættes det tal, hvis absolutte værdi er mindst, lig med det tal hvis absolutte værdi er størst. På denne måde søger vi også at sikre os, at den fremkomne elipsoide indeholder hele den rigtige elipsoide.

Selv med denne nødløsning bliver præcisionen ikke god nok. Ved bare lidt større systemer, giver de ophobede fejl på beregningen af E-matricen problemer. Det viser sig, når det ved beregningen af

$$\sum_{j=1}^{n} (a^i)^t e^j a_{ij} \quad i=1,..,m$$

i procedure BEREGNM fremkommer negative tal, som der så skal uddrages kvadratrødder af. De negative tal opstår formodentligt, fordi der under summationen adderes tal med modsat fortegn, hvis absolutte værdier er næsten ens. Ved sådanne additioner sker der et stort tab af præcision, og små fejl på de mindstbetydende ciffre kan få stor betydning.

For at forebygge dette har vi prøvet at fortage særskilt optælling af positive og negative tal, som så blev adderet til sidst. Vi har også prøvet at runde de beregnede tal op, således at x'>x, før de blev summeret. Ingen af metoderne synes dog at have hjulpet nævneværdigt, og de er udeladt af programteksten.

Bland(1981) nævner som et af problemerne ved den valgte repræsentation af elipsoiden, at beregningen er numerisk ustabil. Derfor er det ikke underligt, at der er opstået alle disse problemer. En forslået løsning er, at faktorisere B-matricen. Der siges dog også

at være store numeriske problemer forbundet med denne metode, så vi er ikke gået dybere i de betragtninger.

En anden mulighed er at eksperimentere med alternative datastrukturer, så som multipel præcisions-aritmetik, interval aritmetik eller brøker.

Selv om man får en implementation der fungere, vil løsningen af L.P.-problemer v.h.a. det duale problem, give meget store kørsler. Ved løsning af det duale problem vokser dimensionen af problemt til n+m, hvilket medfører at antallet af iterationer stiger. En måde at reducere dimensionen på er at opdele løsningen af L.P.-problemet i tre faser.

Først undersøges, om $Ax \leq b$ overhovedet har en løsning x^k. Hvis der eksisterer et løsningsområde kan det være ubegrænset opad. Derfor undersøges om det duale $A^t y \geq c$ har en løsning. I så fald eksisterer $\max(c^t x)$. Til slut kan man så med udgangspunkt i x^k benytte E.M. - i retningen c eller $B^k c$ - suppleret med en bisektionsalgoritme. Førtst nu løses hele det duale-problem med alle båndene.

Denne metode har den fordel, at den kan udføres med den oprindelige dimension.

7. Elipsoide metodens betydning.

7.1. Indledning.

I de foregående afsnit er Simplex og implementeringen af samme blevet beskrevet. Det er blevet vist at Simplex er en eksponential-tids algoritme og E.M. er en polynomial-tids algoritme. I dette afsnit vil nogle af de aspekter, E.M. rejser, blive diskuteret i forhold til NP-teorien. Afsnit 7.3. omhandler den praktiske betydning E.M. vil få som (implementeringsgrundlag) for programmer, der løser L.P.-problemer. Afsnit 7.4. er en opsamlende konklusion.

7.2. E.M.'s teoretiske betydning.

Et af problemerne i NP-teorien har været placeringen af L.P.. Man har ment, at L.P. tilhørte klassen P, men der har ikke været nogle polynomielle algoritmer til løsning af L.P.-problemer. Bl.a. disse overvejelser har ført til definitionen af klasser af problemer, f.eks. co-NP, som vi ikke har behandlet i vores afsnit om NP-teorien. Udvidelsen af den grundlæggende NP-teori kan læses i f.eks. (Papadimitrou 1982).

Det vigtigste ubesvarede spørgsmål i NP-teorien er, om P=NP. D.v.s. om det kan vises, at problemer som DRS har en polynomial-tids løsning. Fremkomsten af E.M. har på flere måder betydning for dette spørgsmål.

Man har forsøgt at vise, at L.P.-problemet tilhørte klassen af NP-komplette problemer. Hvis det lykkes, har man med fremkomsten af E.M. vist , at P=NP.

L.P.-problemets placering har som nævnt været et uafklaret spørgsmål. E.M.'s fremkomst placerer entydigt L.P. i klassen P. Med denne nye viden må man nu underkaste udvidelserne af NP-teorien en grundig analyse for at se, om de stadig er gyldige. Specielt om der stadig eksisterer en mellemzone af problemer i NP.

7.3. E.M.'s praktiske betydning.

De kilder, som vi har benyttet i denne rapport, har det til

fælles, at de alle stiller sig tvivlende overfor den praktiske betydning af E.M., sådan som den kan implementeres i dag. E.M. stiller krav om meget stor nøjagtighed, hvilket også kan kræve meget stort lager. Antallet af iterationer, som E.M. bruger, vokser ganske vist polynomielt, men tallet er meget stort. Vores kørselsresultater tyder dog på, at tallet er mindre end det maksimale, som algoritmen tillader. De enkelte iterations skridt kan også være ret tidskrævende, hvis der skal tages højde for f.eks. manglende præcision.

Men det, at der er fremkommet en polynomial-tids algoritme til løsning af L.P.-problemer, vil formodentligt sætte gang i udviklingen af forbedrede metoder. Det har i følge vores kilder vist sig, at når først der er kommet en polynomial-tids algoritme til løsning af et problem, vil den blive effektiviseret betydeligt af andre forskere.

Et spørgsmål, som fremkomsten af E.M. rejser,er, om definitionen af polynomial-tids algoritmer er rimelig. Det viser sig jo her i praksis, at det tal L, som indgår i p(n), er et meget stort tal, om også vokser voldsomt med størrelsen af n. Så er der rimeligt overhovedet at tale om, at man nu har en polynomial-tids algoritme til løsning af L.P.-problemer.

7.4. Konklusion.

Selvom det er svært at måle, hvor stor omtale i medierne fremkomsten af en ny algoritme fortjener, har E.M. opnået megen opmærksomhed. Desværre har disse informationer oftest været vildledende og medført store forventninger til E.M.. Denne rapport er et forsøg på at finde ind bag de store overskrifter i aviserne.

Aviserne hævdede, at E.M. ville få revulotionerende betydning for numerisk løsning af L.P.-problemer. I de foregående kapitler har vi prøvet at give en baggrund for at vurdere denne påstand. Vi har beskrevet Simplex-metoden, som er effektiv. selvom det,at det er en eksponential-tids algoritme sætter en grænse for problemets størrelse. Vi har gennemgået E.M.'s teoretiske baggrund og vist, at der er problemer forbundet med en praktisk anvendelse. Vi har endvidere beskrevet NP-teorien, som omhandler algoritmers komplexitet og som giver baggrund for at forstå forskellene imellem Simplex og E.M.. På den baggrund mener vi, at E.M. i sin nuværende

form ikke vil få nogen praktisk betydning, men metoden kan selvfølgelig inspirerer til nye undersøgelser og forbedringer af algoritmer til løsning af L.P.-problemer.

De videnskabelige artikler, vi har stiftet bekendtskab emd, fremhæver, at E.M. er meget interessant i forhold til NP-teorien. Vi tror, at det er her, E.M. vil få den største betydning.

Litteraturliste.

Berresford G.C., Rockett A.M., Stevenson J.C.,
"Khachiyan's Algorithm" Part 1+2.
Byte, august+september 1980.

Hacijan L.G.,
"A Polynomial Algorithm in Linear Programming."
Sovjet Math. Dokl. 20:1 1979, p. 191-194.

Lawler E.L.,
"The Great Mathematical Sputnik of 1979."
The Sciences, september 1979.

Garey M.R., Johnson D.S.,
"Computers and Intractability."
1979, Bell Telephone Laboratories Inc.

Papadimitriou C.H., Steiglitz K.,
"Combinatorial Optimization."
1982, Printice-Hall, New Jersey.

Krarup J., Pruzan P.M.,
"Optimeringsmetoder I."
1974, Polyteknisk Forlag.

Gacs P., Lovaz L.,
"Khachiyan's Algorithm for Linear Programming."
1981, Math. Program. Stud. 14, p. 61-68.

Bland R.G., Goldfarb D., Todd M.J.,
"The Elipsoid Method: A Survey."
1981, Operations Research, p. 1039-1091.

Aspvall B., Stone R.E.,
"Khachiyan's Linear Programming Algorithm."
1980, Journal of Algorithms no.1, p. 1-13.

Bergendal G. Brinch I.,
"Linjar Algebra"
1970, Studenterlitteratur, Lund.

Bilag I.

I det følgende gentages og bevises de sætninger, som anvendes til beviset for E.M. i afsnit 5.5..

SÆTNING 1: Ethvert ekstrempunkt v på polyederet defineret ved

$$a^i x \leq b_i, \qquad i=1,..,m$$
$$x \geq 0$$

tilfredsstiller betingelsen $|v| < 2^L/n$. Dens koordinater er rationelle tal, hvis nævnere højst antager værdien 2^L.

BEVIS: Gangen i beviset går på at regne på de enkelte koordinater til v. Det vises at både tæller og nævner i de brøker, som udgør koordinaterne til v, ligger intervallet $1 \leq v_i \leq 2^L/n$.

Betegn $v=(v_1,..,v_n)$. Ved brug af Cramers regel kan hvert v_i udtrykkes, som $v_i=D_i/D$, hvor D_i og D er determinanter, hvis indgange er $0,1,a_{ij}$ og b_i. Da D og D_i er heltal, må $|D| \geq 1$. Hvis (d_{ij}) betegner matricen, hvis determinant er D, fås ved brug af Hadamard's ulighed følgende:

$$|D| \leq \prod_{i=1}^{m} \left(\sum_{j=1}^{m} d_{ij}^2 \right)^{\frac{1}{2}} < 2^L/mn < 2^L/n$$

og det samme gælder for D_i'erne.

SÆTNING 2: Hvis ulighedssystemet

$$a^i x < b_i, \qquad i=1,..,m \ , a^i \in Z^n, \ b_i \in Z$$

har en løsning, så er volumnet af dets løsninger, indeholdt i kuben: $|x_i| \leq 2^L$, mindst $2^{-(n+1)L}$.

BEVIS: Ideen i beviset går ud på at vise, at et polyeder med n+1 ekstrempunkter mindst har volumnet $2^{-(n+1)L}$.

Det forudsættes, at ulighedssystemet har en løsning $x^0 > 0$. Så polyederen

$$a^i x \leq b_i, \qquad i=1,..,m$$
$$x \geq 0$$

har et indre punkt. Da løsningsområdet jo er begrænset har polyederen et ekstrempunkt $v=(v_1,..,v_n)$. Sætning 1 fortæller, at $v_i < 2^L/n < \lfloor 2^L \rfloor$. Da polyederet har et indre punkt $x=(x_1,..,x_n)$, hvor $x_j < \lfloor 2^L \rfloor$, så har polyederet

$$a^i x \leq b_i, \qquad i=1,..,m$$
$$x \geq 0$$
$$x_j \leq \lfloor 2^L \rfloor, \qquad j=1,..,n$$

også et indre punkt. Da polyederet har n+1 ekstrempunkter $v_0,..,v_n$, som ikke udgør et hyperplan, har ovenstående ulighedssystem volumnet:

$$\frac{1}{n!} \left| \det \begin{pmatrix} 1 & 1 & \dots & 1 \\ v_0 & v_1 & & v_n \end{pmatrix} \right| .$$

Fra sætning 1 fås, at $v_i = \frac{1}{D_i} u_j$, hvor u_j er en heltalsvektor og D_i et heltal mindre end $2^L/n$. Videre fås:

$$\left| \det \begin{pmatrix} 1 & & 1 \\ v_0 & \dots & v_n \end{pmatrix} \right| = \frac{1}{|D_0| \cdot \cdot |D_n|} \left| \det \begin{pmatrix} D_0 & & D_n \\ u_0 & \dots & u_n \end{pmatrix} \right| \geq \frac{1}{|D_0| \cdot \cdot |D_n|} \geq 2^{-(n+1)L} n^{n+1},$$

da deternimanten i determinanten i det andet udtryk er et heltal forskelligt fra 0. Derfor er volumnet af polyederet mindst

$$(2^{-(n+1)L} n^{n+1})/n! > 2^{-(n+1)L}.$$

SÆTNING 3: Forudsæt at ulighedssystemet

$$a^i x < b_i + 2^{-L}, \quad i=1,..,m$$

har en løsning. Så har ulighedssystemet

$$a^i x \leq b_i , \qquad i=1,..,m$$

også en løsning.

BEVIS: Ideen i beviset er, at der eksisterer et x således at, der for nogen bånd gælder, at $a^i x = b_i$. (Dette er ekvivalent med, at $a^i x < b_i + 2^{-L}$). Det vises, at de a-vektorer, som tilfredsstiller dette udspænder samtlige a-vektorer. Herefter vises det, at de a-vektorer der udgør basen, svarer til de største $a^i x - b_i$ værdier. Det må derfor gælde at alle $a^i x \leq b_i$.

For $x \in R^n$, sæt $\Theta_i(x) = a^i x - b_i$. Lad $x^0 \in R^n$ være arbitrær.

PÅSTAND: Der eksisterer et $x' \in R^n$, sådan at

(1) $\Theta_i(x') \leq \max(0, \Theta_i(x^0)), \qquad i=1,..,m$

(2) vektorerne $\{ a^i : \Theta_i(x') \geq 0 \}$ udspænder enhver anden vektor a^i.

Bevis af påstanden: For at bevise påstanden er det tilstrækkeligt at vise, at hvis x^0 ikke tilfredsstiller (2), så findes der en vektor x', sådan at x' tilfredsstiller (1) og $\Theta_i(x') \geq 0$ for andre index i end $\Theta_i(x^0) \geq 0$. Ved at gentage dette højst m gange opnås et x', der

tilfredsstiller både (1) og (2).

Sæt $\theta_1(x^0),..,\theta_k(x^0)\geq 0,\theta_{k+1},..,\theta_m(x^0)<0$. Forudsæt at a^v $(v>k)$ er en ikke linæer kombination af $a^1,..,a^k$. Så er ligningssystemet

$$a^i y=0, \qquad i=1,..,k$$

$$a^v y=1$$

løsbart. Lad y^0 være en løsning og betragt $x^1=x^0+ty^0$, hvor

$$t=\max\{s\in R: sa^j y^0+\theta_j\leq 0, \quad j=k+1,..,k\}$$

t er mindre eller lig med $-\theta_v$. Så ved valg af t fås

$$\theta_i(x^1)=ta^i y^0+\theta_i(x^0)=\theta_i(x^0) \qquad \text{hvis } 1\leq i\leq k$$

$$" \quad " \qquad \leq 0 \qquad \text{hvis } k+1\leq i\leq m$$

og lighed opnås for mindst et $k+1\leq i\leq m$.

Forudsæt nu et x^0, sådan at

$$a^i x^0 < b_i+2^L, \qquad i=1,..,m$$

Lad $a^i x^0\geq b_i$ for $i=1,..,k$. Vælg en ordning, så at $a^1,..,a^r$ er lineært uafhængige, mens $a^{r+1},..,a^k$ er uspændt af dem. Fra påstanden kan det forudsættes, at $a^{k+1},..,a^n$ også er udspændt af $a^1,..,a^r$.

Lad nu z være en løsning til ligningssystemet

$$a^i z=b_i \qquad i=1,..,r.$$

Det skal nu vises, at z tilfredsstiller $a^i z\leq b_i$ for ethvert $1\leq i\leq m$.

Det er kendt, at

$$a^i=\sum_{j=1}^{r}\lambda_j a^j.$$

Fra Cramers regel fås, at $\lambda_j=D_j/D$, hvor D_j og D er heltallige determinanter med absolut-værditer mindre end $2^L/n$. Derfor

$$D(a^i z-b_i)=\sum_{j=1}^{r}D_j a^j z-Db_i=\sum_{j=1}^{r}D_j b_j-Db_i.$$

For at estimere højresiden:

$$\sum_{j=1}^{r}D_j b_j-Db_j=\sum_{j=1}^{r}D_j(a^j x^0-\theta_j(x^0))-D(a^i x^0-\theta_i(x^0))$$

$$=D\theta_i(x^0)-\sum_{j=1}^{r}D_j\theta_j(x^0)$$

$$\leq d2^{-L}+\sum_{j=1}^{r}|D_j|2^{-L}<1$$

og da venstresiden er heltallig fås $\sum_{j=1}^{r}D_j b_j-Db_j\leq 0$.

SÆTNING 4: $\frac{1}{2}E_a \in E^a$

BEVIS: Der konstrueres en enhedselipsoide. Udfra denne beregnes efter algoritmens formler et nyt center $x^{0'}$ og en ny matrix A'. Det vises at dette $(x^{0'},A')$ udgør en elipsoide.

Det forudsættes, at $x^0=0$ og $A=I$ (f.eks. elipsoiden er enhedssfæren om origo) og $a=(-1,0,..,0)^t$, da sætningen er invariant under affine transformationer i rummet. (Dette vises i (Aspwall 1980)). Så

$$x^{0'}=(\frac{1}{n+1},0,..,0)^t \text{ og}$$

$$A'=\text{diag}(\frac{n^2}{(n+1)^2},\frac{n^2}{n^2-1},..,\frac{n^2}{n^2-1}).$$

Forudsæt, at $x \in \frac{1}{2}E_a$. Så er $|x| \leq 1$, $1 \geq \xi_1 = -a^t x \geq 0$. Det skal så vises, at

$$(x-x^0)^t A'^{-1}(x-x^{0'}) \leq 1.$$

Men $(x-x^{0'})^t A'^{-1}(x-x^0) = x^t A'^{-1}x - 2x^t A'^{-1}x^{0'} + x^{0'^t} A'^{-1}x^{0'}$

$$= \frac{n^2-1}{n^2}x^2 - \frac{2n+2}{n^2}\xi_1^2 - 2\frac{n+1}{n^2}\xi_1 + \frac{1}{n^2}$$

$$= \frac{n^2-1}{n^2}(x^2-1) + \frac{2n+2}{n^2}\xi_1(\xi_1-1) + 1 \leq 1.$$

SÆTNING 5: $\lambda(E^a) = c(n)\lambda(E)$, hvor $c(n) = (\frac{n^2}{n^2-1})^{(n-1)/2}\frac{n}{n+1} < e^{-(\frac{1}{2}(n+1))}$, og $\lambda(X)$ er volumnet af mængden X.

BEVIS: Det vises v.h.a. volumenformler for simple elipsoider, at $\lambda(E^a) = \lambda(E)\cdot\sqrt{\det A'}$. Det vises herefter, at $\sqrt{\det A'}$ er ekvivalent med udtrykket for c(n) i sætningen. Begrænsningen på c(n) fås ved et estimat af $\sqrt{\det A'}$.

Det forudsættes igen, at E er enhedssfæren om origo og $a=(1,0,..,0)^t$, da affine transformationer ikke ændrer proportionerne af volumnerne. Fra formlen for en elipsoides volumen fås

$$\lambda(E^a) = \frac{\sqrt{\det A'}}{\sqrt{\det A}}\lambda(E) = \lambda(E)\cdot\sqrt{\det A'}$$

$$= \frac{n}{n+1}(\frac{n^2}{n^2-1})^{(n-1)/2}\cdot\lambda(E) = c(n)\lambda(E).$$

For at estimere denne faktor bruges, at

$$\frac{n^2}{n^2-1} = 1+\frac{1}{n^2-1} < e^{1/(n^2-1)} \text{ og } \frac{n}{n+1} = 1-\frac{1}{n+1} < e^{-1/(n+1)}.$$

Ved indsættelse fås $c(n) < e^{-1/(2(n+1))}$.

Bilag II.

```
1  begin
2
3    procedure LI(N, M);
4    integer N, M;
5    begin
6      integer array A(1: M, 1: N);
7      long real array E(1: N, 1: N), X(1: N), B, MALFA(1: M);
8      integer L, MAXIT, K, ALFABAAND;
9      long real ALFAMAX, MI;
10
11     procedure INDLAES;
12     begin
13       integer I, J;
14       ! Proceduren indlaeser a raekke for raekke og b;
15       for I:= 1 step 1 until M do
16         for J:= 1 step 1 until N do
17           A(I, J):= ININT;
18       for I:= 1 step 1 until M do
19         B(I):= INREAL;
20     end *indlaes*;
21
22     procedure INITIALISER;
23     begin
24       integer I, J;
25       long real LL;
26       for I:= 1 step 1 until M do
27         for J:= 1 step 1 until N do
28           L:= L+LOG(ABS(A(I, J))+1)/LOG(2);
29       for I:= 1 step 1 until M do
30         L:= L+LOG(ABS(B(I))+1)/LOG(2);
31       L:= L+LOG(M*N)/LOG(2)+1;
32       MAXIT:= ((N+1)**2)*4*L;
33       LL:= 2**(2*L);
34       for I:= 1 step 1 until N do
35         E(I, I):= LL;
36     end *initialiser*;
37
38     procedure BEREGNM;
39     begin
40       integer H, I, J;
41       for I:= 1 step 1 until M do
42         begin
43           MALFA(I):= 0;
44           for J:= 1 step 1 until N do
45             begin
46               long real MTEMP;
47               for H:= 1 step 1 until N do
48                 MTEMP:= MTEMP+A(I, H)*E(H, J);
49               MALFA(I):= MALFA(I)+MTEMP*A(I, J);
50             end;
51           MALFA(I):= SQRT(MALFA(I));
52         end;
53     end *beregnm*;
54
55     procedure FINDALFA;
56     begin
57       integer I, J;
58       ALFAMAX:= 0;
59       for I:= 1 step 1 until N do
60         ALFAMAX:= ALFAMAX+A(1, I)*X(I);
61       ALFAMAX:= (ALFAMAX-B(1))/MALFA(1);
62       ALFABAAND:= 1;
63       for I:= 2 step 1 until M do
```

```
64          begin
65            long real ALFA;
66            for J:= 1 step 1 until N do
67              ALFA:= ALFA+A(I, J)*X(J);
68            ALFA:= (ALFA-B(I))/MALFA(I);
69            if ALFA > ALFAMAX
70            then
71              begin
72                ALFAMAX:= ALFA;
73                ALFABAAND:= I;
74              end;
75          end;
76        MI:= MALFA(ALFABAAND);
77      end *findalfa*;
78
79      boolean procedure STOP;
80      begin
81        K:= K+1;
82        STOP:= K > MAXIT or ALFAMAX <= 0 or ALFAMAX > 1;
83      end *stop*;
84
85      procedure NYTCENTER;
86      begin
87        integer I, J;
88        long real SP, XTEMP;
89        SP:= (1+N*ALFAMAX)/(N+1);
90        for I:= 1 step 1 until N do
91          begin
92            XTEMP:= 0;
93            for J:= 1 step 1 until N do
94              XTEMP:= XTEMP+E(I, J)*A(ALFABAAND, J);
95            XTEMP:= XTEMP*SP/MI;
96            X(I):= X(I)-XTEMP;
97          end;
98      end *nytcenter*;
99
100     procedure NYELIPSOIDE;
101     begin
102       integer I, J;
103       long real EP, DP;
104       long real array V(1: N);
105       EP:= (N**2/(N**2-1))*(1-ALFAMAX**2);
106       DP:= (2*(1+N*ALFAMAX))/((N+1)*(1+ALFAMAX));
107       for I:= 1 step 1 until N do
108         for J:= 1 step 1 until N do
109           V(I):= V(I)+A(ALFABAAND, J)*E(I, J);
110       for I:= 1 step 1 until N do
111         for J:= 1 step 1 until N do
112           E(I, J):= EP*(E(I, J)-(DP*(V(I)/MI)*(V(J)/MI)));
113       for I:= 1 step 1 until N-1 do
114         for J:= I+1 step 1 until N do
115           if ABS(E(I, J)) < ABS(E(J, I))
116           then
117             E(I, J):= E(J, I)
118           else
119             E(J, I):= E(I, J);
120     end *nyelipsoide*;
121
122     procedure UDSKRIV;
123     begin
124       integer I, J;
125       OUTTEXT("L:");
126       OUTINT(L, 8);
```

```
127         OUTTEXT("   maxit:");
128         OUTINT(MAXIT, 8);
129         OUTIMAGE;
130         OUTTEXT("Initialisering af tabellerne:");
131         OUTIMAGE;
132         OUTIMAGE;
133         OUTTEXT("A-matrissen:");
134         OUTIMAGE;
135         for I:= 1 step 1 until M do
136           begin
137             for J:= 1 step 1 until N do
138               OUTINT(A(I, J), 5);
139             OUTIMAGE;
140           end;
141         OUTIMAGE;
142         OUTIMAGE;
143         OUTTEXT("b-vektoren:");
144         OUTIMAGE;
145         for I:= 1 step 1 until M do
146           OUTINT(B(I), 5);
147         OUTIMAGE;
148         OUTIMAGE;
149         OUTTEXT("Programmet stopper efter ");
150         OUTINT(K-1, ENTIER(LOG(K)+1));
151         OUTTEXT(" iterationer.");
152         OUTIMAGE;
153         if ALFAMAX <= 0
154         then
155           begin
156             OUTIMAGE;
157             OUTTEXT("x-vektoren:");
158             OUTIMAGE;
159             for I:= 1 step 1 until N do
160               begin
161                 OUTREAL(X(I), 8, 20);
162                 OUTIMAGE;
163               end;
164           end
165         else
166           OUTTEXT("Der er ingen loesning.");
167         OUTIMAGE;
168       end *udskriv*;
169
170       INDLAES;
171       INITIALISER;
172       BEREGNM;
173       FINDALFA;
174       while not STOP do
175         begin
176           NYTCENTER;
177           NYELIPSOIDE;
178           BEREGNM;
179           FINDALFA;
180         end;
181       UDSKRIV;
182     end *li*;
183
184     LI(ININT, ININT);
185   end;
```

Bilag III.

Vi var tre datalogistuderende, der havde aftalt at starte på matematikstudiet i januar 83. Vi samledes i slutningen af januar og blev enige om at starte på model modulet (mat.-studiets II.modul). Vi havde fået tildelt Mogens Brun Heefelt (MBH) som vejleder, og vi samledes hurtigt om emnet E.M., som udgangspunkt for projektet.

Det var en artikel i Byte, der var vores første kilde om E.M.. Selvom denne artikel skulle være let tilgængelig, havde vi svært ved at forstå de matematiske omskrivninger. MBH foreslog så, at vi læste lineær algebra først og gik til eksamen i denne emnekreds. Det var både fagligt og studieplansmæssigt en stor fordel for os. Vi brugte derefter ca. en måned på at læse lineær algebra.

Først i marts, blev hovedvægten lagt på implementeringen, da vi på det tidspunkt, så det, som den største udfordring. Vi havde på dette tidspunkt en samtale med Krarup fra DIKU, der henviste og til noget mere dybtgående litteratur. Vi kiggede så på det og afsluttede kursusstoffet med en gennemgang af Simplex-metoden. I slutningen af marts hoppede det ene gruppemedlem fra, og vi var to tilbage i gruppen.

I begyndelsen af april gik vi for alvor igang med læsning af artikler om E.M., da vi havde fået et rimeligt begrebsapparat. Vi blev hurtigt opmærksomme på, at E.M. har stor betydning for NP-teorien. Vi syntes det var et spændende område og bestilte derfor noget litteratur om emnet. Derefter lavede vi en første implementation af E.M.. Vi tog udgangspunkt i et BASIC program fra Byte-artiklen som vi oversatte til Simula.

I slutningen af april opstillede vi en disposition for rapporten, som lå meget tæt op ad den endelige rapport. Vi blev på dette tidspunkt klar over, at vi ikke ville få tid til en dybtgående undersøgelse af de enkelte delområder.

Omkring 1. maj havde vi opstillet algoritmen. Vi brugte resten af måneden til at programmere, skrive og læse lineær algebra til den skriftelige eksamen i begyndelsen af juni. Fra midten af maj og til afslutningen af projektet hen i juli har vi ikke haft de store principielle diskusioner af projektet. Vi har programmeret

og skrevet udfra den disposition, vi lagde os fast på i slutningen af april.

Emnet E.M. er meget relevant, men vore hovedkilder stammer hovedsageligt fra årene 1979-81. Flere af dem er endda først udkommet i lokale rapporter, før de er blevet trykt i alment tilgængelige tidsskrifter. Derfor er vores behandling og opsummering af viden om E.M. baseret på 2-4 år gamle kilder.

1/78 "TANKER OM EN PRAKSIS" - et matematikprojekt.
Projektrapport af Anne Jensen, Lena Lindenskov, Marianne Kesselhahn og Nicolai Lomholt.
Vejleder: Anders Madsen.

2/78 "OPTIMERING" - Menneskets forøgede beherskelsesmuligheder af natur og samfund.
Projektrapport af Tom J. Andersen, Tommy R. Andersen, Gert Kreinøe og Peter H. Lassen.
Vejleder: Bernhelm Booss.

3/78 "OPGAVESAMLING", breddekursus i fysik.
Lasse Rasmussen, Aage Bonde Kræmmer, Jens Højgaard Jensen.

4/78 "TRE ESSAYS" - om matematikundervisning, matematiklæreruddannelsen og videnskabsrindalismen.
Mogens Niss.
Nr. 4 er p.t. udgået.

5/78 "BIBLIOGRAFISK VEJLEDNING til studiet af DEN MODERNE FYSIKS HISTORIE".
Helge Kragh.

6/78 "NOGLE ARTIKLER OG DEBATINDLÆG OM - læreruddannelse og undervisning i fysik, og - de naturvidenskabelige fags situation efter studenteroprøret".
Karin Beyer, Jens Højgaard Jensen og Bent C. Jørgensen.

7/78 "MATEMATIKKENS FORHOLD TIL SAMFUNDSØKONOMIEN".
B.V. Gnedenko.
Nr. 7 er udgået.

8/78 "DYNAMIK OG DIAGRAMMER". Introduktion til energy-bound-graph formalismen.
Peder Voetmann Christiansen.

9/78 "OM PRAKSIS' INDFLYDELSE PA MATEMATIKKENS UDVIKLING". - Motiver til Kepler's:"Nova Stereometria Doliorum Vinarioum".
Projektrapport af Lasse Rasmussen.
Vejleder: Anders Madsen.

10/79 "TERMODYNAMIK I GYMNASIET".
Projektrapport af Jan Christensen og Jeanne Mortensen.
Vejledere: Karin Beyer og Peder Voetmann Christiansen.

11/79 "STATISTISKE MATERIALER"
red. Jørgen Larsen

12/79 "LINEÆRE DIFFERENTIALLIGNINGER OG DIFFERENTIALLIGNINGSSYSTEMER".
Mogens Brun Heefelt

13/79 "CAVENDISH'S FORSØG I GYMNASIET".
Projektrapport af Gert Kreinøe.
Vejleder: Albert Chr. Paulsen

14/79 "BOOKS ABOUT MATHEMATICS: History, Philosophy, Education, Models, System Theory, and Works of Reference etc. A Bibliography".
Else Høyrup.
Nr. 14 er p.t. udgået.

15/79 "STRUKTUREL STABILITET OG KATASTROFER i systemer i og udenfor termodynamisk ligevægt".
Specialeopgave af Leif S. Striegler.
Vejleder: Peder Voetmann Christiansen.

16/79 "STATISTIK I KRÆFTFORSKNINGEN".
Projektrapport af Michael Olsen og Jørn Jensen.
Vejleder: Jørgen Larsen.

17/79 "AT SPØRGE OG AT SVARE i fysikundervisningen".
Albert Christian Paulsen.

18/79 "MATHEMATICS AND THE REAL WORLD", Proceedings of an International Workshop, Roskilde University Centre, Denmark, 1978. Preprint.
Bernhelm Booss & Mogens Niss (eds.).

19/79 "GEOMETRI, SKOLE OG VIRKELIGHED".
Projektrapport af Tom J. Andersen, Tommy R. Andersen og Per H.H. Larsen.
Vejleder: Mogens Niss.

20/79 "STATISTISKE MODELLER TIL BESTEMMELSE AF SIKRE DOSER FOR CARCINOGENE STOFFER".
Projektrapport af Michael Olsen og Jørn Jensen.
Vejleder: Jørgen Larsen.

21/79 "KONTROL I GYMNASIET - FORMÅL OG KONSEKVENSER".
Projektrapport af Crilles Bacher, Per S. Jensen, Preben Jensen og Torben Nysteen.

22/79 "SEMIOTIK OG SYSTEMEGENSKABER (1)".
1-port lineært response og støj i fysikken.
Peder Voetmann Christiansen.

23/79 "ON THE HISTORY OF EARLY WAVE MECHANICS - with special emphasis on the role of realitivity".

24/80 a+b "MATEMATIKOPFATTELSER HOS 2.G'ERE".
1. En analyse. 2. Interviewmateriale.
Projektrapport af Jan Christensen og Knud Lindhardt Rasmussen.
Vejleder: Mogens Niss.
Nr. 24 a+b er p.t. udgået.

25/80 "EKSAMENSOPGAVER", Dybdemodulet/fysik 1974-79.

26/80 "OM MATEMATISKE MODELLER".
En projektrapport og to artikler.
Jens Højgaard Jensen m.fl.
Nr. 26 er p.t. udgået.

27/80 "METHODOLOGY AND PHILOSOPHY OF SCIENCE IN PAUL DIRAC's PHYSICS".
Helge Kragh.

28/80 "DIELEKTRISK RELAXATION - et forslag til en ny model bygget på væskernes viscoelastiske egenskaber".
Projektrapport, speciale i fysik, af Gert Kreinøe.
Vejleder: Niels Boye Olsen.

29/80 "ODIN - undervisningsmateriale til et kursus i differentialligningsmodeller".

Projektrapport af Tommy R. Andersen, Per H.H. Larsen og Peter H. Lassen.

Vejleder: Mogens Brun Heefelt

30/80 "FUSIONSENERGIEN - - - ATOMSAMFUNDETS ENDESTATION".

Oluf Danielsen.

Nr. 30 er udgået.

Udkommer medio 1982 på Fysik-, Matematik- og Kemilærernes forlag.

31/80 "VIDENSKABSTEORETISKE PROBLEMER VED UNDERVISNINGSSYSTEMER BASERET PA MÆNGDELÆRE".

Projektrapport af Troels Lange og Jørgen Karrebæk.

Vejleder: Stig Andur Pedersen.

Nr. 31 er p.t. udgået

32/80 "POLYMERE STOFFERS VISCOELASTISKE EGENSKABER - BELYST VED HJÆLP AF MEKANISKE IMPEDANSMALINGER OG MOSSBAUEREFFEKTMALINGER".

Projektrapport, speciale i fysik, af Crilles Bacher og Preben Jensen.

Vejledere: Niels Boye Olsen og Peder Voetmann Christiansen.

33/80 "KONSTITUERING AF FAG INDEN FOR TEKNISK-NATURVIDENSKABELIGE UDDANNELSER. I-II".

Arne Jakobsen.

34/80 "ENVIRONMENTAL IMPACT OF WIND ENERGY UTILIZATION".

ENERGY SERIES NO.1.

Bent Sørensen.

Nr. 34 er udgået.

Publ. i "Renewable Sources of Energy and the Environment", Tycooli International Press, Dublin, 1981.

35/80 "HISTORISKE STUDIER I DEN NYERE ATOMFYSIKS UDVIKLING".

Helge Kragh.

36/80 "HVAD ER MENINGEN MED MATEMATIKUNDERVISNINGEN ?".

Fire artikler.

Mogens Niss.

37/80 "RENEWABLE ENERGY AND ENERGY STORAGE".

ENERGY SERIES NO.2.

Bent Sørensen.

38/81 "TIL EN HISTORIETEORI OM NATURERKENDELSE, TEKNOLOGI OG SAMFUND".

Projektrapport af Erik Gade, Hans Hedal, Henrik Lau og Finn Physant.

Vejledere: Stig Andur Pedersen, Helge Kragh og Ib Thiersen.

39/81 "TIL KRITIKKEN AF VÆKSTØKONOMIEN".

Jens Højgaard Jensen.

40/81 "TELEKOMMUNIKATION I DANMARK - oplæg til en teknologivurdering".

Projektrapport af Arne Jørgensen, Bruno Petersen og Jan Vedde.

Vejleder: Per Nørgaard.

41/81 "PLANNING AND POLICY CONSIDERATIONS RELATED TO THE INTRODUCTION OF RENEWABLE ENERGY SOURCES INTO ENERGY SUPPLY SYSTEMS".

ENERGY SERIES NO.3.

Bent Sørensen.

42/81 "VIDENSKAB TEORI SAMFUND - En introduktion til materialistiske videnskabsopfattelser".

Helge Kragh og Stig Andur Pedersen.

43/81 1. "COMPARATIVE RISK ASSESSMENT OF TOTAL ENERGY SYSTEMS".

2. "ADVANTAGES AND DISADVANTAGES OF DECENTRALIZATION".

ENERGY SERIES NO.4.

Bent Sørensen.

44/81 "HISTORISK UNDERSØGELSE AF DE EKSPERIMENTELLE FORUDSÆTNINGER FOR RUTHERFORDS ATOMMODEL".

Projektrapport af Niels Thor Nielsen.

Vejleder: Bent C. Jørgensen.

45/82

46/82 I+II "EKSEMPLARISK UNDERVISNING OG FYSISK ERKENDELSE - ILLUSTRERET VED TO EKSEMPLER".

Projektrapport af Torben O. Olsen, Lasse Rasmussen og Niels Dreyer Sørensen.

Vejleder: Bent C. Jørgensen.

47/82 "BARSEBÄCK OG DET VÆRST OFFICIELT-TÆNKELIGE UHELD".

ENERGY SERIES NO.5.

Bent Sørensen.

48/82 "EN UNDERSØGELSE AF MATEMATIKUNDERVISNINGEN PA ADGANGSKURSUS TIL KØBENHAVNS TEKNIKUM".

Projektrapport af Lis Eilertzen, Jørgen Karrebæk, Troels Lange, Preben Nørregaard, Lissi Pedersen, Laust Rishøj, Lill Røn, Isac Showiki.

Vejleder: Mogens Niss.

49/82 "ANALYSE AF MULTISPEKTRALE SATELLITBILLEDER".

Projektrapport af Preben Nørregaard.

Vejledere: Jørgen Larsen & Rasmus Ole Rasmussen.

50/82 "HERSLEV - MULIGHEDER FOR VEDVARENDE ENERGI I EN LANDSBY". ENERGY SERIES NO.6.

Rapport af Bent Christensen, Bent Hove Jensen, Dennis B. Møller, Bjarne Laursen, Bjarne Lillethorup og Jacob Mørch Pedersen.

Vejleder: Bent Sørensen.

51/82 "HVAD KAN DER GØRES FOR AT AFHJÆLPE PIGERS BLOKERING OVERFOR MATEMATIK?"

Projektrapport af Lis Eilertzen, Lissi Pedersen, Lill Røn og Susanne Stender.

52/82 "DESUSPENSION OF SPLITTING ELLIPTIC SYMBOLS"

Bernhelm Booss & Krzysztof Wojciechowski.

53/82 "THE CONSTITUTION OF SUBJECTS IN ENGINEERING EDUCATION".

Arne Jakobsen & Stig Andur Pedersen.

54/82 "FUTURES RESEARCH" - A Philosophical Analysis of Its Subject-Matter and Methods.

Stig Andur Pedersen & Johannes Witt-Hansen.

55/82 "MATEMATISKE MODELLER" - Litteratur på Roskilde Universitetsbibliotek.

En bibliografi.

Else Høyrup.

Vedr. tekst nr. 55/82:
Se også tekst 62/83.

56/82 "ÉN - TO - MANGE" -
En undersøgelse af matematisk økologi.

Projektrapport af Troels Lange.

Vejleder: Anders Madsen.

57/83 "ASPECT EKSPERIMENTET" -
Skjulte variable i kvantemekanikken?

Projektrapport af Tom Juul Andersen.

Vejleder: Peder Voetmann Christiansen.

Nr. 57 er udgået.

58/83 "MATEMATISKE VANDRINGER" - Modelbetragtninger over spredning af dyr mellem småbiotoper i agerlandet.

Projektrapport af Per Hammershøj Jensen & Lene Vagn Rasmussen.

Vejleder: Jørgen Larsen.

59/83 "THE METHODOLOGY OF ENERGY PLANNING".

ENERGY SERIES NO. 7.

Bent Sørensen.

60/83 "MATEMATISK MODEKSPERTISE" - et eksempel.

Projektrapport af Erik O. Gade, Jørgen Karrebæk og Preben Nørregaard.

Vejleder: Anders Madsen.

61/83 "FYSIKS IDEOLOGISKE FUNKTION", som et eksempel på en naturvidenskab - historisk set.

Projektrapport af Annette Post Nielsen.

Vejledere: Jens Høyrup, Jens Højgaard Jensen og Jørgen Vogelius.

62/83 "MATEMATISKE MODELLER" - Litteratur på Roskilde Universitetsbibliotek.

En bibliografi. 2. rev. udgave

Else Høyrup

63/83 "CREATING ENERGY FUTURES: A SHORT GUIDE TO ENERGY PLANNING".

ENERGY SERIES No. 8

David Crossley & Bent Sørensen

64/83 "VON MATHEMATIK UND KRIEG".

Bernhelm Booss og Jens Høyrup

65/83 "ANVENDT MATEMATIK - TEORI ELLER PRAKSIS".

Projektrapport af Per Hedegård Andersen, Kirsten Habekost, Carsten Holst-Jensen, Annelise von Moos, Else Marie Pedersen, Erling Møller Pedersen.

Vejledere: Bernhelm Booss & Klaus Grünbaum

66/83 "MATEMATISKE MODELLER FOR PERIODISK SELEKTION I ESCHERICHIA COLI".

Projektrapport af Hanne Lisbet Andersen, Ole Richard Jensen og Klavs Frisdahl.

Vejledere: Jørgen Larsen og Anders Hede Madsen

ISSN 0106-6242