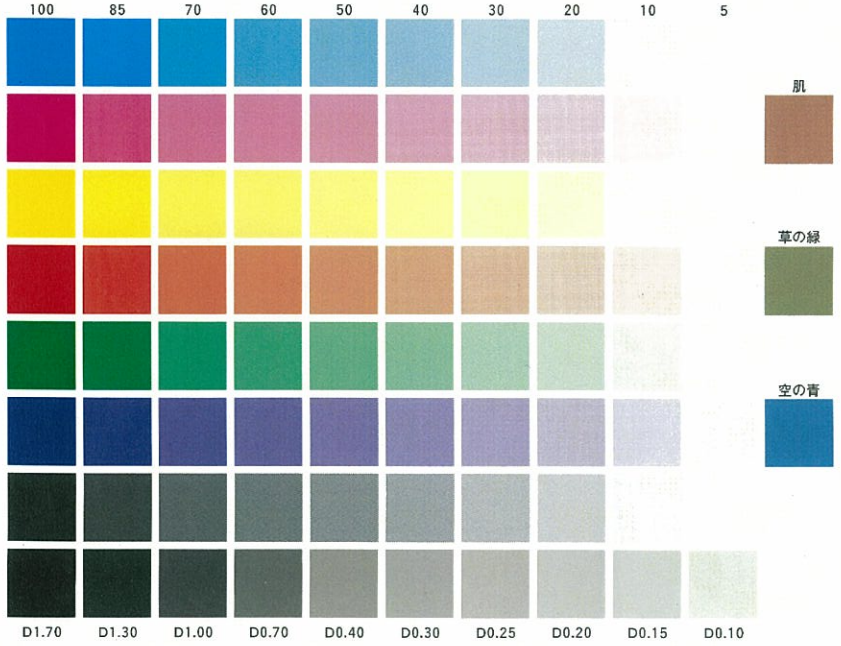


代理制約法における代理乗数決定法とその改良に関する研究

２００５年９月

木　村　作　郎

代理制約法における代理乗数決定法とその改良に関する研究

論文要旨

組合せ最適化(combinatorial optimization)は数理計画の重要な分野のひとつであり，組合せ的な・離散的な制約条件のもとで，離散的な値をとる目的関数を最小あるいは最大にするという最適化を扱う．組合せ最適化問題の具体例としては，整数計画法(integer programming problem)，最短路問題，ナップザック問題(knapsack problem)，スケジューリング問題，最大フロー問題，巡回セールスマン問題，集合被覆問題，施設配置問題など多岐に亘る．この種の問題は，オペレーションズ・リサーチ，システム工学，コンピュータサイエンスなどの理工学から，数理経済学，社会学や心理学に至る広範な領域で見られる．この組合せ最適化問題は，連続変数の最適化手法との間で本質的な相違がある．連続性や微分の概念にのっとった古典的最適化手法を直接利用することができず，列挙法的なアプローチをとらざるを得ない．したがって，組合せ最適化問題は，一般に変数や制約条件式の個数が多くなれば解くことが困難になる．

本論文においては，組合せ最適化問題の解法の一つである代理制約法(surrogate constraint method)を取り上げる．代理制約法は，代理乗数を用いることで原問題の複数制約条件式を単一条件式とした代理問題に変換して解く手法である．原問題が準凸であるときは，複数の制約条件式に乗ずる代理乗数を適当に決定すれは，代理問題の最適解は原問題の最適解と一致することが示されている．しかし，離散問題を緩和して考えた代理双対問題は，代理双対ギャップ(surrogate duality gap)が存在することが多い．この場合には，代理問題の最適解は一般に原問題の最適解と一致せず，原問題の実行可能解になるとは限らない．単に原問題の目的関数の上限値を与えるだけである．最近，仲川は，この代理双対ギャップを閉じ，原問題の厳密解を求めることができる改良代理制約法（ISC 法，Improved Surrogate Constraint Method）を提案した．この ISC 法により制約条件式が 5，6 個で，1000 変数規模の変数分離型の非線形整数計画問題を厳密に解くことが可能となった．

第１章では，序論として，組合せ最適化問題に関する従来の研究の状況を述べ，本研究の位置づけを明確にし，かつ本研究の意義及び研究内容の概要について述べている．

組合せ最適化問題を解くとき，その最適解の上界あるいは下界の限界値(bound)があらかじめ得られていれば，探索を制御するための有効な情報となる．代理制約法を用いて整数計画法を解くときの限界操作において重要な役割を担うのは，その問題の上界値(upper bound)あるいは下界値(lower bound)である．したがって，組合せ最適化問題を効率的に解くには，より良い上界値を求める方法が必要である．この上界値あるいは下界値を求めるための一般的な方法は，変数が整数であるという条件を除いてできあがった線形計画問題，すなわち線形緩和問題を解くことにより求められる．Sinha and Zoltners は，代理制約法

において，線形緩和問題の最適解であるＬＰ解をもとに，より良い限界値を計算する方法を用いている．

第２章においては，組合せ最適化問題のうちの非線形ナップザック問題を解くための，より効率の良いアルゴリズムを構築するために，Sinha and Zoltners よりも更によい上界値を計算する方法を提案している．その方法の有効性を確かめるために，簡単な例題及び実用規模の非線形ナップザック問題を解き，Sinha and Zoltners の方法による上界値と比較，検討し，提案した新上界値がよりよいことが示されている．

非線形整数計画法に対する厳密解法である ISC 法を効率的に実行して，実用的な時間内で解を得るためには，その中に含まれる最適代理乗数の決定アルゴリズムにより最適な代理乗数を求めることが必要である．しかし，制約条件式の個数が多いとき，最適な代理乗数を決定するためのアルゴリズムにおいて，計算時間が膨大になり，さらに計算時に使用する作業領域の量が多大になるため，最適な代理乗数が実用的な時間内で得られないことがある．そこで，第３章においては，代理制約法あるいは ISC 法における最適な代理乗数を決定するアルゴリズムとして，Dyer アルゴリズムと仲川の COP(Cutting-Off Polyhedron) アルゴリズムとを取り上げている．計算機実験を通して両アルゴリズムが比較，検討され，これら代理乗数決定法の長所と短所が明らかにされている．さらに，第３章では Dyer アルゴリズムの短所を改善する改良 Dyer アルゴリズムが提案され，計算機実験により，その有効性が明らかにされている．この改良 Dyer アルゴリズムは，代理乗数を決定する過程において，内接する球の半径の大きさの決定に全く関係していない切断面，及び関係しているがその割合が小さい切断面を削除する改良である．この改良により，代理乗数を計算するときに現れる切断面の候補数が減少するため，使用する作業領域（メモリ）量が減少し，計算時間も短縮される．

第４章では，改良 Dyer アルゴリズムにおいて用いられるパラメータである RP(Reducing Plane) 率，及び RP 周期による，最適解を求めるための計算時間，及び実行回数への影響が明らかにされている．さらに，改良 Dyer アルゴリズムによる代理乗数を用いた代理制約法あるいは ISC 法により，大規模な分離型非線形整数計画問題を解くために有効となるＲＰ率とＲＰ周期の値の組合せも示されている．

第５章では，代理双対ギャップがあるために，既存の解法では厳密解が求められなかったシステムの信頼性設計問題に ISC 法を適用して，その厳密解が求められている．解かれた問題は，２制約式を持つ直列システムの信頼性最適化問題，４制約式を持つ直列システムの信頼性最適化問題，及び変数の数が多い４制約式を持つ直列システムの信頼性最適化問題である．その際，第２章で提案した新上界値，及び第３章で提案した改良 Dyer 法による代理乗数を用いることにより，本論文で提案された新上界値の決定法及び改良 Dyer 法が有用であることが明らかにされている．

第６章では，結論として本研究の全般的な総括を行い，得られた主な成果がまとめられている．

目　次

第1章　序　論

1．1　研究の背景と目的

与えられた制約条件のもとで，ある目的関数を最大あるいは最小にするという最適化問題として数理計画法は，1940 年代に G.B.Dantzig による線形計画法のシンプレックス法の提案以来，意欲的に研究され，コンピュータの進歩とともに発展してきている．

組合せ最適化(combinatorial optimization)は数理計画法の重要な分野のひとつであり，組合せ的な制約条件のもとで，離散値をとる目的関数を最小あるいは最大にするという最適化を扱う．この組合せ最適化を取扱う組合せ最適化問題と，離散最適化問題(discrete optimization problem)，組合せ計画問題(combinatorial programming problem)，離散計画問題(discrete programming problem)なども，ほぼ同じ意味で用いられる．組合せ最適化問題の具体例としては，整数計画問題(integer programming problem)，最短路問題，ナップザック問題(knapsack problem)，スケジューリング問題，最大フロー問題，巡回セールスマン問題，集合被覆問題，施設配置問題など多岐に亘る．この種の問題は，オペレーションズ・リサーチ，システム工学，コンピュータサイエンスなどの理工学から，数理経済学，社会学や心理学に至る広範な領域で見られる．この組合せ最適化問題は，連続変数の最適化手法との間で本質的な相違がある．連続性や微分の概念にのっとった古典的最適化手法を直接利用することができず，列挙法的なアプローチをとらざるを得ず，一般に変数や制約条件式の個数が大きくなれば解くことが困難になる．

この中の整数計画法は，1958 年 R. Gomory[1]により，線形計画法を用いた緩和法の一種である切除平面法(cutting plane method)を用いて解かれた．しかし，この切除平面法は，小型の問題に対しても極めて大量の計算が必要であり，計算誤差のため収束しない例もたくさん報告され，実用性に欠けることになった． A.H. Land and A.G. Doig[2] は，1960 年に列挙法の一種である分枝限定法(branch-and-bound method)を発表した．その後，R. Dakin[3] や E. Beale and R. Small [4] らにより改良が加えられ，一般的な問題を解く汎用アルゴリズムであるコマーシャルコードに採用されるまでになった．また，1965 年 E. Balas[5]が 0-1 変数問題に対する列挙法である部分列挙法を考案し，F.Glover[7]や A. Geoffrion et al.[8]の改良を経て，大型の集合分割問題などに適用され成功を収めた．数理計画法において，Glover[7]によって初めて提案された代理制約法(surrogate constraint method)は，原問題の複数制約条件式を代理乗数(surrogate multiplier)を用いて，代理制約(surrogate constraint)と呼ばれる新しい単一制約式に変換する手法である．限界値を計算するためにより多くの努力が必要ではあるが，複数の制約条件式を有する多次元ナップザック(multidimensional knapsack problem.MKP)問題を解くには， Lagrange 緩和法よりも代理制約法は，より有効であることが示された[18]．Greenberg and Pierskalla[9]は，一

般的な数理計画問題において代理制約法をはじめて適用し，Glover[7,10], Karwan and Rardin[11]および Dyer[15]は，この代理制約法に関する多くの研究を行った．一般的な整数計画問題に対する代理乗数を求める方法は， Karwan and Rardin[11-13],Karwan et al.[14],Dyer[15],Glover[16]および仲川ら[17]によって提案されている．

複数の制約条件式を有する多次元非線形整数計画問題を効率的に解くことは近年ますます重要になっている．上述の通り，複数個の制約条件のもとで，目的関数の値を最大化あるいは最小化する問題は，あらゆる分野で発生し適用範囲が広いからである．その反面，この問題を解くには列挙法的なアプローチをとらざるを得ず，変数の個数が大きくなれば一般的に容易ではない．多次元非線形整数計画問題は次のように定式化される．

$$
\begin{aligned}
[P] \quad & \text{Maximize} \quad f(\mathbf{x}) \\
& \text{subject to} \quad \mathbf{g}(\mathbf{x}) \leq \mathbf{b}, \\
& \mathbf{x} \in \mathbf{X},
\end{aligned}
$$

ここで，$\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$ は，決定空間 $\mathbf{X}$ での N 次元整数値変数ベクトル，$f(x)$ は整数値目的関数，$\mathbf{g}(\mathbf{x}) = (g_1(x), g_2(x), \ldots, g_M(x))^T$ は M 次元整数値ベクトル制約関数，$\mathbf{b} = (b_1, b_2, \ldots, b_M)^T$ は M 次元整数値ベクトル制約許容量，$\mathbf{X}$ はN次元離散決定空間である．

本研究では，組合せ最適化問題の解法の一つである代理制約法(surrogate constraint method)を取上げる．代理制約法は，代理乗数を用いることで原問題の複数制約条件式を単一条件式とした代理問題に変換して解く手法である．原問題が準凸であるときは，複数の制約条件式に乗ずる代理乗数を適当に決定すれば，代理問題の最適解は原問題の最適解と一致することが示されている．しかし，離散問題を緩和して考えた代理双対問題は，代理双対ギャップが存在することが多い．この場合，代理双対問題の最適解は一般に原問題の最適解と一致せず，原問題の実行可能解になるとは限らない．このときは，単に原問題の目的関数の上限値を与えられているだけである．最近，仲川は，この代理双対ギャップを閉じ，原問題の厳密解を求めることができる改良代理制約法（ISC 法，Improved Surrogate Constraint Method）を提案した[25,26]．この ISC 法により制約条件式が 5，6 個で，1000 変数規模の変数分離型の非線形整数計画問題を厳密に解くことが可能となった．

組合せ最適化問題を解くとき，その最適解の近似値や上界値（あるいは下界値）があらかじめ得られていれば，探索を制御するための有効な情報となる．したがって，より良い上界値を求める方法が必要である．

また，整数計画問題を代理制約法あるいは ISC 法を用いて解くとき，代理乗数の決定が最適解を求める際の重要な要因となる．代理乗数の代表的な決定法として，Dyer アルゴリズム[27]や COP(Cutting-Off Polyhedron) アルゴリズム[28]が挙げられる．しかし，これらの代理乗数決定方法の長所，短所に関する研究はほとんど行われていない．代理制約法

をより効率的に解くためには，それらを明らかにする必要がある．さらに，代理制約法の改良に関する研究は行われているが，代理乗数を決定する方法がもつ欠点の改良についてはほとんど研究が行われていない．上記の決定方法の短所を改善し，代理制約法あるいはＩＳＣ法を用いて整数計画問題を解くときに，代理乗数をより速く，より正確に求めて，効率的に最適解を求める必要がある．

１．２　本論文の構成

離散最適化問題を解く際，たとえば代理制約法を用いて整数計画法を解くときの限界値操作において重要な役割を担うのは，その問題の上界あるいは下界の限界値(bound)である．したがって，組合せ最適化問題を効率的に解くには，より良い上界値(upper bound)を求める方法が必要である．限界値を求めるための一般的な方法は，変数が整数であるという条件を除いてできあがった線形計画問題，すなわち線形緩和問題を解くことにより求める．Sinha and Zoltners[24]は，線形緩和問題の最適解であるＬＰ解をもとに，より良い限界値を計算する方法を用いている．そこで，第２章においては，離散最適化問題のうちの非線形ナップザック問題を解くために，より効率の良いアルゴリズムを構築するために，Sinha and Zoltners よりも更によい上界値を計算する方法を提案する．その方法の有効性を確かめるために，簡単な例題及び実用規模の非線形ナップザック問題を解き，Sinha and Zoltners の方法による上界値と比較・検討し，提案した新上界値がよりよいことを示す．

多次元非線形整数計画問題に代理制約法を適用した場合，代理双対問題に代理双対ギャップが存在することが多く，その場合には代理問題の最適解は一般に原問題の最適解に一致せず，実行可能解になるとは限らない．このときは，単に原問題の目的関数の上界値を与えるだけである．非線形整数計画法に対する厳密解法である ISC 法を効率的に実行して実用的な時間内で解を得るためには，その中に含まれる最適代理乗数決定アルゴリズムにより最適な代理乗数を求めることが必要である．しかし，この方法においても，制約条件式の個数が多いとき最適な代理乗数を決定するためのアルゴリズムにおいて，計算時間が膨大になり，計算時に使用する作業領域の量が多大になるため，最適な代理乗数が実用的な時間内で得られないことがある．そこで，第３章において，代理制約法あるいは ISC 法における最適な代理乗数を決定するアルゴリズムとして，Dyer アルゴリズム[27]と仲川の COP アルゴリズム[28]とを取り上げ，計算機実験を通して両アルゴリズムを比較検討し，これらの長所と短所を明らかにする．さらに，本論文では Dyer アルゴリズムの短所を改善する改良 Dyer アルゴリズムを提案し，計算機実験により，その有効性を明らかにする．

この改良 Dyer アルゴリズムは，代理乗数を決定する過程において，内接する球の半径の大きさの決定に全く関係していない切断面，及び関係しているがその割合が小さい切断面を削除する改良である．この改良により，代理乗数を計算するときに現れる切断面の候補

数が減少するため使用メモリ量が減少し，計算時間も短縮される．

第４章において，提案された改良 Dyer アルゴリズムにおいて用いられるパラメータ（RP 率及び RP 周期）による最適解を求めるための実行時間，及び実行回数への影響を明らかにする．さらに，大規模な分離型非線形整数計画問題を解くために有効となるＲＰ率とＲＰ周期の値の組合せも示す．

第５章では，代理双対ギャップがあるために，既存の解法では厳密解が求められなかったシステムの信頼性設計問題に ISC 法を適用して，その厳密解を求める．解かれた問題は，２制約式を持つ直列システムの信頼性最適化問題，４制約式を持つ直列システムの信頼性最適化問題，及び変数の数が多い４制約式を持つ直列システムの信頼性最適化問題である．その際，第２章で提案される新上界値，及び第３章で提案される改良 Dyer 法による代理乗数を用いることにより，本論文で提案した新上界値の決定法及び改良 Dyer 法が有用であることを示す．

最後に，第６章において，結論として本研究の全般的な総括を行い，得られた主な成果をまとめる．

本研究で提案した，新上界値計算法を用いた「よりよい」上界値，及び改良 Dyer 法による最適代理乗数を用いて，代理制約法あるいは ISC 法を組合せ最適化問題に適用すれば，実用規模の，及び大規模な問題に対して，より効率的に最適解が求められることが明らかにされる．

参考文献

[1] R.E. Gomory, " Outline of an Algorithm for Integer Solution to Linear Programs," Bulletin of American Mathematical Society, Vol.64, pp.275−278,1958.

[2] A.H. Land and A.G. Doig, " An automatic method for solving discrete planning problems," Econometrica, Vol.28, pp.497−550, 1960.

[3] R. Dakin, " A tree-search algorithm for mixed integer programming problems," Computer Journal, Vol.8. pp.250−255, 1965.

[4] E.M.L. Beale and R.E. Small," Mixed integer programming by branch and bound technique," Proc. IFIP Congress, W. Klerich et. al. eds., 2, Spartan Press, 1965.

[5] E. Baker, "An additive algorithm for solving linear programs with zero-one variables," Operations Research, Vol.13, pp.517−546, 1965.

[6] F. Glover, "A multiphase-dual algorithm for the zero-one integer programming problem, Operations Research, Vol.13,pp.879−919, 1965.

[7] F. Glover, "Surrogate constraints," Operations Research, Vol.16, pp.741-749, 1968.

[8] A.M. Geoffrion, "An improved implicit enumeration approach for integer programming," Operations Research, Vol.17, pp.437−454, 1969.

[9] H. Greenberg, W. Pierskalla, "Surrogate mathematical programs," Operations Research, Vol.18, pp.924−939, 1970.

[10] F. Glover, "Surrogate constraints duality in mathematical programming," Operations Research, Vol.23, pp.434−451, 1975.

[11] M.H. Karwan, R.L. Rardin, "Some relationships between Lagrangian and surrogate duality in integer programming," Mathematical Programming, Vol.17, pp.320−334, 1979.

[12] M.H. Karwan, R.L. Rardin, "Searchability of the composite and multiple surrogate dual functions," Operations Research, Vol.28, pp.1251−1257, 1980.

[13] M.H. Karwan, R.L. Rardin, "Surrogate dual multiplier search procedures in integer programming," Operations Research, Vol.32, pp.52−69, 1984.

[14] M.H. Karwan, R.L. Rardin, S. Sarin, "A new surrogate dual multiplier search procedure," Naval Research Logistics, Vol.34, pp.431−450, 1987.

[15] H.E. Dyer, "Calculating surrogate constraints," Mathematical Programming, Vol.19, pp.255−278, 1980.

[16] F. Glover, "Tutorial on surrogate constraint approaches for optimization in graphs, "Working paper, Hearin Center for Enterprise Science, University of Mississippi, USA,

2001.
[17] 仲川 勇二、疋田 光伯. 鎌田 弘. "代理双対問題を解くためのアルゴリズム". 電子情報通信学会論文誌. Vol. J67-A. No. 1. pp.53－59. Jan. 1984.
[18] A.Freville, "The multidimensional 0-1 knapsack problem: An overview", European Journal of Operational Research, Vol.155, pp.1－21,2004.
[19] K.M. Bretthauser and B. Shetty, " The nonlinear knapsack problem— algorithms and applications" , European Journal of Operational Research, Vol.138,pp.459－472,2002.
[20]今野　浩，鈴木久敏，「整数計画法と組合せ最適化」，日科技連出版社，1993.
[21]西川禅一，三宮信夫，茨木俊秀，「最適化」，岩波講座　情報科学 19，岩波書店，東京，1982.
[22]坂和正敏，「離散システムの最適化」，森北出版，東京，2000.
[23]茨木俊秀，「組合せ最適化－分枝限定法を中心として－」，講座・数理計画法 8，産業図書，東京，1983.
[24]P.Sinha and A.Zoltners,"The Multiple － Choice Knapsack Problem ",Oper. Res,Vol.27,p.125－131,1979.
[25] Y. Nakagawa, "A reinforced surrogate constraints method for separable nonlinear integer programming", RIMS, Kokyuroku 1068 Kyoto Univ.,pp.194－202, 1998.
[26] Y. Nakagawa, "An improved surrogate constraints method for separable nonlinear integer programming", Journal of the Operations Research Society of Japan, Vol.46, No.2, pp.145－163, 2003.
[27] M.E. Dyer, "Calculating surrogate constraints", Math. Program., Vol.19, pp.255－278,1980.
[28] 仲川 勇二. 疋田 光伯. 鎌田 弘. "代理双対問題を解くためのアルゴリズム". 電子情報通信学会論文誌. Vol. J67－A No. 1. pp.53－59. Jan. 1984.
[29] S. Martello and P. Toth, Knapsack Problems Algorithms and Computer Implementations, John Wiley & Sons, New York, 1990.
[30] T. Ibaraki and N. Katoh, Resource Allocation Problem Algorithmic Approaches, MIT Press, London,1988.

第２章　非線形ナップザック問題における新上界値計算法[13,14]

２．１　概説

決定すべき空間が離散であるような最適化問題は離散最適化問題と呼ばれる．離散最適化問題を効率よく解くためには，効率のよい限界値操作が必要となる．

離散最適化問題を解く際の限界値操作において重要な役割を担うのは，その問題の上界あるいは下界の限界値(bound)である．限界値を求めるための一般的な方法は，変数が整数であるという条件を除いて，できあがった線形計画問題，すなわち線形緩和問題を解くことである．Sinha and Zoltners[4]は線形緩和問題の最適解であるＬＰ解をもとに，よりよい限界値を計算する方法を用いている．

本論文においては，離散最適化問題のうちの非線形ナップザック問題を解くために，モジュラ法(Modular Approach .MA)[1]に基づく，より効率の良いアルゴリズムを構築するために，Sinha and Zoltners[4]が用いた方法よりも更によい上界値(upper bound)を計算する方法を提案する． 文献[6]は変数を1つ固定することによってよい上界値を求めているが，本手法は複数の変数を固定して，更によい上界値を求めようとしている．

非線形ナップザック問題の具体例としては，(1)Bretthauser and Shetty [7]によって名づけられた非線形資源配分問題(nonlinear resource allocation problem)，(2)資源配分問題（例えば，Ibaraki and Katoh [8]），(3)多重選択ナップザック問題[9]，(4)分散コンピュータシステムにおけるプロセッサ及びデータベース割当[10]，(5)プロジェクト選択と積荷作業問題[11],(6)カッティングストック問題 [12]，(7)信頼性問題などがある．

2．2　非線形ナップザック問題

単一制約をもち，分離可能な非線形離散最適化問題は，非線形ナップザック問題

$$[P^0]: \textit{Maximize}\ \ f^0(\mathrm{x}) = \sum_{i=1}^{n} f_i^0(x_i) \qquad (2.1)$$

$$\textit{subject to}\ \ g^0(\mathrm{x}) = \sum_{i=1}^{n} g_i^0(x_i) \le b^0$$

$$x \in K_i^0 \quad \textit{for}\ \ i \in N$$

(nonlinear knapsack problem) と呼ばれる.

ここで,

$$\mathrm{x} = (x_1, x_2, \ldots, x_n), \qquad K_i^0 = \{1,2,\ldots,k_i^0\}, \quad N = \{1,2,\ldots,n\}$$

である．一般性を失うことなしに,

$$f_i^0(x_i) \ge 0 \quad \textit{for} \quad x_i = 1,\ldots,k_i^0, \quad i = 1,\ldots,n$$

$$g_i^0(x_i) \ge 0 \quad \textit{for} \quad x_i = 1,\ldots,k_i^0, \quad i = 1,\ldots,n$$

を仮定する.

仲川［1］は，1 制約の離散最適化問題を解くための新解法としてモジュラ法 (Modular Approach, MA) を提案した．モジュラ法は，次のステップ (1) と (2) を繰り返して実行する.

(1) 変数の決定空間を縮小するように深測操作を適用する.

(2) 問題の変数の数を減らすために，２変数を新しい１変数に統合する.

文献［2］では，仲川らはモジュラ法を用いたアルゴリズムで $n = 2000, k_i^0 = 20$ の規模の問題を実用的な時間で解いている．ただ難しいタイプの問題に対しては，計算効率がかなり低下することも分かっている．このアルゴリズムの計算効率を向上するには，よりよい上界値が必要である.

2．3　深測操作(fathoming operation)

モジュラ法(MA)は．$s \in N, k \in K_s^0$ に対する部分問題 $[P^0 : x_s = k]$ に対して,3 つの深測操作(fathoming operation)を行い，決定空間を縮小する.

(ｉ)実行可能性操作(feasibility operation)

部分問題が実行可能解を含むかどうかをテストする.

(ⅱ)限定操作(bounding operation)

他の部分問題と比較して，その部分問題が明らかに劣っていないかどうか，すなわち原問題の最適解を含むかどうかをテストする.

(ⅲ)優越操作(dominance operation)（整数優越）

部分問題の限界値を求め，現在の暫定解の値と比較して暫定解よりも良い解を含むかどうかを判定する.

整数優越（integer dominance）（優越性操作）

問題 $[P^0]$ において，$f_s^0(k') \ge f_s^0(k)$ ，$g_s^0(k') \le g_s^0(k)$ となる $k' \in K_s^0$ が存在するならば,部分解 $x_s = k$ は優越され，深測される.

原問題 $[P^0]$ に整数優越を適用すると次式が得られる.

$$[P^A]: \textit{Maximize}\ \ f^A(x) = \sum_{i=1}^{n^A} f_i^A(x_i) \qquad (2.2)$$

$$\textit{subject to}\ \ g^A(x) = \sum_{i=1}^{n^A} g_i^A(x_i) \le b^A,$$

$$x \in K_i^A \quad \textit{for}\ \ i \in N^A$$

ここで，$\mathrm{N}^A = \{1,2,\ldots,n^A\}$，$\mathbf{K}_i^A = \{1,2,\ldots,k_i^A\}$ である．また，任意の $i \in N^A$ に対して，

$f_i^A(1) < f_i^A(2) < \ldots < f_i^A(k_i^A)$，$g_i^A(1) = 0, g_i^A(1) < g_i^A(2) < \ldots < g_i^A(k_i^A)$ である.

更に，次の DGR 優越を考える．DGR 優越は，多重選択ナップザック問題(Multiple-choice Knapsack Problem)に対する LP 優越と同じである.

DGR優越（Decreasing Gain Ratio Dominance）

$$\frac{f_i^A(k)-f_i^A(k')}{g_i^A(k)-g_i^A(k')} \le \frac{f_i^A(k'')-f_i^A(k)}{g_i^A(k'')-g_i^A(k)} \qquad (2.3)$$

となる$k', k'' \in K_i^A$が存在するならば，解$x_i = k$は優越される．ただし，$k' < k < k''$である．

2．4　上界値の計算

問題$[P^A]$の上界値を計算するために，問題$[P^A]$ において DGR 優越（LP 優越）を適用して，次の DGR 型の非線形ナップザック問題を考える．

$$[P^B(b^B)]: Maximize \quad f^B(x) = \sum_{i=1}^{n^A} f_i^B(x_i) \qquad (2.4)$$

$$subject \ to \quad g^B(x) = \sum_{i=1}^{n^A} g_i^B(x_i) \le b^B,$$

$$x \in K_i^B \quad for \quad i = 1, \ldots, n^A,$$

$$K_i^B = \{1, 2, \ldots, k_i^B\} \quad for \quad i = 1, \ldots, n^A$$

ここで，

$$f_i^B(k) < f_i^B(k+1) \quad for \quad k \in \{1, 2, \ldots, k_i^B - 1\}, i = 1, \ldots, n^A,$$

$$g_i^B(k) < g_i^B(k+1) \quad for \quad k \in \{1, 2, \ldots, k_i^B - 1\}, i = 1, \ldots, n^A,$$

$$w_i(k) > w_i(k+1) \quad for \quad k \in \{2, 3, \ldots, k_i^B - 1\}, i = 1, \ldots, n^A,$$

$$w_i(k) = \frac{f_i^B(k) - f_i^B(k-1)}{g_i^B(k) - g_i^B(k-1)}$$

である．

原問題$[P^0]$から問題$[P^A]$及び$[P^B]$を導くことを，簡単な例で示す．原問題の係数を表−2.1 とする．

表−2.1　原問題$[P^0]$の係数の例

k	1	2	3	4
$f_1^0(k)$	26	25	27	75
$g_1^0(k)$	33	41	43	59

整数優越により$f_1^0(1) \ge f_1^0(2)$, $g_1^0(1) \le g_1^0(2)$となる$k'(=1) \in K_s^0$が存在するので，部分解$x_s = k(=2)$は優越され，深測されて問題$[P^A]$（表−2.2）が得られる．

表−2.2　問題$[P^A]$の係数の例

k	1	2	3
$f_1^A(k)$	26	25	75
$g_1^A(k)$	33	43	59

更に，DGR 優越を考える．

$$\frac{f_i^A(k)-f_i^A(k')}{g_i^A(k)-g_i^A(k')}\left(=\frac{27-26}{43-33}=0.1\right)\le\frac{f_i^A(k'')-f_i^A(k)}{g_i^A(k'')-g_i^A(k)}\left(=\frac{75-27}{59-43}=3.0\right)$$

となる $k'(=1), k''(=3)\in K_s^0$ が存在するので，部分解 $x_i=k(=2)$ は優越され，問題 $[P^B]$ (表－2.3) が得られる．

表－2.3　問題 $[P^B]$ の係数の例

k	1	2
$f_1^B(k)$	26	75
$g_1^B(k)$	33	59

次に，よく知られた greedy アルゴリズム（Fox [3]）を用いて，問題 $[P^B(b^B)]$ の greedy 解 $\mathbf{x}^G$ が得られる．

$$\min_{i=1,2,\ldots,n^A}\{w_i(x_i^G)\}>w_{i^*}(x_{i^*}^G+1)\ge\max_{i=1,2,\ldots,n^A}\{w_i(x_i^G+1\},\qquad(2.5)$$

$$0\le b^B-\sum_{i=1}^{n}g_i^B(x_i^G)<g_{i^*}^B(x_{i^*}^G+1)-g_{i^*}^B(x_{i^*}^G)$$

ここで i^* は，

$$w_{i^*}(x_{i^*}^G+1)=\max_{i=1,2,\ldots,n^A}\{w_i(x_i^G+1)\}$$

で定義される．更に，次式を仮定する．

$$w_i(1)=\infty,\ w_i(k_i^B+1)=0$$

問題 $[P^B(b^B)]$ の上界値 f^R は，greedy 解 $\mathbf{x}^G$ を用いて次式で与えられる (付録 A 参照)．

$$f^R=\sum_{i=1}^{n^A}f_i^B(x_i^G)+w_{i^*}(x_{i^*}^G+1)\{b^B-\sum_{i=1}^{n^A}g_i^B(x_i^G)\}\qquad(2.6)$$

次に，Sinha and Zoltners [4] が用いた，より厳密な上界値について述べる．問題 $[P^A]$ に対して $p\in K_{i^*}^A$ は，

$$g_{i^*}^A(p)\le h<g_{i^*}^A(p+1)\qquad(2.7)$$

を満たすものとする．ここで，

$$h=g_{i^*}^B(x_{i^*}^G)+b^B-\sum_{i=1}^{n^A}g_i^B(x_i^G)\qquad(2.8)$$

である．次に，

$$w_{i^{Prv}}(x_{i^{Prv}}^G)=\min_{i=1,2,\ldots,n^A,i\neq i^*}\{w_i(x_i^G)\}\qquad(2.9)$$

及び

$$w_{i^{Nxt}}(x_{i^{Nxt}}^G+1)=\max_{\substack{i=1,2,\ldots,n^A\\ i\neq i^*}}\{w_i(x_i^G+1)\}\qquad(2.10)$$

となる i^{Prv} と i^{Nxt} を決定する．Sinha and Zoltners が用いた上界値 f^{UB} は，

$$f^{UB}=\max\{U^1,U^2\}\qquad(2.11)$$

である．ここで，

$$U^1=f^R-\{w_{i^*}(x_{i^*}^G+1)-w_{i^{Nxt}}(x_{i^{Nxt}}^G+1)\}\{h-g_{i^*}^A(p)\},\qquad(2.12)$$

$$U^2=f^R-\{w_{i^{Prv}}(x_{i^{Prv}}^G)-w_{i^*}(x_{i^*}^G+1)\}\{g_{i^*}^A(p+1)-h\}\qquad(2.13)$$

である．

線形緩和問題をＬＰで解くと最適値は実数になる．DGR 優越を考慮した目的関数は凸関数であるので，その変数を整数に固定すると，他が実数に変わる．整数に固定することにより制約条件が加わり，その中に整数解を含む解空間は小さくなり，その解空間から求められる上界値はより良くなる．Sinha and Zoltners よりも良い上界値を求めるために，複数の変数を固定すると原問題に複数の制約条件が加わり，整数解が含まれる解空間が小さくなり，求まる上界値がよりよくなる．なお，１変数の場合，Sinha and Zoltners よりもよい上界値が得られることは文献[６]で証明されている．

以下において，よりよい上界値の計算方法（変数固定式上界値計算法）を提案する．

[変数固定式上界値計算法]

問題 $[P^B(b^B)]$ において，

(1) $K_i^B(i\in N)$ を 2 グループに分ける．

$$K_i^U=\{k\in K_i^B\mid w_i(k)>w_{i^*}(x_{i^*}^G)\}$$
$$K_i^L=\{k\in K_i^B\mid w_i(k)<w_{i^*}(x_{i^*}^G)\}$$

(2) $K_i^U(i\in N)$ に対して，数列

$$N^U=\{s(1),s(2),\ldots,s(n^U)\}\subseteq N\qquad\text{を}$$

$$\min_{k\in K^U_{s(l)}}\{w_{s(l)}(k)\} < \min_{k\in K^U_{s(l+1)}}\{w_{s(l+1)}(k)\} \qquad (l=1,\ldots,n^U-1)$$

となるように決定する．同様に，K^L_i に対して，数列

$$N^L=\{t(1),t(2),\ldots,t(n^L)\}\subseteq N \qquad \text{を}$$

$$\max_{k\in K^L_{t(l)}}\{w_{t(l)}(k)\} > \max_{k\in K^L_{t(l+1)}}\{w_{t(l+1)}(k)\} \qquad (l=1,\ldots,n^L-1)$$

となるように決める．

(3) N^U, N^L に同じ要素がある場合，例えば，$s(l^U)=t(l^L)$ の場合，

①$l^U < l^L$ であれば，$t(l^L)$ を N^L から除く．

②$l^U = l^L$ であれば，N^U と N^L の要素数の多い方から除く．

(4) 問題$[P^A]$に対して固定する変数の数だけ，前から順番に N^U，N^L を取る．固定された変数に関して，その変数の代替案を順番に選択する．それぞれの代替案の組み合わせについて f^R を計算し，その中の最大値を問題$[P^B(b^B)]$の上界値とする．

[上界値計算の効率化]

(4) の計算において次の定理を利用すると，上界値の計算が効率的になる．

[定理 2-1]　問題$[P^B(b^1)]$，$[P^B(b^2)]$の greedy 解をそれぞれ x^1，x^2 とする．もし，$b^1 < b^2$ ならば，$x^1 \le x^2$ である．

（証明）x^1 は$[P^B(b^1)]$の greedy 解であるから，

$$\min_{i=1,2,\ldots,n^A}\{w_i(x^1_i)\} > \max_{i=1,2,\ldots,n^A}[\max\{w_i(k)\mid k\in\{x^1_i+1,x^1_i+2,\ldots\ldots,k^B_i\}\}],$$

$$\sum_{i=1}^{n} g_i(x^1_i) < b^2 \qquad for \quad i=1,\ldots,n$$

したがって，$x^1 \le x^2$ となる．（証明終り）

定理 2-1 は，greedy アルゴリズムにおいて問題$[P^B(b^2)]$の初期解として x^1 が利用可能であることを示している．したがって，問題$[P^B:x_s=k]$（$[P:\bullet]$は問題$[P]$に制約$\bullet$を付け加えた問題）と$[P^B:x_s=k+1]$は制約式の右辺，すなわち $b^B-g^B_s(k)$ と $b^B-g^B_s(k+1)$

が異なるだけであるから，$s\in\{1,\ldots,n^A\}$ に対する問題　$[P^B:x_s=1]$, $[P^B:x_s=2]$,…,

$[P^B:x_s=k^A_s]$の上界値は，定理 2-1 を利用すると効率的に計算できる．この上界値の計算手法は，Marsten and Morin [5] の resource-space tour の特別な場合である．すなわち，計算された部分問題の上界値を利用して，その次の部分問題の上界値を効率的に求める方法である．

[例題]　問題$[P^B(b^B)]$において，$n^A=7$, $k^B_i=3\,(i=1,\ldots,n^A)$，係数 $f^B_i(x_i), g^B_i(x_i)$ の値は表−2.4 の数値とし，コスト制約は $b^B=114$ とする．

表−2.4　例題の係数

i	1	2	3	4	5	6	7
$f_i(1)$	10	8	29	13	21	8	13
$f_i(2)$	34	37	54	43	33	39	17
$f_i(3)$	38	69	56	64	49	68	20
$g_i(1)$	0	0	0	0	0	0	0
$g_i(2)$	19	20	30	5	21	16	7
$g_i(3)$	28	48	51	20	29	43	9

表−2.5　例題に対する値 $w_i(k)$

i	1	2	3	4	5	6	7
$w_i(1)$	∞	∞	∞	∞	∞	∞	∞
$w_i(2)$	1.2632	1.45	0.8333	6	0.9655	1.9375	0.7778
$w_i(3)$	0.4444	1.1429	0.0952	1.4	−	1.0741	−
$w_i(4)$	0.0	0.0	0.0	0.0	0.0	0.0	0.0

表−2.4から$w_i(k)$を計算すると表−2.5が得られる．表−2.5中の−印は，DGR優越により深測され，その後の手順において考慮の対象から除外できる項目である．コスト制約$b^B(=114)$から値$w_i(k)$の降順にしたがって，それに対応する係数$g_i(x_i)$を順番に引いていくと次のようになる．

$$b^1 = 114 - 43 = 71, \quad b^2 = 114 - 16 = 98, \quad b^3 = 114 - 0 = 114$$

$$\Delta = b^B - \sum_{i=1}^{n'} g_i^B(x_i^G) = 114 - 5 - 16 - 20 - (20 - 5) - 19 - (48 - 20) = 11$$

Δを比例配分すると，式(2.6)より上界値f^Rは，

$$f^R = 34 + 69 + 29 + 64 + 21 + 39 + 13 + 11 \times 1.07 = 280.8148$$

となる．

次に，本論文で提案した変数固定式上界値計算法を使って上界値を計算する．上の計算においてgreedy解が含まれていたx_6のみを固定する（1変数固定）．まず，代替案3を選択すると，上と同様に，

$$b^1 = 114 - 43 = 71, \quad f^1 = 68$$

$$\Delta^1 = b^1 - 5 - 20 - (20 - 5) - 19 = 12$$

$$f^{R1} = f^1 + 34 + 37 + 29 + 64 + 21 + 13 + 12 \times 1.14285 = 279.7143$$

となる．代替案2を選択すると，

$$b^2 = 114 - 16 = 98, \quad f^2 = 39$$

$$\Delta^2 = b^2 - 5 - 20 - (20 - 5) - 19 - (48 - 20) = 11$$

$$f^{R2} = f^2 + 34 + 69 + 29 + 64 + 21 + 13 + 11 \times 0.96552 = 279.6207$$

となる．代替案1を選択すると，次のようになる．

$$b^3 = 114 - 0 = 114, \quad f^3 = 8$$

$$\Delta^3 = b^3 - 5 - 20 - (20 - 5) - 19 - (48 - 20) = 27$$

$$f^{R3} = f^3 + 34 + 69 + 29 + 64 + 21 + 13 + 27 \times 0.96552 = 264.0690$$

計算された上界値の候補f^{R1}, f^{R2}, f^{R3}を比較し，その最大値

$$\max\{f^{R1}, f^{R2}, f^{R3}\} = f^{R3} = 279.7143$$

が1変数固定の場合の上界値となる．

次に，3変数を固定した場合の上界値を計算する．

$$N^U = \{2, 1, 4\}$$

$$N^L = \{5, 3, 7, 1\}$$

であるから，上の計算で固定されたgreedy解が含まれるx_6，及びそれを挟むようにN^U, N^Lから取ったx_2とx_5の3変数を固定する．固定した変数に対する代替案を選択するすべての組み合わせ（$3^3 = 27$通り）のうちいくつかの組み合わせは，前述の「上界値計算の効率化」により，f^Rを計算する必要がない．この計算結果を表−2.6に示す．表−2.6において，第1列目は代替案の組合せを，第2列目は「b^Bから，選択した3代替案に対する$g_i^B(x_i)$を引いた値」を，第3列目は「選択した3代替案に対する$f_i^B(x_i)$の和」を，第4列目は式(2.6)の右辺の中括弧の値を，第5列目はf^Rの値をそれぞれ示している．表中−印は上界値の候補から除かれるため，計算されてないことを示す．

表−2.6は第2列目の要素について降順に並べられている．表−2.6より，3変数を固定した場合の上界値は，f^Rの最大値278.1667となる．

以上より，例題において，0変数固定，及び変数固定式上界値計算法を用いた，1変数固定，3変数固定の場合の上界値を比較すると，変数固定式上界値計算法がより強く，しかも固定する変数の数が増えるにしたがって，上界値はより強くなることがわかる．

上界値＝
- 0変数固定，実変数の場合：280.8148
- 変数固定式上界値計算法
  - 1変数固定の場合：279.7143
  - 3変数固定の場合：278.1667

表－2.6　例題に対する上界値の計算（3変数固定の場合）

$x_2x_6x_5$	b^C	$\sum f_i^B(x_i)$	$b^B-\sum g_i^B(x_i)$	f^R
111	114	37	6	215
112	98	68	11	245.0476
211	94	66	—	—
121	93	49	—	—
131	85	65	—	—
212	78	97	9	269
122	77	80	—	—
221	73	78	—	—
113	71	97	—	—
132	69	96	—	—
311	66	98	27	260.5
231	65	94	—	—
222	57	109	18	264
213	51	126	12	276
123	50	109		
312	50	129	11	278.1667
232	49	125	—	—
321	45	110	—	—
133	42	125	—	—
331	37	126	—	—
223	30	138	10	266.6316
322	29	141	9	268.3684
313	23	158	3	277.7895
233	22	154	—	—
332	21	157	—	—
323	2	170	2	247
333	-6	—	—	—

2．5　計算機実験による上界値の比較

本論で提案した変数固定式上界値計算法の有効性を確かめるため，擬似乱数を生成したテスト問題を用いて，Sinha and Zoltners が用いた上界値との比較検討を行う．2代替案(資本予算問題，Capital budgeting problem)及び3代替案の問題を考える．原問題の変数の数が多い場合，モジュラ法と組み合わせて用いることにより，原問題の変数の数を減らすことができる．更に，変数の数が少なくなったとき，解の組み合わせの爆発が生じるので，そのとき「よりよい」上界値が必要になる．したがって，変数の数は 10,20,30,40,50 変数とする．テスト問題の係数は次の擬似乱数とする．

$0 \le f_i(k) < f_i(k+1) \le 256k_i \quad (k=1,2,\ldots\ldots,k_{i-1},\ i=1,2,\ldots\ldots,n),$

$0 \le g_i(k) < g_i(k+1) \le 256k_i \quad (k=1,2,\ldots\ldots,k_{i-1},\ i=1,2,\ldots\ldots,n)$

$f_i(k), g_i(k)$はすべて整数であり，

$$b=\left\lfloor \frac{1}{2}\sum_{i=1}^{n}\{g_i(1)+g_i(k_i)\} \right\rfloor \quad (i=1,2,\ldots\ldots,n)$$

である．

表－2.7は，2代替案問題（資本予算問題）及び3代替案問題において，変数が 10,20,.... ,50 としたとき，0，1,2,3,5 変数固定したときの変数固定式上界値計算法による上界値（以下，変数固定式上界値と略す）に対する Sinha and Zoltners が用いた上界値の相対誤差（%）

$$\frac{Sinha-Zoltnersの上界値 - 変数固定式上界値}{変数固定式上界値}\times 100$$

を表している．それぞれの上界値を求めるとき，擬似乱数の種を１０回変化させて上界値

表－2.7 変数固定式上界値に対する Sinha and Zoltners が用いた上界値の相対誤差（%）

問題		10 変数	20 変数	30 変数	40 変数	50 変数
2代替案	0変数固定	0.27956	0.05740	0.02150	0.02091	0.01214
	1 変数固定	0.27956	0.05740	0.02150	0.02091	0.01214
	2 変数固定	0.38319	0.09082	0.02266	0.03956	0.01558
	3 変数固定	0.93039	0.12103	0.04175	0.04031	0.02279
	5 変数固定	1.36430	0.21308	0.05495	0.05435	0.02901
3代替案	0 変数固定	0.09441	0.07253	0.01676	0.01861	0.00676
	1 変数固定	0.09441	0.07253	0.01676	0.01861	0.00676
	2 変数固定	0.13478	0.08780	0.04101	0.03227	0.00853
	3 変数固定	0.23202	0.12258	0.05841	0.04771	0.00911
	5 変数固定	0.32133	0.15756	0.07275	0.06715	0.01731

を求め，その平均値を上界値とした．

表－2.7より，いずれの場合においても相対誤差が正であり，変数固定式上界値がSinha and Zoltnersが用いた上界値よりも小さい，「よりよい」上界値であることがわかった．また，固定する変数の数が増えるに従って相対誤差が大きくなった．固定する変数の数が増加するにつれて，変数固定式上界値がよりよくなることがわかった．

表－2.8は，10,...,50変数からなる2代替案問題（資本予算問題）および3代替案問題において，0変数固定，及び変数固定式上界値計算法において1，2，3，5変数固定して，それぞれの上界値を求めるために必要な平均計算時間を示している．擬似乱数の種を10回変化させて，それぞれ100,000回ループさせて生成したテスト問題の上界値を計算する総経過時間から平均計算時間（単位：秒）を求めた．

表－2.8　問題（2代替案，3代替案）の平均計算時間（単位：秒）

問題		10変数	20変数	30変数	40変数	50変数
2代替案	0変数固定	0.000030	0.000042	0.000054	0.000073	0.000083
	1変数固定	0.000020	0.000041	0.000054	0.000074	0.000084
	2変数固定	0.000023	0.000037	0.000052	0.000068	0.000080
	3変数固定	0.000024	0.000039	0.000052	0.000073	0.000085
	5変数固定	0.000040	0.000059	0.000070	0.000087	0.000103
3代替案	0変数固定	0.000046	0.000079	0.000081	0.000110	0.000133
	1変数固定	0.000047	0.000077	0.000081	0.000109	0.000132
	2変数固定	0.000030	0.000060	0.000080	0.000110	0.000130
	3変数固定	0.000040	0.000068	0.000088	0.000116	0.000139
	5変数固定	0.000180	0.000233	0.000236	0.000269	0.000290

表－2.8より，2代替案から3代替案へ，また10変数から50変数へ増加するに従って平均計算時間は増加しているが，いずれの場合も十分小さな計算時間で上界値を求められることがわかった．0変数固定した場合の平均計算時間と比較しても，変数固定式上界値計算法による平均計算時間は，1変数固定から5変数固定の場合には顕著な差異は認められなかった．これは，前述の「上界値計算の効率化」によって，上界値の計算を行わない組み合わせが除かれたためである．したがって，変数固定式上界値計算法において1～5変数固定で上界値を求めても，0変数固定で上界値を求める場合の計算時間とほぼ同程度の計算時間で，よりよい上界値を求めることができることがわかった．

2．6　結語

本論文では，非線形ナップザック問題の最適解を求める際，重要な役割を担う『よい』上界値を求めるための計算法(変数固定式上界値計算法)を提案した．同様な計算式はSinha and Zoltners[4]も用いているが，変数固定式上界値計算法の方がより良い上界値を与えることを示した．

簡単な例題及び実用規模の非線形ナップザック問題を解くことにより，変数固定式上界値計算法を用いて得られる上界値は，Sinha and Zoltnersが用いた上界値よりも，よりよい上界値であることが明らかになった．

なお，本章におけるすべての計算は，DOS/V パソコン（CPU:PentiumIII600MHz,メモリ128MB）を用いた．

参考文献

[1]仲川勇二, "離散最適化問題のための新解法", 電子情報通信学会論文誌, Vol. J73-A No. 3, pp. 550－556, March 1990.

[2]Y. Nakagawa and A. Iwasaki, "Modular approach for solving nonlinear knapsack problems", IEICE Trans. Fundamentals, Vol. E82－A, NO. 9, Sep. 1999.

[3]B. Fox, "Discrete optimization via marginal analysis", Manag. Sci, Vol. 13, 1966.

[4]P. Sinha and A. Zoltners, "The Multiple-Choice Knapsack Problem", Oper. Res, Vol. 27 pp. 125－131, 1979.

[5]R. E. Marsten and T. L. Morin, "A hybrid approach to discrete mathematical programming", Math. Programming, Vol. 14, pp. 21－40, 1978.

[6]仲川勇二, 北尾匡史, 辻　光宏, 寺岡義伸, "多重選択ナップザック問題の上界値計算, " 電子情報通信学会論文誌, Vol. I83-A, No. 4, pp. 396-401, Apr. 2000.

[7]K. M. Bretthauer and B. Shetty, "The Nonlinear Resource Allocation Problem," Oper. Res., Vol. 43, pp. 670－683, 1995.

[8]T. Ibaraki and N. Katoh, "Resource Allocation Problems," MIT Press, Cambridge, Mass, 1988.

[9]R. M. Nauss, "The 0-1 Knapsack Prolem with Multiple Choice Constraints, "Eur. J. of Oper. Res., 2, pp. 125－131, 1978.

[10]B. Gavish and H. Pirkul, "Allocation of Database and Processors in a Distributes Computing Systems, "In Managemen of Disributed Data Processing, ed. J. Akoka, pp. 215－231, North－Holland, 1982.

[11]W. Shih, "A Branch and Bound Method for the Multiconstraint Zero-One Knapsack Problem, "J. of the Oper. Res. Soc., Vol. 30, pp369－378, 1979.

[12]P. C. Gilmore and R. E. Gomory, "The Theory and Computation of Knapsack Functions, " Oper. Res., Vol. 14, pp. 1045－1075, 1966.

[13] 木村作郎, 森部浩至, 仲川勇二, "非線形ナップザック問題における上界値の改良, " 京都大学数理解析研究所 講究録, Vol. 1194, pp. 111－115, March, 2001.

[14] 木村作郎, 森部浩至, 仲川勇二, "非線形ナップザック問題における新上界値計算法, " 電子情報通信学会論文誌 A, Vol. J86－A No. 1, pp. 38－45, Jan. 2003.

付録Ａ　上界値式(2.6)の算出

問題$[P^B(b^B)]$に対応した次の区分線形問題$[P^C(b^C)]$を考える．

$$[P^C(b^C)]: Maximize \quad f^C(\xi)=\sum_{i=1}^{n^A}\sum_{k=1}^{k_i^B}\Delta f_i^C(k)\xi_{ik} \tag{A1}$$

$$subject\ to\ g^C(\xi)=\sum_{i=1}^{n^A}\sum_{k=1}^{k_i^B}\Delta g_i^C(k)\xi_{ik}\le b^C,$$

$$\xi_{i1}=1\ (i\in I)$$

$$\xi_{i1}\ge\xi_{i2}\ge\cdots\cdots\ge\xi_{ik_i^B}\ge 0\ ,\ integer\ (i\in I)$$

ただし，

$$\Delta f_i^C(k)=\begin{cases} f_i^C(k) & (k=1)\\ f_i^C(k)-f_i^C(k-1) & (k\ge 2)\end{cases} \tag{A2}$$

$$\Delta g_i^C(k)=\begin{cases} g_i^C(k) & (k=1)\\ g_i^C(k)-g_i^C(k-1) & (k\ge 2)\end{cases} \tag{A3}$$

$$\frac{\Delta f_i^C(k)}{\Delta g_i^C(k)}>\frac{\Delta f_i^C(k+1)}{\Delta g_i^C(k+1)}\quad (k=1,2,\ldots,n^A-1,\ i\in I) \tag{A4}$$

である．

定理Ａ1．問題$[P^B(b^B)]$と$[P^C(b^C)]$は等価である．

(証明)変数変換

$$f_i^B(k_i)=\begin{cases} f_i^C(k)\ \xi_{ik} & (k=1)\\ \{f_i^C(k)\ -\ f_i^C(k-1)\}\ \xi_{ik} & (k\ge 2)\end{cases} \tag{A5}$$

$$g_i^B(k_i)=\begin{cases} g_i^C(k)\ \xi_{ik} & (k=1)\\ \{g_i^C(k)\ -\ g_i^C(k-1)\}\ \xi_{ik} & (k\ge 2)\end{cases} \tag{A6}$$

を行うと問題$[P^B(b^B)]$（式(2.4)）より問題$[P^C(b^C)]$（式(A 1)）が得られる．

定理Ａ2．問題$[P^C(b^C)]$において$\xi_{i1}\ge\xi_{i2}\ge\cdots\cdots\ge\xi_{ik_i^B}\ge 0$を

$0\le\xi_{ik}\le 1\ (k=2,\ldots,n^A-1, i\in I)$と置き換えても最適解は変化しない．

(証明) 式 (A 4) より，置き換えられた問題の最適解は常に $\xi_{i1} \geq \xi_{i2} \geq \cdots\cdots \geq \xi_{ik^B} \geq 0$ を満足する．

定理 A2 より問題 $[P^C(b^C)]$ は，非線形ナップザック問題のＬＰ緩和問題 $[P^D(b^D)]$ と等価である．

$$[P^D(b^D)]: \textit{Maximize}\ \ f^D(\xi) = \sum_{i=1}^{n^A}\sum_{k=1}^{k_i^B} \Delta f_i^D(k)\xi_{ik} \qquad (A7)$$

$$\textit{subject to}\ \ g^D(\xi) = \sum_{i=1}^{n^A}\sum_{k=1}^{k_i^B} \Delta g_i^D(k)\xi_{ik} \leq b^D,$$

$$\xi_{i1} = 1\ \ (i \in I)$$

$$1 \geq \xi_{i1} \geq \xi_{i2} \geq \cdots\cdots \geq \xi_{ik_i^B} \geq 0\ ,\ \text{integer}\ \ (i \in I)$$

この上界値は，greedy 解 x^G を用いて簡単に求めることができる．

$$f^R = \sum_{i=1}^{n^A} f_i^B(x_i^G) + w_{i^*}(x_{i^*}^G + 1)\{b^B - \sum_{i=1}^{n^A} g_i^B(x_i^G)\} \qquad (2.6)$$

第３章　代理制約法における最適代理乗数の決定法［15］

３．１　概説

複数制約条件付きの変数分離型非線形整数計画問題（多次元非線形整数計画問題）の品質の良い解を得ることは，近年ますます重要になってきている．

代理乗数を用いて，原問題の複数制約条件式を単一制約条件式とした代理問題に変換して原問題を解く代理制約法は，大変計算効率のよい解法である[1]．代理制約法は，Glover によって数理計画法において初めて用いられた[1]．原問題が準凸計画問題であるときは代理乗数を適当に決定すれば，代理問題の最適解は原問題の最適解と一致することが示されている[2]．Karwan and Rardin は整数線形計画法における代理制約法の有効性に関する実証的な研究を行っている[3]．しかし，多次元非線形整数計画問題に代理制約法を適用した場合，代理双対問題に代理双対ギャップ(surrogate duality gap)が存在することが多く，その場合には代理問題の最適解は一般に原問題の最適解に一致せず，実行可能解になるとは限らない．このときは，単に原問題の目的関数の上界値を与えるだけである．この代理双対ギャップを閉じ，原問題の厳密解を求めることができる改良代理制約法（ISC 法，Improved Surrogate Constraint Method）が，最近仲川によって提案された[4．5]．この ISC 法により制約条件式が 5，6 個で，1000 変数規模の変数分離型の非線形整数計画問題を厳密に解くことが可能となった．ISC 法を効率的に実行して実用的な時間内で解を得るためには，その中に含まれる最適代理乗数決定アルゴリズムにより最適な代理乗数を求めることが必要である．しかし，制約条件式の個数が多いとき最適な代理乗数を決定するためのアルゴリズムにおいて，実行時間が膨大になり，さらに計算時に使用する作業領域の量が多大になるため，最適な代理乗数が実用的な時間内で得られないことがある．

最適な代理乗数を決定するためのアルゴリズムがいくつか提案されている．本論文では，ISC 法における最適な代理乗数を決定するアルゴリズムとして，Dyer アルゴリズム[6]と仲川の COP(Cutting-Off Polyhedron) アルゴリズム[7]を採用する．なお，これらの方法は代理制約法でよく用いられるが，ISC 法にも適用できる．そして，計算機実験を通して両アルゴリズムを比較検討し，これらの長所と短所を明らかにする．

さらに，本論文では Dyer アルゴリズムの短所を改良する改良 Dyer アルゴリズムを提案し，計算機実験により，その有効性を明らかにする．

3．2　問題

3．2．1　多次元非線形整数計画問題

本論文で解くべき変数分離型の多次元非線形整数計画問題は次のように定式化される．

$$[P] \quad \text{maximize} \quad f(\mathbf{x}) \qquad (3.1)$$
$$\text{subject to} \quad \mathbf{g}(\mathbf{x}) \le \mathbf{b},$$
$$\mathbf{x} \in \mathbf{X},$$

ここで，$\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$ は，決定空間 $\mathbf{X}$ での N 次元整数値変数ベクトル，$f(x)$ は整数値目的関数，$\mathbf{g}(\mathbf{x}) = (g_1(x), g_2(x), \ldots, g_M(x))^T$ は M 次元整数値ベクトル制約関数，$\mathbf{b} = (b_1, b_2, \ldots, b_M)^T$ は M 次元整数値ベクトル制約許容量で，$\mathbf{X}$ はN次元離散決定空間である．この原問題[P]を代理乗数 $\mathbf{u} = (u_1, u_2, \ldots, u_M)^T$ を用いて次の代理問題 $[P^S(\mathbf{u})]$ に変換する．

$$[P^S(u)] \quad \text{maximize} \quad f(\mathbf{x}) \qquad (3.2)$$
$$\text{subject to} \quad \mathbf{u}^T\mathbf{h}(\mathbf{x}) \le 0$$
$$\mathbf{x} \in \mathbf{X},$$

ただし，

$$\mathbf{h}(\mathbf{x}) = \mathbf{g}(\mathbf{x}) - \mathbf{b}$$
$$\mathbf{u} \in \mathbf{U} = \{\mathbf{u} \in \mathbf{R}^M \mid \sum_{m=1}^{M} u_m = 1, u \ge 0\}$$

である．制約条件の第１番目の不等式を代理制約式と呼ぶ．複数の制約条件式を単一の制約条件式で代理させて，複数制約の問題を解く方法を代理制約法と呼ぶ．

原問題［P］の代理双対問題 $[P^{SD}]$ は次のようになる．

$$[P^{SD}] \quad \min\{\text{opt}[P^S(\mathbf{u})] : \mathbf{u} \in \mathbf{U}\} \qquad (3.3)$$

ただし，$\text{opt}[P^S(\mathbf{u})]$ は代理問題 $[P^S(\mathbf{u})]$ の最適解の目的関数値である．

代理問題 $[P^S(\mathbf{u})]$ に対して次の定理が成り立つ[5]．

［定理 3-1］$\mathbf{x}^k$ を代理乗数ベクトル $\mathbf{u}^k \in \mathbf{U}$ に対する問題 $[P^S(\mathbf{u}^k)]$ の最適解とする．$\mathbf{u}^T\mathbf{g}(\mathbf{x}^k) \le \mathbf{u}^T\mathbf{b}$ となる任意の $\mathbf{u} \in \mathbf{U}$ に対して，次式が成り立つ．

$$\text{opt}[P^S(\mathbf{u})] \ge f(\mathbf{x}^k) \qquad (3.4)$$

証明 略[5].

この定理 3-1 は，最適な代理乗数が属する多面体は領域 $\{\mathbf{u} \in \mathbf{U} : \mathbf{u}^T\mathbf{g}(\mathbf{x}^k) \le \mathbf{u}^T\mathbf{b}\}$ を多面体 $\mathbf{U}$ から除外した多面体となることを意味する．この定理を利用したアルゴリズム[7]は，代

理双対問題 $[P^{SD}]$ を厳密に解くことができる．このアルゴリズムは，初期多面体を $\mathbf{U}^1 = \mathbf{U}$ とする．代理乗数 $\mathbf{u}^k$ として k 番目の多面体 $\mathbf{U}^k$ の頂点の重心を用いると，代理問題 $[P^S(\mathbf{u}^k)]$ の最適解 $\mathbf{x}^k$ が得られ，切断面を含む切断領域は $\mathbf{u}^T\mathbf{g}(\mathbf{x}^k) \ge \mathbf{u}^T\mathbf{b}$ となる．この切断面の超平面(hyperplane)によって，次の多面体 $\mathbf{U}^{k+1}$ を作るために多面体 $\mathbf{U}^k$ を切断する．

定理 3-1 から $\mathbf{U}$ のすべてをカバーする代理乗数の有限集合多面体 $\mathbf{u}^1, \mathbf{u}^2, \ldots, \mathbf{u}^k$ が得られる．

代理乗数 $\mathbf{u}^*$ は，

$$\{\text{opt}[P^S(\mathbf{u}^*)] = \min\{\text{opt}[P^S(\mathbf{u}^1)], \text{opt}[P^S(\mathbf{u}^2)], \ldots, \text{opt}[P^S(\mathbf{u}^k)]\} \qquad (3.5)$$

が代理双対問題 $[\mathbf{P}^{SD}]$ に対して最適であるように定義される[5]．

［定理 3-2］もし問題 $[P^S(\mathbf{u}^*)]$ の最適解 x^{SD} が，もとの複数制約問題［P］に対して実行可能ならば，$\mathbf{x}^{SD}$ は［P］の厳密な最適解である[5]．

証明 略[5].

3．2．2　代理双対ギャップの解消 [4, 5]

ISC 法においては，代理双対ギャップを閉じるために $f^{\mathrm{Target}}(0 < f^{\mathrm{Target}} < \infty)$ に対する標的問題[14]を考える．

$$[P^{\mathrm{Target}}(f^{\mathrm{Target}})]: \text{Enumerate all solutions } \mathbf{x} \text{ hitting a target } f(x) \geq f^{\mathrm{Target}} \quad (3.6)$$
$$\text{subject to } \mathbf{u}^{*T}\mathbf{g}(\mathbf{x}) \leq \mathbf{u}^{*T}\mathbf{b},$$
$$\mathbf{x} \in \mathbf{X}$$

ただし，$\mathbf{u}^*$ は元の複数制約問題 [P] に対応する代理双対問題 $[P^{SD}]$ の最適代理乗数である．標的値 f^{Target} 以上の目的関数値をもつ実行可能解を標的解(target solution)と呼ぶ．標的解に対して，次の定理が成り立つ[5]．

[定理 3-3] もし $f^{\mathrm{Target}} \leq \mathrm{opt}[P]$ ならば，[P] の最適解のすべては標的問題 $[P^{\mathrm{Target}}(f^{\mathrm{Target}})]$ に対する標的解である．

証明 略[5]．

定理 3-3 は，もし $f^{\mathrm{Target}} \leq \mathrm{opt}[P]$ であり，問題 $[P^{\mathrm{Target}}(f^{\mathrm{Target}})]$ の標的解を列挙することができれば，原問題[P]の厳密な最適解がすべて得られることを意味する．

3．3　代理乗数決定アルゴリズムの概要

3．3．1　切断面[7]

ある多面体 U^k の重心 $\mathbf{u}^k \in \mathbf{R}^M$ に対する代理問題 $[P^S(\mathbf{u}^k)]$ はすでに解かれているものとする．

[定理 3-4] 代理問題 $[P^S(\mathbf{u}^k)]$ の解を x^k とし，

$$\overline{\mathbf{H}}^k = \{\mathbf{u} \in \mathbf{U}^k : \mathbf{u}^T\mathbf{h}(\mathbf{x}^k) \leq 0\} \quad (3.7)$$

とする．このとき，任意の $\mathbf{u} \in \overline{\mathbf{H}}^k$ において

$$\mathrm{opt}[P^S(\mathbf{u}^k)] \leq \mathrm{opt}[P^S(\mathbf{u})] \quad (3.8)$$

が成り立つ[7]．

[系 3-1] $\mathbf{H}^k = \{\mathbf{u} \in \mathbf{U} : \mathbf{u}^T\mathbf{h}(\mathbf{x}^k) > 0\}$ とする．もし，$\mathrm{opt}[P^S(\mathbf{u})] < \mathrm{opt}[P^S(\mathbf{u}^k)]$ となる $\mathbf{u} \in \mathbf{U}$ が存在すれば，$\mathbf{u} \in \mathbf{U} \cap \mathbf{H}^k$ が成り立つ．

系 3-1 により，多面体 $\mathbf{U}^k$ を切断して新しい多面体 $\mathbf{U}^{k+1}$ を作る代理制約式は，$\mathbf{u}^T\mathbf{h}(\mathbf{x}^k) > 0$ である．

3．3．2　代理制約法のアルゴリズム

代理制約法のアルゴリズムを示す．

1：$k \leftarrow 1$；

$\mathbf{u}^1 \leftarrow (1/M, 1/M, \ldots, 1/M)$；

$\mathbf{U}^1 \leftarrow \{\mathbf{u} \in \mathbf{R}^M \mid \mathbf{u} \geq 0, \sum_{m=1}^{M} u_m = 1\}$；

2：Repeat

3：　代理問題$[P^S(\mathbf{u}^k)]$の最適解$\mathbf{x}^k$を求める；

4：　If($\mathbf{x}^k$が問題[P]の実行可能解である)　then

5：　　$\mathbf{x}^k$が問題[P]の厳密解である；　停止；

6：　EndIf

7：　$\mathbf{x}^k$を用いて多面体$\mathbf{U}^k$の切断面を含む切断領域

$\mathbf{u}^T\mathbf{h}(\mathbf{x}^k) \geq 0$

を求め，COPアルゴリズムまたはDyerアルゴリズムを用いて，次の代理乗数$\mathbf{u}^{k+1}$を求める；

8：　$k \leftarrow k+1$；

9：　Until（$\mathbf{U}^{k-1}$が空である）

10：　Return($\mathbf{u}^k$)；

このアルゴリズムで得られた$\mathbf{u}^k$が代理双対問題の最適な代理乗数である．モジュラ法(MA, Modular Approach)[8,9]は，次の手順を繰り返して代理問題$[P^S(\mathbf{u}^k)]$の厳密解を高速に求めることができる．

(1) 深測操作(fathoming operation)を用いて変数の決定空間を縮小する．

(2) 2つの変数を統合して1つの変数にすることによって，問題の変数の個数を減らす．

3．3．3　代理乗数の決定アルゴリズム

ここでは，多面体$\mathbf{U}^k$のもとで次の代理乗数$\mathbf{u}^k$を決定するための方法として，Dyerアルゴリズムと COP アルゴリズムについて説明する．

（1）Dyerアルゴリズム

Dyer [6] によって提案された方法の記述法を変更して，分かりやすく説明する．本論文では初期多面体として

$$\mathbf{U}^1 \leftarrow \{\mathbf{u} \in \mathbf{R}^M \mid \mathbf{u} \geq 0, \sum_{m=1}^{M} u_m = 1\} \tag{3.9}$$

を用いる．Dyerの論文では，初期多面体としてこの$\mathbf{U}^1$を用いなかったために，得られた初期の代理乗数$\mathbf{u}^k$が多面体$\mathbf{U}^k$の中心部分に存在しないという問題点があった．この問題点を回避するため，Dyerアルゴリズムは得られた$\mathbf{u}^k$を中心部へ移動する操作が別途必要であった．また，この操作には経験的な数値設定も必要であった．本論文では，初期多面体として$\mathbf{U}^1$を用いることによって，このような移動操作は必要なくなり，Dyerのアルゴリズムはより単純でより高速なものになる．

多面体$\mathbf{U}^k$に内接する最大の球の中心を求めるために，まず$\mathbf{U}^k$内の点$\mathbf{u}'$と切断領域

$$\mathbf{u}^T\mathbf{h}(\mathbf{x}^k) \geq 0 \quad (k = 1, 2, \ldots, K) \tag{3.10}$$

との間の距離を考える．取り扱いを簡単化するために，M 次元空間$\mathbf{u} \in \mathbf{U} \subseteq \mathbf{R}^M$を M−1 次元空間$\mathbf{z} \in \mathbf{Z} \subseteq \mathbf{R}^{M-1}$へ変換する次の直交変換$\Lambda \in \mathbf{R}^{M \times (M-1)}$を用いる．

$$\mathbf{u} = \Lambda\mathbf{z} + \frac{1}{M}\mathbf{e} \tag{3.11}$$

$$(\Lambda, \frac{1}{\sqrt{M}}\mathbf{e})\begin{pmatrix} \Lambda^T \\ \frac{1}{\sqrt{M}}\mathbf{e}^T \end{pmatrix} = 1$$

ただし，$\mathbf{e}$は M 次元単位ベクトルである．この変換により，多面体$\mathbf{U}^1$の中心が回転された座標軸の原点となる[10]（図−3.1 参照）．また，点 $\mathbf{u}'$は

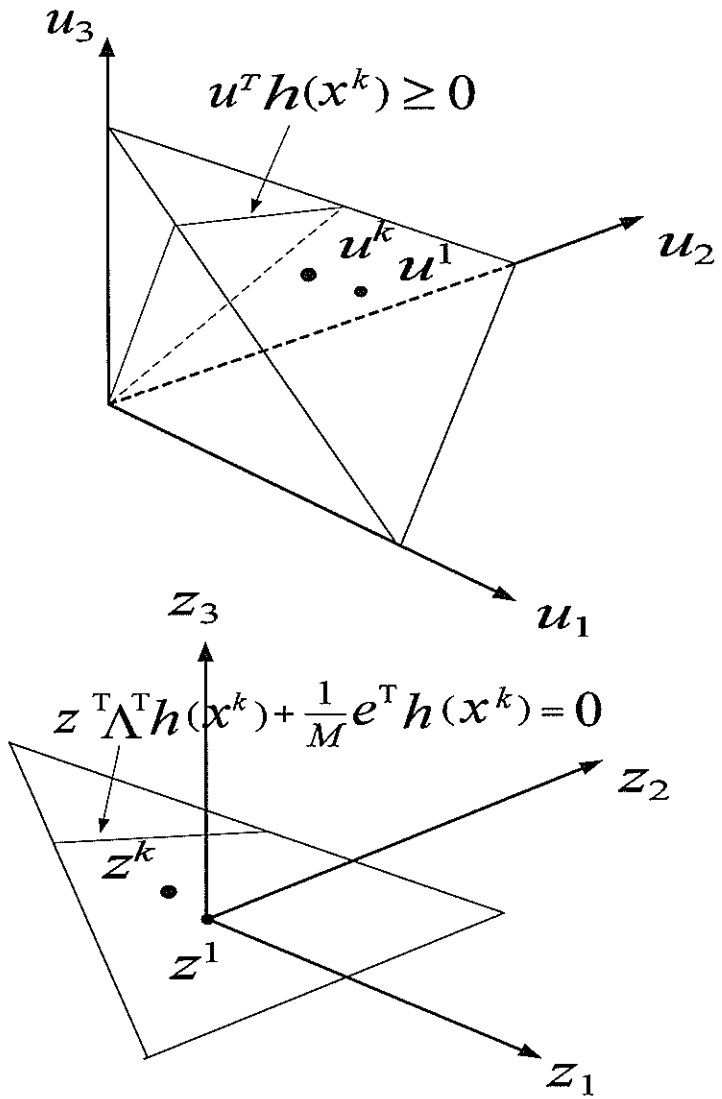
$$\mathbf{z}' = \Lambda^T(\mathbf{u}' - \frac{1}{M}\mathbf{e}) \tag{3.12}$$

となり，切断面を含む切断領域$\mathbf{u}^T h(\mathbf{x}^k) \geq 0$と多面体$U^k$との交線は M−1 次元空間内では

$$\mathbf{z}^T\Lambda^T\mathbf{h}(\mathbf{x}^k) + \frac{1}{M}\mathbf{e}^T\mathbf{h}(\mathbf{x}^k) = 0 \tag{3.13}$$

と表せる．

縮小された多面体の内接円の半径$\mathbf{d}(\mathbf{u}')$は，式（3.13）となる$\mathbf{z} \subseteq \mathbf{U}^k$に対して



図－3.1　代理乗数の回転（M=4）

$$\mathbf{d}(\mathbf{u}') = |\mathbf{u} - \mathbf{u}'| = |\mathbf{z} - \mathbf{z}^k| \tag{3-14}$$

と表される．　このとき，$\mathbf{z} - \mathbf{z}^k$ は $\Lambda^T \mathbf{h}(\mathbf{x}^k)$ に対して平行であるから

$$\mathbf{z} - \mathbf{z}^k = \kappa \Lambda^T \mathbf{h}(\mathbf{x}^k), \quad (\kappa \neq 0) \tag{3.15}$$

と表せる．　また．$\mathbf{z}$ は，式（3.13）の上にあるから

$$\kappa = \frac{\mathbf{z}^T \Lambda^T \mathbf{h}(\mathbf{x}^k) + \frac{1}{M} \mathbf{e}^T \mathbf{h}(\mathbf{x}^k)}{|\Lambda^T \mathbf{h}(\mathbf{x}^k)|^2} \tag{3.16}$$

が成り立つ．それゆえ，$\mathbf{u}'$ と切断領域 $\mathbf{u}^T \mathbf{h}(\mathbf{x}^k) \geq 0$ との距離は，

$$\mathbf{d}^k(\mathbf{u}') = \frac{\mathbf{u}^T h(\mathbf{x}^k)}{\gamma(\mathbf{x}^k)}, \tag{3.17}$$

となる．ここで，

$$\gamma(\mathbf{x}^k) = \sqrt{\mathbf{h}^T(\mathbf{x}^k)\mathbf{h}(\mathbf{x}^k) - (\mathbf{e}^T \mathbf{h}(\mathbf{x}^k))^2 / M}$$

である．このようにして，問題は次の線形計画問題を解くことに帰着される．

$$[LP1] \quad \max \quad 0 \cdot u_1 + 0 \cdot u_2 + \cdots + 0 \cdot u_{M-1} + y \tag{3.18}$$

subject to

$$(h_M - h_1)u_1 + (h_M - h_2)u_2 + \ldots + (h_M - h_{M-1})u_{M-1} \leq h_M$$

$$\mathbf{u} \geq 0, \quad \sum_{j=1}^{M-1} u_j \leq 1$$

しかし，Dyer が用いたこの線形計画問題では，最大の内接球を求める時の切断領域として式(3.18)の制約条件の第１式のみを用いているため，k が小さいとき，求まる最適解は多面体の内部点にはならず，切断面上の点になる問題点があった．Dyer は，その最適解にパラメータを乗じることにより，多面体の内部へ移動し内点を求めている．

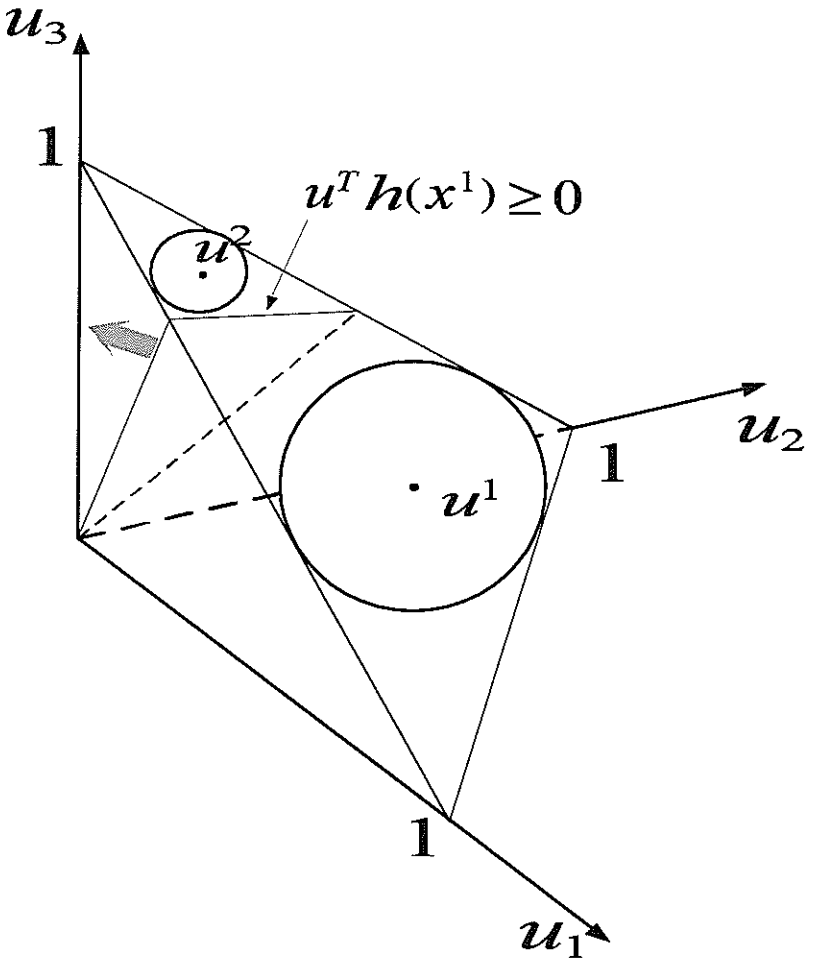
（2）Dyer アルゴリズムの改良 1

本論文では，最大の内接円を求める時に考慮すべき切断領域として式(3. 19)の制約条件の第 1 式から第 4 式を加えることによって，最適解として $\mathbf{U}^k$ の内部点を直接求めることができるようにした次の線形計画問題を用いる．

$$
\begin{aligned}
&[LP2] \quad \max \quad 0\cdot u_1 + 0\cdot u_2 + \cdots + 0\cdot u_{M-1} + y \qquad (3.19)\\
&\text{subject to}\\
&\quad -u_1 + 0\cdot u_2 + \ldots + 0\cdot u_{M-1} + \sqrt{1-1/M}\cdot y \le 0\\
&\quad 0\cdot u_1 - u_2 + \ldots + 0\cdot u_{M-1} + \sqrt{1-1/M}\cdot y \le 0\\
&\qquad \vdots\\
&\quad 0\cdot u_1 + 0\cdot u_2 + \ldots - u_{M-1} + \sqrt{1-1/M}\cdot y \le 0\\
&\quad u_1 + u_2 + \ldots + u_{M-1} + \sqrt{1-1/M}\cdot y \le 1\\
&\quad (h_M(x^k) - h_1(x^k))u_1 + (h_M(x^k) - h_2(x^k))u_2 + \ldots\\
&\qquad + (h_M(x^k) - h_{M-1}(x^k))u_{M-1} + \sqrt{\sum_{j=1}^{M}(h_j(x^k)) - (\sum_{j=1}^{M}h_j(x^k))^2/M}\cdot y \le h_M(x^k)\\
&\qquad u \ge 0, \quad y \ge 0
\end{aligned}
$$

ここで得られた最適解 $\mathbf{u}$ を，次の代理乗数 $\mathbf{u}^{k+1}$ とする．Dyer アルゴリズムを用いた代理乗数 $\mathbf{u}$ の更新過程を図－3. 2.に示す．

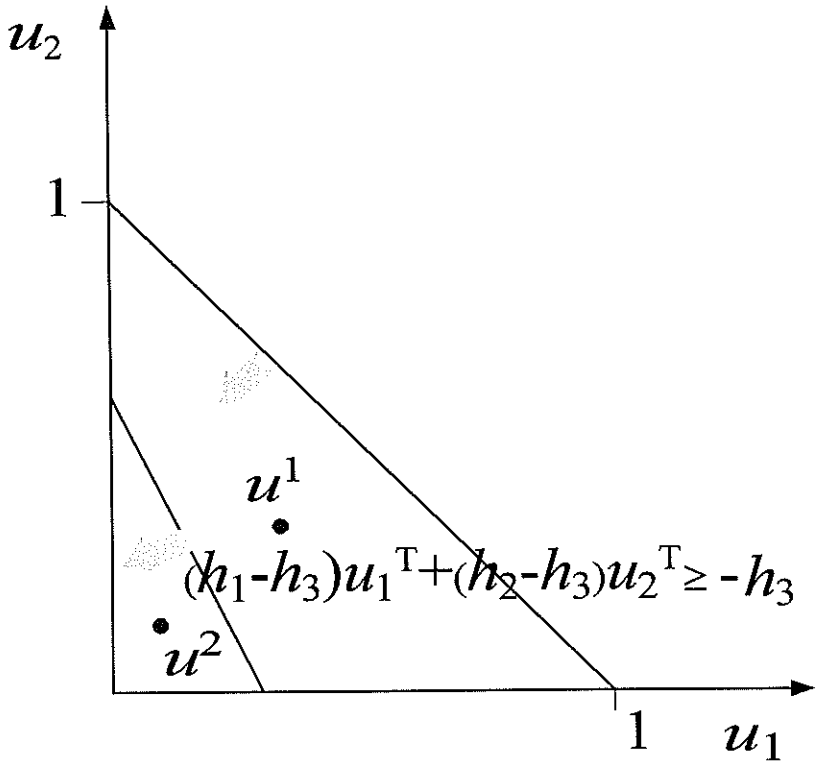
なお，以降で記す Dyer アルゴリズムにはこの Dyer アルゴリズムの改良 1 を施しているものとする．



図－3.2　Dyer アルゴリズムによる代理乗数の更新

（3）COP アルゴリズム[7]

多面体$\mathbf{U}^1$から出発する．第 k 番目の多面体$\mathbf{U}^k$において，$\mathbf{U}^K$のすべての頂点は単位質量の質点系とみなす．このとき，多面体$\mathbf{U}^k$の質点系の重心を代理乗数$\mathbf{u}^k$として用いる．COP アルゴリズムを用いた代理乗数$\mathbf{u}$の更新過程を図－3.3 に示す．



図－3.3　COP アルゴリズムによる代理乗数の更新

3．4　計算機実験の結果と考察

本章で提示した，最適代理乗数を決定するための両アルゴリズムの長所と短所を比較するために，計算機実験を行った．テスト問題は，擬似乱数を生成した問題，及び文献[5]において ISC 法で用いられている総数７１の問題である．なお，本章でのすべての実験において使用した計算機は CPU:Pentium4 1.7GHz のパーソナルコンピュータである．

3．4．1　代理乗数の更新過程

擬似乱数を用いたテスト問題の係数は，Sinha and Zolthers [11] に基づいて次式のように生成した．

$$0 \le f_i(k) \le f_i(k+1) \le 256\,k_i \qquad (3.20)$$
$$(k = 1,2,\ldots,k_{i-1},\ i = 1,2,\ldots,n),$$
$$0 \le g_{ji}(k) \le g_{ji}(k+1) \le 256\,k_i$$
$$(k = 1,2,\ldots,k_{i-1},\ i = 1,2,\ldots,n,\ j = 1,2,\ldots,m),$$
$$b_j = \left\lfloor \frac{1}{2}\sum_{i=1}^{n}\left(g_{ji}(1) + g_{ji}(k_i)\right) \right\rfloor$$
$$(i = 1,2,\ldots,n,\quad j = 1,2,\ldots,m)$$

ただし，f_i, g_{ji}はそれぞれ目的関数，制約関数式の係数であり，すべて正整数である．k_iは変数x_nに対して選択することができる項目数である．

例題として，制約条件の数が 3，変数の数が 5，各変数に対して選択することができる項目数が４である問題の係数を表－3.1～3.3 に示す．

表－3.1　例題の係数 $f_n(a)$

$f_n(a)$	a			
	1	2	3	4
$f_1(a)$	43	62	96	99
$f_2(a)$	39	49	88	118
$f_3(a)$	39	52	57	110
$f_4(a)$	19	57	108	112
$f_5(a)$	51	54	89	94

表－3.2　例題の係数 $g_{mn}(a)$

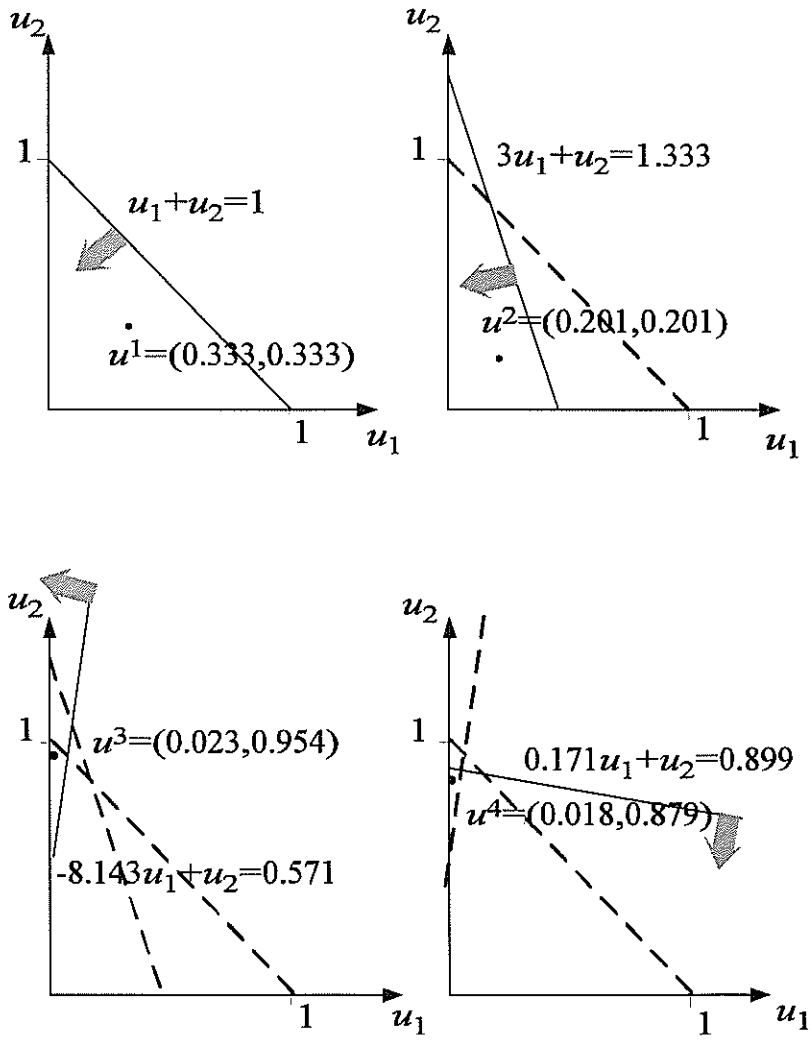
$g_{mn}(a)$		a			
		1	2	3	4
$g_{1n}(a)$	$g_{11}(a)$	51	60	72	127
	$g_{12}(a)$	26	70	74	92
	$g_{13}(a)$	14	57	100	116
	$g_{14}(a)$	56	86	89	106
	$g_{15}(a)$	26	114	124	128
$g_{2n}(a)$	$g_{21}(a)$	32	48	69	126
	$g_{22}(a)$	4	29	46	105
	$g_{23}(a)$	50	81	94	116
	$g_{24}(a)$	25	53	54	81
	$g_{25}(a)$	29	33	94	95
$g_{3n}(a)$	$g_{31}(a)$	58	83	109	114
	$g_{32}(a)$	13	51	92	93
	$g_{33}(a)$	6	86	94	127
	$g_{34}(a)$	18	31	49	97
	$g_{35}(a)$	19	67	101	111

表－3.3　例題の係数 b_m

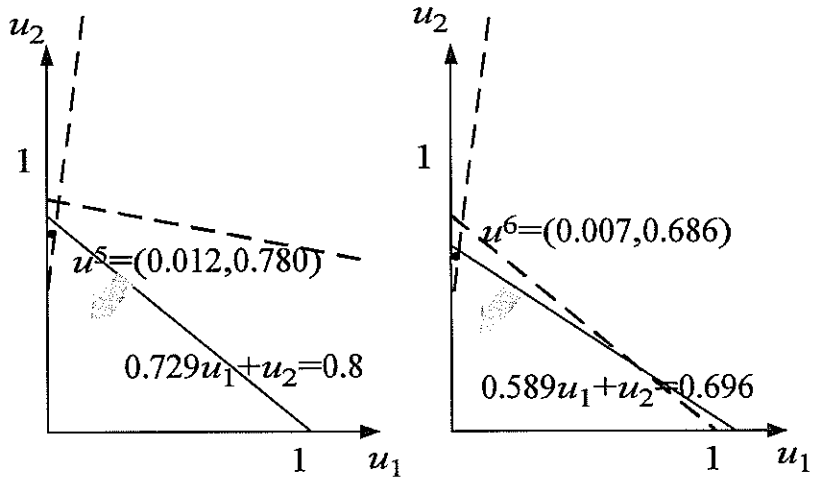
	b_m
1	371
2	331
3	328

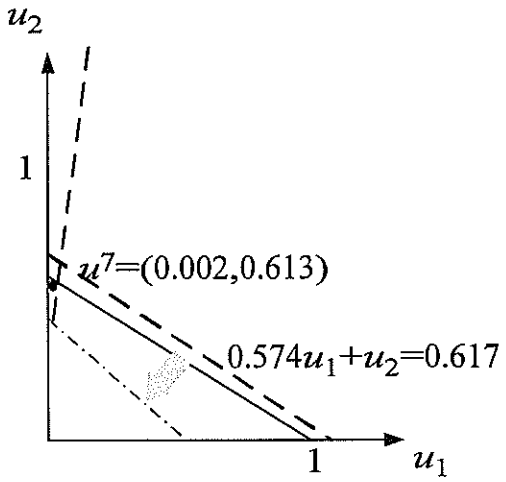
この問題を Dyer アルゴリズムと COP アルゴリズムとを用いて，最適な代理乗数が求まるまでの更新過程をそれぞれ 図－3.4 ～3.6 に示す．各図において，切断面が順次更新されて多面体が縮小され，代理乗数が更新されている．図－3.5 及び図－3.6 の最後の図における一点鎖線は，代理乗数が得られたあとに得られる切断面を表している．この切断面は，縮小された多面体とは交差せず空集合となるので，その前の切断面で得られた代理乗数が最適である．

図－3.4～3.6 から，最適な代理乗数が求まるまでの更新回数は，Dyer アルゴリズムを用いるときの方が，COP アルゴリズムを用いるときよりも多いことが分かる．これは，多面体を切断して縮小していく過程で，Dyer アルゴリズムにおいて内接する円の半径の大きさが変化しない状態が続き，多面体の領域が狭まらなくなるためである．

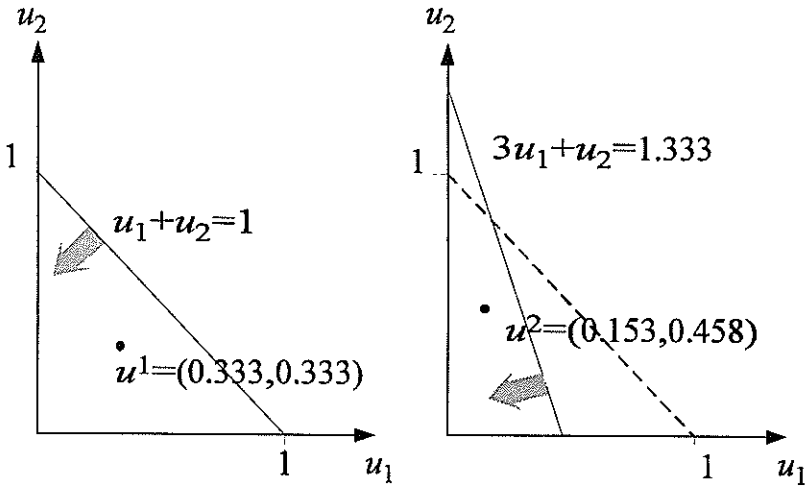


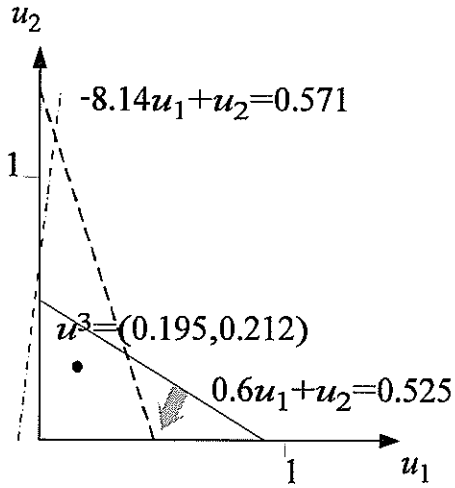
図－3.4　例題における Dyer アルゴリズムによる代理乗数の更新過程（1）





図－3.5　例題における Dyer アルゴリズムによる代理乗数の更新過程（2）





図－3.6　例題における COP アルゴリズムによる代理乗数の更新過程

3．4．2　両アルゴリズムの比較（擬似乱数で生成したテスト問題の場合）

次に，制約条件の数がM=3, 5, 8 ，変数の数が N=10, 20, 50, 80, 100 である場合にSinha and Zolthers の擬似乱数を用いて生成した 10 問のテスト問題について，ISC 法を用いて最適な目的関数値が求めた．その目的関数値が求まるまでの両アルゴリズムの代理乗数の更新回数と実行時間との平均値を表－3.4 ～3.6 に示す．

表－3.4　擬似乱数問題に対する両アルゴリズムの計算比較
（制約条件数が 3 の場合）

	制約条件数	3				
	変数	10	20	50	80	100
更新回数	Dyer	8.5	14.6	15.2	17.8	18.4
	COP	8	11.3	13.2	15.2	15.9
実行時間（秒）	Dyer	0.1	0.3	0.9	1.7	2.9
	COP	0	0.2	0.8	1.8	2.5

表－3.5　擬似乱数問題に対する両アルゴリズムの計算比較
（制約条件数が 5 の場合）

	制約条件数	5				
	変数	10	20	50	80	100
更新回数	Dyer	17.8	30.3	43.5	42.7	51
	COP	17.2	21.7	29	33.8	31.8
実行時間（秒）	Dyer	0.3	0.8	2.6	4.9	8.6
	COP	0.3	0.4	1.6	4	5.1

表－3.6　擬似乱数問題に対する両アルゴリズムの計算比較
（制約条件数が 8 の場合）

	制約条件数	8				
	変数	10	20	50	80	100
更新回数	Dyer	35.2	59.4	86.6	121	114
	COP	36.5	44.6	58.9	67	65
実行時間（秒）	Dyer	0.6	2.1	9.1	27.7	28.4
	COP	1.5	2.2	10.8	16.3	23.7

これらの結果から，以下のことが分かる．なお，両アルゴリズムによって得られた最適解に対する目的関数値はすべての問題において等しいことが確認されている．

(1) 最適な代理乗数が求まるまでの更新回数は，両アルゴリズムとも変数の数と制約条件の数が増えると増大する．変数の数の増加よりも制約条件の数の増加の方が，更新回数に及ぼす影響は大きい．

(2) 代理乗数の更新回数は，COP アルゴリズムの方が Dyer アルゴリズムよりも少ない．

(3) 制約条件の数が増えると，実行時間は両方のアルゴリズムとも飛躍的に増大する．これは，制約条件の数が増加するにつれて，代理乗数を求めるときの問題のサイズが大きくなるためである．

(4) 実行時間については，COP アルゴリズムの方が Dyer アルゴリズムよりも短い．これは，COP アルゴリズムの更新回数が，Dyer アルゴリズムの更新回数よりも少ないことに起因している．

(5) COP アルゴリズムは，制約条件数が 8 より多い問題に対してはメモリ不足により解くことができなかった．一方，Dyer アルゴリズムは１５制約の問題まで解くことができ，それより多くなると実用的な実行時間内で解けなくなった．これは，COP アルゴリズムが各頂点の位置情報（座標）を保持し，その値から代理乗数を求めているのに対して，Dyer アルゴリズムは各平面の情報（平面式）を保持しているだけであり作業領域量が少ないためである．

(6) COP アルゴリズムは最適な代理乗数を求めるまでの更新回数及び実行時間は Dyer アルゴリズムに比べて優れているが，メモリの作業領域の必要量が Dyer アルゴリズムに比べて多い．

3．4．3　両アルゴリズムの比較（文献［5］のテスト問題の場合）

両アルゴリズムの特性を更に客観的に比較するために，文献[5]で用いられている４つのグループからなる総数 71 のテスト問題について，ISC 法を用いて最適解を求めた．なおテスト問題のすべては，次の Web サイトから得られる．

http://www.res.kutc.kansai-u.ac.jp/~nakagawa/orlib/jorsj/

これらの問題の最適解が求まるまでの両アルゴリズムの代理乗数の更新回数と実行時間に顕著な差がみられた問題（総数 42 問）を表－3.7 に示す．

残りの問題（総数 29 問）は更新回数と実行時間が同一であるか，ほとんど差がなかった．なお，両アルゴリズムによって得られた最適解に対する目的関数値は，すべての問題において等しいことが確かめられている．

表--3 7 より以下の事項が明らかになった．

(1) 更新回数に関して，グループ４の 6 個の問題を除いた 36 個の問題では COP アルゴリズムの更新回数の方が少なかった．最適代理乗数を求めるまでの更新回数について両アルゴリズムを比較すると，両者は等しいか，あるいは COP アルゴリズムの方が少ない傾向にある．

(2) 実行時間については，両アルゴリズムが同じ実行時間である場合が多かった．Dyer アルゴリズムの実行時間が COP アルゴリズムの実行時間よりも長い問題の数は，実行時間が短い問題の数とのあいだに大差はなかった．

(3) Dyer アルゴリズムの実行時間の方が長い場合，COP アルゴリズムの実行時間との差が大きい場合がある．これは，Dyer アルゴリズムにおいて，多面体を切断して縮小していく過程で，切断面に内接する球の半径がほとんど変化しない状態が続き，多面体の領域が狭まらなくなるためであると考えられる．

表－3.7　テスト問題[５]に対する両アルゴリズムの計算比較

問題番号	制約条件	変数の数	項目数	更新回数		実行時間(秒)	
				Dyer	COP	Dyer	COP
Gr1_01	5	39	2	47	24	0	0
Gr1_02	5	50	2	51	25	1	1
Gr1_03	5	39	6	41	30	0	0
Gr1_04	5	50	6	40	34	0	0
Gr1_06	4	50	11	46	26	2	1
Gr2_17	3	250	20	18	14	5	4
Gr2_18	3	250	20	31	16	7	4
Gr2_21	3	500	20	26	17	23	15
Gr2_22	3	500	20	27	17	23	14
Gr2_25	3	500	20	19	16	16	13
Gr2_26	3	1000	20	20	18	74	69
Gr2_27	3	1000	20	23	21	87	82
Gr2_28	3	1000	20	27	21	103	83
Gr2_29	3	1000	20	26	18	99	71
Gr2_30	3	1000	20	23	21	89	83
Gr3_01	3	1000	20	20	15	99	59
Gr3_02	3	1000	20	16	14	62	59
Gr3_05	3	1000	20	26	15	101	63
Gr4_02	8	20	2	31	16	0	0
Gr4_04	8	20	2	42	19	0	1
Gr4_05	8	20	2	10	7	0	0
Gr4_07	8	100	2	40	47	1	1
Gr4_10	8	100	2	29	36	1	1
Gr4_11	8	500	2	60	71	8	11
Gr4_12	8	500	2	70	52	10	11
Gr4_13	8	500	2	135	60	24	13
Gr4_14	8	500	2	99	57	15	15
Gr4_15	8	500	2	87	60	13	12
Gr4_16	8	20	50	25	40	0	1
Gr4_17	8	20	50	22	37	0	0
Gr4_18	8	20	50	40	21	0	0
Gr4_19	8	20	50	48	31	0	1
Gr4_20	8	20	50	39	32	0	1
Gr4_21	8	100	50	28	45	0	1
Gr4_22	8	100	50	35	31	1	1
Gr4_23	8	100	50	73	54	2	3
Gr4_24	8	100	50	67	39	2	1
Gr4_25	8	100	50	109	62	5	5
Gr4_27	8	500	50	93	68	35	27
Gr4_28	8	500	50	99	67	38	27
Gr4_29	8	500	50	95	66	35	26
Gr4_30	8	500	50	77	66	30	30

3．5　Dyer アルゴリズムの改良２

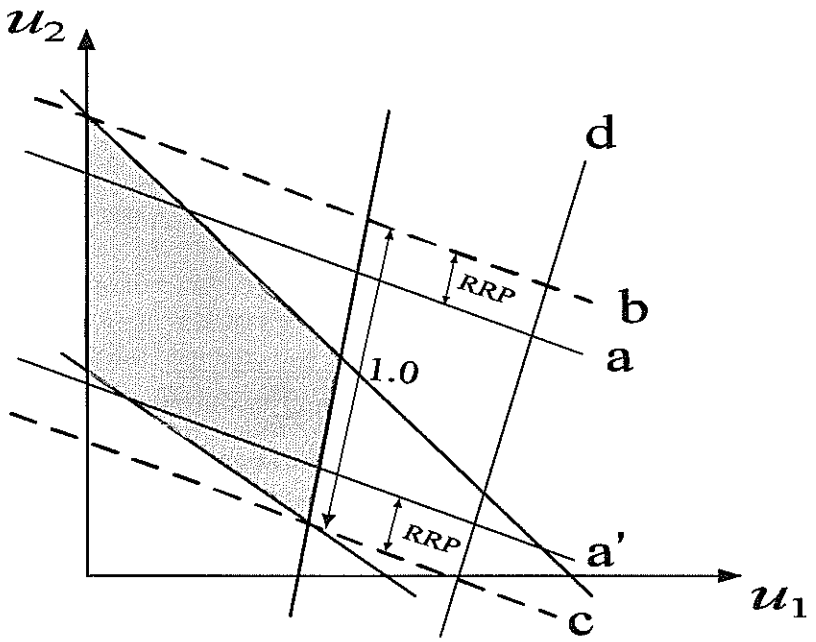
3．5．1　改良 Dyer アルゴリズム

3．4．2において Dyer アルゴリズムでは，原問題における制約条件式の数が 15 より多くなると，代理乗数を求める過程において，計算速度の低下と使用メモリ量が激増することが分かった．これは，制約条件式の個数が増えるにつれて，解くべき問題のサイズが大きくなるためである．

本論文において，制約条件式の数が多い問題を解くために，Dyer アルゴリズムの改良２を行い，計算速度の向上と使用メモリ量を減少させる．この改良を施したアルゴリズムを改良 Dyer アルゴリズムと呼び，３．３．２の Dyer アルゴリズムの改良１と区別する．

改良 Dyer アルゴリズムは，代理乗数を計算する過程において，内接する球（円）の半径の大きさの決定に全く関係していない切断面，及び関係しているがその割合が小さい切断面を削除する．これらの切断面を削除することにより，考慮する制約条件式の候補数が減るため，代理乗数を求めるときの使用メモリ量が減少し，実行時間も短縮される．説明の便宜上，３次元射影した２次元の平面（図－3.7）を考える．まず，対象とする切断面（直線）ａを目的関数とし，その他の切断面（直線）をａに対する制約条件と考える．制約条件を表す切断面のうち目的関数ａが目的関数の最大値をとる切断面をｂとし，最小値をとる切断面をｃとする．切断面ａが制約条件内（実行可能領域内，陰影部分）に収まるかどうかを判定し，収まらなければ削除する．図－3.7 における切断面ｄがこれに該当し，その切断面を削除しても内接する球の半径の最大値が変化せず，代理乗数の決定に関与していない．

上述の代理乗数の決定に全く関与していない切断面は，実際にはごく少数である．制約条件の実行可能領域を満足している切断面についても，次の判定基準値があらかじめ設定した値よりも小さければ，代理乗数の決定に関与する割合が小さいので削除する．判定基準（式（3.22）の左辺）は，切断面ａに対する制約条件の実行可能領域内での目的関数の最大値 Z^U（図－3.7 の切断面ｂ）と最小値 Z^L（図－3.7 の切断面ｃ）を考え，考慮中の切断面ａがｂ－ｃ領域内でどれくらい関与しているかを表している．この判定基準を RP 率，RRP（Ratio of Reducing Plane）と名づける．



図－3.8　改良 Dyer アルゴリズム

$$Z^U = \max \sum_{m=1}^{M-1} \{h_M(\mathbf{x}^{k'}) - h_m(x^{k'})\} u_m \qquad (3.21)$$

OR

$$Z^L = \min \sum_{m=1}^{M-1} \{h_M(\mathbf{x}^{k'}) - h_m(\mathbf{x}^{k'})\} u_m \qquad (3.21')$$

subject to

$$-u_1 + 0 \cdot u_2 + \ldots + 0 \cdot u_{M-1} \le 0$$

$$\vdots$$

$$0 \cdot u_1 + 0 \cdot u_2 + \ldots - u_{M-1} \le 0$$

$$u_1 + u_2 + \ldots + u_{M-1} \le 1$$

$$\sum_{m=1}^{M-1} \{h_M(\mathbf{x}^k) - h_m(\mathbf{x}^k)\} u_m + \sqrt{\sum_{j=1}^{M} (h_j(x^k)) - (\sum_{j=1}^{M} h_j(x^k))^2 / M} \cdot y \le h_m$$

$$(k = 1, \ldots, K, k \ne k')$$

$$\mathbf{u} \ge 0$$

この線形計画問題を解いて，最大値$\mathbf{Z}^U$と最小値$\mathbf{Z}^L$を求め，RRPの値r_{RP}(式(3.22)の右辺)をあらかじめ設定すると，

$$\frac{Z^U - h_m}{Z^U - Z^L} \le r_{RP} \quad , \quad \frac{h_m - Z^L}{Z^U - Z^L} \le r_{RP} \qquad (3.22)$$

となるh_mを表す切断面(図－3.7のaあるいはa')が削除される．

代理乗数の決定にあまり関与しない切断面を削除する操作は，代理乗数を計算するたびに行うのではなく，Dyerアルゴリズムを何回か実行した後，周期的に切断面を削除する操作を行う．この削除の操作と次の削除の操作との間に実行するDyerアルゴリズムの回数をRP周期，PRP（Period of Reducing Plane）と呼び，その設定する値をp_{RP}と表す．

3．5．2　計算機実験の結果と考察

本論文で提案した改良Dyerアルゴリズムの有効性を確かめるために，3．4．1で用いた擬似乱数のテスト問題（式(3.20))を用いる．

変数の数nは50.100.200.300.400.500，制約条件式の数 m は15及び20，項目の数A_nは10とする．改良Dyerアルゴリズムを用いて解くためのパラメータとして，RP率(r_{RP})を0.2，RP周期(p_{RP})を30とする．本論文でのDyerアルゴリズムについては，改良Dyerアルゴリズムにおいて RP周期の値を非常に大きな値 $p_{RP} = 5000$ として実験を行った．

擬似乱数を用いて生成した10問の代理問題に対して，Dyerアルゴリズム及び改良Dyerアルゴリズムを用いて最適解を求めた．その代理最適解が求まるまでの実行時間の平均値を表－3.8に示す．表中*印は，Dyerアルゴリズムの実行途中において，プログラムの配列オーバーのエラーが生じ，最適解が求められなかったことを示す．前述のとおり，Dyerアルゴリズムと改良Dyerアルゴリズムは同じプログラムを用いているが，Dyerアルゴリズムにおいて制約条件式の数が多い15.20のとき，変数の数が増えるにしたがって，式(3.21)で示される線形計画法のサイズが大きくなったため，エラーが生じたものと考えられる．

表－3.8　Dyerアルゴリズム及び改良Dyerアルゴリズムによる平均実行時間（単位：秒）

制約条件の数	変数の数	Dyerアルゴリズム	改良Dyerアルゴリズム r_{RP}=0.2 p_{RP}=30
15	50	54.2	6.9
	100	366.6	33.6
	200	*	84.9
	300	*	141.9
	400	*	224.6
	500	*	330.8
20	50	292.4	36.3
	100	*	178
	200	*	607.5
	300	*	1623.8
	400	*	3386.9
	500	*	4076.3

表－3.8より，本論文で提案した改良Dyerアルゴリズムは Dyerアルゴリズムよりも短時間で代理問題を解くことができることが分かった．さらに，Dyerアルゴリズムでは制約条件式及び変数の数が多いため解けなかった問題が，改良Dyerアルゴリズムを用いることにより，実用的な時間内で解けることが分かった．

3．6　結語

変数分離型の多次元非線形整数計画問題に代理制約法を用いて解く場合，代理双対ギャップが存在することが多く，代理双対問題の最適解は原問題の実行可能解とはならない．最近，仲川は，変数分離型の多次元非線形整数計画問題の厳密解を求めることができる改良代理制約法（ISC 法）を提案した．この ISC 法を効率的に実行するためには，その中で最適な代理乗数を求めることが必要である．

本論文では，この ISC 法あるいは代理制約法において最適な代理乗数を決定するためのアルゴリズムとして知られる，Dyer アルゴリズムと COP アルゴリズムを示し，計算機実験を通して比較検討した．テスト問題としては，擬似乱数を生成した問題，及び文献[5]において ISC 法で用いられている総数 71 の問題を用いた．

これらの結果から，以下のことが明らかになった．

(1) 最適な代理乗数が求まるまでの更新回数及び実行時間は，両アルゴリズムとも変数の数と制約条件の数が増えると増大する．これらに及ぼす影響は，変数の数の増加よりも制約条件の数の増加の方が大きい．

(2) 最適な代理乗数が得られるまでの更新回数及び実行時間ともに，COP アルゴリズムの方が Dyer アルゴリズムよりも少ない場合が多い．

(3) 制約条件式の数が増えると，Dyer アルゴリズムでは最適な代理乗数が得られる問題でも，COP アルゴリズムはメモリ不足で解くことができない場合があった．これは，計算過程において，COP アルゴリズムが多面体の頂点の座標情報を使っているため，切断面の平面式を使う Dyer アルゴリズムよりも計算の作業領域量が多くなるためである．

(4) Dyer アルゴリズムの実行時間が COP アルゴリズムの実行時間よりもかなり長くなる場合があった．これは，Dyer アルゴリズムにおいて，代理乗数を計算する過程において，切断面に内接する球の半径がほとんど変化しない状態が続き，多面体の領域が狭まらなくなるためであると考えられる．

本研究では，上記の Dyer アルゴリズムの短所を改善するため改良 Dyer アルゴリズムを提案した．この改良 Dyer アルゴリズムは，代理乗数を決定する過程において，内接する球の半径の大きさの決定に全く関係していない切断面，及び関係しているがその割合が小さい切断面を削除する．この改良により，代理乗数を計算するときに現れる切断面の候補数が減少するため使用メモリ量が減少し，実行時間も短縮される．

計算機実験において制約条件式及び変数の数が多い問題を解くことによって，改良 Dyer アルゴリズムが Dyer アルゴリズムに比べて，計算速度及び使用メモリ量ともに優れていることが明らかになった．

参考文献

[1] F. Glover. "Surrogate constraints." Operations Research. Vol.16. pp.741-749. 1968.

[2] D. G. Luenberger, "Quasi−convex programming", SIAM J. of Appl. Math., Vol. 16, pp.1090−1095,1968.

[3] M.H. Karwan, R.L. Rardin, "Surrogate dual multiplier search procedures in integer programming," Operations Research, Vol.32, pp.52−69, 1984.

[4] Y. Nakagawa, "A reinforced surrogate constraints method for separable nonlinear integer programming", RIMS, Kokyuroku 1068 Kyoto Univ., pp.194−202,1998.

[5] Y. Nakagawa, "An improved surrogate constraints method for separable nonlinear integer programming", J. Oper. Res. Soc. Jpn, Vol.46, No.2, pp.145−163 ,2003.

[6] M.E. Dyer, "Calculating surrogate constraints", Mathematical Programming, Vol.19, pp.255−278,1980.

[7] 仲川 勇二. 疋田 光伯. 鎌田 弘. "代理双対問題を解くためのアルゴリズム". 電子情報通信学会論文誌 Vol. J67−A No. 1. pp.53−59. Jan. 1984.

[8] 仲川 勇二. "離散最適化問題のための新解法". 電子情報通信学会論文誌. Vol. J73A. No. 3. pp.550−556. March 1990.

[9] Y. Nakagawa and A. Iwasaki, "Modular Approach for Solving Nonlinear Knapsack Problems, "IEICE Trans.Fundamentals, Vol.E82-A, No.9, pp.1860−1864, Sep. 1999.

[10] P. Wolfe, "The ellipsoid algorithm", OPTIMA(Mathematical Programming Society News)

[11] P. Sinha and A. A. Zoltoner, "The Multiple Choice Knapsack Problem", Oper. Res., Vol. 27, No.3, pp.503−515, May-June 1979.

[12] T. L. Magnanti, J. F. Shapiro and M. H. Wagner, "Generalized Linear Programming solves the Dual", Management Science, Vol. 22, No. 11, pp.1195−1202, July 1976.

[13] T. H. Mattheiss and W. B. Widhelm, "The Generalized Slack Variable Linear Program", Management Science, Vol. 23, No. 8, pp.859−871, April 1977.

[14] I. Miyaji, Y. Nakagawa and K. Ohno, " Decision support system for the composition of the examination problem",European J. Oper. Res., vol.80, pp.130−138, 1995.

[15] 仲川勇二，並川哲郎，木村作郎，太田垣博一，"代理制約法における最適代理乗数の決定法".電子情報通信学会論文誌，Vol. J87−A. No.3. pp.364−374. 2004.

Y. Nakagawa. T. Namikawa. S. Kimura. and H. Ohtagaki." A method for determining optimal multipliers in the surrogate constraint method." Electronics and Communications in Japan (Part III: Fundamental Electronic Science). Vol.88. Issue 8 . pp.38−48. published Online: 18 Apr 2005.

第４章　代理制約法における代理乗数決定のための改良Dyerアルゴリズムの特性評価[15]

４．１　概説

近年，複数制約条件付きの分離型非線形整数計画問題に対して，品質のよい解を得ることがますます重要になってきている．

代理制約法は，代理乗数を用いて原問題の複数制約条件式を単一制約条件式とした代理問題に変換する解法であり，大変計算効率がよい[1,2]．原問題が準凸であるときは代理乗数を適当に決定すれば，代理問題の最適解は原問題の最適解と一致することが示されている[3]．しかし，離散問題に代理制約法を適用した場合，代理双対問題に代理双対ギャップ(surrogate duality gap)が存在することが多く，その場合の代理双対問題の最適解は原問題の最適解に一致せず，また実行可能解にはならない．この代理双対ギャップを閉じ、原問題の厳密解を求めることができる改良代理制約法(ISC法, Improved Surrogate Constraint Method）が，最近仲川によって提案された[5, 6]．しかし，このISC法においても，制約条件式あるいは変数の個数が多いとき最適な代理乗数を決定するためのアルゴリズムにおいて，実行時間や計算時に必要とする作業領域の量が多大になり，代理乗数を決定することができなくなることが多い．

第３章において，代理制約法において最適な代理乗数を決定するアルゴリズムとして，Dyerアルゴリズムと COP(Cutting-Off Polyhedron)アルゴリズムを示し，計算機実験により，両アルゴリズムの長所と短所を明らかにした．

(1) 最適な代理乗数が求まるまでの更新回数及び実行時間は，両アルゴリズムとも変数の数と制約条件式の数が増えると増大する．これらに及ぼす影響は，変数の数の増加よりも制約条件式の数の増加の方が大きい．

(2) 最適な代理乗数が得られるまでの更新回数及び実行時間ともに，COP アルゴリズムの方がDyerアルゴリズムよりも少ない場合が多い．

(3) 制約条件式の数が増えると，Dyer アルゴリズムでは最適な代理乗数が得られる問題でも，COPアルゴリズムはメモリ不足で解くことができない場合があった．これは，計算過程において，COPアルゴリズムが多面体の頂点の座標情報を使っているため，切断面の平面式を使うDyerアルゴリズムよりも計算の作業領域量が多くなるためである．

(4) Dyer アルゴリズムの実行時間が COP アルゴリズムの実行時間よりもかなり長くなる場合があった．これは，Dyer アルゴリズムにおいて，代理乗数を計算する過程において，切断面に内接する球の半径がほとんど変化しない状態が続き，多面体の領域が狭まらなくなるためであると考えられる．

さらに，第３章において，上記のDyerアルゴリズムの短所を改善するため改良Dyerアルゴリズムを提案した．この改良Dyerアルゴリズムは，代理乗数を決定する過程において，内接する球の半径の大きさの決定に全く関係していない切断面，及び関係しているがその割合が小さい切断面を削除する（代理乗数の計算を行わない）改良である．この改良により，代理乗数を計算するときに検討する切断面の候補数が減少するため使用作業量が減少し，実行時間も短縮される．３章での計算機実験において，制約条件式及び変数の数が多い問題を解くことによって，改良Dyerアルゴリズムが Dyerアルゴリズムに比べて，計算速度及び使用メモリ量に関して優れていることが明らかになった．

本章において，制約条件式と変数の数を増やした計算機実験を行い，改良Dyerアルゴリズムが計算速度及び使用メモリ量に関して Dyer アルゴリズムよりも優れていることを示す．また，改良Dyerアルゴリズムにおいて用いられるパラメータ（RP率及びRP周期）による，最適解を求めるための実行時間，及び実行回数に対する影響を示す．さらに，大規模な変数分離型非線形整数計画問題を解くときに有効となるＲＰ率とＲＰ周期の値の組合せも示す．

4．2　問題

本論文で解くべき変数分離型の多次元非線形整数計画問題は次式のように定式化される．

$$[P] \quad \text{maximize} \quad f(\mathbf{x}) \qquad (4.1)$$
$$\text{subject to} \quad \mathbf{g}(\mathbf{x}) \le \mathbf{b},$$
$$\mathbf{x} \in \mathbf{X},$$

ここで，$\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$ は，決定空間 $\mathbf{X}$ での N 次元整数値変数ベクトル，$f(\mathbf{x})$ は整数値目的関数，$\mathbf{g}(\mathbf{x}) = (g_1(x), g_2(x), \ldots, g_M(x))^T$ は M 次元整数値ベクトル制約関数，$\mathbf{b} = (b_1, b_2, \ldots, b_M)^T$ は M 次元整数値ベクトル制約許容量で，$\mathbf{X}$ はN次元離散決定空間である．この原問題[P]を代理乗数 $\mathbf{u} = (u_1, u_2, \ldots, u_M)^T$ を用いて次の代理問題に変換される．

$$[P^S(u)] \quad \text{maximize} \quad f(\mathbf{x}) \qquad (4.2)$$
$$\text{subject to} \quad \mathbf{u}^T\mathbf{h}(\mathbf{x}) \le 0$$
$$\mathbf{x} \in \mathbf{X},$$

ただし，

$$\mathbf{h}(\mathbf{x}) = \mathbf{g}(\mathbf{x}) - \mathbf{b}$$
$$\mathbf{u} \in U = \{\mathbf{u} \in \mathbf{R}^M \mid \sum_{m=1}^{M} u_m = 1, u \ge 0\}$$

である．代理制約法は，複数の制約条件式を単一の制約条件式で代理させて，複数制約の問題を解く方法である．

原問題 [P] の代理双対問題 $[P^{SD}]$ は次のようになる．

$$[P^{SD}] \quad \min\{opt[P^S(\mathbf{u})] : \mathbf{u} \in \mathbf{U}\} \qquad (4.3)$$

ただし，$opt[P^S(\mathbf{u})]$ は代理問題 $[P^S(\mathbf{u})]$ の最適解の目的関数値である．

代理問題 $[P^S(u)]$ に対して次の定理が成り立つ[6]．

［定理 4-1］ $\mathbf{x}^k$ を代理乗数ベクトル $\mathbf{u}^k \in \mathbf{U}$ に対する問題 $[P^S(\mathbf{u}^k)]$ の最適解とする．$\mathbf{u}^T\mathbf{g}(\mathbf{x}^k) \le \mathbf{u}^T\mathbf{b}$ となる任意の $\mathbf{u} \in \mathbf{U}$ に対して，次式が成り立つ．

$$opt[P^S(\mathbf{u})] \ge f(\mathbf{x}^k) \qquad (4.4)$$

この定理 4-1 は，最適な代理乗数が属する多面体は領域 $\{\mathbf{u} \in \mathbf{U} : \mathbf{u}^T\mathbf{g}(\mathbf{x}^k) \le \mathbf{u}^T\mathbf{b}\}$ を多面体 U から除外した多面体となることを意味する．この定理を利用したアルゴリズム[8]は，代理双対問題 $[P^{SD}]$ を厳密に解くことができる．このアルゴリズムにおいては，初期多面体を $\mathbf{U}^1 = \mathbf{U}$ とする．代理乗数 $\mathbf{x}^k$ として k 番目の多面体 $\mathbf{U}^k$ の頂点の重心を用いると，代理問題 $[P^S(\mathbf{u}^k)]$ の最適解 $\mathbf{x}^k$ が得られ，切断面を含む切断領域は $\mathbf{u}^T\mathbf{g}(\mathbf{x}^k) \ge \mathbf{u}^T\mathbf{b}$ となる．この切断面の超平面(hyperplane)によって，次の多面体 $\mathbf{U}^{k+1}$ を作るために多面体 $\mathbf{U}^k$ を切断する．

定理 4-1 から U のすべてをカバーする代理乗数の有限集合多面体 $u^1, u^2, \ldots, u^k$ が得られる．代理乗数 $\mathbf{u}^*$ は，

$$\text{opt}[P^S(\mathbf{u}^*)] = \min\{\text{opt}[P^S(\mathbf{u}^1)], \text{opt}[P^S(\mathbf{u}^2)], \ldots, \text{opt}[P^S(\mathbf{u}^k)]\} \qquad (4.5)$$

が代理双対問題 $[P^{SD}]$ に対して最適であるように定義される[6]．

［定理 4-2］もし問題 $[P^S(u^*)]$ の最適解 $\mathbf{x}^{SD}$ が，元の複数制約問題[P]に対して実行可能ならば，$\mathbf{x}^{SD}$ は[P]の厳密な最適解である[6]．

4．3　代理乗数決定アルゴリズムの概要

4．3．1　切断面

ある多面体U^kの重心$u^k \in R^M$に対する代理問題$[P^S(\mathbf{u}^k)]$はすでに解かれているものとする．そこで，次の定理が成り立つ[8].

[定理 4-3]　$[P^S(\mathbf{u}^k)]$の解を$\mathbf{x}^k$とし，

$$\overline{\mathbf{H}}^k = \{\mathbf{u} \in \mathbf{U}^k : \mathbf{u}^T\mathbf{h}(\mathbf{x}^k) \leq 0\} \tag{4.6}$$

とする．このとき，任意の$u \in \overline{H}^k$において

$$\mathrm{opt}[P^S(\mathbf{u}^k)] \leq \mathrm{opt}[P^S(\mathbf{u})] \tag{4.7}$$

が成り立つ[8].

[系 4-1]　$\mathbf{H}^k = \{\mathbf{u} \in \mathbf{U} : \mathbf{u}^T\mathbf{h}(\mathbf{x}^k) > 0\}$とする．もし，$\mathrm{opt}[P^S(\mathbf{u})] < \mathrm{opt}[P^S(\mathbf{u}^k)]$となる$\mathbf{u} \in \mathbf{U}$が存在すれば，$\mathbf{u} \in \mathbf{U} \cap \mathbf{H}^k$が成り立つ．

系 4-1により，多面体$\mathbf{U}^k$を切断して新しい多面体$\mathbf{U}^{k+1}$を作る平面は，$\mathbf{u}^T\mathbf{h}(\mathbf{x}^k) > 0$である．

4．3．2　代理制約法のアルゴリズム[7]

代理制約法のアルゴリズムは，以下のとおりである．

1：$k \leftarrow 1$；
　$u^1 \leftarrow (1/M, 1/M, \ldots, 1/M)$；
　$U^1 \leftarrow \{u \in R^M \mid u \geq 0, \sum_{m=1}^{M} u_m = 1\}$；
2：Repeat
3：　代理問題$[P^S(u^k)]$の最適解$\mathbf{x}^k$を求める；
4：　If($\mathbf{x}^k$が問題[P]の実行可能解である)　then
5：　　$\mathbf{x}^k$が問題[P]の厳密解である；　停止；
6：　Endlf
7：　$\mathbf{x}^k$を用いて多面体U^kの切断面を含む切断領域
　　$\mathbf{u}^T\mathbf{h}(\mathbf{x}^k) \geq 0$
　　を求め，Dyerアルゴリズムまたは COP アルゴリズムを用いて，次の代理乗数$\mathbf{u}^{k+1}$を求める；
8：　$k \leftarrow k+1$；
9：　Until（$\mathbf{U}^{k-1}$が空である）
10：　Return($\mathbf{u}^k$)；

このアルゴリズムによって得られた$\mathbf{u}^k$が，代理双対問題の最適な代理乗数である．モジュラ法(Modular Approach)［10］は次の手順を繰り返して，ステップ3における代理問題$[P^S(u^k)]$の厳密解を高速に求めることができる．

(1) 深測操作(fathoming operation)を用いて変数の決定空間を縮小する．
(2) 2つの変数を統合して1つの変数にすることによって，問題の変数の個数を減らす．

4．3．3　代理乗数決定のための Dyer アルゴリズム[7]

多面体 $\mathbf{U}^k$ のもとで次の代理乗数 $\mathbf{u}^{k+1}$ を決定するための Dyer アルゴリズムを示す．
初期多面体として

$$\mathbf{U}^1 \leftarrow \{\mathbf{u} \in \mathbf{R}^M \mid \mathbf{u} \geq 0, \sum_{m=1}^{M} u_m = 1\} \tag{4.8}$$

を用いる[7]．この多面体 $\mathbf{U}^1$ において，半径が最大となる内接する球の中心の座標を代理乗数 $\mathbf{u}^1$ とする．多面体 $\mathbf{U}^k$ に内接する最大の球の中心を求めるには，次の線形計画問題を解くことに帰着される．

$$[LP] \quad \max \quad 0\cdot u_1 + 0\cdot u_2 + \cdots + 0\cdot u_{M-1} + y \tag{4.9}$$

subject to

$$-u_1 + 0\cdot u_2 + \ldots + 0\cdot u_{M-1} + \sqrt{1-1/M}\cdot y \leq 0$$

$$0\cdot u_1 - u_2 + \ldots + 0\cdot u_{M-1} + \sqrt{1-1/M}\cdot y \leq 0$$

$$\vdots$$

$$0\cdot u_1 + 0\cdot u_2 + \ldots - u_{M-1} + \sqrt{1-1/M}\cdot y \leq 0$$

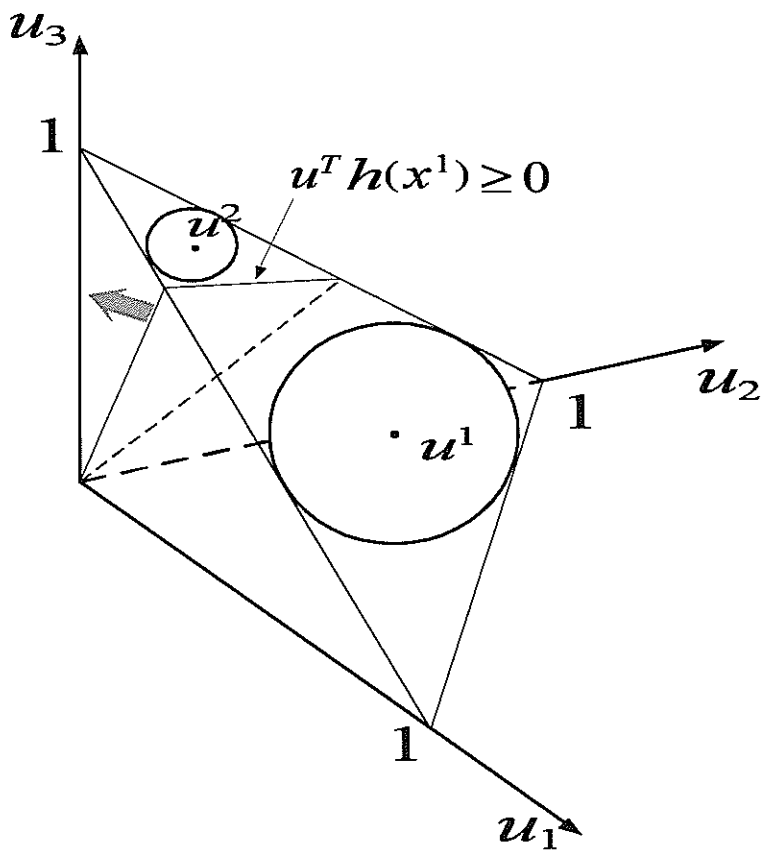
$$u_1 + u_2 + \ldots + u_{M-1} + \sqrt{1-1/M}\cdot y \leq 1$$

$$\sum_{m=1}^{M-1}\{h_M(x^k) - h_m(x^k)\}u_m + \sqrt{\sum_{j=1}^{M}(h_j(x^k)) - (\sum_{j=1}^{M} h_j(x^k))^2/M}\cdot y \leq h_m \quad (k = 1,2,\ldots,K)$$

$$u \geq 0, \quad y \geq 0$$

ここで得られた最適解 $\mathbf{u}$ が，次の代理乗数 $\mathbf{u}^{k+1}$ にする．この Dyer アルゴリズムを用いた代理乗数 $\mathbf{u}$ の更新過程を図－4.1 に示す．

Dyer の論文[4]において示されている多面体 $\mathbf{U}^k$ に対する代理乗数 $\mathbf{u}^k$ は，k が小さいとき多面体の内部に存在せず，切断面上の点になるという問題があった．そのため，得られた $\mathbf{u}^k$ を中心部へ移動する操作が別途必要であり，この操作には経験的な数値設定も必要であった．上述の初期多面体（式（4.8）），及び式（4.9）を解いて得られる多面体 U^k の代理乗数 $\mathbf{u}^k$ では，この問題点を解消して直接内部点を求めることができ，Dyer アルゴリズムをより単純でより高速なものになる．



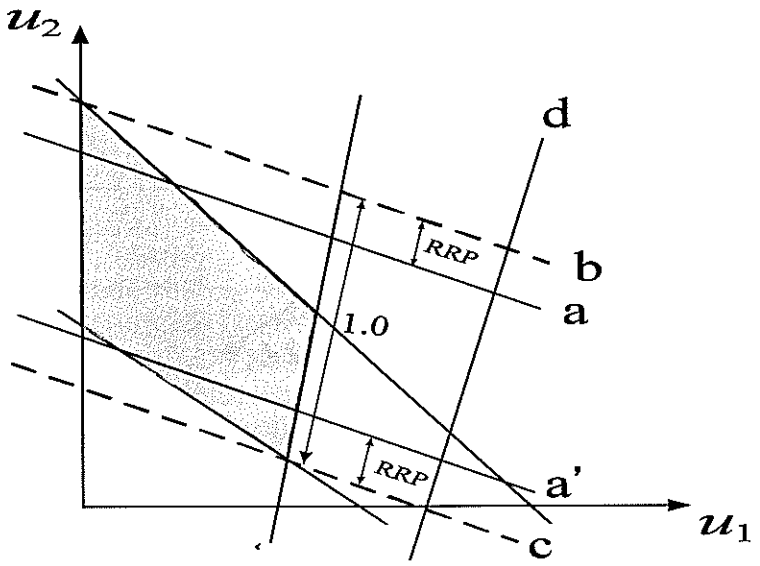
図－4.1　Dyer アルゴリズムによる代理乗数の更新

4．4　改良 Dyer アルゴリズム

Dyer アルゴリズムでは，原問題における制約条件式の数が増加すると代理乗数を求める過程において，計算速度が低下することと使用する作業領域の量が増加することがわかった[7]．これは，制約条件式の個数が増えるにつれて，解くべき問題のサイズが大きくなるためである．

第３章において，制約条件式の数が多い問題を解くために，Dyer アルゴリズムを改良した改良 Dyer アルゴリズムを提案し，計算速度を向上させ，使用メモリ量を減少させることができた．

改良 Dyer アルゴリズムは，代理乗数を計算する過程において，内接する球（円）の半径の大きさの決定に全く関係していない切断面，及び関係しているがその割合が小さい切断面を削除する．これらの切断面を削除することにより，考慮する制約条件式の候補数が減るため，代理乗数を求めるときの使用メモリ量が減少し，実行時間も短縮される．



図－4.2　改良 Dyer アルゴリズム

まず，対象とする切断面（直線）ａを目的関数とし，その他の切断面（直線）をａに対する制約条件と考える．制約条件を表す切断面のうち目的関数ａが目的関数値の最大値をとる切断面をｂとし，最小値をとる切断面をｃとする．切断面ａが制約条件内（実行可能領域内，陰影部分）に収まるかどうかを判定し，収まらなければ削除する．図－4.2 における切断面ｄがこれに該当し，その切断面を削除しても内接する球の半径の最大値が変化せず，代理乗数の決定に関与していない．

制約条件の実行可能領域に含まれる切断面についても，次の判定基準値（式（4.11）の左辺）があらかじめ設定した値よりも小さければ，代理乗数の決定に関与する割合が小さいので削除する．判定基準は，切断面ａに対する制約条件の実行可能領域内での目的関数値の最大値 Z^U（図－4.2 の切断面ｂ）と最小値 Z^L（図－4.2 の切断面ｃ）を考え，考察中の切断面ａがｂ－ｃ領域内でどれくらい関与しているかを表している．この判定基準を RP 率，RRP（Ratio of Reducing Plane）と名付ける．

$$Z^U = \max \sum_{m=1}^{M-1} \{h_M(x^{k'}) - h_m(x^{k'})\} u_m \qquad (4.10)$$

OR

$$Z^L = \min \sum_{m=1}^{M-1} \{h_M(x^{k'}) - h_m(x^{k'})\} u_m \qquad (4.10')$$

subject to

$$-u_1 + 0 \cdot u_2 + \ldots + 0 \cdot u_{M-1} \le 0$$
$$\vdots$$
$$0 \cdot u_1 + 0 \cdot u_2 + \ldots - u_{M-1} \le 0$$
$$u_1 + u_2 + \ldots + u_{M-1} \le 1$$
$$\sum_{m=1}^{M-1} \{h_M(x^k) - h_m(x^k)\} u_m + \sqrt{\sum_{j=1}^{M} (h_j(x^k)) - (\sum_{j=1}^{M} h_j(x^k))^2 / M} \cdot y \le h_m$$
$$(k = 1, \ldots, K, k \ne k')$$
$$u \ge 0$$

この線形計画法を解いて，最大値 Z^U と最小値 Z^L とを求め，RRP の値 r_{RP}（式(4.11)の右辺）をあらかじめ設定すると，

$$\frac{Z^U - h_m}{Z^U - Z^L} \le r_{RP} \quad , \quad \frac{h_m - Z^L}{Z^U - Z^L} \le r_{RP} \qquad (4.11)$$

となる h_m をあらわす切断面(図－4.2 のａあるいはａ')が削除される．

代理乗数の決定に関与しない切断面を削除する操作は，代理乗数を計算するたびに行うのではなく，Dyer アルゴリズムを何回か実行した後，周期的に切断面を削除する操作を行う．この削除の操作と次の削除の操作との間に実行する Dyer アルゴリズムの回数を RP 周期，PRP（Period of Reducing Plane）と呼び，その設定する値を p_{RP} と表す．

4．5　計算機実験と考察

4．5．1　テスト問題

改良 Dyer アルゴリズムの特性を評価するために，テスト問題を Dyer アルゴリズムと改良 Dyer アルゴリズムを用いて解く．擬似乱数を用いたテスト問題の係数は，Sinha and Zoltners[11]に基づいた次式のように生成した．

$$0 \le f_i(k) < f_i(k+1) \le 256k_i \quad (k=1,2,\ldots\ldots,k_{i-1},\ i=1,2,\ldots\ldots,n), \qquad (4.12)$$

$$0 \le g_{ji}(k) < g_{ji}(k+1) \le 256k_i \quad (k=1,2,\ldots\ldots,k_{i-1},\ i=1,2,\ldots\ldots,n,\ j=1,2,\ldots\ldots,m)$$

$$b_j = \left\lfloor \frac{1}{2}\sum_{n=1}^{n}\{g_{ji}(1) + g_{ji}(k_i)\} \right\rfloor \quad (i=1,2,\ldots\ldots,n,\ j=1,2,\ldots\ldots,m)$$

ただし，f_i, g_{ji}はそれぞれ目的関数，制約関数式の係数であり，すべて正整数である．k_iは変数 x_i に対して選択することができる項目数である．変数の数 n を 50, 100, 200, 300, 400, 500，制約条件式の数 m は 5, 10, 15, 20，項目の数を 10 とした．改良 Dyer アルゴリズムを用いて解くためのパラメータとして，ＲＰ率（r_{RP}）は 0.1, 0.2, 0.3, 0.4，ＲＰ周期（p_{RP}）は 10, 30, 50, 80, 100 とした．本章での計算機実験において，Dyer アルゴリズムと改良 Dyer アルゴリズムは同一のプログラムを用いた．Dyer アルゴリズムについては，切断面の削除操作が実施されないように，ＲＰ周期を p_{RP} = 5000 と十分大きくした．

擬似乱数の異なる 10 個の種を用いて作成された代理問題に，Dyer アルゴリズム及び改良 Dyer アルゴリズムを用いて最適解を求めた．その最適解が得られるまでの実行時間と実行回数との平均値を表－4.1 と表－4.2 に示す．

なお，変数の数と制約条件式の数及び擬似乱数の種の値が等しいテスト問題において，Dyer アルゴリズム，並びに改良 Dyer アルゴリズムにおけるＲＰ率及びＲＰ周期のすべての場合において，得られた代理双対問題の最適解はすべて等しいことが確認されている．擬似乱数の種を変化させて作成された 10 問のうち，実行回数が 10,000 回を越えても最適解が得られなかったテスト問題があった場合，最適代理乗数が存在しないとして，表中に*印で表した．また，表中†印は，10 問のうちいくつかの問題において，Dyer アルゴリズムの実行途中にプログラムの配列オーバーエラーが生じたことを示す．これは，Dyer アルゴリズムにおいて制約条件式の数が 15, 20 と多くなり，変数の数が増えるにしたがって式(4.10)あるいは式(4.10′)で示される線形計画問題のサイズが大きくなるためだと考えられる．

なお，本章のすべての計算実験は，Windows XP 上で DOS/V コンピュータ (Pentium4 2.2GHz, メモリ 512MB) を用いて行った．すべてのアルゴリズムは，仲川 [14] によって開発された Modula2 風の C 言語により記述されている．

表－4.1　Dyer アルゴリズム及び改良 Dyer アルゴリズムによる平均実行時間（単位：秒）（制約条件数が 5 の場合）（つづく）

制約条件	変数	Dyer法	10				30			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
5	50	0.2	0.3	0.3	0.7	*	0.4	0.3	0.3	0.4
	100	1.0	1.0	1.3	3.1	*	0.9	0.8	0.9	1.2
	200	3.2	4.2	7.8	25.3	*	3.2	3.2	3.4	4.2
	300	7.0	11.2	18.5	107.6	*	6.9	7.0	8.3	13.5
	400	13.0	17.4	44.8	180.6	*	13.3	13.5	14.5	18.5
	500	22.0	44.6	69.3	*	*	22.1	22.1	24.8	38.7
制約条件	変数	Dyer法	50				80			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
5	50	0.2	0.3	0.3	0.2	0.2	0.4	0.4	0.3	0.4
	100	1.0	1.0	0.8	0.9	0.7	0.8	0.7	0.9	0.8
	200	3.2	3.2	3.2	3.3	3.4	3.2	3.1	3.2	3.2
	300	7.0	7.0	7.1	7.1	7.1	7.1	7.0	7.0	6.8
	400	13.0	13.5	13.3	13.7	15.0	13.1	12.9	12.9	13.1
	500	22.0	22.0	22.1	23.5	24.9	21.8	21.7	21.7	21.8
制約条件	変数	Dyer法	100							
			0.1	0.2	0.3	0.4				
5	50	0.2	0.3	0.3	0.2	0.3				
	100	1.0	0.7	0.9	0.9	0.8				
	200	3.2	3.1	3.0	3.2	3.1				
	300	7.0	7.1	6.8	7.0	6.9				
	400	13.0	13.1	13.0	13.1	13.1				
	500	22.0	21.7	21.7	21.7	21.8				

備考：＊　10,000 回以内で収束しないテスト問題がある

†　メモリ不足で代理乗数を求められないテスト問題がある

表－4.1　Dyer アルゴリズム及び改良 Dyer アルゴリズムによる平均実行時間（単位：秒）
（制約条件数が 10 の場合）（つづき）

制約条件	変数	Dyer法	10				30			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
10	50	3.2	2.1	1.9	*	*	2.5	1.4	1.7	*
	100	16.0	6.5	14.5	*	*	7.5	5.2	6.4	*
	200	25.4	15.9	109.4	*	*	15.9	14.6	85.3	*
	300	49.6	56.7	273.8	*	*	33.2	34.0	41.6	*
	400	48.0	50.8	129.4	*	*	44.0	54.6	64.6	*
	500	74.5	97.6	690.7	*	*	69.7	69.8	224.2	*

制約条件	変数	Dyer法	50				80			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
10	50	3.2	2.7	1.9	1.8	6.2	2.7	2.3	2.5	3.7
	100	16.0	8.3	6.0	5.9	111.2	11.4	8.0	7.9	13.0
	200	25.4	17.3	14.2	14.5	87.2	20.5	16.8	16.3	64.5
	300	49.6	34.5	32.0	33.2	*	40.8	34.4	34.0	*
	400	48.0	45.5	44.7	45.7	309.7	46.3	45.3	44.5	53.5
	500	74.5	72.1	68.6	93.8	282.9	74.6	71.7	73.3	85.9

制約条件	変数	Dyer法	100			
			0.1	0.2	0.3	0.4
10	50	3.2	3.8	3.2	3.2	3.4
	100	16.0	11.0	9.0	9.4	12.3
	200	25.4	23.8	19.4	17.9	23.7
	300	49.6	38.0	34.5	33.2	122.8
	400	48.0	46.1	45.2	44.5	47.9
	500	74.5	71.9	70.5	70.7	83.0

備考：＊　10,000 回以内で収束しないテスト問題がある
　　†　メモリ不足で代理乗数を求められないテスト問題がある

表－4.1　Dyer アルゴリズム及び改良 Dyer アルゴリズムによる平均実行時間（単位：秒）
（制約条件数が 15 の場合）（つづき）

制約条件	変数	Dyer法	10				30			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
15	50	54.2	16.0	7.3	*	*	15.5	6.9	6.6	*
	100	366.6	*	*	*	*	146.9	33.6	27.1	*
	200	†	596.6	*	*	*	397.4	84.9	*	*
	300	†	740.7	*	*	*	445.1	141.9	*	*
	400	†	942.4	*	*	*	602.6	224.6	392.5	*
	500	†	*	*	*	*	773.3	330.8	*	*

制約条件	変数	Dyer法	50				80			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
15	50	54.2	20.1	9.8	7.5	*	31.9	17.0	12.4	51.1
	100	366.6	170.9	41.7	27.4	*	216.0	68.4	35.9	*
	200	†	407.4	106.5	66.7	*	513.0	160.2	79.2	*
	300	†	447.7	171.6	110.9	*	571.5	225.8	131.7	*
	400	†	588.4	239.9	228.3	*	673.4	312.1	197.1	*
	500	†	778.9	343.1	266.4	*	891.5	412.9	294.3	*

制約条件	変数	Dyer法	100			
			0.1	0.2	0.3	0.4
15	50	54.2	34.7	20.2	17.7	72.8
	100	366.6	266.2	91.1	51.9	*
	200	†	565.8	203.4	99.1	*
	300	†	612.1	282.4	164.1	*
	400	†	729.4	352.0	228.0	*
	500	†	1014.7	481.6	319.9	*

備考：＊　10,000 回以内で収束しないテスト問題がある
　　†　メモリ不足で代理乗数を求められないテスト問題がある

表−4.1　Dyer アルゴリズム及び改良 Dyer アルゴリズムによる平均実行時間（単位：秒）
（制約条件数が 20 の場合）（つづき）

制約条件	変数	Dyer法	10				30			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
20	50	292.4	137.9	35.0	*	*	105.3	36.3	20.4	*
	100	†	1880.4	*	*	*	1037.8	178.0	*	*
	200	†	7814.6	*	*	*	4013.9	607.5	*	*
	300	†	22112.1	*	*	*	10236.3	1623.8	*	*
	400	†	48990.5	*	*	*	18902.5	3386.9	*	*
	500	†	*	*	*	*	28277.2	4067.3	*	*

制約条件	変数	Dyer法	50				80			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
20	50	292.4	128.0	41.4	20.7	*	178.7	72.3	35.6	*
	100	†	946.5	212.9	135.0	*	1185.8	350.8	123.0	*
	200	†	3666.7	664.9	298.2	*	3825.3	908.4	310.7	*
	300	†	8557.8	1509.7	*	*	7929.6	1692.4	544.6	*
	400	†	17143.4	3080.5	811.7	*	14969.8	3721.9	905.1	*
	500	†	21799.7	3734.2	*	*	20008.5	3956.7	1265.6	*

制約条件	変数	Dyer法	100			
			0.1	0.2	0.3	0.4
20	50	292.4	182.0	86.4	52.4	101.5
	100	†	1298.1	432.1	166.0	*
	200	†	4131.3	1039.7	489.6	*
	300	†	8530.3	2069.8	630.6	*
	400	†	15456.6	3540.1	1096.7	*
	500	†	20307.4	4408.3	1298.9	*

備考：＊　10,000 回以内で収束しないテスト問題がある
　　†　メモリ不足で代理乗数を求められないテスト問題がある

表−4.2　Dyer アルゴリズム及び改良 Dyer アルゴリズムによる平均実行回数
（制約条件数が 5 の場合）（つづく）

制約条件	変数	Dyer 法	10				30			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
5	50	29.8	30.7	36.4	71.3	*	29.9	30.3	30.7	34.2
	100	34.7	35.2	46.6	134.8	*	34.7	35.2	39.0	44.2
	200	41.0	51.5	107.7	366.2	*	41.3	42.4	45.3	56.6
	300	43.5	73.0	123.4	721.7	*	43.5	44.2	51.6	86.0
	400	48.5	66.1	174.6	713.3	*	48.5	50.0	53.2	69.3
	500	51.8	108.6	168.7	*	*	52.0	52.5	59.2	93.4

制約条件	変数	Dyer 法	50				80			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
5	50	29.8	29.8	29.8	29.8	29.8	29.8	29.8	29.8	29.8
	100	34.7	34.7	34.7	34.7	34.7	34.7	34.7	34.7	34.7
	200	41.0	41.0	41.5	42.6	43.0	41.0	41.0	41.0	41.0
	300	43.5	43.7	43.7	44.5	45.3	43.5	43.5	43.5	43.5
	400	48.5	49.3	49.4	50.9	55.8	48.5	48.5	48.5	48.5
	500	51.8	52.1	52.3	55.6	58.9	51.8	51.8	51.8	51.8

制約条件	変数	Dyer 法	100			
			0.1	0.2	0.3	0.4
5	50	29.8	29.8	29.8	29.8	29.8
	100	34.7	34.7	34.7	34.7	34.7
	200	41.0	41.0	41.0	41.0	41.0
	300	43.5	43.5	43.5	43.5	43.5
	400	48.5	48.5	48.5	48.5	48.5
	500	51.8	51.8	51.8	51.8	51.8

備考：＊　10,000 回以内で収束しないテスト問題がある
　　†　メモリ不足で代理乗数を求められないテスト問題がある

表－4.2　Dyer アルゴリズム及び改良 Dyer アルゴリズムによる平均実行回数
（制約条件数が 10 の場合）（つづき）

制約条件	変数	Dyer 法	10				30			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
10	50	101.4	107.6	148.0	*	*	102.2	106.8	134.8	*
	100	153.2	159.8	527.3	*	*	153.2	155.5	217.8	*
	200	160.7	170.6	1418.9	*	*	141.3	170.4	1123.6	*
	300	176.6	340.4	1783.7	*	*	179.5	206.1	259.3	*
	400	152.0	181.0	487.9	*	*	152.2	198.9	237.6	*
	500	160.1	231.3	1709.0	*	*	160.4	163.7	550.4	*

制約条件	変数	Dyer 法	50				80			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
10	50	101.4	102.2	103.8	116.0	430.9	101.4	101.9	108.5	160.9
	100	153.2	153.2	157.9	170.9	3891.0	153.2	154.1	171.2	312.9
	200	160.7	160.8	156.7	173.1	1125.5	160.9	162.5	172.4	712.8
	300	176.6	176.9	179.5	199.7	*	178.3	179.5	187.9	*
	400	152.0	152.2	155.9	163.6	1145.9	152.2	153.1	153.5	187.8
	500	160.1	160.4	160.3	223.6	689.1	161.9	161.4	168.5	201.2

制約条件	変数	Dyer 法	100			
			0.1	0.2	0.3	0.4
10	50	101.4	101.9	102.1	104.6	118.5
	100	153.2	153.2	153.6	162.4	236.3
	200	160.7	160.8	161.3	170.5	235.5
	300	176.6	176.6	177.1	177.4	665.2
	400	152.0	152.0	151.8	151.8	165.4
	500	160.1	160.1	160.4	162.6	191.7

備考：＊　10,000 回以内で収束しないテスト問題がある
　　†　メモリ不足で代理乗数を求められないテスト問題がある

表－4.2　Dyer アルゴリズム及び改良 Dyer アルゴリズムによる平均実行回数
（制約条件数が 15 の場合）（つづき）

制約条件	変数	Dyer 法	10				30			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
15	50	209.5	250.3	322.6	*	*	209.9	216.9	368.2	*
	100	332.2	*	*	*	*	333.2	346.7	654.4	*
	200	†	583.4	*	*	*	428.6	428.2	*	*
	300	†	865.3	*	*	*	443.4	438.0	*	*
	400	†	557.3	*	*	*	455.7	489.5	1319.0	*
	500	†	*	*	*	*	468.1	514.0	*	*

制約条件	変数	Dyer 法	50				80			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
15	50	209.5	213.1	219.8	289.3	*	209.5	209.4	268.1	1462.3
	100	332.2	332.3	327.6	468.8	*	332.3	324.6	414.4	*
	200	†	423.5	423.2	561.0	*	420.9	425.6	482.4	*
	300	†	425.9	443.9	538.0	*	424.4	438.9	516.5	*
	400	†	439.7	444.1	712.1	*	439.9	445.2	512.7	*
	500	†	467.3	472.0	554.5	*	467.2	467.9	542.0	*

制約条件	変数	Dyer 法	100			
			0.1	0.2	0.3	0.4
15	50	209.5	209.5	215.4	258.8	1332.0
	100	332.2	332.2	326.0	406.5	*
	200	†	422.3	416.7	466.2	*
	300	†	426.1	438.4	511.0	*
	400	†	439.4	435.4	515.6	*
	500	†	467.3	469.7	523.0	*

備考：＊　10,000 回以内で収束しないテスト問題がある
　　†　メモリ不足で代理乗数を求められないテスト問題がある

表－4.2　Dyer アルゴリズム及び改良 Dyer アルゴリズムによる平均実行回数
（制約条件数が 20 の場合）（つづき）

制約条件	変数	Dyer 法	10				30			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
20	50	304.9	308.7	363.3	*	*	305.1	356.0	630.8	*
	100	†	634.4	*	*	*	506.5	543.5	*	*
	200	†	772.5	*	*	*	748.5	779.1	*	*
	300	†	937.4	*	*	*	837.4	863.7	*	*
	400	†	1352.3	*	*	*	911.6	939.8	*	*
	500	†	*	*	*	*	1075.1	1159.6	*	*

制約条件	変数	Dyer 法	50				80			
			0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4
20	50	304.9	304.9	319.2	433.4	*	305.0	312.7	394.2	*
	100	†	503.9	516.5	1632.8	*	503.9	514.1	693.4	*
	200	†	749.4	774.2	1914.7	*	748.3	772.9	1138.2	*
	300	†	838.4	857.3	*	*	837.0	824.4	1129.1	*
	400	†	911.1	942.0	1498.0	*	911.1	930.1	1232.8	*
	500	†	1022.4	1510.6	*	*	1022.0	1006.2	1473.9	*

制約条件	変数	Dyer 法	100			
			0.1	0.2	0.3	0.4
20	50	304.9	304.9	308.1	402.8	1166.2
	100	†	503.9	518.2	648.3	*
	200	†	750.2	747.5	1442.3	*
	300	†	837.1	836.2	1066.3	*
	400	†	911.1	901.6	1218.3	*
	500	†	1021.7	1010.5	1242.0	*

備考：＊　10,000 回以内で収束しないテスト問題がある
　　†　メモリ不足で代理乗数を求められないテスト問題がある

4．5．2　計算結果に対する考察

（1）Dyer アルゴリズムと改良 Dyer アルゴリズムとの比較

表－4.1 において，Dyer アルゴリズムと改良 Dyer アルゴリズムとの平均実行時間を比較すると，制約条件の数が 5 のとき，実行時間にほとんど差がなかった．制約条件の数が 10 より多くなると，平均実行時間はほとんどの場合，改良 Dyer アルゴリズムの方が短かった．制約条件の数が 15，20 と多くなると，Dyer アルゴリズムでは配列オーバーのエラーが生じて代理双対問題の最適解は求まらなかった．これは，式(4.10)あるいは式(4.10′)で表される線形計画問題を解くときにエラーが生じるためだと考えられる．これらのテスト問題においても，改良 Dyer アルゴリズムでは最適解が求められた．したがって，改良 Dyer アルゴリズムは，計算速度及び使用する作業領域量に関して Dyer アルゴリズムよりも優れていることがわかる．

Dyer アルゴリズムにおいて，配列オーバーエラーとなった制約条件の数が 15 で，変数の数が 300，400，500 のいくつかのテスト問題において，Dyer アルゴリズムで解けたテスト問題について改良 Dyer アルゴリズムとの比較を表－4.3 に示す．この場合においても，

表－4.3　擬似乱数のそれぞれの種における Dyer アルゴリズムと
改良 Dyer アルゴリズムの比較

15 制約，300 変数　　（p_{RP} = 50，r_{RP} = 0.3 ）

種	Dyer アルゴリズム		改良 Dyer アルゴリズム	
	実行回数	実行時間(秒)	実行回数	実行時間(秒)
2	384	525	441	86
3	456	1098	685	130
5	266	133	272	58
7	368	458	374	79
8	340	319	330	64
9	481	1747	680	159
10	302	201	385	70

15 制約，400 変数　　（p_{RP} = 50，r_{RP} = 0.3 ）

種	Dyer アルゴリズム		改良 Dyer アルゴリズム	
	実行	実行	実行	実行

	回数	時間(秒)	回数	時間(秒)
1	361	458	399	121
3	353	431	410	133
5	381	543	433	144
6	370	460	477	140
7	363	444	365	119
10	278	201	290	95

15 制約，500 変数　　(p_{RP} = 50,　r_{RP} = 0.3)

	Dyer アルゴリズム		改良 Dyer アルゴリズム	
種	実行回数	実行時間(秒)	実行回数	実行時間(秒)
3	412	772	471	217
4	337	423	394	188
6	443	1001	490	223
7	361	490	395	189
9	340	427	335	153
10	380	622	360	170

実行回数は増えているが，実行時間はすべての場合で改良 Dyer アルゴリズムのほうが Dyer アルゴリズムよりもかなり短いことがわかった．

以上の結果から，制約条件式が少ないときには Dyer アルゴリズムは有効であるが，制約条件の数が 10, 15, 20 と多い場合に Dyer アルゴリズムを用いると実行時間が非常に長くなるか，あるいは最適解が得られないことがある．その場合，適当なＲＰ周期とＲＰ率とが設定された改良 Dyer アルゴリズムを用いれば，Dyer アルゴリズムよりも短時間で，しかもより少ない作業領域量で代理双対問題の最適解が得られることがわかった．

（２）ＲＰ周期及びＲＰ率による実行時間及び実行回数への影響

制約条件式の数が多いとき，各ＲＰ周期においてＲＰ率が増加すると実行時間は短くなる傾向がある．ＲＰ周期が 50, 80, 100 のとき，ＲＰ率が 0.1 から 0.3 に増加するにしたがって，実行時間は単調減少する．

制約条件の数が 5 のときは，変数の数がいずれの場合も，ＲＰ周期，ＲＰ率の違いによる実行時間の差違はほとんどなかった．

ＲＰ周期が 30,50,80,100 のいずれの場合においても，ＲＰ率が 0.1 及び 0.2 の場合はいずれも 10,000 回以内で代理乗数が収束して最適解が得られた．ＲＰ率を 0.3 とした場合，ＲＰ周期が 80,100 のとき最適解は得られたが，ＲＰ周期が 50,30,10 と小さくなるにつれて，10,000 回以内で代理最適解が得られないテスト問題があった．特に，ＲＰ周期が 10 のときには，制約条件数が 10 のときからすべての場合で，代理乗数が収束しないテスト問題があった．

ＲＰ率を 0.4 に設定すると，制約条件及び変数の数が増えると，10,000 回以内で代理乗数が収束しなかった．特に，制約条件数が 15 を超えると 100 変数以上のすべての場合で代理乗数が収束しないテスト問題があった．これはＲＰ率が大きくなると，最適解を求める途中において代理乗数の多面体のほとんどが削除され，実行回数が 10,000 回以内で最適解が得られなかったと考えられる．

制約条件の数が 5 のとき，いずれの変数の数においても，ＲＰ周期，ＲＰ率の違いによる実行回数の差違はほとんどなかった．

ＲＰ周期が 10,...,100 のいずれの場合も，ＲＰ率が 0.1 から 0.4 に増加するにしたがって，実行回数は増加する．これは縮小する切断面を選択するための目的関数値の割合を大きくすることにより，選択される候補の数が減るため最適解を得られるまでの実行回数が増えるためである．

ＲＰ率を 0.4 のとき，実行回数が多くなって代理乗数が得られるか，あるいは 10,000 回以内でも得られない場合が多くなった．ＲＰ周期が大きい（80 及び 100）ときは，ＲＰ率が 0.4 で，しかも制約条件式の数が多く，変数の数が 100 より大きいとき 10,000 回以内で収束しなかった．また，ＲＰ周期が小さくなるにしたがって制約条件式がより少ない問題でも代理乗数が得られなくなり，さらにＲＰ周期が小さくなるとＲＰ率が 0.3 の場合にも，収束しなかったテスト問題があった．さらに，ＲＰ周期 30 ではＲＰ率 0.4 で，ＲＰ周期 10 ではＲＰ率 0.2 や 0.1 においても代理乗数が得られない場合があった．

（３）有効なＲＰ率とＲＰ周期の組合せ

改良 Dyer アルゴリズムを用いるとき，ＲＰ周期とＲＰ率の有効な組合せについて考察する．制約条件式及び変数の数のそれぞれの問題規模において，擬似乱数を用いて作成された 10 個のテスト問題を解いたとき，最少の実行時間で代理双対問題の最適解が得られたＲＰ周期とＲＰ率の組合せの問題数を表－4.4 に示す．

表－4.4　各問題規模において RP 周期と RP 率の組み合わせに対して最短時間で代理最適解が求まった問題の個数（制約条件数が 5 の場合）

	5制約									
RP	50 変数					100 変数				
周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10	0.61	0.48	0.51	0.28	0.08	0.13	0.13	0.20	0.09	
30		0.41	0.51	0.48	0.40		0.22	0.34	0.34	0.27
50		0.48	0.48	0.59	0.63		0.13	0.34	0.25	0.72
80		0.47	0.45	0.53	0.49		0.51	2.52	0.38	0.33
100		0.49	0.54	0.59	0.53		1.34	0.22	0.22	1.34
RP	200 変数					300 変数				
周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10	0.42		0.42			0.58	0.35	0.22		
30		0.67	0.88	0.34	0.20		0.80	0.71	0.16	
50		0.40	0.79	0.46	0.54		0.51	0.46	0.58	0.81
80		0.40	0.74	0.78	0.65		0.46	0.75	0.74	0.94
100		0.57	0.71	0.46	0.57		0.43	0.43	0.49	0.58
RP	400 変数					500 変数				
周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10	0.65	1.23	0.06			0.34	1.13			
30		0.26	0.52	0.29			0.19	0.34	2.63	0.13
50		0.32	0.19	0.69	0.23		0.27	0.14	0.14	0.27
80		0.49	0.61	1.45	0.52		0.47	0.47	0.97	0.34
100		0.52	0.95	0.52	0.52		0.47	0.52	0.59	0.59

（制約条件数が 10 の場合）

	10制約									
RP	50 変数					100 変数				
周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10		1.36	1.00					2.00		
30		0.11	2.44	1.50				3.83	0.33	
50		0.11	0.11	1.11				2.50	1.33	
80		0.33	0.33	0.25	0.25					
100		0.25	0.11	0.11	0.61					
RP	200 変数					300 変数				
周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10		0.33								
30		0.50	3.17				0.25	4.92	1.00	
50			3.00	2.67				0.58	1.33	
80		0.17	0.17					0.33	0.33	
100									1.25	
RP	400 変数					500 変数				
周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10		0.33					0.33			
30		2.33	0.33	1.33			1.50	2.83		
50			1.00	2.50				2.33	1.00	
80				1.67				0.50	0.50	1.00
100				0.50						

（制約条件数が 15 の場合）

	15制約									
RP	50 変数					100 変数				
周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10			2.00							
30			4.50	3.50				1.00	6.00	
50									3.00	
80										
100										
RP	200 変数					300 変数				
周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10										
30			1.00	2.00				0.50		
50				7.00					7.50	
80									2.00	
100										
RP	400 変数					500 変数				

周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10										
30			2.00					1.00	1.00	
50				7.00					7.00	
80				1.00					1.00	
100										

（制約条件数が 20 の場合）

RP	20制約									
	50 変数					100 変数				
周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10			2.33							
30			1.00	4.83				1.00	4.00	
50				1.83					5.00	
80										
100										
RP	200 変数					300 変数				
周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10										
30				1.00					2.50	
50				8.00					4.00	
80				1.00					3.50	
100										
RP	400 変数					500 変数				
周期	Dyer	0.1	0.2	0.3	0.4	Dyer	0.1	0.2	0.3	0.4
10										
30										
50				8.00					3.00	
80				2.00					5.00	
100									2.00	

最少の実行時間が同一であるＲＰ周期とＲＰ率の組合せが t 個あった場合，それぞれの問題数は 1／t として計数する．表−4.4 から制約条件式の数が少ないときには，ＲＰ周期とＲＰ率の組合せに顕著な差はみられず，ほとんどすべての組合せにおいて，最少の実行時間で最適解が得られていることがわかる．制約条件式の数が多くなるにしたがって，最少の実行時間で最適解が得られるＲＰ周期とＲＰ率との組合せが限られてくる．すなわち，ＲＰ周期は 30，50 あるいは 80 で，ＲＰ率は 0.3 に設定したときに最少の実行時間で最適解が得られることがわかる．表−4.4 において最短時間で最適解が得られる問題が最多の組合せはＲＰ周期 50，ＲＰ率 0.3 である．しかし，表−4.1 及び表−4.2 からこの組合せにおいて代理乗数が収束しなかったテスト問題があるので，実行時間が多く要するが，最適な代理乗数がより確実に得られるためにはＲＰ周期 80，ＲＰ率 0.3 がより適当であろう．

4．6　結語

多次元非線形整数計画問題を解くために代理制約法及びISC法を用いるとき，最適な代理乗数が必要になる．しかし，制約条件式の個数が多いとき，代理乗数を決定するためのアルゴリズムにおいて，必要な作業領域（メモリ）の量が多大になり，実行時間も実用的でなくなり代理乗数を決定することができなくなることが多い．

第3章において，代理乗数を決定するアルゴリズムの一つであるDyerアルゴリズムを改良した改良Dyerアルゴリズムを提案した．これは，代理乗数を決定する過程において，内接する球の半径の大きさの決定に全く関係していない切断面，及び関係しているがその割合（ＲＰ率，RRP）が小さい切断面を削除するものである．しかも，この操作はDyerアルゴリズムを一定回数（ＲＰ周期，PRP）行った後，周期的に実行する．

本章において，改良Dyerアルゴリズムの特性を評価するために，擬似乱数を用いた大規模な多次元非線形整数計画問題の計算機実験を行った．テスト問題の係数は，Sinha and Zoltnersによる式を用いた．テスト問題の変数の数は50, 100, 200, 300, 400, 500とし，制約条件式の数は5, 10, 15, 20とし，項目の数は10とした．擬似乱数の異なる10個の種を用いて作成されたテスト問題に，Dyerアルゴリズム及び改良Dyerアルゴリズムを用いて最適解を求めた．その最適解が得られるまでの実行時間と実行回数との平均値を比較した．なお，変数の数と制約条件式の数及び擬似乱数の種の値が等しい問題において，Dyer アルゴリズム，並びに改良Dyerアルゴリズムにおけるｒｐ率及びＲＰ周期のすべての値の場合において，得られた代理双対問題の最適解はすべて等しいことが確認されている．

その結果，制約条件式の数及び変数の数が少ないときは両アルゴリズムにおける平均実行時間にほとんど差はなかった．しかし，それらの数が多くなる大規模な多次元非線形整数計画問題においては，改良Dyer アルゴリズムの方がDyer アルゴリズムよりも少ない実行時間及び作業領域量で最適解が求められることがわかった．

また，ＲＰ率とＲＰ周期の値による実行時間，実行回数の影響を示した．制約条件式の数が多いとき，各ＲＰ周期においてＲＰ率が増加すると実行時間は短くなる傾向がある．ＲＰ周期が50, 80, 100のとき，ＲＰ率が0.1から0.3に増加するにしたがって，実行時間は単調減少する．ＲＰ周期がいずれの値の場合も，ＲＰ率が0.1から0.4に増加するにしたがって，実行回数は増加する．これは縮小する切断面を選択するための目的関数値の割合を大きくすることにより，選択される候補の数が減るため最適解を得られるまでの実行回数が増えるためである．

さらに，大規模な変数分離型非線形整数計画問題を，改良Dyerアルゴリズムを用いて解くために有効となるＲＰ率とＲＰ周期の値の組合せとしては，実行時間が多少多く要するが，最適な代理乗数がより確実に得られるＲＰ周期80，ＲＰ率0.3がより適当であることがわかった．

参考文献

[1] F. Glover, "A multiphase-dual algorithm for the zero-one integer programming problem", Oper. Res., Vol.13, pp.879−919, 1965.

[2] F. Glover, "Surrogate constraint Duality in mathematical programming", Oper. Res., Vol.23, pp.433−451, 1975.

[3] D. G. Luenberger,"Quasi-convex programming", SIAM J. of Appl. Math., Vol. 16, pp.1090−1095, 1968.

[4] M.E. Dyer, "Calculating surrogate constraints", Math. Program., Vol.19, pp.255−278, 1980.

[5] Y. Nakagawa, "A reinforced surrogate constraints method for separable nonlinear integer programming", RIMS, Kokyuroku 1068 Kyoto Univ., pp.194−202, 1998.

[6] Y. Nakagawa, "An improved surrogate constraints method for separable nonlinear integer programming", Journal of the Operations Research Society of Japan, Vol.46, No.2, pp.145−163, 2003.

[7] 仲川勇二，並川哲郎，木村作郎，太田垣博一，"代理制約法における最適代理乗数の決定法".電子情報通信学会論文誌，Vol. J87−A. No.3. pp.364−374. 2004.

[8] 仲川 勇二. 疋田 光伯. 鎌田 弘，"代理双対問題を解くためのアルゴリズム". 電子情報通信学会論文誌 (A) Vol. J67−A No. 1. pp.53−59. 1984.

[9] 仲川 勇二，"離散最適化問題のための新解法". 電子情報通信学会論文誌(A). Vol. J73A. No. 3. pp.550−556. 1990.

[10] Y. Nakagawa and A. Iwasaki, "Modular Approach for Solving Nonlinear Knapsack Problems, "IEICE Trans.Fundamentals, Vol.E82−A, No.9, pp.1860−1864, 1999.

[11] P. Sinha and A. A. Zoltoner, "The Multiple Choice Knapsack Problem", Oper. Res., Vol. 27, No.3, pp.503−515, 1979.

[12] T. L. Magnanti, J. F. Shapiro and M. H. Wagner, "Generalized Linear Programming solves the Dual", Management Science, Vol. 22, No. 11, pp.1195−1202, 1976.

[13] T. H. Mattheiss and W. B. Widhelm, "The Generalized Slack Variable Linear Program", Management Science, Vol. 23, No. 8, pp.859−871, 1977.

[14] 仲川 勇二，「やさしいＣプログラミング」，朝倉書店，東京. 1996.

[15] 木村作郎，太田垣博一，仲川勇二，"代理制約法における代理乗数決定のための改良 Dyer アルゴリズムの特性評価，" 日本経営工学会論文誌，Vol.55，No.5，pp.252−261，Dec.2004.

第５章　改良代理制約法を用いた信頼性最適化問題[12, 13]

５．１　概説

複数の制約条件をもつ非線形整数計画問題に対して品質の良い解を，実用的な時間内で得ることは，近年ますます重要になってきている．複数の制約条件式を単一の制約条件式で代理させる代理制約法は，大変計算効率の良い解法である．しかし，代理双対ギャップが存在することが多く，その場合代理双対問題の最適解は原問題の実行可能解とはならず，厳密な最適解をもたらさない．仲川は，この代理ギャップを閉じ，原問題の厳密解を求めることができる改良代理制約法（ISC 法，Improved Surrogate Constraint Method）を最近提案した[10]．

本章において，代理双対ギャップがあるために，既存の解法で厳密解が求められなかったシステムの信頼性設計問題に，ISC 法を適用して，その厳密解を求める．その際，第２章で提案した新上界値，及び第３章で提案した改良 Dyer 法による代理乗数を用いることにより，本論文で提案された新上界値の決定法及び改良 Dyer 法が有用であることが明らかにする．

５．２　信頼性設計の問題への適用

仲川が提案した改良代理制約法であるISC法の有効性を確かめるために，代理双対ギャップがあるため，従来厳密解を求めることができなかった２種類の信頼性最適化問題を解く．２制約式を持つ直列システムの信頼性の最適化問題，及び４制約式を持つ直列システムの信頼性の最適化問題である．

システムの信頼性を改良するには，より信頼できる構成部品を使用すること，または並列冗長な構成部品を提供することによって実現することができる． システムの各ステージにおいて，複数の異なる信頼度と費用の構成部品が利用可能であり，または並列冗長性が利用可能な場合，システムの最適化問題は非線形整数計画法問題として定式化される[5.11]．

システムの信頼性は，決定変数$x_1, x_2, \ldots, x_n$（非負整数）からなる関数$f(x_1, x_2, \ldots, x_n)$と書ける．構成部品を選択し，並列冗長の選択によりシステムの信頼性を最大化する問題は，次のように定式化される．

$$[P^0] \quad \text{Maximize} \quad R = f(x_1, x_2, \ldots, x_n) \tag{5.1}$$

subject to

$$\sum_{i=1}^{n} g_{ji}(x_i) \le b_j \quad \text{for} \quad i = 1, 2, \ldots, n$$

$$l_i \le x_i \le u_i \quad \text{for} \quad j = 1, 2, \ldots, m$$

ここで，$f(x_1, x_2, \ldots, x_n)$はx_i個の構成部品がステージiで直列に配列され，各ステージで並列冗長の選択可能なときの信頼性である．nはシステムのリソース数であり，x_iはステージiにおける並列冗長性の数（非負整数）である．l_iとu_iはそれぞれx_iの下界値と上界値である．制約式は変数分離可能である．直列システムの並列冗長配分問題において，対数を用いると目的関数は変数分離型に書き換えられる．

本論文では，原問題$[P^0]$の困難度を測定するために PGC(Percent Gap Closure)値[4]を用いる．

$$PGC = \frac{f^{\mathrm{Real}} - \mathrm{opt}[\mathrm{P}^{\mathrm{SD}}]}{f^{\mathrm{Real}} - \mathrm{opt}[\mathrm{P}]} \times 100 \tag{5.2}$$

ここで，f^{Real}は原問題$[P^0]$に対応する線形整数計画問題の線形計画緩和値であり，opt[•]は問題[•]の最適な目的関数の値を意味する．PGC 値が小さいほど代理双対ギャップが深いことを意味し，PGC 値 100%は代理双対問題$[\mathrm{P}^{\mathrm{SD}}]$の最適関数値は$[P^0]$の最適値と等しい．

5．2．1　2制約式をもつシステムの信頼性最適化問題

総システム費用及び総重量の許容範囲内で最大の信頼性を達成するために，各ステージにおける並列冗長のユニット数を決定する問題を考える [3 ,8].

$$[\mathrm{P}^1]\ \text{Maximize}\ R=\prod_{i=1}^{14}[1-\{(1-r_i(k_i)\}^{d_i}] \qquad (5.3)$$

$$\text{subject to}\ \sum_{i=1}^{14} d_i\, c_i(k_i)\le C,$$

$$\sum_{i=1}^{14} d_i\, w_i(k_i)\le W$$

$$1\le k_i\le 4, 1\le d_i\ ;\ k_i, d_i\ \text{integers.}$$

ここで，変数k_iはステージiにおいて採択される設計案の番号であり，変数d_iはステージiにおけるk_iのユニット数である．$r_i(k_i), c_i(k_i), w_i(k_i)$は，設計案$k_i$のそれぞれ信頼度，費用，重量である．$C$と$W$はシステムの許容費用と許容重量であり，ここでは$C=130$とする．$r_i(k_i), c_i(k_i), w_i(k_i)$の値は表－5．1に示されている[3]．表中＊印は，設計代替案がないことを意味する．$1\le k_i\le 4$であり，更に$1\le d_i\le 5$と仮定すると，28 個の変数k_iと$d_i(i=1,\ldots,14)$の組合せの組は，14 個の変数x_i $(1\le x_i\le 20)$に置き換えられる[3]．目的関

表－5．1　$r_i(k_i), c_i(k_i), w_i(k_i)$の値(文献[3]による)

Stage	Design Alternative (k_j)											
	1			2			3			4		
No.	$r_i(1)$	$c_i(1)$	$w_i(1)$	$r_i(2)$	$c_i(2)$	$w_i(2)$	$r_i(3)$	$c_i(3)$	$w_i(3)$	$r_i(4)$	$c_i(4)$	$w_i(4)$
1	0.90	1	3	0.93	1	4	0.91	2	2	0.95	2	5
2	0.95	2	8	0.94	1	10	0.93	1	9	*	*	*
3	0.85	2	7	0.9	3	5	0.87	1	6	0.92	4	4
4	0.83	3	5	0.87	4	6	0.85	5	4	*	*	*
5	0.94	2	4	0.93	2	3	0.95	3	5	*	*	*
6	0.99	3	5	0.98	3	4	0.97	2	5	0.96	2	4
7	0.91	4	7	0.92	4	8	0.94	5	9	*	*	*
8	0.81	3	4	0.9	5	7	0.91	6	6	*	*	*
9	0.97	2	8	0.99	3	9	0.96	4	7	0.91	3	8
10	0.83	4	6	0.85	4	5	0.9	5	6	*	*	*
11	0.94	3	5	0.95	4	6	0.96	5	6	*	*	*
12	0.79	2	4	0.82	3	5	0.85	4	6	0.9	5	7
13	0.98	2	5	0.99	3	5	0.97	2	6	*	*	*
14	0.90	4	6	0.92	4	7	0.95	5	6	0.99	6	9

* means no design alternatives.

数の対数をとることにより，変数分離可能な多次元非線形ナップザック問題に書き換えられる．

仲川ら[8]において，Wの値を 191,190,...,159 に変化させた３３種のテスト問題が，改良された代理制約法（N＆M法）を用いて解かれている．しかし，W＝190,189,187の３問は代理双対ギャップがあるため，N＆M法を用いても最適解を求めることができなかった．代理双対ギャップが深いため，既存の解法では厳密解が求められなかったこれらの３問題について，ISC 法を適用して厳密解を求める．

表－5.2 は求められた厳密解（各ステージのユニット数d_i及び設計案k_i）並びにそのときの総システム費用，総重量，及び信頼度である．更に，原問題 [P]の代理双対ギャップによる困難度を表す PGC 値も示されている．表中括弧内の数値は，N＆M法を用いて求められた解である[20]．

したがって,代理双対ギャップがあるために従来厳密解が求められなかった３問とも ISC 法を用いて，厳密解を求めることができた．計算時間はいずれも１秒未満であった．

表－5.2　代理双対ギャップを有する Fyffe らの問題の厳密な最適解

W	Stage No. i	1	2	3	4	5	6	7	8	9	10	11	12	13	14
190	d_i	3	1	4	3	2	2	1	1	1	2	3	1	1	3
	k	3	2	3	4	3	2	3	4	2	4	2	4	2	2
189	d_i	3	1	4	3	2	2	1	1	1	3	3	1	1	3
	k	3	2	3	4	3	2	3	4	2	3	2	4	2	2
187	d_i	3	1	4	3	2	2	1	1	1	3	1	1	2	3
	k	3	2	3	3	3	2	3	4	2	3	3	4	2	2

W	System Cost	System Weight	Maximum Reliability	PGC (%)
190	130	190	0.985225	72.00
	(132)	(189)	(0.9854*)	
189	129	188	0.984738	54.66
	(131)	(188)	(0.9850*)	
187	125	187	0.983568	38.89
	(133)	(186)	(0.9840*)	

( ) means the values solved by Nakagawa & Miyazaki.

* upper bound values obtained by the N&M algorithm.

5．2．2　4制約式をもつシステムの信頼性最適化問題

Prasad and Kuo が用いた，システムの並列冗長配分による信頼性最適化のテスト問題は，以下のとおりである［11］.

$$[\mathrm{P}^2]: \text{Maximize}\quad R=\prod_{i=1}^{n}[1-(1-r_i)^{x_i}] \tag{5.4}$$

subject to

$$\sum_{i=1}^{n}\alpha_i\ x_i^2 \le \left(1+\frac{\theta}{100}\right)\sum_{i=1}^{n}\alpha_i\ t_i^2\ ;$$

$$\sum_{i=1}^{n}\beta_i\ \exp(\ x_i/2) \le \left(1+\frac{\theta}{100}\right)\sum_{i=1}^{n}\beta_i\ \exp(t_i/2)\ ;$$

$$\sum_{i=1}^{n}\gamma_i\ x_i \le \left(1+\frac{\theta}{100}\right)\sum_{i=1}^{n}\gamma_i\ t_i\ ;$$

$$\sum_{i=1}^{n}\delta_i\ \sqrt{x_i} \le \left(1+\frac{\theta}{100}\right)\sum_{i=1}^{n}\delta_i\ \sqrt{t_i}\ ;$$

$$1 \le x_i \le 10 \qquad \text{for}\quad i=1,2,\ldots,n.$$

ここに，x_iはステージiにおける並列冗長の構成部品の数であり，t_iはb_jを求める際のx_iの基準値である．係数$\alpha_i,\beta_i,\gamma_i,\delta_i$は，それぞれ[6,10],[1,5],[11,20],[21,40]の範囲内で生成させる一様乱数である．表−5．3は各ステージiにおけるユニットの不信頼度（unreliability，$1-r_i$）及び，$\alpha_i,\beta_i,\gamma_i,\delta_i$の値を示している[11]．なお，構成部品の信頼度$r_i$は一様乱数で，範囲[0.95,1.0]で生成されている[11]．θはt_iに基づく各構成部品の最少必要量を与えるための百分率を表す．本論文では，文献［1］と同じ$\theta=33$，$t_i=1$の場合と，新たに$\theta=3$，$t_i=4$の場合を考える．平均最少必要量はステージiにおける必要な校正部品jの下界値の総計で

$$\sum_{i=1}^{n}g_{j,i}(t_i) \tag{5.5}$$

となり，

$$b_j=\left(1+\frac{\theta}{100}\right)\sum_{i=1}^{n}g_{j,i}(t_i)\quad \text{for}\quad n=50$$

表−5.3　数値データ（文献[11]による）

i	$1-r_i$	α_i	β_i	γ_i	δ_i	i	$1-r_i$	α_i	β_i	γ_i	δ_i
1	0.005	8	4	13	26	18	0.023	8	3	19	38
2	0.026	10	4	16	32	19	0.027	10	5	18	36
3	0.035	10	4	12	23	20	0.028	7	4	13	26
4	0.029	6	3	12	24	21	0.03	6	2	15	30
5	0.032	7	1	13	26	22	0.027	6	2	12	24
6	0.003	10	4	16	31	23	0.018	7	2	20	40
7	0.02	9	2	19	38	24	0.013	8	5	19	38
8	0.018	9	3	15	29	25	0.006	9	5	15	29
9	0.004	7	4	12	23	26	0.029	8	1	18	35
10	0.038	6	4	16	31	27	0.022	8	3	16	32
11	0.028	6	5	14	28	28	0.017	9	3	15	29
12	0.021	10	3	15	30	29	0.002	10	1	18	35
13	0.039	9	1	17	34	30	0.031	9	2	19	37
14	0.013	10	4	20	39	31	0.021	7	5	14	28
15	0.038	7	4	14	28	32	0.023	9	5	11	22
16	0.037	10	2	13	25	33	0.03	6	3	15	29
17	0.021	10	1	15	29	34	0.026	7	3	14	27

i	$1-r_i$	α_i	β_i	γ_i	δ_i
35	0.009	6	5	15	29
36	0.019	10	5	17	33
37	0.005	9	5	19	37
38	0.019	10	5	11	22
39	0.002	6	2	17	34
40	0.015	8	3	17	33
41	0.023	10	5	17	33
42	0.04	8	3	18	35
43	0.012	8	1	18	35
44	0.026	6	4	19	38
45	0.038	6	4	13	26
46	0.015	8	1	19	37
47	0.036	7	4	14	28
48	0.032	10	2	19	37

49	0.038	8	3	15	30
50	0.013	10	2	11	22

の平均値が表－5.4 に示されている．問題$[P^2]$の目的関数は対数をとることにより，変数分離可能な多次元非線形ナップザック問題に書き換えられる．

表－5.4　資源最小必要量の平均値とそれに対するb_j

θ	33				3			
j	1	2	3	4	1	2	3	4
$g_{ji}(l_i)$	408	265.44	782	1540	6528	1189.64	3128	3080
b_j	542.64	353.04	1040.06	2048.2	6723.84	1225.33	3221.84	3172.4

θ＝33，t_i＝1の場合，ISC 法を用いて問題$[P^2]$を解くと，各ステージにおける並列冗長の構成部品の最適数，目的関数値である最適信頼度(0.4053895),制約条件式における各式の左辺の値は表－5.5 の上段に示されており，これらの結果は Prasad and Kuo[11]の結果とすべて同一である．この問題の PGC 値は 100%であり，代理双対ギャップはないことがわかった．なお，文献[11]における表－5.4 に対応する表の値は制約条件式を満たしておらず，明らかに記載間違いである．

次に，他の係数は同じ値で，$\theta=3, t_i=4$の場合の信頼性最適化問題を考える．この問題は代理双対ギャップが深いため，従来の解法では厳密解を求めることができなかった．ISC 法を用いて解かれた厳密解，そのときの目的関数値(最適信頼度)，及び制約条件式の左辺の値は，それぞれ表－5.5 の下段に示されている．この問題のＰＧＣ値は 10.8%となり，代理双対ギャップがある問題である．深い代理双対ギャップがあるために従来厳密解が求められなかった問題が，ISC 法を用いて厳密解を求めることができた．計算時間はいずれも１秒未満であった．

表－5.5　厳密な最適解と目的関数値

θ	最適解											f	$\sum \alpha x_j^2$	$\beta_j \exp(x_j)$
33	x_j	1	1	1	2	1	1	1	1	1	2	0.40539	540	291.11358
	x_j	1	1	1	1	2	1	1	1	1	1	$\sum \gamma_j x_j$	$\sum \delta_j \sqrt{x_j}$	PGC(%)
	x_j	2	1	1	1	1	1	1	1	1	1	881	1621.2	100
	x_j	1	1	2	1	1	1	1	1	1	1			
	x_j	1	1	1	1	2	1	2	1	1	1			
3	x_j	3	4	4	4	5	3	4	4	3	5	f	$\sum \alpha x_j^2$	$\beta_j \exp(x_j)$
	x_j	4	4	5	4	5	5	4	4	4	4	0.999985	6712	1217.4595
	x_j	4	4	4	4	3	4	4	4	3	4	$\sum \gamma_j x_j$	$\sum \delta_j \sqrt{x_j}$	PGC(%)
	x_j	4	4	4	4	3	4	3	4	3	4	3141	3078.8	10.8
	x_j	4	5	4	4	5	4	4	5	5	4			

5．2．3　大規模な信頼性最適化問題

ISC 法の有効性を確かめるために，変数の数が多い大規模な信頼性最適化問題を解く．問題$[P^3]$の係数は，5．2．2で用いられた表−5.3 に基づいて作成される．この大規模な問題は 250 変数で，その係数は表−5.6 に示されている．紙面の関係上，表中の係数は省略されている．この大規模問題を，ISC 法を用いて解いたときの各ステージにおける冗長成分の最適数，最大信頼度（0.912227），及び各制約式の左辺の値は表−5.7 に示されてい

表−5.6　数値データ（n=250）（文献[11]による）

i	$1-r_i$	α_i	β_i	γ_i	δ_i
1	0.005	10	4	12	26
2	0.026	10	3	13	31
3	0.035	6	1	16	38
:	:	:	:	:	:
49	0.038	10	4	16	23
50	0.013	8	4	12	24
51	0.005	10	3	13	31
:	:	:	:	:	:
99	0.038	8	4	12	24
100	0.013	10	4	12	26
101	0.005	6	1	16	38
:	:	:	:	:	:
149	0.038	10	4	12	26
150	0.013	10	3	13	31
151	0.005	7	4	19	29
:	:	:	:	:	:
199	0.038	10	3	13	31
200	0.013	6	1	16	38
201	0.005	10	2	15	23
:	:	:	:	:	:
248	0.032	10	3	13	31
249	0.038	6	1	16	38
250	0.013	7	4	19	29

表−5.7　大規模な問題（n=250）の最適解と目的関数値

	最	適	解												
j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x_j	2	2	3	2	2	2	2	2	2	3	2	2	3	2	2
j	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
x_j	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2
j	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
x_j	2	2	2	2	2	2	2	2	2	2	2	3	2	2	3
j	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
x_j	2	3	3	3	2	2	2	3	2	2	2	2	2	2	3
j	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75
x_j	2	2	3	2	3	3	2	2	2	2	2	2	2	2	2
j	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90
x_j	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
j	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105
x_j	2	3	2	2	3	2	3	2	3	2	2	2	3	2	2
j	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120
x_j	2	2	2	2	3	2	2	3	2	3	3	2	2	2	2
j	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135
x_j	2	2	2	2	2	2	2	2	2	3	2	2	2	2	2
j	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150
x_j	2	2	2	2	2	2	3	2	2	3	2	3	3	3	2
j	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165
x_j	2	2	2	2	3	2	2	2	2	3	2	2	3	2	3
j	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180
x_j	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2
j	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195
x_j	2	2	2	2	2	2	2	2	2	2	2	3	2	2	3
j	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210
x_j	2	3	2	3	2	2	2	2	2	3	2	2	2	2	3
j	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225
x_j	2	2	3	2	3	3	2	2	2	2	2	2	2	2	2
j	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240

x_j	2	2	2	2	3	2	2	2	2	2	2	2	2	2	2
j	241	242	243	244	245	246	247	248	249	250					
x_j	2	3	2	2	3	2	3	2	3	2					

f	$\sum \alpha x_j^2$	$\beta_j \exp(x_j)$	$\sum \gamma_j x_j$	$\sum \delta_j \sqrt{x_j}$	PGC(%)
0.912227	10135	2456.254773	8538	11341.09116	56.6

る．この問題の PGC 値は 56.6%であり，代理制約ギャップがあることを意味する．この問題に対する計算時間は，338 秒であった．

5．3　結語

代理双対ギャップがあるために，既存の解法では厳密解が求められなかった 3 種類の信頼性設計の最適化問題を解いた．システムの信頼性を改良するには，より信頼できる構成部品を使用すること，または並列冗長な構成部品を提供することによって実現することができる．システムの各ステージにおいて，複数の異なる信頼度と費用の構成部品が利用可能であり，または並列冗長性が利用可能な場合，システムの最適化問題は非線形整数計画法問題として定式化される．

本章において厳密解を求めた信頼性設計の最適化問題は，2 制約式を持つ直列システムの信頼性の最適化問題，4 制約式を持つ直列システムの信頼性，及び変数の数が多い 4 制約式を持つ直列システムの信頼性の最適化問題である．既存の解法では厳密解が求められなかったいずれの問題とも ISC 法を用いて厳密解を得ることができた．それらの計算時間に関しては，前者 2 種はいずれも 1 秒未満で，変数の多い後者の問題も実用的な計算時間で解くことができた．信頼性設計の最適化問題における原問題の困難度を測定するために，本章においては PGC 値を用いた．PGC の値により代理双対ギャップの有無が明らかになった．

代理双対ギャップがある信頼性問題に対しても，ISC 法が有効であることが明らかになった．また，ISC 法を用いる際，第 2 章で提案した新上界値，及び第 3 章で提案した改良 Dyer 法を用いて最適な代理乗数を決定した．したがって，ISC 法において本論文で提案した新上界値の決定法及び改良 Dyer 法を用いれば，より有用であることが明らかになった．

参考文献

[1] R. E. Barlow and F. Proschan. Mathematical Theory of Reliability. John Wiley & Sons, New York. 1965.

[2] M. E. Dyer, "Calculating surrogate constraints," Math. Programming, Vol. 19, pp. 255－278, 1980.

[3] D. E. Fyffe, W. W. Hines, and N. K. Lee, " System reliability allocation and a computation algorithm", IEEE Trans. Reliab., Vol. R－17, pp. 64－69, 1968.

[4] M. H. Karwan, R. L. Rardin, and S. Sarin, "A new surrogate dual multiplier search procedure," Naval Res. Logist., Vol. 34, pp. 431－450, 1987.

[5] W. Kuo and V. R. Prasad, "An annotated overview of system-reliability optimization," IEEE Trans. Reliab., Vol. 49, No. 2, pp. 176－187, 2000.

[6] Y. Nakagawa, and K. Nakashima, "A heuristic method for determining optimal reliability allocation," IEEE Trans. Reliab., Vol. R-26, pp. 156－161, 1977.

[7] Y. Nakagawa and Y. Hattori, " Reliability optimization with multiple properties and integer variables," IEEE Trans. Reliab., Vol. R-28, pp. 73－78, 1979.

[8] Y. Nakagawa, and S. Miyazaki, " Surrogate constraints algorithm for reliability optimization problems with two constraints," IEEE Trans. Reliab., Vol. R－30, pp. 175－180, 1981.

[9] 仲川勇二, "離散最適化問題のための新解法," 電子情報通信学会論文誌, Vol. J73-A, No. 3, pp. 550－556, Mar. 1990.

[10] Y. Nakagawa, "An improved surrogate constraints method for separable nonlinear integer programming", Journal of the Operations Research Society of Japan, Vol. 46, no. 2, pp. 145－163, 2003.

[11] V. R. Prasad and W. Kuo, "Reliability optimization of coherent systems," IEEE Trans. Reliab., Vol. 49, No. 3, pp. 323－330, Sep. 2000.

[12] 木村作郎, 伊佐田百合子, 仲川勇二, "改良代理制約法を用いたシステム信頼性最適化問題", 電子情報通信学会論文誌 A, Vol. J86－A, No. 10, pp. 1088－1092, Nov. 2003.

[13] S. Kimura, Ross J. W. James, J. Ohnishi, Y. Nakagawa , "The System Reliability Optimization Problems by Using an Improved Surrogate Constraint Method," Advanced Reliability Modeling, Proceedings of the 2004 Asian International Workshop (AIWARM2004), World Scientific Publishing Co. Pte. Ltd., pp. 261－268, 2004. 8

[14] 仲川勇二, 並川哲郎, 木村作郎, 太田垣博一, "代理制約法における最適代理乗数の決定法", 電子情報通信学会論文誌,Vol. J87－A, No.3, pp.364－374, 2004.

[15] D.W.Coit and A.E.Smith," Reliability Optimization of Series-Parallel Systems Using a Genetic Algorithm," IEEE Transactions on Reliability, Vol.45, pp.254－260,1996.

[16] W.Kuo, V.R.Prasad, F.A. Tillman and C-L Hwang,"Optimal Reliability Design -Fundamentals and applications-," Cambridge University Press 2001.

[17] H. Pham (Editor), Handbook of Reliability Engineering, Springer

[18] Y.Hsieh."A linear approximation for redundant reliability problems with multiple component choices." Computers and Industrial Engineering. Vol.44 . pp.91－103. 2002.

[19] D.W.Coit and A.E.Smith."Penalty guided genetic search for reliability design optimization." Computers and Industrial Engineering. Vol.30 . pp.895－904. 1996.

[20] H.Kim. C.Bae and S.Park. "Simulated annealing algorithm for redundancy optimization with multiple component choices." Proceedings of the 2004 Asian International Workshop. pp.237－244. 2004.

[21] Y.Nakagawa. "Studies on optimal design of high reliability system: single and multiple objective nonlinear integer programming." Doctoral thesis. Kyoto University .1978.

[22] Y-C.Liang and A.Smith. " An ant colony optimization algorithm for the redundancy allocation problem (RAP)." IEEE Transactions on Reliability. Vol.53 . pp.417－423. 2004.

第6章　結　論

組合せ最適化は数理計画法の重要な分野のひとつであり，組合せ的な制約条件のもとで，離散値をとる目的関数を最小あるいは最大にするという最適化を扱う．組合せ最適化問題は，理工学から数理経済学，社会学，心理学などの多方面の分野において多くの種類の問題に見られ，効率的に解くことは近年ますます重要になっている．

この組合せ最適化問題は，連続変数の最適化手法との間で本質的な相違がある．連続性や微分の概念にのっとった古典的最適化手法を直接利用することができず，列挙法的な探索アプローチをとらざるを得ない．したがって，組合せ最適化問題は，一般に変数や制約条件式の個数が多くなれば解くことが困難になる．

代理制約法は，組合せ最適化問題の解法の一つであり，代理乗数を用いて原問題の複数制約条件式を単一条件式とした代理問題に変換して解く手法である．原問題が準凸であるときは，複数の制約条件式に乗ずる代理乗数を適当に決定すれば，代理問題の最適解は原問題の最適解と一致する．しかし，離散問題を緩和して考えた代理双対問題は，代理双対ギャップが存在することが多い．この場合には，代理問題の最適解は原問題の最適解と一致せず，原問題の実行可能解になるとは限らない．そこで，仲川が最近提案した改良代理制約法（ISC 法，Improved Surrogate Constraint Method）は，代理双対ギャップを解消し，原問題の厳密解を求めることができるようになったため，有力な解法手段となった．

組合せ最適化問題を解くとき，その最適解の上界値（あるいは下界値）があらかじめ得られていれば，探索を制御するための有効な情報となるため，よりよい上界値を求める方法が重要になる．本研究では，よりよい上界値を求めるための方法を提案した．

また，整数計画問題を代理制約法あるいは ISC 法を用いて解くときに，代理乗数の決定が重要な要因となる．その決定方法の代表的なものとして，Dyer アルゴリズムや COP アルゴリズムが挙げられる．しかし，これらの代理乗数決定方法の長所，短所に関する研究はほとんど行われていない．代理制約法をより効率的に用いるためには，それらを明らかにする必要がある．さらに，ISC 法のように代理制約法の改良に関する研究は行われているが，代理乗数を決定する方法がもつ欠点の改良についてはほとんど研究が行われていない．代理乗数の決定方法の短所を改善し，代理制約法あるいは ISC 法を用いて整数計画問題を解くときに，代理乗数をより速く，より正確に求めて，効率的に最適解が求められるようにする必要がある．

本研究では，代理乗数決定法として，Dyer アルゴリズム及び COP アルゴリズムを示し，それらの長所，短所を明らかにした．さらに，Dyer アルゴリズムの短所を改善した改良 Dyer アルゴリズムを提案し，その有用性を明らかにした．

第2章では，組合せ最適化問題のうちの非線形ナップザック問題に対する最適解を求める際，重要な役割を担う『よい』上界値を求めるための計算法として，変数固定式上界値計算法を提案した．この方法の原理は次のとおりである．線形緩和問題を線形計画法で解くと最適値は実数になる．DGR 優越を考慮した目的関数は凸関数であるので，その変数の複数個を整数に固定すると，他が実数に変わる．整数に固定することにより複数の制約条件が加わり，その中に整数解を含む解空間は小さくなり，その解空間から求められる上界値はよりよくなるためである．同様な上界値の計算式は Sinha and Zoltners も用いているが，本論文で提案した変数固定式上界値計算法の方がよりよい上界値を与えることを示した．また，本論文において上界値を計算する際，計算された部分問題の上界値を利用して，その次の部分問題の上界値を計算するという効率化を図った．計算機実験として簡単な例題及び実用規模の非線形ナップザック問題を解くことにより，提案した変数固定式上界値計算法を用いて得られる上界値は，Sinha and Zoltners が用いた上界値よりも，よりよい上界値であることが明らかになった．

第3章では，代理制約法あるいは ISC 法において最適な代理乗数を決定するためのアルゴリズムとして知られる，Dyer アルゴリズムと COP アルゴリズムを示し，計算機実験を通して，それらの長所及び短所を明らかにするために比較検討した．その結果，以下のことが明らかになった．

(1) 最適な代理乗数が求まるまでの更新回数及び実行時間は，両アルゴリズムとも変数の数と制約条件の数が増えると増大する．これらに及ぼす影響は，変数の数の増加による影響よりも制約条件の数の増加による影響の方が大きい．

(2) 最適な代理乗数が得られるまでの更新回数及び実行時間ともに，COP アルゴリズムの方が Dyer アルゴリズムよりも少ない場合が多い．

(3) 制約条件式の数が増えると，Dyer アルゴリズムでは最適な代理乗数が得られる問題でも，COP アルゴリズムはメモリ不足で解くことができない場合があった．これは，計算過程において，COP アルゴリズムが多面体の頂点の座標情報を使っているため，切断面の平面式を使う Dyer アルゴリズムよりも計算の作業領域量が多くなるためである．

(4) Dyer アルゴリズムの実行時間が COP アルゴリズムの実行時間よりもかなり長くなる場合があった．これは，Dyer アルゴリズムにおいて，代理乗数を計算する過程において，切断面に内接する球の半径がほとんど変化しない状態が続き，多面体の領域が狭まらなくなるためであると考えられる．

さらに，第3章において，上記(4)で記述された Dyer アルゴリズムの短所を改善するため，改良 Dyer アルゴリズムを提案した．この改良 Dyer アルゴリズムは，代理乗数を決定する過程において，内接する球の半径の大きさの決定に全く関係していない切断面，及び関係しているがその割合（ＲＰ率）が小さい切断面を削除するものである．しかも，この操作は Dyer アルゴリズムを一定回数（ＲＰ周期）行った後，周期的に実行する．この改良により，代理乗数を計算するときに現れる切断面の候補数が減少するため，使用メモリ量が減少し，実行時間も短縮される．計算機実験において制約条件式及び変数の数が多い問題を

解くことによって，改良 Dyer アルゴリズムが Dyer アルゴリズムに比べて，計算速度及び使用メモリ量に関して優れていることが明らかになった．

第４章では，第３章で提案した改良 Dyer アルゴリズムの有効性を確かめ，さらにその特性を評価するために計算機実験を行った．計算機実験としては，制約条件式及び変数の個数が実用規模から大規模のテスト問題に対して，擬似乱数の種を変化させて作成した 10 問を，ＲＰ率，ＲＰ周期の値を変化させて解き，そのときの平均実行回数と平均実行時間を計算した．その結果，Dyer アルゴリズムと改良 Dyer アルゴリズムのそれぞれを比較検討し，改良 Dyer アルゴリズムが Dyer アルゴリズムに比べて，計算速度及び使用メモリ量に関して優れていることが明らかになった．さらに，ＲＰ率とＲＰ周期の値による実行時間，実行回数の影響を示した．

(1)　制約条件式の数が多いとき，各ＲＰ周期においてＲＰ率が増加すると実行時間は短くなる傾向がある．RP 周期が大きいとき，ＲＰ率が 0.1 から 0.3 に増加するにしたがって，実行時間は単調増加する．

(2)　ＲＰ周期が 30, 50, 80, 100 のいずれの場合においても，ＲＰ率が 0.1 及び 0.2 の場合はいずれも 10,000 回以内で代理乗数が収束して最適解が得られた．ＲＰ率を 0.3 とした場合，ＲＰ周期が 80, 100 のとき最適解は得られたが，ＲＰ周期が 50, 30, 10 と小さくなるにつれて，10,000 回以内で代理最適解が得られないことが起こった．特に，ＲＰ周期が 10 のときには，制約条件数が 10 のときからすべての場合で，代理乗数が収束しなかった．

(3)　ＲＰ率を 0.4 に設定すると，制約条件及び変数の数が増えると，10,000 回以内で代理乗数が収束しなかった．特に，制約条件数が 15 を超えると 100 変数以上のすべての問題で代理乗数が収束しなかった．これはＲＰ率が大きくなると，最適解を求める途中において代理乗数の多面体のほとんどが削除され，10,000 回以内で最適解が得られなかったと考えられる．

(4)　ＲＰ周期が 10, . . . , 100 のいずれの場合も，ＲＰ率が 0.1 から 0.4 に増加するにしたがって，実行回数は増加する．これは縮小する切断面を選択するための目的関数値の割合を大きくすることにより，選択する候補の数が減るため最適解が得られるまでの実行回数が増えるためである．

さらに，第４章では，大規模な変数分離型非線形整数計画問題を解くために有効となるＲＰ率とＲＰ周期の値の組合せも示した．実行時間は多少多く要するが，最適な代理乗数がより確実に得られるためにはＲＰ周期 80，ＲＰ率 0.3 がより適当であろう．

第５章では，代理双対ギャップがあるために，既存の解法では厳密解が求められなかった３種類の信頼性設計の最適化問題を解いた．それらは，２制約式を持つ直列システムの信頼性の最適化問題，４制約式を持つ直列システムの信頼性，及び変数の数が多い４制約式を持つ直列システムの信頼性の最適化問題である．いずれの問題とも ISC 法を用いて厳密解を得ることができた．それらの計算時間に関しては，前者２種はいずれも１秒未満であり，変数の多い後者の問題も実用的な計算時間であった．信頼性設計の最適化問題における原問題の困難度を測定するために，本論文においては PGC(Percent Gap Closure)値を用いた．PGC 値により代理双対ギャップがあることが明らかになった．

代理双対ギャップがある信頼性問題に対しても，ISC 法が有効であることが明らかになった．また，ISC 法を用いる際，第２章で提案した新上界値，及び第３章で提案した改良 Dyer 法を用いて最適な代理乗数を決定した．したがって，ISC 法において本論文で提案した新上界値の決定法及び改良 Dyer 法を用いれば，より有用であることが明らかになった．

謝　辞

本研究を結ぶにあたり，本研究を遂行する上でご指導とご支援をいただいた方々に感謝の意を表します．

関西大学総合情報学部　仲川勇二教授には，筆者が関西大学高槻キャンパス事務室に勤務している際，研究活動を再開する機会を与えていただきました．さらに，本研究の構想から，遂行にわたって暖かいご指導とご鞭撻をいただきました．ここに深甚なる感謝の意を表します．また，毎月の研究会に参加させていただき，そこで他大学の先生方と議論する機会を得て，研究を進めるにあたり非常に有意義でした．重ねて深く感謝いたします．

本論文をまとめるにあたり，ご多忙中にもかかわらず，暖かくかつ適切なご助言，ご指導を賜りました関西大学工学部システムマネジメント工学科　森　健一教授に厚くお礼申し上げます．また，ご多忙中にもかかわらず適切なご助言を賜り，ご鞭撻いただきました関西大学工学部先端情報電気工学科　原　武久教授，ならびに同システムマネジメント工学科　中井暉久教授に心よりお礼申し上げます．

岡山理科大学工学部電子工学科　太田垣博一教授には，研究を進めるにあたり，様々な議論をしていただき，また貴重なご助言をいただきましたことに心より感謝いたします．

学位論文を申請することに対して快く了承し，暖かく励ましていただきました，高槻キャンパス事務局長　浅尾　勇氏，ならびに前高槻キャンパス事務長　山田　賢一氏に心より感謝いたします．また，高槻キャンパス事務室の職員諸氏，とりわけネットワークセンターの職員諸氏には，筆者が日常業務を遂行するにあたり協力，援助していただいたことに感謝いたします．

最後に，本研究を進めることに対して，家庭にあって筆者のわがままを許し，好きなことをすることに寛容であった妻　正美と娘達　仁美と聡江に深く感謝いたします．