資源配分問題

—計算の複雑さの立場から—

茨木　俊秀

1. まえがき

システムのモデル化に際し，その最適化を前提にする場合が多い．この目的に，数理計画問題へのモデル化はきわめて自然であり，標準的なアプローチの1つである．しかし，数理計画問題への記述で話が終了するわけではなく，その最適解を計算するというプロセスを忘れてはならない．ある問題は，きわめて効率よく解けるかも知れないが，他の問題はそうではないだろう．したがって，場合によっては，実用的な計算量で最適解が得られるように，モデルの精密さ等を犠牲にしなければならないこともある．“モデルの複雑さ”を論じるとき，そのモデルを解く“計算の複雑さ”についても十分な知識をもつことが不可欠であるといえる．

計算の複雑さの理論（theory of complexity）は，ここ10年ほどの間に爆発的に発展した学問領域で，効率よいアルゴリズムを開発すること，あるいはその逆に，コンピュータを用いても実用的な計算時間では解けないような難しい問題であることを理論的に明らかにすること，を目的としている．その成果については，各所で報告されており（たとえば［22］や成書［1, 10, 12］），筆者自身も少し述べたことがある［13］．いささか屋上屋を架すようでもあるが，ここでは，その具体例ということで，きわめて単純な整数計画問題である資源配分問題をとりあげ，目標関数の複雑さに応じて，最適解を求める計算の複雑さがどのように変化するか眺めてみたい．その結果，資源配分問題へのモデル化に際して，計算の複雑さが本質的な役割を果していることを理解していただけるのではないかと思う．

さて，ここでいう資源配分問題とは，Koopmans［19］に始まる．

$$\text{目標関数}\quad f(x_1, x_2, \cdots, x_n) \longrightarrow \text{最小}$$

$$\text{拘束条件}\quad \sum_{j=1}^{n} x_j = N \qquad (1)$$

$$x_j：\text{非負整数},\ j=1, 2, \cdots, n.$$

なる整数計画問題である．きわめて簡単な拘束条件を1個だけもつのが特徴であって，要はN個の離散的な資源を，n種の活動にどのように配分すれば最適か？　という問題である．資源として具体的に種々のものを考えることができ，マンパワープラニング，投資計画，生産計画，最適観測計画，最適軍備計画等々の簡単な場合を含む．また，後で少し詳しく触れるが，議員定数の最適配分問題もこのタイプである．

式(1)では，目標関数fの形を特に指定していないが，実際には，その応用に応じて，特別な形をとるのが普通である．一般性のある関数形を仮定すればするほど，広範囲の応用があるが，最適解の計算はそれだけ難しくなる．以下ではfの形に

いばらき　としひで　京都大学　工学部

何の仮定もおかない場合から始め，分離形，分離形かつ凸の場合と次第に特殊な関数を考察し，最適解を計算するアルゴリズムの効率が向上していく様子を具体的に示そう．

このように簡単な拘束条件をもつ問題でも，難しい問題から，非常に効率よく解けるものまで，複雑さの階層のあざやかな断面を見せてくれるのは，計算の複雑さの理論を身近かに感じさせてくれる意味で，興味深いのではなかろうか．

2. 一般の場合

目標関数fに何の仮定もおかない場合，資源配分問題(1)は非常に難しい問題である．もちろん，拘束条件を満たすすべてのn次元整数ベクトルxを列挙し，その中で$f(x)$を最小にするものを見出せば最適解を計算できるが，この方法はnとNが少し大きくなれば，もはや実用的ではない．以下の議論は，本質的に，このような列挙法以外のアルゴリズムが存在しないことを示唆するものと解釈してもよい．

さて，難しいことを示すために，ここではいささかずるい論法を用いる．すなわち，すでに難しいことが証明されている問題Aをもってきて，それ以上に難しいことを証明するのである．問題Aの役割を，ここでは集合分割問題 (set partitioning problem) に果してもらう．これは，m要素からなる集合Sの部分集合$S_1, S_2, \cdots, S_n$が与えられているとき，この中から適当なN個$S_{j1}, S_{j2}, \cdots, S_{jN}$を選んで，

$$S_{ik} \neq S_{jl}, \quad k \neq l$$
$$\bigcup_{k=1}^{N} S_{jk} = S$$

とできるかどうか判定せよという問題である．Sの第i要素がS_jに属すとき$a_{ij}=1$，属さないとき$a_{ij}=0$と定めれば，以下のように整数計画問題にも書ける．

$$\sum_{j=1}^{n} x_j = N$$
$$\sum_{j=1}^{n} a_{ij} x_j = 1, \quad i=1,2,\cdots,m \qquad (2)$$
$$x_j : \text{非負整数}, \quad j=1,2,\cdots,n.$$

この問題に対応して，以下の資源配分問題を定義しよう．

$$\text{目標関数} \quad f(x) = \sum_{i=1}^{m} \left(\left(\sum_{j=1}^{n} a_{ij} x_j \right) - 1 \right)^2 \longrightarrow \text{最小}$$
$$\text{拘束条件} \quad \sum_{j=1}^{n} x_j = N \qquad (3)$$
$$x_j : \text{非負整数}, \quad j=1,2,\cdots,n.$$

容易にわかるように，問題(3)の最適値は，式(2)が解をもつときかつその時に限り0になる．したがって，資源配分問題を解くアルゴリズムがあれば，それを用いて問題(3)を解き，その結果式(2)の解の存在を知り，集合分割問題を解くことができる．式(2)から式(3)への変換に要する計算はほとんど無視できるものであるから，以上の結果は，集合分割問題の複雑さが資源配分問題以下であることを示している．

ところで，上の論法が有効であるためには，少なくとも最初の問題に対して別の議論でその複雑さを示さねばならない．この目的に大きな役割を果したのが非決定性計算を道具にしたNP完全性 (NP-completeness) の概念である．ここでは詳しい説明は略すが（たとえば[1, 10]参照），現在，数千にものぼるNP完全問題が発見されている．どのNP完全問題に対しても，現在のところ列挙法に類する効率の悪い方法以外のアルゴリズムは知られていない．NP完全問題の1つでも効率よく解ければ（厳密には，入力データ長の多項式オーダー時間で解ければということ），すべてのNP完全問題を効率よく解けるという性質があるので，これはNP完全問題がすべて難しく，効率よいアルゴリズムをもたないことの傍証であると考えられている．

NP完全性が最初にわかった記念すべき問題は，命題論理式の充足性の判定問題である．その後何段かの帰着を経て，集合分割問題のNP完全性もわかっているので，上の式(2)(3)に関する議論は資源配分問題のNP完全性を言ったことに

相当する．すなわち，資源配分問題は，ＮＰ完全問題という，数学的にきちんと定義されたクラスの一員であり，簡単には解けない（だろう）というお墨付をもらったわけである．

式(３)の目標関数は２次式であり，しかも凸関数である(凸関数の定義は後述)．x_j が実数変数であれば，このような非線形計画問題は比較的処理しやすいものであるが，ＮＰ完全性の結果は，変数の整数性のためにそのような特性を十分生かすことができないことを示している．こんなに簡単にみえる問題に対し，効率よい解法が存在しないのがかえって不思議な気がするぐらいである．

目標関数 f において，変数相互間の結合があまり密でない場合，いわゆる非直列動的計画法(non-serial dynamic programming) という手法がある[４]．これは，変数間の独立性を利用して，列挙する部分をできるだけ排除しようとするものである．特に，n 変数が全部独立で，いわゆる分離形の目標関数になる場合，通常の動的計画法と等しく，計算効率も高くなる．そこで，次節では，この特別な場合を調べてみよう．

3. 分離形目標関数の場合

f が１変数関数の和に分離される場合，すなわち，

$$\text{目標関数}\quad f(x)=\sum_{j=1}^{n} f_j(x_j)\longrightarrow\text{最小}$$

$$\text{拘束条件}\quad \sum_{j=1}^{n} x_j=N \qquad (4)$$

$$x_j：\text{非負整数},\ j=1,2,\cdots,n.$$

を考える．この問題は，動的計画法の標準的な適用例としていろいろな教科書([3, 5, 21]その他)によくとりあげられているものである．動的計画法の適用法はいくつか考えられる．ここではその１つを紹介しよう．

まず，次の関数を定義する．

$$f^{(k)}(y)=\min\{\sum_{j=1}^{k} f_j(x_j)\,|\,\sum_{j=1}^{k} x_j=y,\ x_j：\text{非負整数}\} \qquad (5)$$

$$k=1,2,\cdots,n,\ y=0,1,\cdots N.$$

すなわち，$f^{(k)}(y)$ は変数を x_1 から x_k，資源量を y に限ったときの最適値であり，$f^{(n)}(N)$ を最終的に求めたい．

まず，$k=1$ の場合，次式の成立は定義より明らかであろう．

$$f^{(1)}(y)=f_1(y),\ y=0,1,\cdots,N. \qquad (6)$$

ここから始め，次の漸化式を用いて，一般の $f^{(k)}(y)$ を計算できる．

$$f^{(k)}(y)=\min\{f^{(k-1)}(y-l)+f_k(l)\,|\,l=0,1,\cdots,y\} \qquad (7)$$

$$k=2,3,\cdots,n,\ y=0,1,\cdots,N.$$

すなわち，$k=2,3,\cdots,n$ の順に，それぞれについて $y=0,1,\cdots,N$ に対する $f^{k}(y)$ を求めていけばよい．

式(７)は動的計画法の考え方そのものであるといってよく，いわゆる最適性の原理を書き表わしたものである．その正当性は以下のように説明できる．ベクトル $x'=(x_1',x_2',\cdots,x_n')$ が式(５)の右辺の条件を満たし，$f^{(k)}(y)$ の値を与えるとしよう．このとき，

$$\sum_{j=1}^{k-1} f_j(x_j')=f^{(k-1)}(y-x_k') \qquad (8)$$

を示せば，式(７)の正当性がいえる．まず，x' が $\sum_{j=1}^{k-1} x_j'=y-x_k'$ の条件を満たすことから $\sum_{j=1}^{k-1} f_j(x_j')\geq f^{(k-1)}(y-x_k')$ は明らかである．そこで $\sum_{j=1}^{k-1} f_j(x_j')>f^{(k-1)}(y-x_k')$ を仮定すると，$f^{(k-1)}$ の定義より，

$$\sum_{j=1}^{k-1} f_j(x_j'')=f^{(k-1)}(y-x_k')$$

$$\sum_{j=1}^{k-1} x_j''=y-x_k'$$

を満たす非負整数ベクトル $(x_1'',x_2'',\cdots,x_{k-1}'')$ が存在するはずである．このとき，ベクトル $x^*=(x_1'',x_2'',x_{k-1}'',x_k')$ は式(５)の条件を満たし，さらに，

$$\sum_{j=1}^{k} f_j(x_j^*)=f^{(k-1)}(y-x_k')+f_k(x_k')$$

$$<\sum_{j=1}^{k} f_j(x_j')=f^{(k)}(y)$$

となり，x' が定義(５)の $f^{(k)}(y)$ を実現するとい

う仮定に矛盾する．これは式(8)の成立を示すものである．

ところで，式(8)が正しいことがわかっても，x_k'の値を前もって知ることはできない．そこで，$x_k'=l$のすべての可能性を調べることにすれば，式(7)が得られるわけである．

すべての$f^{(k)}(y)$を上の手順で求めるには，kとyの各組に対し，$y+1$回の加算とy回の比較($y+1$要素の最小値を見出すため)を必要とするから，総計算時間（計算ステップ数，計算手間などと言いかえてもよい）は$O(nN^2)$になる†．これは列挙法によれば，どうしてもnかNの指数オーダーになってしまうことを考えると，大変効率的な解法であるといえる．その理由は式(7)の再帰的計算に帰され，最終的には最適性の原理の効果であるといえる．

動的計画法が適用できる資源配分問題の目標関数は，1変数関数の和にとどまらず，たとえば，

$$f(x)=\max\{f_1(x_1), f_2(x_2), f_3(x_3)\}+f_4(x_4)$$

のような関数でもよい．これらを含めた一般理論が多くの研究者によって展開されているが，ここではこれ以上立入らない．むしろ，目標関数の形を限定することで，計算効率をさらに改善できることを次節で述べたい．

4. 分離形凸関数の場合

式(4)の分離形資源配分問題において，さらに各f_jが凸関数であると仮定しよう．$f_j(x_j)$が凸であるとは，任意の$x_j^{(1)}$と$x_j^{(2)}$および$0\leq\lambda\leq1$に対し，

$$f_j(\lambda x_j^{(1)}+(1-\lambda)x_j^{(2)})\leq\lambda f_j(x_j^{(1)})+(1-\lambda)f_j(x_j^{(2)})$$

が成立することをいう．図1に示すように，勝手な2点を結ぶ線分がf_j曲線の上方に位置するこ

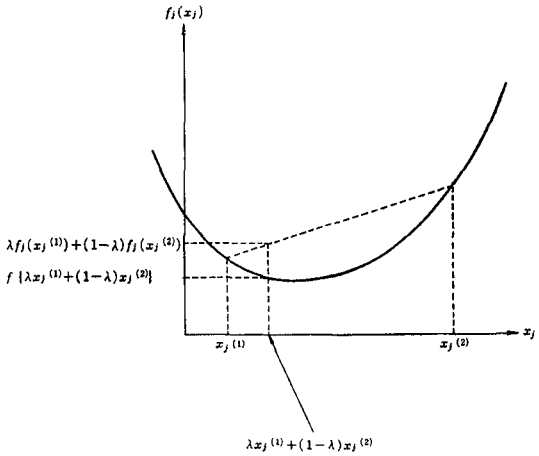

図 1 凸関数の例

と，要は，f_jが下に張り出した形をしていることである．f_jの増分を，

$$d_j(y)=f_j(y)-f_j(y-1) \qquad (9)$$

と定義するとき，凸関数のいちじるしい性質として，

$$d_j(1)\leq d_j(2)\leq\cdots\leq d_j(N) \qquad (10)$$

が成立する．この性質を利用すれば，動的計画法によらないもっと直接的な解法が可能になる．以下，Everettの提案になる一般化ラグランジュ乗数法[6]を経て，そのような解法を導いてみよう．

補題 1 資源配分問題(4)のラグランジュ緩和問題を次のように定義する．

$$\text{目標関数}\quad \sum_{j=1}^{n}f_j(x_j)-\lambda\sum_{j=1}^{n}x_j \qquad (11)$$

拘束条件　x_j：非負整数，$j=1,2,\cdots,n$.

ただし，λは実数定数（ラグランジュ乗数）である．このとき，式(11)の最適解x^*は，次の資源配分問題の最適解でもある（拘束条件の右辺が変化していることに注意）．

$$\text{目標関数}\quad \sum_{j=1}^{n}f_j(x_j)\longrightarrow\text{最小}$$

$$\text{拘束条件}\quad \sum_{j=1}^{n}x_j=\sum_{j=1}^{n}x_j^* \qquad (12)$$

x_j：非負整数，$j=1,2,\cdots,n$.

証明　まず，x^*は明らかに式(12)の許容解である．さらに，問題(12)の任意の許容解x'に対し，

† 一般に$O(g(n,N))$はオーダー$g(n,N)$と読み，$g(n,N)$の定数倍$cg(n,N)$で上から押えられていることを示す．

$$\sum_{j=1}^{n} f_j(x_j^*)-\lambda\sum_{j=1}^{n} x_j^* \le \sum_{j=1}^{n} f_j(x_j')-\lambda\sum_{j=1}^{n} x_j'$$

が成立し（x^* が式(11)の最適解であるから），$\sum_{j=1}^{n} x_j^*=\sum_{j=1}^{n} x_j'$ に注意すれば，

$$\sum_{j=1}^{n} f_j(x_j^*) \le \sum_{j=1}^{n} f_j(x_j')$$

を得る．これは x^* が式(12)の最適解であることを示している．□

この補題の結果，$\sum_{j=1}^{n} x_j^*=N$ が成立するように λ を定めることができれば，ラグランジュ緩和問題を通して資源配分問題を解くことができる．資源配分問題の場合，そのような λ が比較的簡単に求まるのである．

定理 1 式(4)の分離形資源配分問題で，各 f_j が凸関数の場合を考える．D を $d_j(y)$ $(j=1,2,\cdots,n,\ y=1,2,\cdots,N)$ の小さいものから順に N 個集めた集合とする．ただし，$d_j(y-1)=d_j(y)$ の場合，$d_j(y-1)$ を優先して D に選ぶ．このとき，次式で定義される x^* は最適解である．

$$x_j^*=\begin{cases} 0, & d_j(1)\notin D \text{ の場合} \\ N, & d_j(N)\in D \text{ の場合} \\ y, & d_j(y)\in D \text{ かつ } d_j(y+1)\notin D \text{ の場合.} \end{cases}$$

証明 ラグランジュ緩和問題(11)の目標関数を $d_j(y)$ を用いて書くと，

$$\sum_{j=1}^{n}(f_j(x_j)-\lambda x_j)$$
$$=\sum_{j=1}^{n}[f_j(0)+(d_j(1)-\lambda)+\cdots+(d_j(x_j)-\lambda)]$$

となる．そこで，λ を $d_j(y)\in D$ の最大のものに等しくおくと，

$(d_j(y)-\lambda)\ge 0$，$d_j(y)\notin D$ の場合

$(d_j(y)-\lambda)\le 0$，$d_j(y)\in D$ の場合，

が成立するので，式(10)を考慮して x^* がラグランジュ緩和問題の最適解であることがわかる．さらに，この x^* は，

$$\sum_{j=1}^{n} x_j^*=|D|=N$$

を満たし（$|D|$ は集合 D の位数）資源配分問題の許容解でもある．したがって，補題1から x^* は資源配分問題の最適解である．□

定理1にしたがえば，$d_j(y)$ の作る行列

$$\begin{bmatrix} d_1(1) & d_2(1) & \cdots & d_n(1) \\ d_1(2) & d_2(2) & \cdots & d_n(2) \\ \vdots & \vdots & & \vdots \\ d_1(N) & d_2(N) & \cdots & d_n(N) \end{bmatrix} \qquad (13)$$

から最小の N 要素を選ぶことで資源配分問題を解ける．一般に nN 要素からそのような N 要素を選ぶには最低 $O(nN)$ ステップは必要である．しかし，上の行列の各列は式(10)に示したように，すでに $d_j(1)\le d_j(2)\le\cdots\le d_j(N)$ の形に整列しているので，この性質を利用すれば計算時間をさらに短縮できる可能性がある．

最も直接的な方法は，いわゆる増分法（increment method）あるいは欲張り法（greedy method）と呼ばれる方法である[11, 7]．$x=(0,0,\cdots,0)$ から始め，

$$d_{j0}(x_{j0}+1)=\min_{i\le j\le n} d_j(x_j+1) \qquad (14)$$

を満たす j_0 を見出し，x_{j0} を1増分し，得られたベクトルをふたたび x とみなすという手順を，$\sum_{j=1}^{n} x_j=N$ が成立するまでりく返すものである．$d_j(x_j+1)$，$j=1,2,\cdots,n$ をヒープ(heap，整列2分木ともいう[1])など適当なデータ構造で記憶しておけば，式(14)の j_0 の計算や，それにともなうヒープの修正が $O(\log n)$ 時間でできるので，N 回のくり返しの総時間は $O(N\log n+n)$ である（f_j の各 x_j に対する値 $f_j(x_j)$ の計算は定数時間で可能と仮定している）．後半の n は，解ベクトル x や初期ヒープの準備に要する計算時間を示している．ヒープなど効率よいデータ構造の開発は，計算の複雑さの理論の中心をなすものの1つであって，多彩な成果が得られていることはご承知の読者も多いであろう（たとえば[1，12]参照）．

例 1 資源配分問題の応用例として，議員総数 N 人を n 選挙区に割当てる問題を考えよう．選挙区 j の有権者数を p_j，定数を x_j とする．各選挙区の1票の重みを x_j/p_j で評価し，この1票の重

みをできるだけ均一化したい．x_j が整数値でなければならないという制限のため，すべての j に対し x_j/p_j を同一にすることは一般には不可能で，どうしてもばらつきを生じる．そこでばらつきの評価の1つとして，x_j/p_j の平均値

$$b=N/\sum_{j=1}^{n} p_j$$

からの誤差の2乗和をとってみよう．この結果，次の資源配分問題が得られる．

$$\text{目標関数}\quad \sum_{j=1}^{n} p_j\left(\frac{x_j}{p_j}-b\right)^2 \longrightarrow \text{最小}$$

$$\text{拘束条件}\quad \sum_{j=1}^{n} x_j=N$$

$$x_j：\text{非負整数},\ j=1,2,\cdots,n.$$

この場合，各 $f_j(x_j)=p_j\left(\frac{x_j}{p_j}-b\right)^2$ は凸関数だから，増分法が使える．簡単な計算で，

$$d_j(x_j+1)=((2x_j+1)/p_j)-2b$$

となる．式(14)にしたがって $d_j(x_j+1)$ の最小のものを見出すことは，

$$p_j\Big/\left(x_j+\frac{1}{2}\right)$$

の最大の j を見出すことと言いかえてもよい．この解法は，実は，議員定数問題に対し，Huntington 法の一種 Webster 法[2]として古くから(1910年の Sainte-Lagüe[23]の結果が最初か？)用いられているものである．□

増分法は効率のよい計算法ではあるが，行列(13)を他の観点から眺めると，場合によってはもっと効率のよいアルゴリズムを構成できる．その1つを以下に紹介してみよう．

実数 λ および $j=1,2,\cdots,n$ に対し，

$p_j(\lambda)$：$d_j(y)<\lambda$ を満たす最大の y

$q_j(\lambda)$：$d_j(y)\leq\lambda$ を満たす最大の y

$$p(\lambda)=\sum_{j=1}^{n} p_j(\lambda)$$

$$q(\lambda)=\sum_{j=1}^{n} q_j(\lambda)$$

と定めよう．容易にわかるように，定理1の D の計算は，

$$p(\lambda)<N\leq q(\lambda) \tag{15}$$

を満たす $\lambda=d_j(y)$ を発見することと言いかえてもよい．

式(10)の性質を利用すると，与えられた λ に対する $p_j(\lambda)$ の計算を，2分探索 (binary search) を用いて $O(\log N)$ 時間で行なえる．すなわち，まず，$d_j(1),d_j(2),\cdots,d_j(N)$ の中点 $d_j(\lfloor N/2\rfloor)$ を選ぶ†．$d_j(\lfloor N/2\rfloor)\geq\lambda$ ならば $d_j(y)<\lambda$ を満たす最大の $y(=p_j(\lambda))$ は区間 $[1,\ \lfloor N/2\rfloor-1]$ 内に，また $d_j(\lfloor N/2\rfloor)<\lambda$ ならば区間 $[\lfloor N/2\rfloor,\ N]$ 内にある．そこで，次はその区間の中点を調べ，同様の基準で，くり返すごとに区間長を2分してゆくのである．これを $\lceil\log N\rceil$ 回くり返すと，区間長は1になり正しい $p_j(\lambda)$ が求まるわけである．$q_j(\lambda)$ の計算も同様にできるから，結局，上の手順を $j=1,2,\cdots,n$ に適用して，$O(n\log N)$ 時間で $p(\lambda)$ および $q(\lambda)$ を計算できる．

同様な考察は，λ の探索にも適用できる．すなわち，ある j に対し，式(15)を満たす $\lambda=d_j(y)$ の存在を調べるとき，まず，中点 $\lambda=d_j(\lfloor N/2\rfloor)$ をテストする．$p(\lambda)\geq N$ ならば，式(15)を満たす λ は(あるとすれば)区間 $[1,\ \lfloor N/2\rfloor-1]$ に，$q(\lambda)<N$ ならば(あるとすれば)区間 $[\lfloor N/2\rfloor+1,\ N]$ に存在する．この2分割をくり返せば，やはり $O(\log N)$ 回の試行で，$d_j(1),d_j(2),\cdots,d_j(N)$ の中に目ざす λ が存在するかどうかを判定できる．この手順を $j=1,2,\cdots,n$ のそれぞれに適用すれば，$O(n\log N)$ 回以内の試行で，必ず式(15)を満たす $\lambda=d_j(y)$ を発見できるわけである．

以上の手順の総時間は，各 $\lambda=d_j(y)$ ごとに条件(15)を調べるのに $O(n\log N)$ 時間かかり，最大 $O(n\log N)$ 個の $\lambda=d_j(y)$ をテストすることを考えると，$O(n^2(\log N)^2)$ である．増分法の $O(N\log n+n)$ と比べると，N が n にくらべ大きい場合に有利である．またこれは，問題の入力データ長 $O(n+\log N)$ (整数 N を入力するのに $O(\log N)$ ビット必要，各 f_j の入力には定数長を仮定)の多項式オーダー時間で解けることを示している

† $\lfloor\cdot\rfloor$ は整数部分を示す記号．また，$\lceil\cdot\rceil$ は内容より小さくない最小の整数を示す．

意味でも興味深い．計算の複雑さの理論の中心課題の1つは，それぞれの問題が多項式オーダーで解けるかどうかを判定することにあり，本節のタイプの資源配分問題に対し，その答を与えているからである．

上の2分探索を用いるアルゴリズムは，3年ほど前われわれのグループで考案したものであるが(文献[16])，実は同時期によそでも同様の研究が進行していたことを後で知った(文献[9，8]など)．現在知られている最も高速のアルゴリズムは，Fredeckrickson と Johnson になるもので，

$O(n(1+\log N/n))$，$n\leq N$ の場合

$O(n)$，$n\geq N$ の場合　　(16)

にまで改良されている．さらに，オーダーの意味で，これ以上早いアルゴリズムを作ることはできないこともわかっている．スピード競走の熾烈さは，まさにあきれるばかりである．

5. その他の目標関数

現実に遭遇する目標関数は，以上の他にもいろいろあろう．たとえば，先の例1で述べた議員定数問題でも，公平さの基準を，1票の重み最大区と最小区に注目して，

$$\max_j x_j/p_j \longrightarrow \text{最小} \qquad (17)$$

$$\min_j x_j/p_j \longrightarrow \text{最大} \qquad (18)$$

$$(\max_j x_j/p_j - \min_j x_j/p_j) \longrightarrow \text{最小} \qquad (19)$$

$$(\max_j x_j/p_j)/(\min_j x_j/p_j) \longrightarrow \text{最小} \qquad (20)$$

などとすることができよう．第1はいわゆる minimax タイプ，第2は maximin タイプ，第3は1票の重みの最大差の最小化，第4は1票の重みの最大比の最小化である．

第1の minimax タイプについては，E を $f_j(y)$ $(j=1,2,\cdots,n,\ y=1,2,\cdots,N)$ の最小の N 個の集合 ($f_j(y-1)=f_j(y)$ ならば $f_j(y-1)$ に優先権がある) とするとき，

$$x_j^*=\begin{cases}0, & f_j(1)\notin E \text{ の場合}\\ N, & f_j(N)\in E \text{ の場合}\\ y, & f_j(y)\in E \text{ かつ } f_j(y+1)\notin E \text{ の場合}\end{cases} \qquad (21)$$

で定まる x^* が最適解であることを示せる．したがって，定理1の結果と比較すれば，$d_j(y)$ を $f_j(y)$ に置きかえることで，そのまま拡張できることがわかる．特に，式(16)の計算量は，式(17)の目標関数についても正しい．

maximin タイプについても同様の考察が可能であり，やはり式(16)が成立する．

式(19)(20)の両目標関数も，類似の取り扱いが可能であるが，やや複雑になり，現在知られているアルゴリズム[18]は $O(n^2)$ の手間を要する．文献[17]では，もう少し一般的な形の目標関数について，$O(n\log N+N^2)$ のアルゴリズムが提案されている．

最後に，分数型目標関数

$$\sum_{j=1}^{n} f_j(x_j)\Big/\sum_{j=1}^{n} g_j(x_j) \qquad (22)$$

もよく見かけるものである．この場合には，一般的な分数型最適化問題に提案されている Migid-do の方法[20]と動的計画法を利用することで，$O(nN^3+n^2N^2\log N)$ 時間のアルゴリズムが可能である[15]．さらに，各 f_j が凸関数で，各 g_j が凹関数ならば，$O(n^2(1+\log N/n)^2)$ にまで短縮できる．

これら各種目標関数の場合についての，もう少し詳しい議論が[14]に与えられている．

6. むすび

非常に簡単な拘束条件を特長とする1つの整数計画問題である資源配分問題について述べたが，目標関数の形に応じて，難しいものから簡単に解けるものまで広いスペクトルを与えることをご理解いただけたと思う．同様の性質は拘束条件についても成立つ．ちなみに，拘束条件を少し一般化し，

$$\sum_{j=1}^{n} a_j x_j = N$$

x_j：非負整数，$j=1,2,\cdots,n$

とすると，目標関数が1次関数であるような簡単

な場合（ナップザック問題とよばれている）でもNP完全になり，効率よく解くことは難しくなってしまう．

このような複雑さの分析は，広範囲な組合せ最適化問題に対して行なわれており，資源配分問題はそのほんの一例にすぎない．最近刊行されたGarey と Johnsonによる[10]には数千個の問題についての分類結果が収められている．これらの成果を頭に入れておけば，数理計画問題，特に組合せ最適化問題へのモデル化に際し，その計算の複雑さを含めた議論が可能になり，実用上有用である．小稿がこのための一助になれば幸いである．

最後に日頃ご指導いただく京都大学三根久教授および長谷川利治教授，さらに，いくつかのタイプの資源配分問題のアルゴリズム開発に参加いただいた大阪府立成人病センター加藤直樹氏に深謝したい．

文　献

[1] A.V. Aho, J.E. Hopcroft and J.D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading Mass., 1974; 野崎昭弘，野下浩平訳，アルゴリズムの設計と解析 I，II，サイエンス社，1977.

[2] M.L. Balinski and H.P. Young, On Huntington method of apportionment, SIAM *J. Appl. Math.*, **33**, 607-618, 1977.

[3] R. E. Bellman and S. E. Dreyfus, Applied Dynamic Programming, Princeton University Press, Princeton, N.J., 1962; 小田中敏男，有水彊訳，応用ダイナミックプログラミング，日科技連，1962.

[4] U. Bertelè and F. Brioschi, Nonserial Dynamic Programming, Academic Press, New York, 1972.

[5] S.E. Dreyfus and A.W. Law, The Art and Theory of Dynamic Programming, Academic Press, New York, 1977.

[6] H. Everett, Generalized Lagrange multiplier method for solving problems of optimum allocation of resources, *Operations Research*, **11**, 399-417, 1963.

[7] B.L. Fox, Discrete optimization via marginal analysis, *Management Science*, **13**, 210-216, 1966.

[8] G.N. Frederickson and D.B. Johnson, Optimal algorithms for generating quontile information in X+Y and matrices with sorted columns, CS-79-45, Computer Science Department, The Pennsylvania State University, 1979.

[9] Z. Galil and N. Megiddo, A fast selection algorithm and the problem of optimum distribution of effort, *J. ACM*, **26**, 58-64, 1979.

[10] M.R. Garey and D.S. Johnson, Computers and Intractability : A Guide to the Theory of NP-completeness, Freeman, San Fransisco, 1979.

[11] O. Gross, A class of discrete type minimization problems, RM-1144, RAND Corp., 1956.

[12] E. Horowitz and S. Sahni, Fundamentals of Data Structures, Computer Science Press, Potomac Maryland, 1976.

[13] 茨木俊秀，整数計画はなぜ難しい？，オペレーションズ・リサーチ，**22**，352-358，1977.

[14] T. Ibaraki, Solving mathematical programming problems with fractional objective functions, NATO Advanced Study Institute, Generalized Concavity in Optimization and Economics, 1980.

[15] ——, Resource allocation with a convex objective function and its generalizations, NATO Advanced Study Institute, Generalized Concavity in Optimization and Economics, 1980.

[16] N. Katoh, T. Ibaraki and H. Mine, A polynomial time algorithm for the resource allocation problem with a convex objective function, *J. Operational Research*, **30**, 449-455, 1979.

[17] ——, —— and ——, An algorithm for the equipollent resource allocation problem, Working Paper, Dept. of Applied Mathematics and Physics, Kyoto University, 1980.

[18] ——, —— and ——, Equipollent resource allocation problem with application to optimal apportionment, Working Paper, Dept. of Applied Mathematics and Physics, Kyoto University, 1980.

[19] B. O. Koopmans, The optimum distribution of effort, *Operations Research*, **1**, 52–63, 1952.

[20] N. Megiddo, Conbinatorial optimization problem with rational objective functions, *Math. of Operations Research*, **4**, 414–424, 1979.

[21] 尾形克彦, ダイナミックプログラミング, 培風館, 1973.

[22] 大山達雄, 組合せ問題の計算上の複雑さについて, オペレーションズ・リサーチ, **24**, 427–433, 1979.

[23] Sainte-Lagüe, La représentation et la méthode des moindres carrés, *Compt. Rend. Acad. Sci.* **151**, 377–378, 1910.