Aalto-yliopisto
Perustieteiden korkeakoulu
Teknillisen fysiikan ja matematiikan tutkinto-ohjelma

Lineaaristen monitavoiteoptimointitehtävien ratkaiseminen

Kandidaatintyö
22. marraskuuta 2012

Jerri Nummenpalo

Aalto-yliopisto
Perustieteiden korkeakoulu
Teknillisen fysiikan ja matematiikan koulutusohjelma

KANDIDAATINTYÖN
TIIVISTELMÄ

Tekijä:	Jerri Nummenpalo
Työn nimi:	Lineaaristen monitavoiteoptimointitehtävien ratkaiseminen
Pääaine:	Systeemitieteet
Koodi:	F3010
Vastuuopettaja:	Prof. Ahti Salo
Työn ohjaaja:	TkT Juuso Liesiö

Tiivistelmä:
Monitavoitteinen lineaarinen optimointi (MOLP, Multiple Objective Linear Programming) on monen lineaarisen kohdefunktion optimointia lineaarisilla rajoitusehdoilla. Monitavoitteisia lineaarisia ongelmia esiintyy laajasti esimerkiksi erilaisissa talouden ja tekniikan alan sovelluksissa.

Tässä työssä kuvaillaan MOLP-tehtävän ratkaisut - Pareto-optimaaliset pisteet - etsivän monitavoitteisen Simplex-algoritmin toimintaperiaate sekä tarvittava teoria. Työssä keskitytään tehokkaiden kulmapisteiden ratkaisemiseen, mutta myös menetelmä kaikkien pisteiden löytämiseksi selitetään. Ohjelmoidun algoritmin käyttömahdollisuuksia tutkitaan soveltamalla sitä monitavoitteiseen resurssienallokointitehtävään.

Työn tuloksena saadaan, että esitelty algoritmi on hyvä pienikokoisten ongelmien ratkaisemisessa, joissa päätösmuuttujia on muutamia satoja. Varsinkin pienehköissä resurssienallokointiongelmissa algoritmin avulla voidaan ratkaista tehtävä ja tarjota resurssienallokoinnista päättävälle mielenkiintoisia suosituksia. Suurempia tehtäviä ratkaistaessa vaihtoehtoiset lähestymistavat, esimerkiksi tavoiteavaruudessa toimiva algoritmit, ovat tarpeen.

Päiväys: 22. marraskuuta 2012 **Kieli:** suomi **Sivumäärä:** 28

Avainsanat: optimointi, monitavoiteoptimointi, lineaarinen, tehokkuusanalyysi, LP, MOLP, DEA

Sisältö

Käytetyt symbolit ja lyhenteet

LP	Linear Programming, lineaarinen ohjelmointi
MOLP	Multiple Objective Linear Programming, monitavoitteinen lineaarinen ohjelmointi
LP(λ)	Painovektorin λ avulla LP-tehtäväksi redusoitu MOLP
$\mathbb{R}^p_>$	$\{x \in \mathbb{R}^p \mid x_i > 0 \,\forall\, i = 1, \ldots, p\}$
$\mathbb{R}^p_+$	$\{x \in \mathbb{R}^p \mid x_i \geq 0 \,\forall\, i = 1, \ldots, p\}$
$\mathcal{B}$	Kantaindeksit
$\mathcal{N}$	Ei-kantaindeksit
$\mathcal{EN}$	Tehokkaat ei-kantaindekst
$\mathcal{A}_\mathcal{B}$	Kantamatriisi
$\widetilde{A}$	Matriisi $\mathcal{A}_\mathcal{B}^{-1}A$
$\widetilde{b}$	Vektori $\mathcal{A}_\mathcal{B}^{-1}b$
e	$\{x \in \mathbb{R}^p \mid x_i = 1 \,\forall\, i = 1, \ldots, p\}$
$\mathcal{X}$	Käypä joukko
$\mathcal{X}_E$	Tehokas joukko
$\overline{c}$	Redusoidut kustannukset LP-tehtävässä
$\overline{C}$	Redusoidut kustannukset MOLP-tehtävässä
R	Ei-kantamuuttujien redusoidut kustannukset
$\mathcal{L}_1$	Joukko käsittelemättömiä kantoja
$\mathcal{L}_2$	Joukko käsitteltyjä kantoja
DEA	Data Envelopment Analysis, tehokkuusanalyysi
Λ^{CCR}	Käypien painojen joukko
T	Mahdollisten tuotantoyksiköiden joukko
$Ef(T)$	Tehokkaiden tuotantoyksiköiden joukko

1 Johdanto

Lineaarinen optimointi (LP, Linear Programming) on jatkuvan lineaarisen kohdefunktion optimointia jatkuvilla lineaarisilla yhtälö- ja epäyhtälörajoitteilla. Lineaarisesta optimoinnista puhuttaessa käyttöön on vakiintunut myös termi lineaarinen ohjelmointi.

LP-tehtävät ovat konvekseja tehtäviä, joissa mikä tahansa lokaali optimi on myös globaali optimi. Tämän seurauksena LP-tehtävät ovat ratkaistavissa tehokkaasti. Kun lisäksi monet tehtävät ovat luonnostaan lineaarisia, tai saatettavissa lineaarisiksi, tällä tehtäväluokalla on paljon sovelluskohteita. Yleisiä ongelmatyyppejä ovat esimerkiksi tuotannon- tai reitinsuunnittelutehtävät (Hung ja Leachman, 1996; Ferguson ja Dantzig, 1956) sekä verkostoongelmat (Bazaraa ym., 1990).

Monissa tilanteissa optimointi yhden tavoitteen tai funktion suhteen ei kuitenkaan ole riitä tai ole edes järkevää. Esimerkiksi sijoitusportfolio-optimoinnissa ei yleensä kannata maksimoida ainoastaan tuottoa välittämättä lainkaan riskeistä. Myös ympäristöpäätöksenteon ongelmissa joudutaan monissa tapauksissa pohtimaan useiden eri tavoitteiden samanaikaista optimointia. Tällaiset tehtävät kuuluvat LP-tehtäviä laajempaan tehtäväluokkaan, eli monitavoitteisiin lineaarisiin optimointitehtäviin (MOLP, Multiple Objective Linear Programming). Niiden sovellusmahdollisuudet ovat LP-tehtäviä laajemmat, sillä moni LP-tehtävä on laajennettavissa MOLP-tehtäväksi lisäämällä tavoitteita. Käytännön esimerkkitehtävä on resurssien allokointi tuotantoyksiköille (Korhonen ja Syrjänen, 2004).

LP-tehtävien ratkaisuna on optimi, joka antaa parhaan arvon kohdefuntiolle. Monitavoitteisissa tehtävissä ei kuitenkaan välttämättä saada kaikkia kohdefunktioita optimoitua samanaikaisesti. Optimin sijaan MOLP-tehtävien ratkaisuja ovat tehokkaat pisteet, joita sanotaan myös Pareto-optimaalisiksi pisteiksi. Käypä piste, eli rajoitteet toteuttava piste, on tehokas, jos missä tahansa toisessa käyvässä pisteessä kohdefunktioiden arvot ovat huonompia tai korkeintaan samoja. Koska yhden optimin sijaan onkin etsittävä joukko tehokkaita pisteitä, tehtävän ratkaisemiseksi vaadittu laskentatyö kasvaa paljon. Nykyaikaiset tietokoneet eivät tyypillisesti pysty ratkaisemaan kuin korkeintaan pieniä tai keskikokoisia MOLP-tehtäviä, joissa muuttujia on joitakin satoja, kun taas miljoonien muuttujien LP-tehtäviä on jo pitkään onnistuttu ratkaisemaan hyvinkin nopeasti (kts. esim. Lustig ym., 1994).

MOLP-tehtäviä ratkaisevia ohjelmistoja ei juurikaan ole saatavilla. Fortran-kielellä ohjelmoitu ADBASE (Steuer, 2003) on alan kirjallisuudessa siteera-

tuin ohjelmisto, mutta sekin on saatavilla ainoastaan akateemiseen käyttöön. Ero on merkittävä LP-tehtäviin, joille on olemassa lukuisia ratkaisuohjemistoja niin vapailla kuin kaupallisillakin markkinoilla. Tässä työssä tutustutaan monitavoitteisten lineaaristen ongelmien ratkaisuteoriaan ja tarkastellaan ohjelmoidun MOLP-tehtävän ratkaisevan algoritmin tehokkuutta. Lisäksi tutkitaan algoritmin käyttömahdollisuuksia todellisissa ongelmissa ratkaisemalla kirjallisuudessa esitetty resurssienallokointitehtävä.

Loppuosa työstä on jaoteltu seuraavalla tavalla. Kappaleessa 2 esitellään lineaarisen ohjelmoinnin teoriaa ja käsitteistöä sekä Simplex-algoritmin toimintaa. Kappale 3 käsittelee monitavoitteista lineaarista ohjelmointia ja sen ratkaisemista monitavoitteisella Simplex-algoritmilla. Kappaleessa 4 ratkaistaan monitavoitteinen resurssienallokointitehtävä. Yhteenveto on kappaleessa 5.

2 Lineaarinen ohjelmointi

2.1 Lineaarisen ohjelmoinnin perusmuotoinen tehtävä

Yksitavoitteinen jatkuva lineaarisen ohjelmoinnin tehtävä voidaan esittää muodossa

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.e.} \quad & Ax = b \\ & x \geq 0, \end{aligned} \tag{1}$$

missä c $\in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, ja $b \in \mathbb{R}^m$ (Bertsimas ja Tsitsik lis, 1997). Muuttujaa $x \in \mathbb{R}^n$ kutsutaan *päätösmuuttujaksi*. Rajoitusehdot määräävät n-ulotteiseen avaruuteen konveksin ja suljetun alueen $\mathcal{X} = \{x \in \mathbb{R}^n \mid Ax = b,\ x \geq 0\}$, jota sanotaan *käyväksi alueeksi*. Mikäli käypä alue on tyhjä joukko, sanotaan, ettei tehtävä ole käypä. Tehtävää (1) kutsutaan yleisesti LP-tehtäväksi (Linear Programming).

Määritelmä 1 *Olkoon $\hat{x} \in \mathcal{X}$. Mikäli $\nexists\ x \in \mathcal{X}$ s.e. $c^T x < c^T \hat{x}$, piste $\hat{x}$ on tehtävän (1) optimi.*

Tehtävän (1) ratkaisu on siis vektori $\hat{x} \in \mathcal{X}$, jossa lausekkeen $c^T \hat{x}$ arvo on pienin mahdollinen. Määritelmän 1 mukaisesti tällaista vektoria sanotaan tehtävän (1) optimiksi. Optimaalinen vektori ei välttämättä ole yksikäsitteinen, vaan saman LP-tehtävän saattaa ratkaista joukko pisteitä.

2.2 Simplex-algoritmi

Olkoon annettuna tehtävän (1) mukainen ongelma. Tällöin voidaan yleisyyttä rajoittamatta aina olettaa, että matriisin A aste on m (Bertsimas ja Tsitsik lis, 1997). Matriisin A säännöllistä alimatriisia $A_{\mathcal{B}} \in \mathbb{R}^{m\times m}$ sanotaan *kantamatriisiksi*. Indeksijoukkoa $\mathcal{B} \subset \{1,\ldots,n\}$ kutsutaan *kannaksi*, ja se sisältää ne A:n sarakkeiden indeksit, jotka määrittelevät $A_{\mathcal{B}}$:n. Olkoon lisäksi $\mathcal{N} = \{1,\ldots,n\}\backslash\mathcal{B}$ niiden sarakeindeksien joukko, jotka eivät kuulu kantaan. Päätösmuuttujan x alkion x_i ja indeksin i sanotaan olevan *kannassa*, mikäli $i \in \mathcal{B}$ ja alkiota x_i kutsutaan tällöin *kantamuuttujaksi*. Mikäli $i \in \mathcal{N}$, niin sanotaan, että x_i on *ei-kantamuuttuja*.

Jaottelemalla indeksijoukkojen $\mathcal{B}$ ja $\mathcal{N}$ avulla matriisi A ja vektori x osiin, tehtävän (1) rajoitusehto $Ax = b$ voidaan kirjoittaa muodossa

$$\begin{bmatrix} A_{\mathcal{B}} & A_{\mathcal{N}} \end{bmatrix} \begin{bmatrix} x_{\mathcal{B}} \\ x_{\mathcal{N}} \end{bmatrix} = b, \tag{2}$$

ja koska $A_{\mathcal{B}}$ oli määritelmänsä mukaan kääntyvä, saadaan

$$x_{\mathcal{B}} = A_{\mathcal{B}}^{-1}(b - A_{\mathcal{N}}x_{\mathcal{N}}). \tag{3}$$

Asettamalla $x_{\mathcal{N}} = 0$, saadaan tehtävälle (1) kantaa $\mathcal{B}$ vastaava ratkaisu $\begin{bmatrix} x_{\mathcal{B}} & 0 \end{bmatrix}^T$, jota sanotaan *kantaratkaisuksi*. Mikäli se totetuttaa tehtävän positiivisuusvaatimuksen, sanotaan lisäksi että kyseessä on *käypä kantaratkaisu*, ja että sitä vastaava kanta on käypä. Käypä kantaratkaisu on aina jokin käyvän alueen muodostavan polyhedraalijoukon kulmapiste.

Jaottelemalla myös tehtävän (1) kohdefunktion c^Tx vektorit indeksijoukkojen avulla ja sijoittamalla $x_{\mathcal{B}}$ kaavasta (3), saadaan

$$c^Tx = \begin{bmatrix} c_{\mathcal{B}}^T & c_{\mathcal{N}}^T \end{bmatrix} \begin{bmatrix} x_{\mathcal{B}} \\ x_{\mathcal{N}} \end{bmatrix} = c_{\mathcal{B}}^TA_{\mathcal{B}}^{-1}b + (c_{\mathcal{N}}^T - c_{\mathcal{B}}^TA_{\mathcal{B}}^{-1}A_{\mathcal{N}})x_{\mathcal{N}}. \tag{4}$$

Määritelmä 2 *Redusoitujen kustannusten vektori on* $\overline{c}^T = c^T - c_{\mathcal{B}}^TA_{\mathcal{B}}^{-1}A$.

Määritelmästä 2 seuraa, että aina pätee $\overline{c}_{\mathcal{B}} = 0$, jolloin redusoitujen kustannusten mieleniintoisin osa, $\overline{c}_{\mathcal{N}}$, on tismalleen muuttujan $x_{\mathcal{N}}$ kerroin kaavassa (4). Koska kantaratkaisussa ei-kantamuuttujien arvo on nolla, voidaan redusoiduista kustannuksista tulkita, miten tehtävän (1) kohdefunktion arvo muuttuisi, mikäli ei-kantamuuttujaa poikkeutettaisiin nollasta positiiviseksi. Erityisesti jos jollekin $s \in \mathcal{N}$ pätee $\overline{c}_s \leq 0$, niin tavoitefunktion c^Tx arvo

pienenee (paranee), mikäli x_s saa positiivisia arvoja. Mikäli ainuttakaan tällaista indeksiä s ei ole olemassa, on löydetty tehtävän optimi, joka on sen hetkistä kantaa $\mathcal{B}$ vastaava kantaratkaisu.

Simplex-algoritmi toimii lähtemällä liikeelle käyvästä kannasta, jolle lasketaan redusoidut kustannukset. Mikäli ne ovat kaikille ei-kantamuuttujille positiivisia, ollaan optimissa. Muutoin valitaan jokin ei-kantamuuttuja x_s, jolla on negatiivinen redusoitu kustannus ja kasvatetaan sen arvoa nollasta niin paljon kuin mahdollista. Rajan kasvulle antaa kaava (3), jonka kaikkien alkioiden on pysyttävä positiivisena, jotta ratkaisu pysyy käyvällä alueella. Kun x_s saa suurimman mahdollisen arvonsa, jokin kantamuuttuja x_j, $j \in \mathcal{B}$, saa arvon 0, jolloin päädytään uuteen kantaratkaisuun, jota vastaava kanta on $\mathcal{B}' = (\mathcal{B} \setminus \{j\}) \cup \{s\})$. Tällöin sanotaan, että x_s on *kantaan tuleva muuttuja* ja x_j on *kannasta lähtevä muuttuja*. Muutosta kannasta toiseen kutsutaan *kannavaihdoksi*.

Simplex-algoritmi etenee käyvästä kannasta toiseen edellä kuvatulla tavalla, kunnes jossain vaiheessa kaikki redusoidut kustannukset ovat positiivisia. Tällöin sen hetkistä kantaa vastaava kantaratkaisu on tehtävän optimi. Voidaan myös sanoa, että kyseinen kanta on optimaalinen. Mikäli käypä alue on rajoittamaton, voi käydä kuitenkin niin, että jotakin ei-kantamuuttujaa voidaan kasvattaa rajattomasti, jolloin tehtävä itsessään todetaan *rajoittamattomaksi*.

Tärkeä käsite, ja usein ongelmia aiheuttava ominaisuus, on käyvän alueen kulmapisteen *degeneroituvuus*. Degeneroituneessa kulmapisteessä osa kantaratkaisun $x_\mathcal{B}$ alkioista on nollassa. Ongelmallisuus seuraa siitä, että tällöin useampi eri kanta vastaa samaa kulmapistettä. Simplex-algoritmi saattaisi tällöin joissain tapauksissa tehdä kannanvaihtoja ainoastaan näiden kantojen välillä, jolloin algoritmi ei koskaan päättyisi. Samojen kantojen välillä kiertämisen estämiseksi on kuitenkin olemassa yksinkertaisia sääntöjä. Näitä ovat esimerkiksi Blandin sääntö tai leksikografiset säännöt, jotka estävät tällaisen ikuisen silmukan syntymisen. Yhden tai useamman kulmapisteen ollessa degeneroitunut sanotaan myös, että tehtävä on degeneroitunut.

Redusoitujen kustannusten ja kantaratkaisun kaavojen lyhentämiseksi määritellään matriisi $\widetilde{A} = A_\mathcal{B}^{-1}A$ ja vektori $\widetilde{b} = A_\mathcal{B}^{-1}b$. Tehtävä on rajoittamaton, mikäli matriisissa $\widetilde{A}$ jonkin ei-kantamuuttujaa vastaavan sarakkeen kaikki alkiot ovat negatiivisia. Degeneroituvuus puolestaan havaitaan siitä, että vektorissa $\widetilde{b}$ osa alkioista on nollassa (Bertsimas ja Tsitsik lis, 1997).

3 Monitavoitteinen lineaarinen ohjelmointi

Lineaarisessa monitavoiteoptimoinnissa kohdefunktioita on yhden sijasta monta, mutta käypä alue on edelleen samaa muotoa kuin tehtävässä (1). Yhden kriteerin sijaan halutaankin siis optimoida useampaa tavoitetta samanaikaisesti ja tällaisesta ongelmasta käytetään termiä MOLP (Multiple Objective Linear Programming). Muuttamalla kaikki kohdefunktiot minimoitaviksi saadaan perusmuotoinen p-tavoitteinen ongelma

$$\begin{array}{ll} \min\limits_x & c_1^T x \\ \vdots & \\ \min\limits_x & c_p^T x \\ \text{s.e.} & Ax = b \\ & x \geq 0 \end{array} \quad \Longleftrightarrow \quad \begin{array}{ll} \min\limits_x & Cx \\ \text{s.e.} & Ax = b \\ & x \geq 0. \end{array} \tag{5}$$

Tehtävä (5) koostuu kohdefunktioista $c_i^T x, i = 1, \ldots, p$, joita kaikkia minimoidaan. Kohdefunktioiden kertoimet on tapana esittää matriisina $C \in \mathbb{R}^{p \times n}$, jonka rivit muodostuvat tavoitevektoreista $c_i \in \mathbb{R}^n$.

LP-tehtävän (1) tapauksessa määriteltiin optimipiste, jossa kohdefunktion arvo minimoituu. Monen kohdefunktion MOLP-tehtävissä optimipisteen käsitteestä on luovuttava, sillä tehtävälle ei voida välttämättä löytää ratkaisua, jossa kaikki kohdefunktiot saavuttaisivat globaalin miniminsä samanaikaisesti. Tietyt käyvän alueen pisteet ovat kuitenkin parempia kuin toiset, ja näitä pisteitä kutsutaan tehokkaiksi pisteiksi. Kirjallisuudessa käytetään usein myös termiä Pareto-optimaalinen piste. Tehokkaan pisteen määritelmä on laajennus optimipisteen käsitteestä.

Määritelmä 3 *Piste $\hat{x} \in \mathcal{X}$ on tehokas, mikäli $\nexists x \in \mathcal{X}$ s.e. $Cx \leq C\hat{x}$ ja $Cx \neq C\hat{x}$.*

Toisin sanoen tehokas piste on sellainen, josta ei ole mahdollista liikkua toiseen käyvän alueen pisteeseen, jossa jokin kohdefunktio paranisi ja loput kohdefunktiot pysyisivät vähintään samana. Olkoon $\mathcal{X}_E$ *tehokas joukko*, joka sisältää kaikki tehokkaat pisteet. Tehokas joukko muodostaa tehtävän (5) mahdolliset ratkaisut rationaaliselle päätöksentekijälle, sillä tehottomista pisteistä voidaan aina siirtyä sellaiseen tehokkaaseen pisteeseen, ainakin yhden kohdefunktion arvo paranee. Se, mikä tehokkaista pisteistä on paras riippuu

päätöksentekijän preferensseistä. Päätöksentekijän kannalta olisi suotavaa, jos tehokas joukko, $\mathcal{X}_E$, olisi tiedossa.

Määritelmä 4 *Piste $\hat{x} \in \mathcal{X}$ on heikosti tehokas, mikäli $\nexists x \in \mathcal{X}$ s.e. $Cx < C\hat{x}$.*

Tehokkaiden pisteiden joukko on heikosti tehokkaiden pisteiden osajoukko. Joissain tapauksissa myös heikosti tehokkaita pisteitä voidaan pitää hyvinä ratkaisuina MOLP-tehtävälle. Koska heikosti tehokkaiden pisteiden joukko monissa tilanteissa on huomattavan suuri, usein on järkevämpää keskittyä käsittelemään ainoastaan tehokasta joukkoa (Miettinen, 1999).

Käyvän alueen kuva lineaarikuvauksen C suhteen, $\mathcal{Y} = C\mathcal{X}$, kertoo mistä joukosta kohdefunktiot saavat arvoja. Sitä avaruutta, jonka osajoukko $\mathcal{X}$ on sanotaan *päätösavaruudeksi* ja se avaruus, josta $\mathcal{Y}$ on osajoukko on nimeltään *tavoiteavaruus.*

Määritelmä 5 *Olkoon $\hat{y} \in \mathcal{Y}$. Mikäli on olemassa $\hat{x} \in \mathcal{X}_E$ siten, että $\hat{y} = C\hat{x}$, niin sanotaan, että $\hat{y}$ on ei-dominoitu.*

Määritelmän 5 vektori $\hat{y}$ kertoo kohdefunktioiden arvot tehokkaassa pisteessä $\hat{x}$. Kaikki ei-dominoidut pisteet muodostavat joukon $\mathcal{Y}_E = C\mathcal{X}_E$. Monissa päätöksenteko-ongelmissa joukko $\mathcal{Y}_E$ on kiinnostavampi kuin $\mathcal{X}_E$, koska tavoitefunktioiden arvot ovat niitä kriteerejä, joita tehtävässä (5) pyritään minimoimaan. Lisäksi tavoiteavaruus on dimensioltaan usein pienempi kuin päätösavaruus, jolloin ratkaisujen esittäminen on tavoiteavaruuden puolella helpompaa. Useimmat kirjallisuudessa esitetyt metodit MOLP-tehtävän ratkaisemiseen keskittyvät kuitenkin ratkaisemaan ensin joukon $\mathcal{X}_E$, sillä MOLP-tehtävän rajoitefunktiot on esitetty päätösavaruudessa.

Lause 1 (Ehrgott, 2005) *Tehtävän (5) tehokas joukko on yhtenäinen ja käyvän alueen reunan osajoukko.*

Koska tehtävän (5) käypä alue muodostuu lineaarisista rajoitteista, on kyseessä polyhedri eli konveksi monitahokas. Sen reuna muodostuu kulmapisteistä ja niiden virittämistä moniulotteisista tahkoista (engl. facet). Lauseen 1 seurauksena tehokkaan joukon karakterisoimiseksi riittää tietää ainoastaan $\mathcal{X}_E$:n kulmapisteet.

Esimerkissä 1 on yksinkertainen MOLP-tehtävä, joka havainnollistaa hyvin tehokkaan pinnan käsitettä sekä sen sijoittumista käyvän alueen reunalle.

Esimerkki 1

$$
\begin{array}{rrrrrr}
\min_{x_1,x_2,x_3} & & -2x_2 & x_3 & := & c_1^T x \\
\min_{x_1,x_2,x_3} & & x_2 & -2x_3 & := & c_2^T x \\
\text{s.e.} & -x_1 & -x_2 & & \leq & -2 \\
& 2x_1 & -x_2 & & \leq & 4 \\
& -\frac{1}{2}x_1 & +x_2 & & \leq & 2 \\
& & & x_3 & \leq & 1 \\
& x_1 & , x_2 & , x_3 & \geq & 0
\end{array}
$$

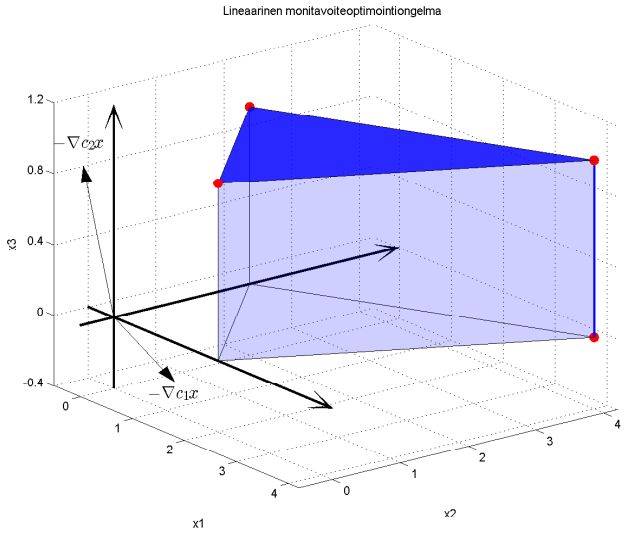
Kuvassa 1 on piirrettynä esimerkin 1 tehtävä päätösavaruudessa ja sen geometrinen ratkaisu. Käypä alue on merkitty vaalealla ja tehokas joukko tummemmalla. Tehokkaan joukon neljä kulmapistettä on merkitty punaisilla palloilla. Eräs tärkeä havainto on, että kaikki käypien pisteiden konveksit kombinaatiot eivät välttämättä ole tehokkaita. Sen sijaan tämän tehtävän tapauksessa $\mathcal{X}_E$ muodostuu kahdesta tahkosta, joista toinen on yksi- ja toinen kaksidimensioinen.

Kuvaan 2 on piirretty vastaavin merkinnöin sama tehtävä tavoiteavaruudessa. Ei-dominoidut kulmapisteet on piirretty puaisilla ympyröillä ja kaikki niiden väliset ei-dominoidut pisteet ovat tummennettuja. Ei-dominoitujen pisteiden määrittäminen tavoiteavaruudessa on geometrisesti helpompaa kuin tehokkaiden pisteiden hahmottaminen päätösavaruudessa. Kuten monissa sovelluksissa, hahmottamista auttaa tavoiteavaruuden pieni dimensio.

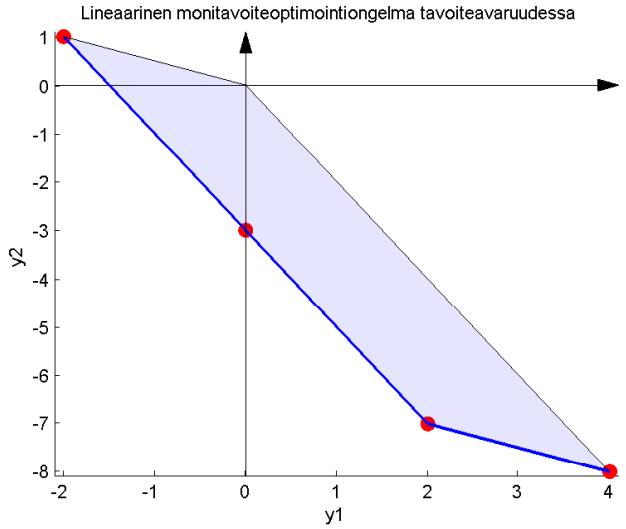
Mikäli käypä alue on rajoittamaton johonkin suuntaan, saattaa LP-tehtävässä käydä niin, että koko tehtävä todetaan rajoittamattomaksi. MOLP-tehtävässä myös rajoittamattomat ratkaisut ovat mahdollisia. Voi nimittäin olla niin, että jostakin tehokkaasta kulmapisteestä on mahdollista liikkua rajattomasti jonkin vektorin suuntaan siten, että kaikki näin määrittyvät pisteet ovat edelleen tehokkaita. Esimerkissä 1 tällaisia rajoittamattomia tehokkaita ratkaisuja syntyisi, jos rajoite, $2x_1 - x_2 \leq 4$, poistettaisiin. Useiden sovellusten kannalta tällaiset ratkaisut eivät kuitenkaan ole mielenkiintoisia, eikä niitä käsitellä tässä työssä, mutta ne voitaisiin helposti ottaa huomioon niin haluttaessa.

3.1 Monitavoitteinen Simplex-algoritmi

Monitavoitteinen Simplex-algoritmi etsii tehokkaan joukon, $\mathcal{X}_E$, kulmapisteet tehtävälle (5). Lineaarisen ohjelmoinnin Simplex-algoritmiin liittyvistä käsitteistä ainoastaan redusoidut kustannukset tarvitsevat laajennetun määritelmän siirryttäessä monitavoitteiseen tapaukseen.



Kuva 1: Kuva esimerkin 1 tehtävästä päätösavaruudessa. Tehtävän tehokas joukko on merkitty tummemmalla sinisellä ja se muodostuu kolmiosta ja janasta. Tehokkaat kulmapisteet ovat näiden kappaleiden kulmat ja ne on merkitty punaisilla palloilla. Kahden tavoitefunktion gradienttien vastavektorit on merkitty nuolilla.



Kuva 2: Kuva esimerkin 1 tehtävästä tavoiteavaruudessa. Ei-dominoitu joukko sijaitsee kuvion alapinnalla ja se on merkitty tummansinisillä janoilla. Kuvan 1 tehokkaita kulmapisteitä vastaavat ei-dominoidut pisteet on merkitty punaisilla palloilla.

Määritelmä 6 *Redusoitujen kustannusten matriisi on* $\overline{C} = C - C_{\mathcal{B}} A_{\mathcal{B}}^{-1} A$.

Matriisi $\overline{C}$ sisältää määritelmän 6 perusteella yksittäisten kohdefunktioiden redusoidut kustannukset riveinään. Lisäksi määritellään matriisi $R = \overline{C}_{\mathcal{N}}$, sillä $\overline{C}_{\mathcal{B}} = 0$, eikä siten ole mielenkiintoinen. Ei-kantamuuttujien redusoidut kustannukset kertovalla matriisilla R on tärkeä rooli monitavoitteisen tehtävän ratkaisemisessa.

MOLP-tehtävä 5 voidaan muuttaa LP-tehtäväksi painottamalla kohdefunktioita positiivisilla painoilla, jolloin tehtäväksi saadaan

$$\begin{aligned} \min_{x} \quad & \lambda^T C x \\ \text{s.e.} \quad & Ax = b \\ & x \geq 0, \\ & \lambda \in \mathbb{R}^p_{>} = \{y \in \mathbb{R}^p | y_i > 0 \, \forall \, i = 1, \ldots, p\}. \end{aligned} \tag{6}$$

Viitataan jatkossa LP-tehtävään (6) merkinnällä LP(λ), missä λ on tehtävän painotusvektori. Tärkeä tulos, joka yhdistää MOLP-tehtävän tehokkaan pisteen ja LP-tehtävän optimin on esitetty lauseessa 2.

Lause 2 (Isermann, 1974) *Käypä ratkaisu* $x^0 \in \mathcal{X}$ *on tehokas ratkaisu tehtävälle (5) jos ja vain jos on olemassa painovektori* $\lambda \in \mathbb{R}^p_{>}$ *siten, että*

$$\lambda^T C x^0 \leq \lambda^T C x, \quad \forall x \in \mathcal{X}. \tag{7}$$

Lauseen 2 perusteella kaikki monitavoitteisen tehtävän (5) tehokkaat ratkaisut ovat tehtävän LP(λ) optimipisteitä jollain painovektorilla λ. Eräs lähestymistapa monitavoitteisten ongelmien ratkaisuun on painottaa kohdefunktioita. Ongelmana kuitenkin on se, että sellaisten painojen löytäminen, joiden avulla saataisiin varmasti etsittyä kaikki tehokkaat pisteet, on käytännössä mahdotonta. Tällöin on tyydyttävä jonkinlaiseen osajoukkoon joukosta $\mathcal{X}_E$. Sitä, kuinka edustava otos koko joukosta saatu osajoukko on, ei voida tietää.

Tehtävän (5) kaikkien tehokkaiden pisteiden ratkaisemiseksi voidaan käydä säännönmukaisesti läpi eri painokertoimia. Lauseen 1 perusteella riittää tutkia ainoastaan käyvän alueen reunaa. Koska käypä alue on lisäksi konveksi monitahokas, reunan määrittävät äärellinen määrä käyvän alueen kulmapisteitä. Täten tehokkaan joukon määrittämiseksikin riittä etsiä ainoastaan tehokkaan joukon tehokkaat kulmapisteet. Samoin kuten Simplex-algoritmissa, käytetään varsinaisten kulmapisteiden sijaan kuitenkin niitä vastaavia kantoja.

Määritelmä 7 *Tehtävän (5) käypää kantaa $\mathcal{B}$ sanotaan tehokkaaksi kannaksi, jos on olemassa $\lambda \in \mathbb{R}^p_>$, jolle pätee $\lambda^T R \geq 0$.*

Olkoon kanta $\mathcal{B}$ tehokas ja olkoon määritelmän 7 mukainen painovektori $\lambda^{\mathcal{B}}$. Tällöin $\mathcal{B}$ on optimikanta tehtävälle LP($\lambda^{\mathcal{B}}$) eli kantaa vastaava kantaratkaisu on tehtävän optimipiste. Tämä johtuu siitä, että painotetun tehtävän LP($\lambda^{\mathcal{B}}$) kaikki redusoidut kustannukset ovat positiivisia ehdon $(\lambda^{\mathcal{B}})^T R \geq 0$ seurauksena, jolloin kyseessä on tehtävän optimi. Tehokasta kantaa vastaava kantaratkaisu on siis lauseen 2 nojalla tehtävän (5) tehokas kulmapiste. Käänteinenkin väite pitää paikkansa: jokaista tehokasta kulmapistettä vastaa vähintään yksi tehokas kanta. Kaikkien tehokkaiden kulmapisteiden määrittämiseksi riittääkin siis etsiä niitä vastaavat tehokkaat kannat.

Kannanvaihdon sanotaan olevan *käypä*, mikäli uutta kantaa vastaava kantaratkaisu on käypä. Lisäksi sanotaan, että kaksi kantaa $\mathcal{B}$ ja $\widehat{\mathcal{B}}$ ovat *vierekkäisiä*, jos kannat eroavat ainoastaan yhden indeksin suhteen, eli $|\mathcal{B} \cap \widehat{\mathcal{B}}| = m-1$.

Määritelmä 8 *Olkoon $\mathcal{B}$ tehokas kanta ja x_j jokin ei-kantamuuttuja. Jos on olemassa $\lambda \in \mathbb{R}^p_>$, jolle on voimassa ehdot $\lambda^T R \geq 0$ ja $\lambda^T r^j = 0$, missä r^j on matriisin R muuttujaa x_j vastaava sarake, niin x_j on tehokas ei-kantamuuttuja.*

Määritelmä 9 *Tehokkaan ei-kantamuuttujan x_j tuomista kantaan käyvällä kannanvaihdolla sanotaan tehokkaaksi kannanvaihdoksi*

Määritelmässä 8 ehdon $\lambda^T r^j = 0$ seurauksena muuttujan x_j redusoitu kustannus on 0 tehtävässä LP(λ). Siten x_j:n tuominen kantaan käyvällä kannanvaihdolla ei muuta redusoitua kustannusta, ja uusi kanta $\widehat{\mathcal{B}}$ on edelleen optimaalinen tehtävälle LP(λ) samalla parametrin λ arvolla.

Lause 3 (Ehrgott, 2005) *Olkoon $\mathcal{B}$ käypä kanta ja x_j tehokas ei-kantamuuttuja. Tällöin mikä tahansa käypä kannanvaihto, jossa x_j tulee kantaan muodostaa uuden tehokkaan ja vierekkäisen kannan $\widehat{\mathcal{B}}$.*

Monen tutkijan (esim. Steuer, 1985; Ehrgott, 2005) esittämä algoritmi kaikkien tehokkaiden pisteiden löytämiseksi perustuu olettamukselle tehokkaiden kantojen kytkeytyneisyydestä. Tällä tarkoitetaan sitä, että mielivaltaisesta tehokkaasta kannasta päästään mihin tahansa toiseen tehokkaaseen kantaan tekemällä ainoastaan tehokkaita kannanvaihtoja. Väite ei kuitenkaan täysin pidä paikkaansa.

Lause 4 (Schechter ja Steuer, 2005) *Olkoon $\mathcal{B}$ tehokas kanta tehtävälle (5) ja $\overline{x}$ jokin tehokas kulmapiste. Lähtien kannasta $\mathcal{B}$ ja suorittamalla tehokkaita kannanvaihtoja, on mahdollista päätyä johonkin kantaan $\mathcal{B}'$, jota*

vastaava kantaratkaisu on $\overline{x}$*.*

Lause 4 ei kumoa väärää oletusta hyödyntäviä algoritmeja, vaan osoittaa niiden olevan oletettua tehokkaampia (Schechter ja Steuer, 2005). Tekemällä ainoastaan tehokkaita kannavaihtoja löydetään lopulta vähintään yksi kanta jokaista tehokasta kulmapistettä kohden. Kuitenkaan kaikkia tehokkaita kantoja ei välttämättä saada selville. Koska kannat eivät ole sellaisenaan mielenkiintoisia, lauseen 4 hyödyntäminen nopeuttaa laskenta-aikoja.

Lauseiden 2-4 avulla voidaan muotoilla monitavoitteinen Simplex-algoritmi, joka etsii käyvän alueen tehokkaat kulmapisteet käymällä läpi järjestelmällisesti tehokkaita kantoja (kts. tarkemmin Ehrgott, 2005). Monitavoitteinen Simplex-algoritmi on kuvailtu kokonaisuudessaan kaaviossa 1.

Vaiheessa 1 algoritmi pyrkii etsimään jonkin käyvän pisteen tehtävälle (5) ratkaisemalla aputehtävän, jossa jokaiselle rajoitteelle annetaan mahdollisuus poiketa yhtäsuuruusrajoitteesta. Näiden positiivisten poikkeamien summaa pyritään minimoimaan ja mikäli ei löydy pistettä, jolla kaikki poikkeamat olisivat nollia, on alkuperäinen käypä alue tyhjä joukko ja algoritmi pysähtyy. Muussa tapauksessa löytyy jokin käyvän alueen piste x^0, jota käytetään seuraavassa vaiheessa.

Vaiheen 2 aputehtävälle ei ole helpposelkoista tulkintaa, sillä se on erään toisen ongelman duaali. Mikäli kyseiselle aputehtävälle ei ole olemassa optimia, voidaan todeta, ettei alkuperäisellä ongelmalla (5) ole tehokkaita pisteitä. Muutoin ratkaisuna saatu vektori $\widehat{w}$ toimii painovektorina tehtävälle LP($\widehat{w}$), josta saatu optimaalinen kanta on tehokas kanta alkuperäiselle tehtävälle. Saatu kanta asetetaan joukoon $\mathcal{L}_1$.

Algoritmin toiminta vaiheessa 3 perustuu lauseisiin 2-4 ja määritelmiin 6-9. Algoritmi ottaa jokaisella iteraatiokierroksella yhden kannan, $\mathcal{B}$, tutkimattomien kantojen joukosta $\mathcal{L}_1$ ja asettaa sen tutkittujen kantojen joukkoon $\mathcal{L}_2$. Tälle kannalle lasketaan tämän jälkeen mahdolliset tehokkaat kannanvaihdot ja niiden seurauksena saatavat uudet kannat. Kaikki sellaiset kannat, joita ei löydy vielä kummastakaan listasta, asetetaan tutkimattomien kantojen joukkoon $\mathcal{L}_1$. Algoritmi jatkaa kantojen tutkimista joukosta $\mathcal{L}_1$, kunnes jossain vaiheessa tutkittavia kantoja ei enää ole. Kaikki algoritmin selvittämät tehokkaat kannat ovat tällöin joukossa $\mathcal{L}_2$, jonka algortimi vaiheen 3 jälkeen palauttaa. Lauseen 4 perusteella joukossa $\mathcal{L}_2$ on vähintään yksi tehokas kanta jokaista tehokasta kulmapistettä kohti.

Vaiheen 3 laskennallisesti työläin osuus on niiden ei-kantamuuttujien selvittäminen, jotka ovat tehokkaita tutkittavana olevalle kannalle $\mathcal{B}$. Sen tekemiseksi ratkaistaan yksitavoitteinen LP-aputehtävä, jonka avulla saadaan tehok-

Kaavio 1 Monitavoitteinen Simplex-algoritmi (Ehrgott, 2005)

Syöte: MOLP-tehtävän (5) määrittelevät matriisit A, b ja C

Alusta: $\mathcal{L}_1 = \emptyset, \mathcal{L}_2 = \emptyset$

Vaihe 1: Ratkaise LP-tehtävä $\min_{x,z}\{e^T z \mid Ax + Iz = b;\ x, z \geq 0\}$. Jos kohdefunktio saa optimissa nollasta poikkeavan arvon, lopeta algoritmi ja totea, että $\mathcal{X} = \emptyset$. Muussa tapauksessa optimipisteen vektori x^0 on käypä kantaratkaisu tehtävälle (5).

Vaihe 2: Ratkaise LP-tehtävä $\min_{w,u}\{u^T b + w^T C x^0 \mid u^T A + w^T C \geq 0;\ w \geq e\}$. Jos tehtävä ei ole käypä, lopeta algoritmi ja totea, että $\mathcal{X}_E = \emptyset$. Muussa tapauksessa olkoon optimiratkaisu $(\widehat{u},\ \widehat{w})$. Etsi LP-tehtävän $\min_x\{\widehat{w}^T C x \mid Ax = b, x \geq 0\}$ optimipisteen kanta $\mathcal{B}$ ja aseta $\mathcal{L}_1 = \{\mathcal{B}\}$.

Vaihe 3:

    while $\mathcal{L}_1 \neq \emptyset$ **do**
        Ota $\mathcal{B} \in \mathcal{L}_1$, aseta $\mathcal{L}_1 = \mathcal{L}_1 \setminus \{\mathcal{B}\}$ ja $\mathcal{L}_2 = \mathcal{L}_2 \cup \{\mathcal{B}\}$
        Laske $\widetilde{A}$, $\widetilde{b}$ ja R kannalle $\mathcal{B}$.
        $\mathcal{EN} = \mathcal{N}$.
        for all $j \in \mathcal{N}$ **do**
            Ratkaise LP $\min_{y,\delta,v}\{e^T v \mid Ry - r^j\delta + Iv = 0;\ y, \delta, v \geq 0\}$.
            Jos LP on rajoittamaton, aseta $\mathcal{EN} = \mathcal{EN} \setminus \{j\}$.
        end for
        for all $j \in \mathcal{EN}$ **do**
            for all $i \in \mathcal{B}$ **do**
                Jos $\mathcal{B}' = (\mathcal{B} \setminus \{i\}) \cup \{j\}$ on käypä, ja $\mathcal{B}' \notin \mathcal{L}_1 \cup \mathcal{L}_2$,
                niin $\mathcal{L}_1 = \mathcal{L}_1 \cup \mathcal{B}'$
            end for
        end for
    end while

Palauta: $\mathcal{L}_2$

kaiden ei-kantamuuttujien joukko $\mathcal{EN}$. Tämän jälkeen, lauseen 3 pohjalta, tutkitaan mahdolliset käyvät kannanvaihdot.

Joukon $\mathcal{L}_2$ kannoista voidaan laskea tehokkaat kulmapisteet joko jo algoritmin suorittamisen aikana, tai jälkikäteen, käyttäen kaavaa (3). Huomionarvoista on, että degeneroituneessa tapauksessa tehokkaiden kantojen määrä on suurempi kuin tehokkaiden kulmapisteiden määrä. Koska kannat itsessään eivät ole kiinnostavia tehtävän ratkaisun kannalta, tekee algoritmi tällöin turhaa työtä. Degeneroituvuutta tulisikin pyrkiä välttämään tehtävän hyvällä formuloinnilla.

3.2 Monitavoitteinen Simplex-algoritmin laskenta-ajan tutkiminen

LP-tehtävän ratkaiseva Simplex-algoritmi on pahimmassa tapauksessa eksponentiaaliaikainen. Tällöin myös monitavoitteinen Simplex-algoritmi on pahimmillaan eksponentiaaliaikainen. MOLP-tehtävän ratkaisemiseen kuluva laskenta-aika on verrannollinen tehokkaiden kulmapisteiden määrään ja sitä kautta tehokkaiden kantojen määrään.

On helppo muodostaa tehtävä, jolla on eksponentiaalinen määrä tehokkaita kulmapisteitä suhteessa muuttujien ja rajoitteiden määrään. Jokaisen kulmapisteen tehokkuus on tutkittava, jolloin tehtävän ratkaisemiseen vaadittu laskenta-aika kasvaa nopeasti liian suureksi. Tämä ongelma ei liity ainoastaan esitettyyn algorimiin, vaan mikä tahansa MOLP-algoritmi joutuu tutkimaan kaikki kulmapisteet. Ehrgott (2005) esittää kirjassaan seuraavan esimerkin

Esimerkki 2

$$\begin{aligned} \min \quad & x_i, \quad i = 1, \ldots, n \\ \min \quad & -x_i, \quad i = 1, \ldots, n \\ \text{s.e.} \quad & x_i \leq 1, \quad i = 1, \ldots, n \\ & -x_i \leq 1, \quad i = 1, \ldots, n. \end{aligned}$$

Esimerkissä 2 minimoidaan kohdefunktioita, jotka osoittavat jokaiseen positiiviseen ja negatiiviseen koordinaattisuuntaan hyperkuution yli $\mathbb{R}^n$:ssä. Selvästi jokainen hyperkuution kulmapiste on tehokas, jolloin tehokkaita pisteitä on yhteensä 2^n kappaletta, vaikka kohdefunktioita sekä rajoitteita on ainoastaan $2n$ kappaletta kutakin. Jo suhteellisen pienillä n:n arvoilla tehtävän ratkaiseminen on ajallisesti erittäin työlästä ratkaista monitavoitteisella Simplex-algoritmilla.

Taulukko 1: *Ohjelmoidun monitavoitteisen Simplex-algoritmin laskenta-aika esimerkin 2 tehtävälle parametrin n eri arvoilla.*

n	7	8	9	10	11	12	13	14
aika (s)	0.51	0.99	2.1	4.7	10.1	23.6	62.9	143
kulmapisteitä	128	256	512	1024	2048	4096	8192	16384

Taulukossa 1 on listattu Matlab-ohjelmistolla toteutetun monitavoitteisen Simplex-algoritmin laskenta-ajat parametrin n eri arvoilla 2.2 GHz kaksiydinprosessorilla varustetulla tietokoneella. Havaitaan, että laskenta-aika enemmän kuin tuplaantuu, kun n kasvaa yhdellä. Selitys tälle on se, että algoritmin vaiheessa 3 ratkaistavien tehtävien määrä tuplaantuu ja samanaikaisesti myös niiden koko kasvaa, mikä lisää laskenta-aikaa. Suurilla n:n arvoilla laskenta-aikaa kasvattaa lisäksi merkittävästi joukkojen $\mathcal{L}_1$ ja $\mathcal{L}_2$ kasvava koko. Koska algoritmissa joudutaan tutkimaan kuuluko tehokas kanta jo joukkoihin $\mathcal{L}_1$ ja $\mathcal{L}_2$, kasvaa tähän tarkistukseen menevä aika joukkojen koon kasvaessa huomattavasti. Etsinnän nopeuttamiseksi ohjelmoitu algoritmi toteutettiin käyttäen Aurovillian ym. (1997) esittämiä tietorakenteita. Hajautustaulukoita käytettiin kantojen tallentamiseen ja järjestämiseen, jonka ansiosta tietyn kannan etsiminen nopeutuu huomattavasti.

Kirjallisuudesta löytyy monia esimerkkejä tehtävän eri parametrien vaikutuksista monitavoitteisen Simplex-algoritmin laskenta-aikaan. Benson (1998) raportoi lyhyesti saamistaan tuloksista satunnaisesti generoitujen tehtävien tehokkaiden pisteiden määrästä. Hänen saamiensa tehokkaiden pisteiden lukumäärät ovat taulukossa 2. Hänen testeissä käyttämässään ADBASE-ohjelmassa (Steuer, 2003) tehokkaiden pisteiden lukumäärän rajana on 200000, jonka jälkeen algoritmi pysähtyy. Taulukosta 2 havaitaan, että tehokkaiden pisteiden määrä riippuu paljon tehtävän koosta ja on todella herkkä eri parametrien, kuten muuttujien sekä rajoitteiden lukumäärälle.

Steuer (1985) on tutkinut tarkemmin eri parametrien muutosten vaikutusta tehokkaiden pisteiden määrään satunnaisissa tehtävissä. Hän raportoi vastaavanlaisista havainnoista kuin Benson. Steuerin tulosten perusteella suurin vaikutus tehokkaiden pisteiden määrään on kohdefunktioiden määrä ja niiden suuntaus. Mikäli kohdefunktioita on monta ja niiden gradientit osoittavat eri suuntiin, on tehokkaiden pisteiden määrä luonnollisesti suurempi, kuin jos kohdefunktioita on vähän, ja niiden suunnassa ei ole paljon vaihtelua. Toisaalta mikäli rajoitefunktioita on hyvin pieni määrä, ei mahdollisia kulmapisteitäkään ole montaa. Hän raportoikin rajoitefunktioiden määrän olevan toisiksi tärkein kriteeri laskenta-ajassa. Myös tehtävän dimensio, eli

Taulukko 2: (Benson, 1998) *Tehokkaiden pisteiden keskiarvo kymmenestä satunnaisesti generoidusita tehtävistä eri parametrien arvoilla. Parametri n on muuttujien määrä ja parametri m on rajoitusten lukumäärä. Jokaisessa tapauksessa kohdefunktiota on 4 kappaletta.*

n	m	Tehokkaiden pisteiden lkm
30	25	7245
50	50	83782
60	50	yli 200000

muuttujien määrä lisää tehokkaiden pisteiden lukumäärää, sillä yhden rajoitteen lisääminen voi muodostaa sitä useamman uuden tehokkaan pisteen, mitä suurempi on tehtävän dimensio.

3.3 Tehokkaiden tahkojen määrittäminen

Monitavoitteisen Simplex-algoritmin avulla saadaan käyvän alueen tehokkaat kulmapisteet. Ne riittävät määräämään koko tehokkaan joukon. Lisätehtäväksi jää ainoastaan määritellä minkälaiset yhdistelyt tehokkaiden kulmapisteiden välillä ovat myös tehokkaita.

Isermann (1977) kuvaa artikkelissaan menetelmän, joka etsii tehokkaat tahkot. Menetelmässä selvitetään aputehtävällä ne tehokkaiden kulmapisteiden maksimaaliset osajoukot, joissa myös pisteiden välinen konveksi kombinaatio on tehokas. Maksimaalinen osajoukko tarkoittaa sitä, että minkä tahansa toisen kulmapisteen lisääminen kyseiseen osajoukkoon rikkoo oletuksen pisteiden konveksien kombinaatioiden tehokkuudesta. Esimerkiksi kuvan 1 tehtävässä kolmionmuotoisen tehokkaan alueen rajaavat kulmapisteet muodostavat tällaisen maksimaalisen joukon. Mikä tahan niiden välinen konveksi kombinaatio on tehokas, mutta lisättäessä joukoon neljäs piste, ei näin enää ole.

Lause 5 (Isermann, 1977) *Olkoon $\mathcal{B}$ tehokas kanta ja $\mathcal{N}^f \subset \mathcal{N}$ niiden ei-kantamuuttujien joukko, jotka on mahdollista tuoda kantaan käyvällä kannanvaihdolla. Olkoon lisäksi $\mathcal{J} \subset \mathcal{N}^f$. Tällöin kaikki indeksijoukon $\mathcal{J}$ mu-*

kaiset ei-kantamuuttujat ovat tehokkaita jos ja vain jos LP-tehtävällä

$$\begin{aligned} \max_{z,\delta,v} \quad & e^T v \\ \text{s.e.} \quad & Rz - R^{\mathcal{J}}\delta + Iv = e \\ & z, \delta, v \geq 0, \end{aligned} \tag{8}$$

on olemassa optimiratkaisu. Tässä yhteydessä $R^{\mathcal{J}}$ tarkoittaa indeksijoukkoa $\mathcal{J}$ vastaavia R:n sarakkeita.

Lauseen 5 tehtävä (8) on tehtävän

$$\begin{aligned} \min_{\lambda} \quad & e^T \lambda \\ \text{s.e.} \quad & R^T \lambda \geq 0 \\ & -R^{\mathcal{J}} \lambda \geq 0 \\ & \lambda \geq e \end{aligned} \tag{9}$$

duaali. Mikäli tehtävällä (8) optimi, tällöin myös tehtävällä (9) on optimiratkaisu. Sen lisäksi, että kaikki indeksijoukkoa $\mathcal{J}$ vastaavat muuttujat ovat tehokkaita, toteuttavat ne tehtävän (9) ehdot samalla muuttujan λ arvolla. Tekemällä tehokkaita kannanvaihtoja nykyisestä kannasta siten, että tuleva muuttuja valitaan indeksijoukosta $\mathcal{J}$, saadaan joukko uusia kantoja, jotka ovat vierekkäisiä nykyiselle kannalle. Koska kaikkia näitä kantoja vastaavat kulmapisteet ovat tehtävän LP(λ) optimiratkaisuja, ovat myös niiden konveksit kombinaatiot tehtävän LP(λ) optimiratkaisuja. Kaikki indeksijoukkoon $\mathcal{B} \cup \mathcal{J}$ kuuluvat tehokkaita kantoja vastaavat kulmapisteet virittävät siis tehokkaan tahkon. Tarkoituksena on kuitenkin etsiä maksimaalisia indeksijoukkoja.

Määritelmä 10 *Sanotaan, että $\mathcal{J}$ on maksimaalinen joukko ei-kantamuuttujien indeksejä, jos ei ole olemassa joukkoa $\mathcal{J}' \subset \mathcal{N}^f$ siten, että $\mathcal{J} \subset \mathcal{J}'$, ja tehtävällä (8) olisi ratkaisu joukolle $\mathcal{J}'$.*

Tehokasta kantaa $\mathcal{B}$ vastaavat maksimaaliset indeksijoukot voidaan löytää lähtemällä liikkeelle monitavoitteisen Simplex-algoritmin vaiheessa 3 löydetyistä tehokkaista kannoista. Ne muodostavat joukon $\mathcal{N}^f$. Tämän jälkeen on mahdollista järjestää kannat puuverkostoksi. Tehokas tapa etsiä kaikki maksimaaliset joukot on esitetty kirjassa Steuer (1985).

Olkoot $\mathcal{B}^\tau$, $\tau = 1, \ldots, t$ tehokkaat kannat, ja $\mathcal{J}^{\tau,\rho}$, $\tau = 1, \ldots, t$, $\rho = 1, \ldots, r$ kaikki maksimaaliset indeksijoukot kannalle $\mathcal{B}^\tau$. Määritellään $\mathcal{Q}^{\tau,\rho} = \mathcal{B}^\tau \cup$

$\mathcal{J}^{\tau,\rho}$, jolloin $\mathcal{Q}^{\tau,\rho}$ sisältää kaikki $\mathcal{B}^{\tau}$:n viereiset kannat, jotka ovat tehokkaita ja samalla tahkolla. Koska kiinnostuksen kohteen on ainoastaan maksimaaliset pinnat, valitaan pienin mahdollinen määrä indeksijoukkoja $\mathcal{U}^1, \ldots, \mathcal{U}^o$, jotka edustavat alkuperäisiä joukkoja $\mathcal{Q}^{\tau,\rho}$ seuraavin säännöin

1. Jokaiselle $\mathcal{Q}^{\tau,\rho}$ on olemassa joukko $\mathcal{U}^s$ s.e. $\mathcal{Q}^{\tau,\rho} \subset \mathcal{U}^s$
2. Jokaiselle $\mathcal{U}^s$ on olemassa joukko $\mathcal{Q}^{\tau,\rho}$ s.e. $\mathcal{U}^s = \mathcal{Q}^{\tau,\rho}$
3. Ei ole olemassa joukkoja $\mathcal{U}^s$ ja $\mathcal{U}^{s'}$, $s \neq s'$, s.e. $\mathcal{U}^s \subset \mathcal{U}^{s'}$

Kukin konstruoitu joukko $\mathcal{U}^s$ vastaa yhtä maksimaalista pintaa tehokkaassa joukossa. Olkoon

$$\mathcal{I}^s = \{\tau \in \{1, \ldots, t\} \mid \mathcal{B}^{\tau} \subset \mathcal{U}^s\}, \quad s \in \{1, \ldots, o\},$$

ja määritellään

$$\mathcal{X}_s = \{x \in \mathcal{X} \mid x = \sum_{\tau \in \mathcal{I}^s} \alpha_\tau x^\tau, \sum_{\tau \in \mathcal{I}^s} \alpha_\tau = 1, \alpha_\tau \geq 0\},$$

missä x^τ on kantaa $\mathcal{B}^{\tau}$ vastaava kantaratkaisu. Joukko $\mathcal{X}_s$ sisältää konveksit kombinaatiot niistä pisteistä, jotka ovat samalla tahkolla. Pistejoukot $\mathcal{X}_s$ ovat tehokkaita ja maksimaalisia, mikäli alkuperäinen MOLP ei ole degeneroitunut (Steuer, 1985). Degeneroituneessa tapauksessa joukot ovat edelleen tehokkaita, mutteivät välttämättä maksimaalisia.

4 Resurssien tehokas allokointi

Tehokkaiden pisteiden määrä näyttäisi satunnaisesti generoiduissa tehtävissä kasvavan hyvin nopeasti liian suureksi. Käytännön ongelmissa on kuitenkin usein jonkinlaista rakennetta, jota satunnaisesti generoiduista tehtävistä puuttuu. Muodostettavat tehtävät saattavat tällöin olla sellaisia, että suuretkin tehtävät olisivat ratkaistavissa kohtuullisessa ajassa. Esimerkkitehtävänä tässä työssä käytetään resurssienallokointitehtävää, jossa on hyödynnetty tehokkuusanalyysiä.

4.1 Tehokkuusanalyysi

DEA (Data Envelopment Analysis; Charnes ym., 1978) on menetelmä, jonka avulla voidaan tutkia ja vertailla esimerkiksi tuotantoyksiköiden tehokkuuksia. Tavallista tehokkuuden mittaria - tuottojen suhdetta käytettyihin resursseihin - ei voida käyttää, koska sekä tuottoja, että resursseja on useaa eri tyyppiä. DEA-mallit yleistävät tehokkuuden käsitteen juuri tällaisiin tilanteisiin.

Tässä yhteydessä esitetään ainoastaan olennaisimmat termit DEA-malleista. Tarkempi johdatus alaan löytyy esimerkiksi kirjasta Cooper ym. (2004). Tutkittavia organisaatioita tai yrityksiä kutsutaan *päätöksentekoyksiköiksi* (DMU, Decision Making Unit). Olkoon tutkittavana joukko päätöksentekoyksiköitä indekseillä $j = 1, \ldots, m$. Jokainen yksikkö kuluttaa $s \in \mathbb{N}$ eri resurssia ja niistä saadaan $n \in \mathbb{N}$ eri tuottoa. Yksikön j resurssien kulutusta merkitään vektorilla $x^j \in \mathbb{R}^s_+$ ja tuottoja vektorilla $y^j \in \mathbb{R}^n_+$. Vektorit muodostavat matriisit $X = [x^1 \ldots x^s] \in \mathbb{R}^{s \times m}$ ja $Y = [y^1 \ldots y^n] \in \mathbb{R}^{n \times m}$. Tässä yhteydessä $\mathbb{R}^p_+ = \{x \in \mathbb{R}^p | x_i \geq 0 \, \forall \, i = 1, \ldots, p\}$.

DEA perustuu päätöksentekoyksiköiden verailuun suhteessa erilaisiin lineaarikombinaatioihin kaikista päätöksentekoyksiköistä. Tässä työssä käytetään DEA-mallia, jossa kaikki lineaarikombinaatiot ovat sallittuja. Pienet ja suuret kauppakeskukset ovat siis keskenään samanarvoisessa asemassa. Tällaista mallia kutsutaan CCR-malliksi tai CRS-malliksi DEA-kirjallisuudessa. Lineaarikombinaatioissa käytettävien painojen joukko CCR-mallissa on

$$\Lambda^{CCR} = \mathbb{R}^m_+.$$

Painot Λ^{CCR} rajaavat kaikkien mahdollisten tuotantoyksiköiden joukon T.

Määritelmä 11 *Mahdollisten tuotantoyksiköiden joukko on*

$$T = \{(x, y) \in \mathbb{R}^s_+ \times \mathbb{R}^n_+ | \exists \lambda \in \Lambda^{CCR} \; s.e. \; x \geq X\lambda, y \leq Y\lambda\}. \qquad (10)$$

Joukko T sisältää kaikki sellaiset kuvitteelliset päätöksentekoyksiköt, jotka saadaan lineaarikombinaationa olemassa olevista yksiköistä CCR-mallin painokertoimilla. Kiinnostavimmat joukkoon T kuuluvat tuotantoyksiköt ovat ne, jotka kuluttavat mahdollisimman vähän resursseja ja jotka tuottavat mahdollisimman paljon. Tällaisten tuotantoyksiköiden muodostamaa joukkoa sanotaan tehokkaaksi rintamaksi, joka on joukon T reunan osajoukko.

Määritelmä 12 *Mahdollisten tuotantoyksiköiden joukon T rajaama tehokas rintama on*

$$Ef(T) = \{(x, y) \in T | \nexists (x', y') \in T \; s.e. \; x' < x, y' > y\}. \tag{11}$$

DEA:ssa päätöksentekoyksiköiden tehokkuutta mitataan etäisyytenä tehokkaasta rintamasta. Usein käytetään mittarina sitä, kuinka paljon resurssien allokointia on vähennettävä tuottojen pysyessä samana, jotta päätöksentekoyksikkö olisi tehokkaalla rintamalla. Vastaavasti voitaisiin pitää resurssien käyttö samana ja tutkia paljonko tuottoja olisi lisättävä. Resurssien vähentämiseen perustuva tehokkuusluku, θ_i, päätöksentekoyksikölle i määritellään lineaarisena optimointitehtävänä

$$\begin{aligned} \theta_i &= \min\{\sigma_i | (\sigma_i x^i, y^i) \in T^f\} \\ &= \min_{\sigma_i, \lambda}\{\sigma_i | \sigma_i x^i \geq X\lambda, y^i \leq Y\lambda, \lambda \in \Lambda^{CCR}\}. \end{aligned} \tag{12}$$

Kaavasta (12) saatu tehokkuusluku kertoo, kuinka paljon resurseja joudutaan skaalaamaan pienemmiksi, jotta päätöksentekoyksikkö olisi tehokkaalla pinnalla. Tehokkuusluvulle pätee $0 < \theta_i \leq 1$, ja yhtäsuuruus toteutuu tehokkailla päätöksentekoyksiköillä. Tehokkuusluvun lisäksi saadaan kaavan (12) optimoinnin seurauksena painojen λ avulla laskettua tehottomille päätöksentekoyksiköille *referenssipiste*, joka sijaitsee tehokkaalla pinnalla.

4.2 Resurssien allokointi

Tehokkuusluvun avulla voidaan vertailla miten eri päätöksentekoyksiköt suoriutuvat suhteessa toisiinsa. Tämän tiedon avulla on siten mahdollista paikallistaa tehottomia yksiköitä ja tehdä toimenpiteitä, joilla huonompien yksiköiden tehokkuutta saadaan parannettua. Muutokset ovat siis yksiköiden sisällä tapahtuvia parannuksia, joilla saadaan tehokkaammin käytettyä olemassa olevat resurssit. Mikäli tällaiset yksiköiden sisäiset muutokset eivät ole mahdollisia, saatetaan kuitenkin pystyä jaottelemaan uudelleen resurssien käyttö yksiköiden kesken, kuten Korhonen ja Syrjänen (2004) ehdottavat artikkelissaan. Tällöin kyseessä on resurssien uudelleenallokointiongelma, jossa resurssit pyritään sijoittamaan sinne, mistä niistä saadaan tehokkaimmat tuotot. Vastaavanlainen ongelma syntyy tilanteessa, jossa tarkoituksena on jakaa lisäresursseja yksiköille mahdollisimman tehokkaasti.

Jotta uudelleenallokointiongelma saadaan esitettyä matemaattisesti, täytyy tehdä oletuksia siitä, miten yksiköiden resurssien käyttöä on mahdollista muuttaa. Siihen liittyen Korhonen ja Syrjänen (2004) esittävät kaksi erilaista mallia. Ensimmäinen malli perustuu oletukseen, että yksikkö voi lisätä ja vähentää tuotantoaan ainoastaan skaalaamalla nykyistä tuotantoa ja resursseja. Toisessa mallissa resurssit puolestaan allokoidaan siten, että kunkin yksikön tehokkuus pysyy samana. Molemmissa tapauksessa uudelleenallokoinnin tavoitteena on parantaa päätöksentekoyksiköiden muodostaman systeemin kokonaistehokkuutta, eli tuottaa kokonaisuutena enemmän.

Taulukossa 3 on esitettynä todellinen tilasto 25 suomalaisen samaan ketjuun kuuluvan kauppakeskuksen käytetyistä resursseista ja tuotoista (Korhonen ja Syrjänen, 2004). Tilannetta tarkastellaan koko kauppaketjun johtajan kannalta, jota kiinnostaa tietää, voidaanko pienemmällä resurssien kokonaiskulutuksella saada suurempia kokonaistuottoja. Toisaalta, jos olisi mahdollista allokoida lisäresursseja tai uudelleenjärjestellä resurssien käyttöpaikat, niin mihin kauppakeskukseen ne kannattaisi sijoittaa, jotta saadut tuotot maksimoituisivat?

Perustuen oletukseen, jossa yksiköt voivat muuttua ainoastaan suhteessa nykyiseen tuotantoonsa, Korhonen ja Syrjänen (2004) johtavat artikkelissaan MOLP-tehtävän

$$
\begin{aligned}
\max_{\hat{x}^i,\hat{y}^i,\delta_i,i=1\ldots m} \quad & \sum_{i=1}^{m} \hat{y}_j^i \qquad j = 1 \ldots s \\
& \hat{x}^i \geq \delta_i x^i, \qquad i = 1, \ldots, m \\
& \hat{y}^i \leq \delta_i y^i, \qquad i = 1, \ldots, m \\
& \hat{x}^i \geq \alpha^i, \qquad i = 1, \ldots, m \\
& \hat{x}^i \leq \beta^i, \qquad i = 1, \ldots, m \\
& \sum_{i=1}^{m} \hat{x}^i \leq r.
\end{aligned}
\tag{13}
$$

Tehtävä (13) ratkaisee resurssienallokointiongelman tehdyin oletuksin. Tehtävän kohdefunktioina ovat eri tuottojen kokonaissummat. Kohdefunktioita on siis yhtä monta kuin tuottoja eli s kappaletta. Päätösmuuttujina ovat kaikkien tuotantoyksiköiden uudet resurssit $\hat{x}^i$, ja tuotot $\hat{y}^i$, sekä kunkin tuotantoyksikön skaalauskerroin δ_i. Tehtävässä olevat vektorit x^i ja y^i ovat päätöksentekoyksikön i havaitut resurssi- ja tuottovektorit. Ensimmäiset kaksi rajoitetta pitävät huolta oletuksesta tuotannon muuttumisesta suhteessa nykyiseen tuotantoon. Koska kyseessä on CCR-malli, kaikki mahdolliset käyvät

Taulukko 3: *Todellinen tilasto 25 suomalaisen kauppakeskuksen käyttämistä resursseista ja tuotoista. (Korhonen ja Syrjänen, 2004)*

	Resurssit		Tuotot	
	Miestyötunnit (10^3 h)	Liikkeen koko (10^3 m^2)	Myynti (10^6 Mk)	Voitto (10^6 Mk)
i	x_1^i	x_2^i	y_1^i	y_2^i
1	79.1	4.99	115.3	1.71
2	60.1	3.3	75.2	1.81
3	126.7	8.12	225.5	10.39
4	153.9	6.7	185.6	10.42
5	65.7	4.74	84.5	2.36
6	76.8	4.08	103.3	4.35
7	50.2	2.53	78.8	0.16
8	44.8	2.47	59.3	1.3
9	48.1	2.32	65.7	1.49
10	89.7	4.91	163.2	6.26
11	56.9	2.24	70.7	2.8
12	112.6	5.42	142.6	2.75
13	106.9	6.28	127.8	2.7
14	54.9	3.14	62.4	1.42
15	48.8	4.43	55.2	1.38
16	59.2	3.98	95.9	0.74
17	74.5	5.32	121.6	3.06
18	94.6	3.69	107	2.98
19	47	3	65.4	0.62
20	54.6	3.87	71	0.01
21	90.1	3.31	81.2	5.12
22	95.2	4.25	128.3	3.89
23	80.1	3.79	135	4.73
24	68.7	2.99	98.9	1.86
25	62.3	3.1	66.7	7.41
Yht	1901.5	102.97	2586.1	81.72

pisteet ovat myös mahdollisten tuotantoyksiköiden joukossa. Tiettyä yksikköä ei käytännön tilanteissa ole mahdollista suurentaa tai pienentää rajattomasti, joten tehtävässä on asetettu ylä- ja alaraja - α^i ja β^i - päätösmuuttujille x^i. Lisäksi viimeisenä rajoitteena on budjettirajoite, missä parametri r määrää allokoitavien resurssien määrän. Mikäli vektori r on ainoastaan jo alunperin havaittujen resurssien summa, kyseessä on ainoastaan resurssien uudelleenjärjestämisongelma.

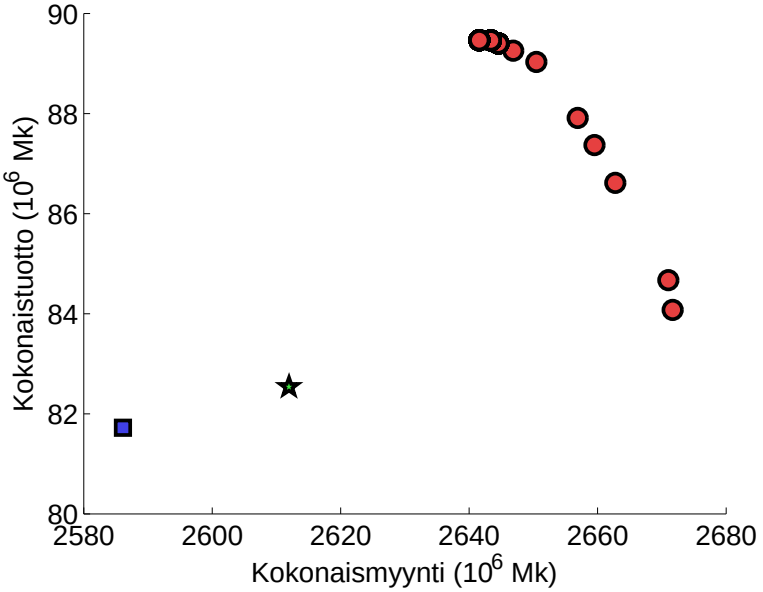
Korhosen ja Syrjäsen esittämässä toisessa mallissa oletetaan, että kunkin tuotantoyksikön tehokkuus pysyy samana, mutta muutoin resursseja saadaan allokoida yksiköiden kesken uudelleen. Tämä malli saadaan myös formuloitua MOLP-tehtävänä korvaamalla tehtävästä (13) kaksi ensimmäistä rajoitetta rajoitteilla

$$\begin{aligned} \theta_i \hat{x}^i &\geq X\lambda_i, \qquad i = 1, \ldots, m \\ \hat{y}^i &\leq Y\lambda_i, \qquad i = 1, \ldots, m \\ \lambda_i &\geq 0, \qquad i = 1, \ldots, m \end{aligned} \tag{14}$$

missä siis päätösmuuttuja δ_i korvautuu vektorilla λ_i. Rajoiteet (14) pitävät huolta siitä, että yksiköt eivät poistu mahdollisten tuotantoyksiköiden joukosta. Lisäksi tehokkuusluvulla θ_i kertominen takaa sen, että yksikön tehokkuus säilyy samana.

4.3 Mallien testaus monitavoitteisella Simplex-algoritmilla

Skaalaukseen perustuvaa resurssien uudelleenallokointitehtävän (13) ratkaisemista tutkittiin taulukon 3 kauppakeskuksille. Tehtävän rajoite- ja resurssiparametreina käytettiin samoja arvoja, kuin Korhonen ja Syrjänen (2004) käyttivät artikkelissaan. Resurssien sallittiin laskevan korkeintaan 10% ($\alpha^i = 0.9x^i$) ja nousevan korkeintaan 30% ($\beta^i = 1.3x^i$). Budjettilisäys oli 1% ($r = 1.01 \sum x^i$). Havaittiin, että tehtävä on ohjelmoidulle algoritmille sellaisenaan liian suuri ratkaistavaksi. Tehtävälle on olemassa kuitenkin ekvivalentti muoto, jossa on vähemmän muuttujia ja vähemmän rajoitteita. Koska jokaisessa tehokkaassa ratkaisussa $\hat{y}^i$-muuttujia sisältävissä rajoitteissa yhtäsuuruus pätee aina, voidaan epäyhtälö muuttaa yhtälöksi. Tällöin $\hat{y}^i$-muuttujat voidaan sijoittaa yhtälöstä kohdefunktioon, jolloin sekä päätösmuuttujien että rajoitefunktioiden kokonaismäärät pienenevät. Lisäksi havaitaan, että optimissa budjetti tulee aina täytettyä kokonaan, joten myös budjettirajoitteen voi muuttaa yhtäsuuruusmuotoiseksi.



Kuva 3: Tehtävän (13) ei-dominoidut pisteet (punaiset ympyrät), alkuperäinen arvo (sininen neliö), ja alkuperäinen arvo 1 %:n lisäyksellä (vihreä tähti).

Uudelleenformuloidun ongelman ratkaisemiseen kului 3.5 GHz neliydinprosessorilla ja 8 GB muistilla varustetulta tietokoneelta aikaa noin tunti, ja sille löytyi 42114 eri tehokasta kantaa. Nämä kannat vastasivat kuitenkin ainoastaan 345 eri tehokasta pistettä, joten tehtävä on voimakkaasti degeneroitunut. Näistäkin pisteistä ainoastaan 67 oli sellaisia, jotka erosivat toisistaan merkittävästi. Tehtävää on kuitenkin hyvin vaikea ellei mahdotonta muotoilla siten, että degeneroituvuus vältettäisiin kokonaan. Ei-dominoidut pisteet, eli kohdefunktioiden arvot eri tehokkaissa ratkaisuissa, on piirretty kuvaan 3. Siitä havaitaan, että erilaisia ei-dominoituja pisteitä on ainoastaan 10 kappaletta. Tällainen ilmiö, missä ei-dominoitujen pisteiden määrä on huomattavasti pienempi kuin tehokkaiden pisteiden määrä on yleinen monissa sovelluksissa (Steuer, 1985, Benson, 1998). Tämä on seurausta siitä, että kohdefunktioiden määrä pysyy useimmiten alle 20 kappaleessa. Mikäli tavoiteavaruuden arvot ovat kiinnostavia, voidaankin todeta, että monitavoitteinen Simplex-algoritmi on laskennallisesti työläs. Suoraan tavoiteavaruudessa toimivat algoritmit (esim. Benson, 1998) saattaisivatkin olla laskennallisesti nopeampia tämänkaltaisissa ongelmissa.

Kuvasta 3 havaitaan myös, että resurssien allokointi on kannattavaa ja tuottaa lisähyötyä. Mikäli neliöllä merkittyyn alkutilanteeseen lisätään tasaisesti

1%:n verran resursseja jokaiselle kauppakeskukselle, päädytään kuvan tähdellä merkittyyn pisteeseen. Lisäämällä sama määrä resursseja, ja uudelleenallokoimalla niiden käyttö, päästään paljon parempaan kokonaismyyntiin ja kokonaistuottoon riippumatta siitä, mikä ei-dominoiduista punaisista palloista valitaan. Kokonaistuottoa on mahdollista parantaa jopa yli 8% suhteessa alkuperäiseen tilanteeseen pelkällä resurssien uudelleensiirtämisellä.

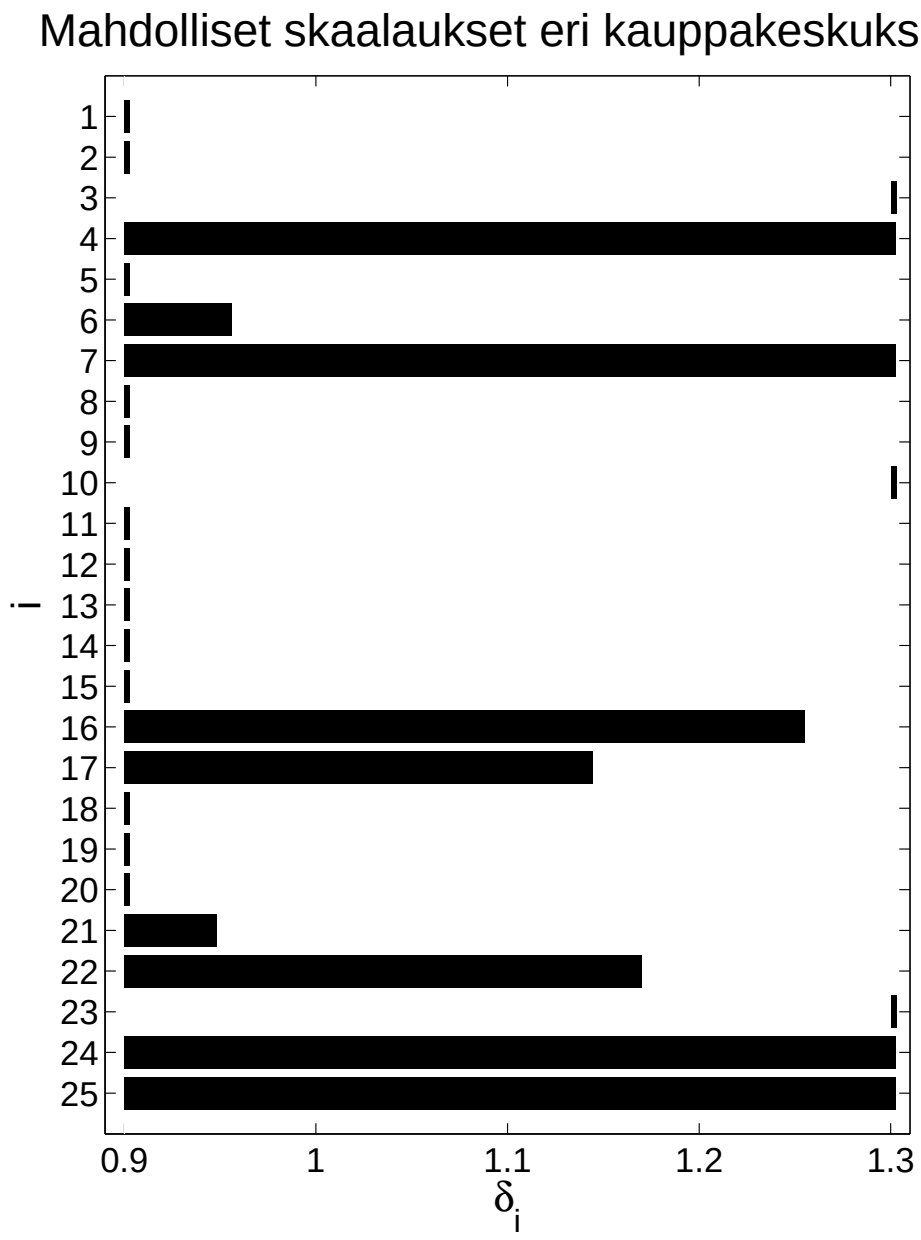
Koska kyseessä oli nykyisen tuotannon skaalaustehtävä, ratkaisussa kiinnostavat myös skaalauskertoimet δ_i. Eräs mielenkiintoinen tieto, joka saadaan ratkaisusta on skaalauskertoimien vaihteluvälit eri ratkaisuissa, jotka on piirretty kuvaan 4. Havaitaan, että tietyille kauppakeskuksille skaalauskerroin on sama jokaisessa tehokkaassa ratkaisussa. Jo tämä tieto kertoo paljon päätöksentekijälle. Eri tavoitteiden tärkeydestä riippumatta voidaan sanoa, että joka tapauksessa kannattaa allokoida resursseja tietyille kauppakeskuksille ja ottaa tietyiltä kauppakeskuksilta resursseja aina pois. Tällaisilla kauppakeskuksilla on kuvassa 4 ainoastaan yksittäinen arvo. Skaalausten alaraja (0.9) ja yläraja (1.3) ovat seurausta käytetyistä parametreista α^i ja β^i.

Monitavoitteisella Simplex-algoritmilla yritettin ratkaista myös tehtävää (14), jossa resursseja allokoidaan pitäen kunkin tuotantoyksikön tehokkuus samana. Se osoittautui kuitenkin liian suureksi tehtäväksi, eikä laskenta tullut valmiiksi useita päiviä kestäneen laskennan jälkeenkään. Myös tälle tehtävälle koitettiin käyttää erilaisia ekvivalentteja muotoja, mutta nekään eivät auttaneet tehtävän ratkaisemisessa.

Koska tehtävässä (14) on muuttujana vektori λ_i jokaista päätöksentekoyksikköä kohti, on jo pelkästään näitä muuttujia kauppakeskusdatalle $25 \cdot 25 = 625$ kappaletta, joka selittää tehtävän ratkaisemisen vaikeuden. Vaihtoehtoinen lähestymistapa olisi laskea tietylle DEA-mallille tehokkaan rintaman kärkipisteet ja muodostaa niiden avulla hypertasojen yhtälöt, jotka rajaavat mahdollisten tuotantoyksiköiden joukon. Näiden hypertasojen avulla voisi korvata rajoitteet (14). Useamman muuttujan tapauksessa pelkästään tehokkaiden muuttujien avulla ei kuitenkaan ole mahdollista muodostaa rajoitteita, jotka määrittäisivät koko tehokkaan rintaman. Tehokkaalle rintamalle kuuluu nimittäin myös heikosti tehokkaita pisteitä.

5 Yhteenveto

Tässä työssä tarkasteltiin monitavoitteisen lineaarisen optimointiongelman täydellistä ratkaisemista eli tehtävän kaikkien tehokkaiden pisteiden määrit-



Kuva 4: Tehtävän (13) tehokkaiden ratkaisujen painojen vaihteluvälit taulukon 3 datalle.

tämistä. Työssä esiteltiin myös kirjallisuudesta tähän tarkoitukseen löytyvän algoritmin toiminta ja siihen liittyvä teoria. Algoritmin toimintaa käytännön tehtävässä tutkittiin resurssienallokoinnin avulla.

Monitavoitteinen optimointi on laskennallisesti työläs jo lineaarisissakin tehtävissä. Tehokkaiden kulmapisteiden lukumäärä kasvaa eksponentiaalisesti tehtävän koon myötä, mikä asettaa rajat kohtuullisessa ajassa ratkaistavissa olevan tehtävän koolle. Tulosten perusteella käytännössä esiintyviä resurssienallokointitehtäviä on kuitenkin mahdollista ratkaista monitavoitteisen Simplex-algoritmin avulla.

Käytännön sovelluksissa päätöksentekijää kiinnostaa usein eri muuttujien mahdolliset vaihteluvälit tehokkaassa joukossa. Tässä työssä selvitettiin vaihteluvälit ratkaisemalla koko MOLP-tehtävä. Tällainen lähestymistapa muuttuu kuitenkin nopeasti epäkäytännölliseksi muuttujien määrän kasvaessa. Vaihtoehtoisina lähestymistapoina ovat myös erilaiset interaktiiviset menetelmät, joissa päätöksentekijä antaa tietoa preferensseistään eri kohdefunktioiden välillä. Tällaista menetelmää on mahdollista käyttää rajaamaan tehokkaiden pisteiden määrää pienemmäksi jo ennen optimointia.

Lukuisten sovellusten kannalta mielenkiintoisin vaihtoehtoinen lähestymistapa suurien MOLP-tehtävien ratkaisemiseksi on kuitenkin tehtävän ratkaiseminen tavoiteavaruudessa päätösavaruuden sijaan. Koska käytännön tapauksissa kohdefunktioiden määrä on usein huomattavasti pienempi, kuin päätösmuuttujien määrä, on käyvän alueen dimensio tavoiteavaruudessa huomattavasti pienempi, kuin päätösavaruudessa. Lisäksi resurssienallokointiongelmassa on monia apumuuttujia, jotka eivät ole tehtävän ratkaisun kannalta mielenkiintoisia, mutta lisäävät turhaan ongelman kokoa. Lisäongelman aiheuttaa se, että useilla erilaisilla apumuuttujien arvoilla saatetaan päätyä samaan pisteeseen varsinaisten muuttujien kohdalla, joka edelleen lisää laskentatyötä tarpeettomasti. Erilaisten menetelmien käyttäminen tehokkuusanalyysissä ja resurssien allokoinnissa vaatii kuitenkin vielä lisätutkimuksia.

Viitteet

Aurovillian, A., Zhang, H., ja Wiecek, M.M. A bookkeeping strategy for multiple objective linear programs. *Computers & operations research*, 24 (11):1033–1041, 1997.

Bazaraa, M. S., Jarvis, J. J., Sherali, H. D., ja Bazaraa, M. S. *Linear programming and network flows*, volume 2. Wiley Online Library, 1990.

Benson, H.P. An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear programming problem. *Journal of Global Optimization*, 13(1):1–24, 1998.

Bertsimas, D. ja Tsitsik lis, J. N. *Introduction to Linear Optimization*. Athena Scientific Series in Optimization and Neural Computation, 6. Athena Scientific, Belmont, MA, 1997.

Charnes, A., Cooper, W.W., ja Rhodes, E. Measuring the efficiency of decision making units. *European Journal of Operational research*, 2(6):429–444, 1978.

Cooper, W.W., Seiford, L.M., ja Zhu, Joe. *Handbook on Data Envelopment Analysis*. Kluwer Academic Publishers, Boston, 2004.

Ehrgott, M. *Multicriteria Optimization*. Springer Verlag, Boston, Dordrecht, London, second edition, 2005.

Ferguson, A. R. ja Dantzig, G. B. The allocation of aircraft to routes-an example of linear programming under uncertain demand. *Management Science*, 3(1):45–73, 1956.

Hung, Y. F. ja Leachman, R. C. A production planning methodology for semiconductor manufacturing based on iterative simulation and linear programming calculations. *Semiconductor Manufacturing, IEEE Transactions on*, 9(2):257–269, 1996.

Isermann, H. Proper efficiency and the linear vector maximum problem. *Operations Research*, 22(1):189–191, 1974.

Isermann, H. The enumeration of the set of all efficient solutions for a linear multiple objective program. *Operational Research Quarterly (1970-1977)*, 28(3):711–725, 1977.

Korhonen, P. ja Syrjänen, M. Resource allocation based on efficiency analysis. *Management Science*, 50(8):1134–1144, 2004.

Lustig, Irvin J., Marsten, Roy E., ja Shanno, David F. Interior point methods for linear programming: Computational state of the art. *ORSA Journal on Computing*, 6(1):1, 1994.

Miettinen, K. *Nonlinear Multiobjective Optimization*. International Series in Operations Research & Management Science. Kluwer Academic Publishers, Boston, 1999.

Schechter, M. ja Steuer, R.E. A correction to the connectedness of the Evans-Steuer algorithm of multiple objective linear programming. *Foundations of Computing and Decision Sciences*, 30(4):351–360, 2005.

Steuer, R. E. *ADBASE: A Multiple Objective Linear Programming Solver for All Efficient Extreme Points and All Unbounded Efficient Edges*. Terry College of Business, University of Georgia, Athens, 2003.

Steuer, R.E. *Multiple Criteria Optimization: Theory, Computation, and Application*. John Wiley & Sons, New York, NY, New York, NY, 1985.