数 値 解 析 36－3
（1991. 3. 15）

優先順位を考慮した大規模一般化割当問題のための近似解法

錦織昭峰　*一森哲男　渡辺展男　金指正和　青木兼一
広島県立大学　*大阪工業大学

本論文では、優先順位に従った割当を行って、割当問題の実行可能解を求める近似解法を提案する。提案法は、利益あるいは費用が金額として明確に与えることができない割当問題を解くのに有効である。提案法では、(i)単調減少しにくい実行不可能和の最小化を行うためには、どのような複数の整数変数を一度に変化させるべきかを探索木を用いて調べており、(ii)得られた探索によって逐次割当を変更する際には、変更したところ以外の探索木は残しておいて、次の探索の際の情報として用いている。大規模な割当問題を用いて数値シミュレーション実験を行った結果、提案法は効率的な方法であることが示された。

AN APPROXIMATION METHOD FOR SOLVING LARGE-SCALE GENERALIZED ASSIGNMENT PROBLEMS WITH PRIORITY ORDER

Akimine Nishikori　*Tetuo Ichimori　Norio Watanabe　Masakazu Kanezashi　Kenichi Aoki

Hiroshima Prefectural University　*Osaka Institute of Technology
Shobara, Hiroshima 727, Japan
*Osaka 540, Japan

This paper presents an approximation method for obtaining feasible solutions of large-scale generalized assignment problems with priority order. The proposed method is applicable to assignment problems whose cost coefficients cannot clearly be determined. In the method, the following two procedures are iteratively carried out to decrease the sum of infeasibilities ; (i) a set of integer variables to be changed is chosen by using search trees, and (ii) the trees are modified after the integer variables are adjusted. The method has proved to be efficient as a result of applying it to large-scale assignment problems.

１．まえがき

割当問題は、整数変数が０あるいは１の値しかとらない０－１整数計画問題[1,2]の一種である。この割当問題では、ある期間帯ｉ（あるいは場所ｉ）に、ある資源ｊ（例えば、人数、設備、資金など）を割り当てたときの、「利益」（あるいは「コスト」）C_{ij}を目的関数の利益係数として用いている。この係数C_{ij}が、「利益」あるいは「コスト」という明確な金額で表すことができず、単に、ある期間帯ｉに、ある資源ｊを割り当てる「重要度」あるいは「優先度」という「重み付け」を表している場合がある。従来このような場合には、資源ｊに対して、期間帯ｉの重要な順

$$i_1, i_2, \cdots, i_m$$

に対して、目的関数に用いられる利益係数C_{ij}を

$$C_{(i1)j} > C_{(i2)j} > \cdots > C_{(im)j}$$

となるように適当な比率を設けて重要な順に重み付けを行って、できるだけ重要度を満足させてきた[4,6]。

しかしながら、このような割当問題で用いられるのは０－１整数変数であるので、資源の割当を表す変数の値は０あるいは１で求まり、必ずしも重み付けに比例した値が求められない。すなわち、このような優先度は本来数値化すること自体が困難であり、優先度をどのように重み付けしたらよいという明確な基準がない。

ところで、このような割当問題では、ある期間帯に、どのような資源の割当をするかしないかという候補が事前に決まっており、更にその候補の中にも、好ましい割当候補と、そうでない割当候補があることが多い。このような場合には、予め決められた複数の割当候補の中で、割当を要望する順序、すなわち「優先順位」に従った割当を行ったほうが良い。このような点を考慮した割当を行う問題を、以降、「優先順位を考慮した割当問題」と呼ぶことにする。この問題は、優先順位を考慮した割当を行って、多数の実行可能解のうちで、ある一つの実行可能解を求める問題となる。筆者らは、電力システムにおいて、このような優先順位付きの最適化問題を考察してきた[3]。従来、利益あるいは費用が金額として明確に与えられた場合に割当を行う方法[1,2,4]は多数提案されているが、優先順位を考慮した割当を行う方法は未だに提案されていない。

また、実行可能解を求めるための方法[2,1]に関して述べると、従来では、不等式制約式の実行不可能和を評価関数にして、整数変数を変化させてその和を減少させて零にしていた。しかしながら、評価関数は多数の実行不可能量の和として表されるため、整数変数を変化させて、ある制約式の実行不可能量を減少させても、別の制約式の実行不可能量が増加することが起こって、非増加ではあるが単調減少しにくくなることがある。このようなときには、更に続いて他のどのような整数変数を変化させることが必要かを組織的に探索してみなければならない。

本論文では、利益あるいは費用が金額として明確に与えられていない割当問題を解く際に、割当を行う順番を表す優先順位を設定し、その優先順位に従った割当を行って実行可能解を求めるアルゴリズムを提案する。この提案法では、

- 単調減少しにくい実行不可能和の最小化を行うためには、どのような複数の整数変数を一度に変化させるべきかを探索木を用いて調べており、
- 得られた探索によって逐次割当を変更する際には、変更したところ以外の探索木は残しておいて、次の探索の際の情報として用いている。

このアルゴリズムでは、実行不可能和を最小化する整数計画問題を繰り返し解く必要があるが、整数変数の個数が非常に多い大規模な割当問題では、真の最適解を効率よく求めることが困難であることが知られている[1,2]。従って、本研究では、現実に起こる非常に大規模な実際問題にも対応できるようにするために、真の最適解を求めることはあきらめて、満足できる実行可能解を少ない計算量で求めることができる近似解法を開発して、提案法で用いている。

本論文での提案法を用いて、乱数により作成した大規模な割当問題を解いた結果、実用的な計算時間内に、優先順位を考慮した実行可能解を求めることができた。

本論文の概要は次のとおりである。第２章では対象とする割当問題の制約条件式を示す。第３章では、優先順位を考慮した割当問題に関する考察を行う。第４章では提案法の概要を説明し、第５章で提案法を記述する。第６章では、提案法による数値シミュレーショ

ン実験を述べる。第７章では本研究のあとがきを記す。

２．制約条件

対象とする割当問題の制約条件を、以下に記す。

$$\sum_{j \in \Gamma^{ik}} x_{ij} \leqq b_{ik} \quad k \in \Lambda^i \ ; \ i=1,2,\ldots,m \qquad (1)$$

$$\sum_{i=1}^{m} x_{ij} = a_j \qquad j=1,2,\ldots,n \qquad (2)$$

$$x_{ij} = 0 \text{ or } 1 \quad i=1,2,\ldots,m \ ; \ j=1,2,\ldots,n \qquad (3)$$

ここでΛ^i とΓ^{ik}は添字の集合、 b_{ik}, a_j は正の整数であり、

$$x_{ij} = \begin{cases} 1 & \text{時間帯 } i \text{ （すなわち節点 } i \text{）に資源 } j \text{ を割り当てるとき} \\ 0 & \text{そうでないとき} \end{cases}$$

である。以降、提案法を説明する際には、「時間帯ｉに割り当てる」という代わりに、「節点ｉに割り当てる」ということもある。

この定式化は、次に述べるような二つの点から、普通の割当問題の制約条件を一般化している。

(i) 普通の割当問題では、 (2)式のａ$_j$ はつねに１であるが、上記の定式化では一般には１でない。これは、同一の資源が複数個ある場合も考慮するためであり、各時間帯ｉと各資源ｊは一般には１対１に対応していない。従って、提案法では、同一複数個の資源ｊを、異なった複数の期間帯ｉに一つずつ割り当てることを考えている。

(ii)各期間帯ｉへ、異なった複数の資源を割り当てることも考慮するが、割り当てることができる資源の総数に上限を設ける。この際に、各資源には種類（あるいはタイプ）があるならば、その種類毎に、割り当てられる資源の総数に上限を設ける。それ故、 (1)式のように各期間帯ｉ毎に制約式番号を表す添字ｋを設けて、複数の制約式を考慮できるようにしている。ここで、各ｉでの制約式の数は $|\Gamma^{ik}|$ である。

例えば、二つの種類の資源ｊ＝１，２と ｊ＝３，４，５があり、期間帯ｉに割当可能な資源の個数は、それぞれ １，２ であるとする。このとき、 (1)式は次のような制約式を表している。

$$\sum_{j \in \{1,2\}} x_{ij} \leqq 1 \quad ; \quad \sum_{j \in \{3,4,5\}} x_{ij} \leqq 2 \qquad (4)$$

３．優先順位を考慮した割当問題に関する考察

割当問題[1,2] の評価関数で用いられる係数C_{ij} は、ｉをｊに（またはｉにｊを）割当てる場合の「利益」あるいは「コスト」を表している。しかしながら、「利益」が割当の「優先度」あるいは「重要度」を表している場合がある。このような例として、

(i) 講義室の数とその大きさに種類がある場合に、各時間帯毎に、各先生の授業科目を開講する講義室を決定する時間割の問題[6,7] 、

(ii)設備などの資源制約を満たしつつ、複数の作業期間帯に、何種類かの仕事を割り当てる生産計画の問題[5]

などがある。

このような優先度は本来数値化すること自体が困難であり、優先度をどのように重み付けしたらよいという明確な基準がない。従って、このような場合には、優先度の総和を最大にするよりもむしろ、優先度の大きい順に割り当てる解を求めた方が良い。

いま、ある資源ｊを、期間帯ｉ＝１，２，･･･，ｍ のどれかに割り当てるとき、

第１候補として　ｉ＝ｉ$_{1j}$

第２候補として　ｉ＝ｉ$_{2j}$

・・・

第ｍ候補として　ｉ＝ｉ$_{mj}$

という候補の順位、すなわち優先順位に従って割当を

行うことを考える。このとき、すべての資源j＝1，2，…，nの第1候補が異なっており、その結果すべての期間帯i＝1，2，…，m に偏りなく割り当てられれば良い。しかしながら、多数の資源の第1候補がいくつかのある特定の期間帯に集中したときには、第2候補以下に繰り下げなければならない。すなわち、優先順位に従って割り当てるというのは、まず最初に、資源jを第1候補のi＝i_{1j}に割り当てることを考える。このとき、すでに期間帯i_{1j}に他の資源が割り当てられており、この期間帯i_{1j}での制約を違反する場合には、資源jを第2候補の期間帯i_{2j}に割り当てることを考える。以下同様にして、制約を違反しない期間帯がみつかるまで、第3候補以下の期間帯に割り当てることを考えていく。このような繰り下げが、多数の期間帯で起こる場合には、各々の資源を割り当てた期間帯の優先順位ができるだけばらつかないように考慮しつつ、各々の資源毎の優先順位にできるだけ添った割当を決定するために、各資源間での割当の調整が必要となる。

各資源間での割当の調整に関する一例を以下に示す。いま、資源j＝1，2，3 に対して、

j＝1の優先順位は　i＝1，2，3 の順

j＝2の優先順位は　i＝2，3，1 の順

j＝3の優先順位は　i＝1，2，3 の順

とし、各期間帯i＝1，2，3 には各々一つずつしか資源の割当ができないとする。単純に、資源j＝1，2，3 の順番に割当をしていくと、割当は

(i,j) = (1,1)，(2,2)，(3,3)

となり、資源j ＝1，2 は優先順位第1位の割当となるが、資源j＝3だけは第3位の割当となる。しかしながら、この割当を調整してやると

(i,j) = (1,1)，(3,2)，(2,3)

の割当となり、すべての資源が第2位までの割当となって、割当のばらつきがなくなる。

本研究では、この「優先順位を考慮した割当問題」のために、次のようなアルゴリズムを提案する。

「優先順位を考慮した割当問題」のアルゴリズム

手順0　各資源jに対して、第r位の優先順位で割り当ててよい期間帯iの集合をI^{jr}と表す。ここで、各I^{jr}（r=1,2,...,r_s）は、同じ要素iを含まず、

$$\bigcup_{r=1}^{r_s} I^{jr} = \{1,2,\cdots,m\} \quad j=1,2,\cdots,n \qquad (5)$$

である。r＝1とし、$I^j := I^{j1}$とする。

手順1　各資源jを割り当てる期間帯iを、集合I^jの範囲内で変更することによって、すべての制約式の実行不可能和 f(X) を最小化する。

手順2　実行不可能和 f(X) が零になったならば、解が求まったので終了する。さもなければ手順3へ。

手順3　r＝r_sならば、実行可能解はなしと判定して終了する。さもなければ、

$$I^j := I^j \cup I^{j(r+1)} \qquad (6)$$

とし、r := r＋1として、資源jが割当可能な期間帯iの数を増やして、手順1へもどる。

手順1で用いられる整数計画の解法について述べると、0－1整数変数の個数が非常に多い大規模な割当問題では、真の最適解を効率よく求めることが困難であることが知られている[1,2]。市販の大型計算機により解ける問題の大きさの限界を述べると、現在のところ、連続変数と整数変数の両方を含む混合整数計画問題では、0－1整数変数の個数が450個程度までならば真の最適解を求め、また、2100個程度までならば実行可能解が求められている[5]。従って、真の最適解を求めることはあきらめて、満足できる実行可能解を少ない計算量で求めることができる近似解法を開発して、手順1で用いることが必要となる[1,2]。

4．提案法の概要

本研究での提案法の概要を、図1に示すような例を用いて説明する。この例では、節点i＝1，2 に、そ

れぞれ、資源 j = 1, 2, 3 と j = 3, 4, 5 がすでに割り当てられている。

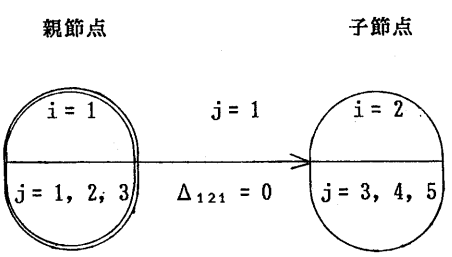

図1　割当変更のための展開

更に、節点 i = 1, 2 の制約条件として、それぞれ、

$$x_{11} + x_{12} + x_{13} \leqq 2 \quad (7)$$

$$x_{11} + x_{13} + x_{14} \leqq 1 \quad (8)$$

$$x_{21} + x_{22} + x_{23} + x_{24} \leqq 2 \quad (9)$$

$$x_{21} + x_{22} + x_{24} + x_{25} \leqq 2 \quad (10)$$

があるとする。このとき、節点1の実行不可能量 $f^1(X^1)$ は、

$$f^1(X^1) = \max\{(x_{11} + x_{12} + x_{13} - 2), 0\} + \max\{(x_{11} + x_{13} + x_{14} - 1), 0\} = 2 \quad (11)$$

であり、節点2の実行不可能量 $f^2(X^2) = 0$ である。ここで、

$$X^1 = \{1, 2, 3\}, \quad X^2 = \{3, 4, 5\} \quad (12)$$

である。実行不可能である節点1は◎で表すことにする。いま、節点1に割り当てられている資源 1, 2, 3 のうちのどれかを、節点2へ移すことを考える。これを、「未展開節点1を展開する」ということにする。資源1を、節点1から2へ移動するならば、節点1、2の実行不可能量の変化量 Δ_{p11}, Δ_{c21} は、

$$\Delta_{p11} = 0 - 2 = -2 \quad (13)$$

$$\Delta_{c21} = 2 - 0 = 2 \quad (14)$$

となり、結局実行不可能和の変化量 Δ_{121} は

$$\Delta_{121} = \Delta_{p11} + \Delta_{c21} = 0 \quad (15)$$

となり変化しない。資源2を節点1から2へ移す場合も同様に、実行不可能和が減少しない。そこで、資源1を節点1から2へ移した後、更に、節点2にすでに割り当てられている資源を、別の節点3に移すことを考える。これを、「深さ2で未展開節点2を展開する」ということにする。いま、図2のように、節点3には、資源 1, 2, 3, 5 が割り当てられており、制約条件として、

$$x_{32} + x_{33} + x_{34} + x_{35} \leqq 3 \quad (16)$$

があるとする。

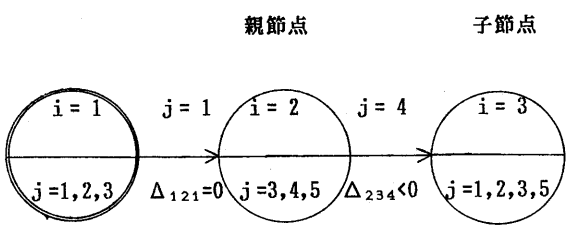

図2　割当変更のための「深さ2」の展開

このとき、資源4を節点2から3へ移すならば、節点2と3の実行不可能量の変化量は、それぞれ、-2, 1 であるので、結局、実行不可能和の変化量 Δ_{234} は、

$$\Delta_{234} = (-2) + 1 = -1 < 0 \quad (17)$$

となり減少する。従って、実行不可能和 f(X) が減少することがわかったので、実際に、節点1から2へ資源1の割当を変更し、節点2から3へ資源4の割当を

変更して、図３のように ⇒ で表すことにする。これを、「根節点１から節点３までのパスによる割当変更」ということにする。

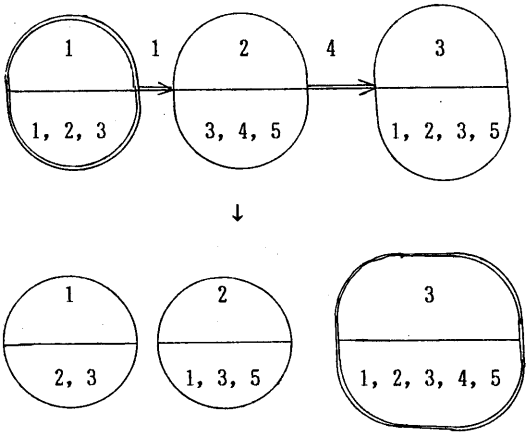

図３　根節点１から節点３までのパスによる割当変更

このような割当の変更を行った結果、節点３では逆に実行不可能になったので、節点３を根とする新しい探索木を設けて、節点３を未展開節点とする。しかしながら、実行不可能和を単調に減少させている。

以下同様にして、実行不可能量が零でない節点に割り当てられている資源を、他の節点へと移動することにより、すべての節点での実行不可能量を零にする。すなわち、図２に示すように、制約を違反した◎の節点ｉを探索木の根とし、ある親節点に割り当てられていた資源を、別の子節点への割当に変更を考えてみるならばアーク → を付けるという手続きを繰り返して探索していく。図４、５では、図６に示すように、深さ３のパス ⇒ で割当の変更を行った一例を示しており、---> は削除されたアークを表している。図５において、節点i_5 から節点i_3 への割当変更の ⟶ が無効となり削除されて--->となったのは、節点i_3 に資源j_3 が割り当てられてしまったために、節点i_5 の資源j_3 を節点i_3 に移すことができなくなったので、節点i_5 を根とする探索木を修正したことを示している。また、資源j_2 を節点i_1 からi_6 へ移す ⟶ が、--->となったのは、節点i_1 に割り当てられていた資源j_2 が、節点i_2 に割り当てられてしまったために、節点i_6 へ割り当てることができなくなったからである。

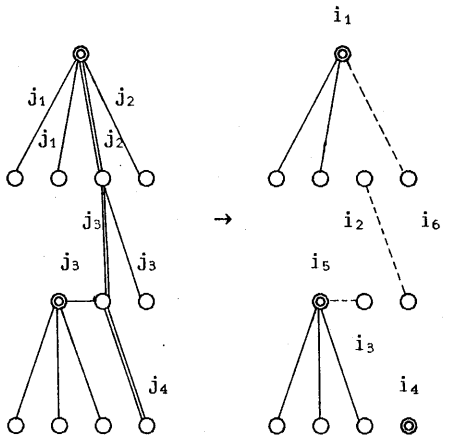

図４　割当変更前の探索木　　図５　割当変更後

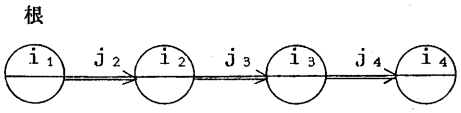

図６　深さ３のパスによる割当変更の例

５．提案法

提案法では、(2),(3) 式を満足させながら、節点ｉへの資源ｊの割当を変更していく。いま、節点ｉに割当られている資源ｊの集合を X^i と表す。すなわち、

$$X^i = \{ j \mid x_{ij} = 1 \ ; \ j=1,2,\ldots,n \} \qquad (18)$$

制約条件の実行不可能量の総和を

$$f(X) = \sum_{i=1}^{m} f^i(X^i) \qquad (19)$$

と表す。ここで、

$$f^i(X^i) = \sum_{k \in \Lambda^i} \max\{ (\sum_{j \in \Gamma^{ik}} x_{ij} - b_{ik}), 0 \} \quad (20)$$

(6)式で定義したように、資源jに対して、第1位から第r位 ($1 \leqq r \leqq r_s$) までの優先順位で $x_{ij} = 1$ としてよい節点iの集合を I^j と表す。

提案法

第3章のアルゴリズムに従った提案法を以下に記す。

ステップ0：初期状態

$f^i(X^i) > 0$、すなわち、制約式が実行不可能である節点iを根とする探索木を設けて、この根を未展開節点とする。優先順位 $r = 1$ とする。

ステップ1：展開節点の選択

未展開節点を一つ選んで i_p と表す。

ステップ2：親節点の展開

未展開節点 $i = i_p$ に対して、$\Delta_{pij} < 0$ となる資源jの集合を J_{ip} と表す。ここで、

$$\Delta_{pij} = f^i(X^{pij}) - f^i(X^i) \quad (21)$$

$$X^{pij} = X^i - \{ j \} \quad (22)$$

ステップ3：子節点の決定

すべての資源 $j \in J_{ip}$ に対して、$\Delta_{(ip)ij}$ ($i \in I^{jr}$) を計算する。ここで、

$$\Delta_{(ip)ij} = \Delta_{p(ip)j} + \Delta_{cij} \quad (23)$$

$$\Delta_{cij} = f^i(X^{cij}) - f^i(X^i) \quad (24)$$

$$X^{cij} = X^i + \{ j \} \quad (25)$$

ステップ4：割当変更の判定

ステップ3で $\Delta_{(ip)ij} < 0$ となる $(i,j)=(i^x,j^x)$ があるならば、次のステップへ。さもなければステップ7へ。

ステップ5：割当の変更

根節点から節点 i^x までのパスによって資源の割当を変更して、X^i を修正し、無効になった節点を削除する。その結果、新たに制約式が実行不可能になった節点 i' に対しては、i' を根とする新しい探索木を設けて、i' を未展開節点とする。

ステップ6：解の判定

探索木が一つもない、すなわち、$f(X) = 0$ ならば、実行可能解が求まったので終了する。さもなければ、ステップ8へ。

ステップ7：新しい未展開節点

ステップ3で $\Delta_{(ip)ij} = 0$ となる $(i,j)=(i^x,j^x)$ があるならば、子節点 i^x をすべて未展開節点とする。

ステップ8：展開節点の変更

未展開節点が一つでもあれば、ステップ1へ。さもなければ、次のステップへ。

ステップ9：解なしの判定

$r = r_s$ ならば、これ以上優先順位を更新することができないので、実行可能解はなしと判定して終了する。さもなければ、次のステップへ。

ステップ10：優先順位の更新

(6)式のように I^j を更新し、$r := r + 1$ とする。すなわち、優先順位を一つ増やして、資源jが割当可能な節点iの数を増やす。更に、制約式が実行不可能である節点を、再び未展開節点とする。ステップ1へ。

6．数値シミュレーション実験

提案法を用いて数値計算を行った結果を表1～4に示す。MS-FORTRANを用いたが、実行可能ファイルのサイズは359KBである。

例1～4の問題は、乗積合同法[8]によって発生させた乱数を用いて、以下に述べるように作成した。ま

ず、$\max_j a_j$ と $\max_i |\Lambda^i|$ を与えて、乱数で a_j の値と集合 Λ^i を決定する。次に、$\max_i \max_k b_{ik}$ を与えて、乱数で b_{ik} の値を決定する。更に、$\max_i \max_k |S_{ik}|$ を与えて、乱数で集合 S_{ik} を決定する。最後に、優先順位を表す集合 I^{jr} を乱数で決定する。

提案法の性能を調べるために、各イテレーション毎に、従来法を用いて実行不可能和をできるだけ減少させた後に、更に提案法により減少させることにする。

従来の実行可能解を求める方法として、いくつかの変数（N個の変数）を選んで一度に変化させるN－最適化法[2]がある。割当問題においては、1－最適化では必ず(2)式を満足しなくなるので、Nは2以上となる。Nが大きくなると、選ぶ変数の組み合わせの候補の数は非常に膨大なものとなるので、従来法では2－最適化を行うこととする。これは、提案法における深さ1の探索と同じ手続きを行っている。なお、従来法では、ある節点に割当られていた資源を、他の節点へ割当を変更しても実行不可能和が減少しないと一度判定されたならば、以後この節点を調べないとする。

例1～4の数値例を解いた結果を、表1～表4に示している。表1の項目Aにあるように、各例の0－1整数変数の個数は、それぞれ、 11000, 8000, 5400, 105000 個である。

表1の項目Bは、資源の総数 $\Sigma_j a_j$ を表し、項目CとDは、それぞれ、すべての期間帯での制約式の総数 $\Sigma_i|\Lambda^i|$、すべての制約式に含まれる x_{ij} の項の総数 $\Sigma_i\Sigma_k|\Gamma^{ik}|$、を表している。また、すべての資源を、優先順位第1位の期間帯に割り当てた状態を初期解とすると、項目Sは、その初期解において、一つ以上の制約式が実行不能である期間帯iの個数を表している。更に、項目Tは実行不可能な制約式の総数であり、項目Uは実行不可能量の総和（すなわち評価関数値）を表している。

表2と3では、例1～4の探索状況を示している。表2で示した例1を説明すると、第3章で記述したアルゴリズムに従って、割当可能な期間帯の優先順位rを一つずつ下げており、例1では優先順位第4位までで実行可能解が求まっている。すなわち、第4回めのイテレーションにおいて、優先順位第1位から第4位までの範囲の期間帯に割り当てた結果、実行可能解が求まっている。表2の上部には、各イテレーションにおいて割当を変更した結果、各優先順位の期間帯がいくつ割り当てられたかを表している。

各イテレーションrにおいて、項目aは、すべての資源において新たに割当可能となった期間帯の総数、すなわち $\Sigma_j I^{jr}$ を表している。各イテレーションrでの探索において、図3に示したような資源割当の変更をした回数（項目b）と、それらの変更のうちで、図6で説明した「深さ」が2以上の変更を行った回数（項目c）、及び、「変更の最大の深さ」とそのときに「変化させた変数の個数」（項目d）を示している。なお、提案法におけるステップ1の未展開節点の選び方は幅優先探索[1]としており、深さの最大値は5として、6以上の深さの探索は打ち切っている。また、その変更によって展開された親節点の個数（項目e）と、子節点の個数（項目f）を示している。更に、すべての節点のうちで、割当を変更した後に残った節点の比率（すなわち、例えば、（図5のアークの数）÷（図4のアークの数））（項目g）を示している。また、そのような割当変更の結果、制約を違反している期間帯の個数（項目s）と、制約を違反している式の個数（項目t）と、評価関数値（実行不可能和）（項目u）が、どのような値になったのかを示している。その実行不可能和の減少量のうち、従来法と提案法による減少量（項目pとq）もまた示している。

表2の項目qからわかるように、例1は従来法だけでは優先順位第4位までで実行可能解を求めることができなかったが、提案法を用いて求めることができた。例2、3、4においても同様に、従来法のみでは、それぞれ、第4位、第5位、第3位以内の割当では求めることができないが、提案法によって可能となった。

表2と3の項目pとqでは、例1～4において、イテレーション回数rが大きくなるにつれて、評価関数値が単調に減少していったことを示している。この事より、提案法では、実行不可能和の単調減少性を保証することによって収束の保証を得ていることがわかる。

表4では、各例で実行可能解が求まるまでの計算時間を示している。この表からわかるように、提案法は、パソコンを用いて、実用的な計算時間内に優先順位付きの割当による解を求めている。

表１　数値例の初期値データ

数値例	例１	例２	例３	例４
A：変数の次元（ = m*n ）	100*110	80*100	60*90	300*350
B：非零 x_{ij}(=1)の個数	1667	1058	984	5620
C：制約式の総数	1293	903	818	4905
D：制約式の非零係数の個数	15959	11609	11499	84089
S：制約を違反した節点の個数	79	64	50	224
T：制約を違反した式の個数	408	328	247	565
U：評価関数値(実行不可能和)	866	653	649	832

表２　例１の探索状況

イテレーション回数 / 優先順位 r	1	2	3	4
1	1667	1389	1235	1136
2	／	278	263	257
3	／	／	169	175
4	／	／	／	99
a：新たに割当可能となった節点の総数		427	428	409
b：割当変更回数		4	13	16
c：深さ２以上の割当変更回数		0	13	16
d：割当変更の深さの最大値		1	5	2
一度に変化させた変数の最大個数		2	10	4
e：親節点の展開回数		2637	2305	369
f：子節点の個数		582	1647	1608
g：節点の平均生存率（単位は％）		93	80	47
p：従来法による減少量		562	183	88
q：提案法による減少量		4	13	16
s：制約を違反している節点の個数		79	54	0
t：制約を違反している式の個数		193	79	0
u：評価関数値(実行不可能和)		300	104	0

表3　例2,3,4の探索状況

例	2			3				4	
イテレーション回数	2	3	4	2	3	4	5	2	3
e：親節点の展開回数	1461	1624	806	1765	929	2582	430	4073	291
f：子節点の個数	371	1236	2559	268	169	2041	1107	3693	863
g：節点の平均生存率（%）	100	74	36	98	100	63	44	92	87
p：従来法による減少量	434	120	53	367	123	77	25	627	154
q：提案法による減少量	4	14	28	2	1	23	31	24	27

表4　数値例の計算時間

	例1	例2	例3	例4
優先順位	4	5	4	3
従来法の計算時間	725	341	535	1012
提案法の計算時間	622	541	909	1105

（単位は秒：データの入力時間を除く）

7．あとがき

本論文では、優先順位に従った割当を行って、割当問題の実行可能解を求めるアルゴリズムを提案した。提案法は、利益あるいは費用が金額として明確に与えることができない割当問題を解くのに有効である。

提案法は、各資源の個数が一定であることを利用した近似解法であり、大規模な割当問題の数値シミュレーション実験を行った結果、提案法は効率的であることが示された。

参考文献

[1] 茨木俊秀、「組合せ最適化　分枝限定法を中心として」、産業図書、昭和58年。

[2] 今野浩、鈴木久敏編、「整数計画法と組合せ最適化」、日科技連、1982年。

[3] K. Aoki and A. Nishikori, " An Algorithm for Constrained Load Flow," IEEE Transactions on Power Apparatus and Systems, Vol. PAS-103, No.5, pp.963-973, May 1984.

[4] G. T. Ross and A. A. Zoltners, " Weighted Assignment Models and their Applications," Management Science, Vol.25, No.7, pp.683-696, July 1979.

[5] 草刈君子、宮崎知明、金指哲也、「富士通ＡＭＰからみた最近の混合整数計画法　－その性能と適用例－」、第二回ＲＡＭＰシンポジウム論文集、pp.11-20, 社団法人日本オペレーションズ・リサーチ学会特殊研究部会数理計画法研究会、1990年。

[6] G. Schmidt, " Timetabling Construction - an Annotated Bibliography,"The Computer Journal, Vol.23, No.4, pp.307-316, 1980.

[7] 藤野喜一、「乱数を使用した学校時間割作成プログラム」、情報処理、Vol.5, No.2, pp.68-76, 1964年。

[8] 吉田茂、「経営シミュレーション」、オーム社、1988年。