数理計画手法によるディスクへのデータベース収容問題の近似解法

野瀬純郎[†] 清田三紀雄[††] 斎藤努[††]

情報システムの最も基本的な構成要素の1つにデータベースがある．データの実体はファイルとして，通常，磁気ディスク記憶装置（DK）に収容されるが，どのファイルをどのDKに収容するか，その計画の良否はシステムの性能に大きく影響する．このため，ファイル収容計画は大規模データベースシステムにおいて，システム設計上の重要な課題となっている．それにもかかわらず，現状では経験に基づき，人手作業により対処しているにすぎず，組合せ最適化問題として取り組まれ，実用化されている例は発表されていない．性能からみたファイル収容の最適化とは，DKごとの使用率をできるだけ小さく，かつ平準化することといえる．一方，設備コストからみたファイル収容の最適化とは，できるだけ少ないDKですべてのファイルを収容することといえる．この問題は多目的最適化問題であるが，我々はDK数をパラメータとしてDKの使用率平準化問題を繰り返し解く，というアプローチをとった．DK数固定のもとでのDKの使用率平準化問題に対し近似解法を開発し，いくつかの実験でその有効性を確認した．さらに，その成果をもとに，最小のDK数を決定できるファイル最適収容計画支援ツールを実用化した．本稿では，近似解法の内容と適用実験の結果について述べるとともに，支援ツールの概要について紹介する．

Heuristic Algorithms for Optimal Database Allocation Problem by Mathematical Programming Method

JUNRO NOSE,[†] MIKIO KIYOTA[††] and TSUTOMU SAITO[††]

The database system is one of the basic components of information processing systems. The data are stored in a magnetic disk drive (DK) as a file. An excellent plan to determine which file should be stored into which DK has an effect on the system performance. Thus, the file allocation planning is very important matter in designing the system. However, in fact, the field operation on the file allocation planning is operated by hand based on some human experiences. Any practical example of treating this problem as the combinatorial optimization problem has not been announced as yet. The maximum system performance requires the minimum and uniform utilization over DKs. On the other hand, the minimum system cost requires the minimum number of DKs. This problem is essentially one of the multiple objects optimization problems. But, we took the approach that to develop the approximate algorithm to uniform the DK's utilization under the fixed number of DKs and try it several times with changing the number of DKs. We have examined the efficiency through some numerical experiments. We developed the practical file allocation planning support tool to determine the minimum number of DKs based on this examination results. In this paper, we propose the outline of our approximate algorithms, feasibility studies, and also our planning tool.

1. はじめに

情報システムの最も基本的な構成要素の1つにデータベースがある．データの実体はファイルとして，通常，磁気ディスク記憶装置（DK）に格納される．業務プログラム（AP）は，処理の途中で必要の都度ファイルにアクセス（読み出し，書き込み）するが，この際，複数のAP間で同一のDKへのアクセスが同時に発生すると待ち行列が生じ，システム性能が劣化する．そのため，性能の観点からは，DKの台数を十分用意してファイルを分散収容し，アクセス競合を避けたいが，設備コストの面から限界がある．このことから，ファイルごとのアクセス頻度と容量が与えられた場合，できるだけ少ないDKで，性能上最適なファイルの収容計画を求める問題が提起される．

情報化社会の進展とともに，情報システムはますま

† NTTソフトウェア株式会社
NTT Software Corporation
†† 株式会社構造計画研究所
Kozo Keikaku Engineering Inc.

す大規模化しており，データベースの容量が数百～数千GB（ギガバイト），APが数百～数千本にも及ぶものもあり，経験に頼って人手で最適解を求めることは困難である．また，ファイルの規模やAPの数，およびAPのファイルアクセス頻度は，中長期的な時間の経過とともに変化（一般には増加）するため，一度収容計画を決定すればそれで終わりというわけにはいかず，状況に応じてファイルの分割，統合，再配置などを実施していく必要がある．以上から分かるように前記最適収容問題に対する実用面でのニーズは非常に大きい．しかしながら，現状では経験に基づき，人手作業により対処しているにすぎず，組合せ最適化問題として取り組まれ，実用化されている例は発表されていない．

性能からみたファイル収容の最適化とは，できるだけ少ないDKで，かつ各DKの使用率を可能な限り平準化することといえる．この多目的問題の解法はまだ発表されていない．著者らは，まず厳密解法の適用を検討したが，計算量からみて実用的ではないと判断した．そこで，近似解法を開発し，いくつかの実験でその有効性を確認した．また，本解法のツール化を行った．

以下，2章で実際のファイル収容計画の紹介と本稿で対象とする問題の定義を行い，3章で新たに開発した近似解法について説明する．4章で実験の概要と実験結果について述べ，最後に，5章で実用化したツールの概要について述べる．

2. ファイル収容計画と問題の定義

データベース構築におけるファイル収容計画の段階では，各APが単位時間あたり何件実行され，1回の実行でAPが各ファイルに平均何回アクセスするかをまとめた，ファイルアクセスプロファイルと呼ばれる表（図11参照）をもとに検討が始まる．

データベースシステムでは，DKに比べて非常に高速な半導体ディスク記憶装置（SFM）が混在して用いられることがある．SFMはDKに比べて容量が一桁以上小さくかつ高価であるため台数が制限される．このため，一般には容量あたりのアクセス頻度の大きなファイルを優先してSFMに収容し，次に残りのファイルについてDKへの収容を検討する．また，性能および容量の異なる数種類のDKを混在して用いる場合があるので，収容問題はさらに複雑になる．

このプロファイルから以下のような特性値を算出したうえで，設計目標値であるDKの使用率および収容率の上限を超えない範囲で，最適なファイル収容計画の検討が行われる．

各ファイルのアクセス頻度および各DKの収容率は，それぞれ次のように計算される．

$$\begin{aligned}&(\text{ファイルのアクセス頻度})\\&\quad=\sum\{(\text{各APの実行頻度})\\&\qquad\times(\text{各APのファイルへのアクセス回数})\}\\&(\text{DKの収容率})\\&\quad=\left\{\sum(\text{DKに収容されている各ファイルの容量})\right\}/(\text{DKの容量})\end{aligned}$$

このようなファイル収容計画の枠組みの中で，前章で述べた問題は以下のように定義できる．

「それぞれ固有のアクセス頻度および容量を持つ複数のファイルと，ファイルを収容すべきDKの性能および容量が与えられたとき，DKの使用率および収容率の上限を超えない範囲で各DKの使用率をできるだけ平準化し，かつ必要なDKの数を最小とするようなファイル収容計画を求める．」

この問題は，「DKの使用率の平準化」と「DK数の最小化」という，異なる2つの目的を持つ多目的問題に分類される．このような多目的最適化問題に対して，次の2つのアプローチが考えられる．問題の定式化に先だって，記号を次のように定める．

添字

i　DKを識別する添字

j　ファイルを識別する添字

係数

m, M　DK数とDKの添え字集合

n, N　ファイル数とファイルの添え字集合

a_j　ファイルjの容量

b_{ij}　DK_iにファイルjを単独で収容したときのDK_iの使用率

これは，あらかじめ以下のように計算できる．

$$b_{ij}=(DK_i\text{の平均アクセス時間})\times(\text{ファイル}j\text{のアクセス頻度})$$

c_i　DK_iの容量

変数

$\boldsymbol{x}=(x_{ij})$　ただし，

$x_{ij}=1$　(DK_iにファイルjを収容する)

$=0$　(DK_iにファイルjを収容しない)

なお，DKの収容率は，通常，システム運用上の柔軟性の観点から決められるもので，以後4章まではDKの容量は収容率の上限を考慮した実質的な容量とする．

アプローチ 1)

DK 数をパラメータとして DK の使用率平準化問題を解き，その中から DK の使用率の上限を超さない最小の DK 数を探索することを考える．ここで，「DK の使用率の平準化」は「DK の最大使用率の最小化」あるいは「DK 使用率の分散の最小化」と見なすことができるが，両者ともに目的関数が非線形であり，厳密解が実用的な時間内で求められる一般的な解法は確立していない[1]．このアプローチをとれば，上で定義した多目的問題の解法として，次のようなパラメータ付きの DK 使用率平準化問題を解くアルゴリズムがあればよい．

目的関数

$$f(\boldsymbol{x}) = \max_{i \in M} \left(\sum_{j \in N} b_{ij} x_{ij} \right)$$

(最大値の最小化の場合)

$$= \sum_{i \in M} \left(\sum_{j \in N} b_{ij} x_{ij} - \bar{b} \right)^2$$

(分散の最小化の場合)

ただし，$\bar{b}$ は DK の使用率の平均値

$$\bar{b} = \left(\sum_{i \in M} \sum_{j \in N} b_{ij} x_{ij} \right) / m$$

これらの記号を用いて問題を以下のように定式化できる．

$$P1(m) \left| \begin{array}{lll} \text{最小化 } f(\boldsymbol{x}) & & (1) \\ \text{条件 } \displaystyle\sum_{j \in N} a_j x_{ij} \leq c_i & \forall i \in M & (2) \\ \displaystyle\sum_{i \in M} x_{ij} = 1 & \forall j \in N & (3) \\ x_{ij} \text{は 0 または 1} & \forall i \in M, \\ & \forall j \in N & (4) \end{array} \right.$$

この問題の目的関数が線形であれば，よく知られている「一般割当問題」(GAP:Generalized Assignment Problem)[2]となる．また，この問題は，変数の数が $(m \times n)$，条件の数が $(m + n)$ の整数計画問題となる．

アプローチ 2)

DK の使用率の上限をパラメータとして DK 数の最小化問題を解き，DK 数最小の解の中から DK 使用率を最小とする解を探索することが考えられる．このアプローチをとれば，上で定義した多目的問題の解法として，次のようなパラメータ付きの DK 数最小化問題を解くアルゴリズムがあればよい．このDK数最小化問題では，各 DK についてそれが必要か否かを表す変数 y_i を導入する．また，あらかじめ設定すべき DK 数 m は収容されるファイル数 n を超えることはない．

追加する係数

b　DK の使用率の上限

追加する変数

$y_i = 1$　(DK_i を用いる)

$= 0$　(DK_i は用いない)

として，問題を以下のように定式化できる．

$$P2(b) \left| \begin{array}{lll} \text{最小化 } \displaystyle\sum_{i \in M} y_i & & (5) \\ \text{条件 } \displaystyle\sum_{i \in M} a_j x_{ij} \leq c_i \cdot y_i & \forall j \in N & (6) \\ \displaystyle\sum_{i \in M} b_{ij} x_{ij} \leq b \cdot y_i & \forall j \in N & (7) \\ \displaystyle\sum_{i \in M} x_{ij} = 1 & \forall j \in N & (8) \\ x_{ij} \text{は 0 または 1} & \forall i \in M, \\ & \forall j \in N & (9) \\ y_i \text{は 0 または 1} & \forall i \in M & (10) \end{array} \right.$$

この問題から制約式 (7) を除いた問題は，2 次元箱詰め問題 (2-dimensional bin-packing Problem)[3]として知られる問題である．この問題は，(DK 数をファイル数に一致させた場合)，変数の数が $n \times (n+1)$，条件の数が $(3 \times n)$ の整数計画問題となる．

実際の問題は DK 数が数十～数百のオーダ，ファイル数が数百～数千のオーダであるので，問題 $P1(m)$ のサイズ (変数および条件の数) に比べて問題 $P2(b)$ のサイズがかなり大きくなってしまう．また，問題 $P1(m)$ のパラメータ m は限られた範囲の整数値をとるので，パラメータ探索の容易性を考え，問題 $P1(m)$ を本研究の中心問題とした．以後に用いる記号の定義は，アプローチ 1 の定義を用いる．

著者らは，問題 $P1(m)$ に類似した構造を持ち，よく知られている GAP に着目して厳密解法の適用を試みたが，報告されているラグランジュ緩和[4]や連続緩和[4]による方法を応用した効率的な解法を開発することはできなかった．[5)~11)]

3. 近似解法の適用

一般的な近似解法である OPT 法[4]の枠組みに準じて新たな近似解法を開発し，計算機実験を試みた．OPT 法は目的関数の形によらないので，DK 使用率の最大

値の最小化問題とともに，DK使用率の分散の最小化問題（以下，分散最小と呼ぶ）も同時に開発した．また，実行可能解を求めるLPT法（Largest Processing Time）[12]も対比のために実験対象に含めた．

3.1 LPT 法

容量条件に着目しながら，実行可能解を求めるための基本的な解法である．

ステップ1) 最も容量の大きいファイルを選ぶ．

ステップ2) そのファイルを収容できるDKの中で，最も余裕のあるDKに収容する．

ステップ3) ファイルが残っていればステップ1に行く．

この方法は，計算量も $n \times m$ で済む．

3.2 2-OPT 法

近似解法は既知の実行可能解 $\bar{x} = (\bar{x}_{ij})$ が与えられたとき，この点の「近傍」を探索して局所最適解を求める方法である．整数計画法で採用される近傍の1つに次の r–近傍がある．

r–近傍　　$N_r(\bar{x}) = \{x | x_{ij} \neq \bar{x}_{ij}$ となる個数は r 個以下 $\}$

OPT法はこの近傍内で局所最適解を探索する．ここではある実行可能解について，単一のファイルの移動および2個のファイルの交換の組合せによって得られる解の集合の和集合を「2–近傍」として用いることとした（一般に r 個のファイルの組合せを探索する方法へ形式的に拡張できるが，問題が少し大きくなると必要な計算時間は指数的に増大してしまう）．この近傍の定義に基づく2-OPT法の手続きを以下のように構成した（**図1**参照）．

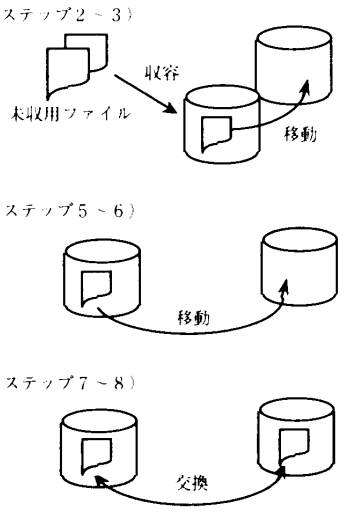

図1 近傍の生成
Fig. 1 Neighbourhood generation.

ステップ1) LPT法により実行可能解 $\bar{x}$ を得る．

ステップ2) LPT法で収容できなかったファイルがある場合，収容済みのファイルを別のDKに移すことによって，未収容のファイルをそのDKに収容することを試みる．

ステップ3) ステップ2を未収容のファイルがなくなるか，可能な交換がなくなるまで繰り返す．

ステップ4) 未収容のファイルが残っていれば，「解なし」として停止する．

ステップ5) 別のDKに移すことで目的関数が小さくなるようなファイルがあれば，そのDKへ移動させる．

ステップ6) ステップ5を可能な移動がなくなるまで繰り返す．

ステップ7) 異なるDKに収容された2つのファイルについて，それを交換することによって目的関数が小さくなれば，交換を行う（最大値最小の場合は使用率が最大となっているDKに収容されているファイルに限定する）．

ステップ8) ステップ5～7を可能な移動がなくなるまで繰り返す．

ステップ9) 最終的に得られた解を局所最適解とする．

3.3 改善 2-OPT 法

2-OPT法は2–近傍の局所最適解を見いだすことによって，次々に可能解を更新していき，2–近傍中で目的関数の改善がなくなった時点で探索を終了する．このような局所探索法では，つねに局所的な最適解に陥ってしまう可能性が存在する．計算時間が十分短かければ，初期解を複数作り出すことによって多点探索を行い，実用的な時間内でよりよい解を得られる可能性を高めることができる．

ここでは，2-OPT法で得られた最終解に対して確率的な操作を加え，一部のファイルが未収用となっている状態を作り，その解を初期解として再度2-OPT法を適用し，よりよい解の探索を行うアルゴリズムを考えた．

「全ファイルに対する未収容ファイル数の割合」（未収容比率と呼ぶことにする）と，「改善試行回数」（改善回数と呼ぶことにする）をパラメータとして，以下のような手続きを構成した（以後，改善2-OPT法と呼ぶことにする）．

ステップ1) LPT法で初期解を求める．

ステップ2) 得られた解に対して2-OPT法のステップ2～ステップ8を適用し，得られた解を暫定解とする．

ステップ3) 暫定解に対して未収容比率の割合に応じて，ファイルを確率的に選び未収容の状態にする．

ステップ4) ステップ3で生成した状態に対して2-OPT法のステップ2～ステップ8を適用し，得られた解を対比解とする．

ステップ5) 暫定解と対比解を比べて良い方を新たな暫定解とする．

ステップ6) 改善回数だけ，ステップ3～5を繰り返す．

ステップ7) 最終的に得られた解を局所最適解とする．

4. 実　　験

LPT法，2-OPT法（最大値最小および分散最小問題）および改善2-OPT法（最大値最小および分散最小問題）の各アルゴリズムに対して，アルゴリズムの精度，問題の大きさによる計算時間の変化，アルゴリズムの持つパラメータの感度を調査するために，以下のような計算機実験を実施した．さらに，実問題に対する適用実験も，あわせて行った．

比較に用いる解の精度として，次の2つの評価基準を用いた．

評価基準 1

最大値の最小化という観点からみて，得られた解のDK使用率の最大偏差（最大値と平均との差）と平均の比を，アルゴリズムによる解の精度として採用する．

精度（最大値基準）

＝DK使用率の最大偏差/DK使用率の平均

評価基準 2

分散最小という観点からみて，得られた解のDK使用率の標準偏差と平均の比（統計のフィールドでは変動係数[13]と呼ばれ，ばらつきの相対的な大きさを測る尺度として用いられる）を，アルゴリズムによる解の精度として採用する．

精度（分散基準）

＝DK使用率の標準偏差/DK使用率の平均

実験では，各DKの容量およびアクセス時間はすべて同じとしたうえで，すべてのDKの使用率が同一になる解が必ず存在するランダム問題を，必要に応じて生成して実験に用いた（付録A）．

すべてのランダム問題には，DKの使用率が同一となる解が少なくとも1つ存在するので，アルゴリズムが厳密解を見いだすことができた場合は，上の2つの精度はともに0となる．

計算機実験には，HPのワークステーション712を用いた．

4.1 実　　験 1

DK数を20，ファイル数を50とした100のランダム問題にそれぞれのアルゴリズムを適用し，各アルゴリズムについて，ある許容精度以内の解が得られる割合を調査した．**図2**に最大値基準，**図3**に分散基準の精度に関する実験結果を示す．いずれのグラフも，実線は分散最小問題，点線は最大値最小問題をそれぞれ表す．

LPT法は許容精度を9×10^{-3}に設定しても，最大値基準では5%以下，分散基準で20%以下の解しか許容精度を満たさない．また，2-OPT法，改善2-OPT法について目的関数別に見ると，いずれの基準を用い

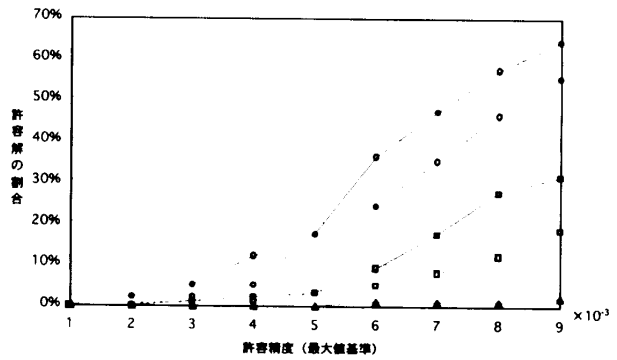

図2 各アルゴリズムによる解の精度（最大値基準）
Fig. 2 Solution accuracy of each algorithm (maximum base).

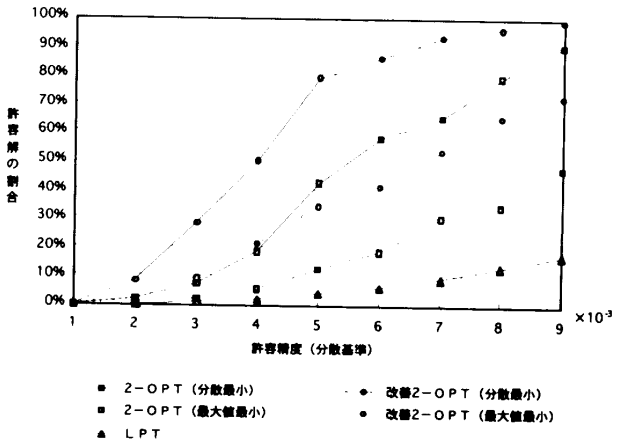

図3 各アルゴリズムによる解の精度（分散基準）
Fig. 3 Solution accuracy of each algorithm (variance base).

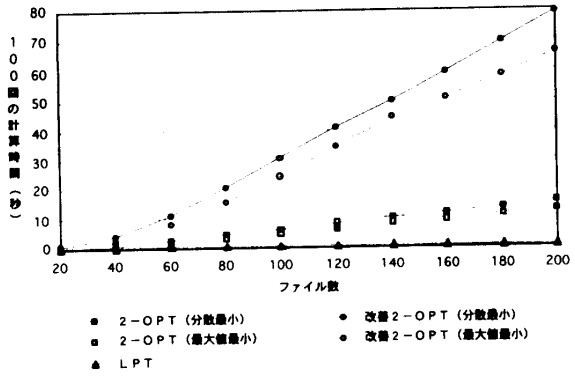

図4 ファイル数の増加にともなう計算時間の変化
Fig. 4 Relationship between the calculation time and the number of files.

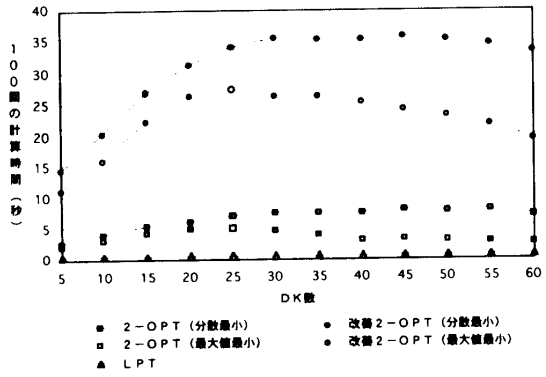

図5 DK数の増加にともなう計算時間の変化
Fig. 5 Relationship between the calculation time and the number of DK's.

ても，許容精度全域にわたって分散最小による解の方が，許容精度以内に収まっている問題が多い．総合的には分散最小の改善2-OPT法が非常に良く，精度が 9×10^{-3} 程度まで許容すれば，最大値基準で65%，分散基準でほぼ100%の問題が許容精度以内で解が求められている．

4.2 実 験 2

DK数を20に固定してファイル数を変化させる実験（**図4**）と，ファイル数を100に固定してDK数を変化させる実験（**図5**）を行い，計算時間の変化を確認した．測定した計算時間は，それぞれ100のランダム問題に対する計算時間の合計である．また，改善2-OPT法においては，未収容比率を0.5，改善回数を10回に固定した．

実験の範囲内ではLPT法が最も早く解け，DK数，ファイル数の変化にもそれほど影響されない．また，目的関数別では2-OPT法，改善2-OPT法ともに，最大値最小の方が分散最小に比べて若干早く解を見いだしている．

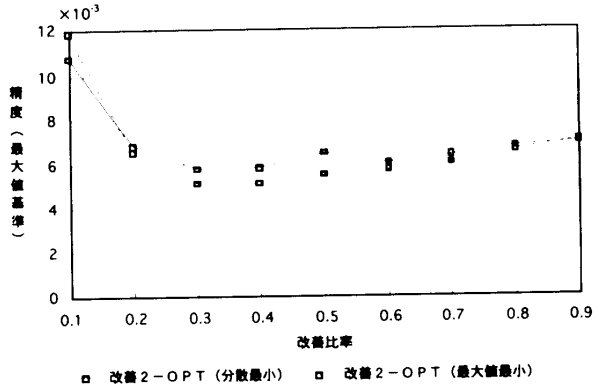

図6 改善比率による精度の比較（最大値基準）
Fig. 6 Relationship between the improvement ratio and the solution accuracy (maximum base).

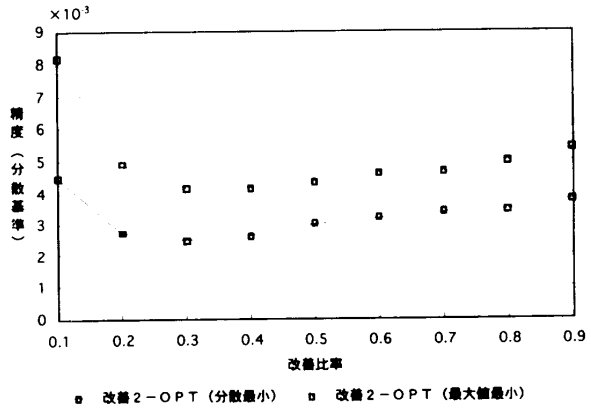

図7 改善比率による精度の比較（分散基準）
Fig. 7 Relationship between the improvement ratio and the solution accuracy (variance base).

図5において，DK数が多くなると計算時間が減少するのは，DKとファイルの可能な組合せの数が減少するためと推定される．

4.3 実 験 3

改善2-OPT法について，改善回数を100回に固定して未収容比率を変化させる実験（**図6**，**図7**）と，未収容比率を0.5に固定して改善回数を変化させる実験（**図8**，**図9**）を行い，アルゴリズム固有のパラメータの変化が解の精度に及ぼす影響を確認した．DK数を20，ファイル数を50に固定した100のランダム問題について適用した．

いずれの基準による精度でも，未収容比率は0．5程度，改善回数は10程度で十分効果が得られることが分かった．また，改善回数が100以上となると，最大値基準で最大値最小問題の精度が分散最小問題の精度を上回るようになるが，その差はごくわずかである．

4.4 実 験 4

DK数が48，ファイル数が119の中規模の実問題

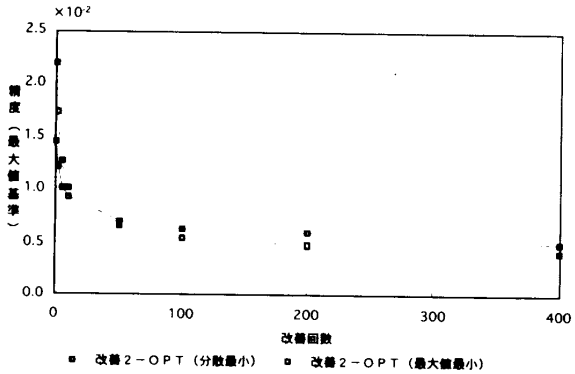

図8 改善回数による精度の比較（最大値基準）
Fig. 8 Relationship between the improvement trials and the solution accuracy (maximum base).

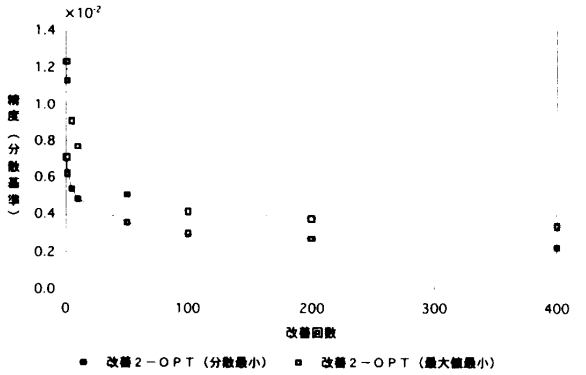

図9 改善回数による精度の比較（分散基準）
Fig. 9 Relationship between the improvement trials and the solution accuracy (variance base).

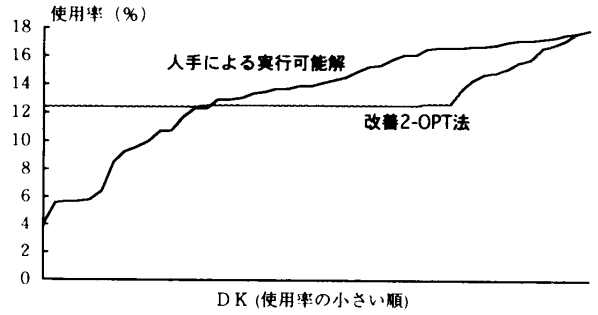

図10 DK使用率平準化の比較（改善2-OPT法と人手）
Fig. 10 Degree of utilization smoothness, the improved 2-OPT vs. human.

に対して，改善2-OPT法を適用した．未収容比率は0.5で改善回数は10回とした．計算時間は約3秒であり，実用上問題ない．人手による実行可能解との比較を**図10**に示す．改善2-OPT法が人手に比べてはるかに平準化された解を提供することが分かる．

5. ファイル最適収容計画支援ツールの概要

実験を通して良い結果が得られ，現実の問題に対しても十分対応できる見通しが得られた分散最小による改善2-OPT法をベースにして，ファイル最適収容計画支援ツール：FALLOCを開発した．以下にその概要を紹介する．

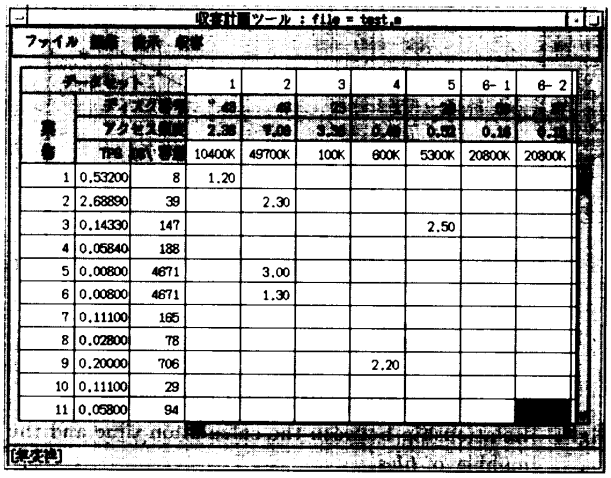

図11 ファイルアクセスプロファイル
Fig. 11 File access profile.

5.1 データの入力

入力データとして次のような情報を入力する．

① ファイルアクセスプロファイル

APごとのアクセスするファイル名とアクセス回数，APごとの実行頻度およびファイルごとの容量を入力する（**図11**参照）．

② ファイル装置構成

DKのほかにSFMも対象とする．それぞれ3種，2種の機種まで扱うことができ，その混在も可能である．容量，台数，平均アクセス時間を指定する．

5.2 計算条件の指定

計算を実行するにあたっての条件として次のような情報を指定する（**図12**参照）．

① ファイル装置使用条件

各DKの収容率の上限と使用率の上限を指定する．

② 解法

分散最小化を目的関数とする改善2-OPT法を用いる．

③ 改善回数（図12では最大計算回数）

改善回数を指定する．未収容比率は0.5に固定した．

④ 適用基準

指定されたDKの台数をパラメータとして計算する（DK使用率平準化問題）か，またはDKの使用率の上限を満たす限り台数を減少させる（DK台数最小かつDK使用率平準化問題）かを指定する．

5.3 計算結果の表示

与えられたデータを指定された条件のもとで計算し

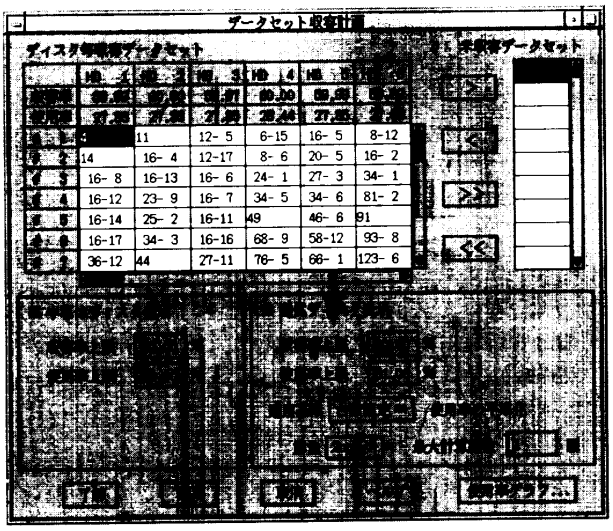

図12 収容計画シート
Fig. 12 Allocation planning sheet.

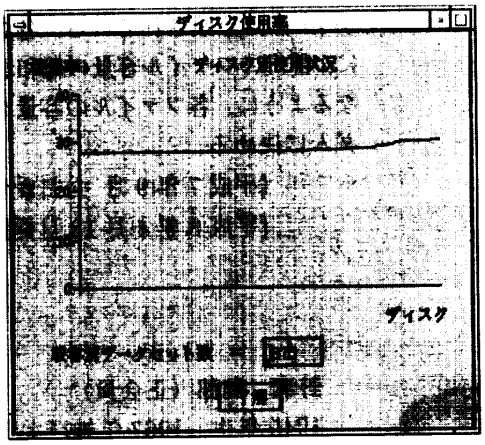

図13 DK使用率
Fig. 13 DK utilization graph of an allocation plan.

た結果を出力する.

① 最適ファイル収容計画案

どのファイルをどの装置（DKおよびSFM）に収容すればよいか，その結果各装置の収容率および使用率がどの程度になるかを表示する（図12参照）.

② DKの使用率

上記収容計画案に従った場合，DKの使用率がどの程度平準化されるかをグラフ表示する（**図13**参照）.

5.4 計算実行例

AP数170，DK数60，SFM数10，ファイル数875の実問題に対して改善2-OPT法（未収容比率0.5，改善回数10回）を適用した場合，Sun SPARCstation20では7分強で図13に示すような良い解が得られた.

6. おわりに

本論文では，情報システムの構築あるいは運用にあたり，できるだけ少ないDKで，性能上最適なファイル収容計画を求める方法について述べた．要約すると以下のとおりである.

(1) ファイル収容計画は重要かつニーズが大きい問題であるにもかかわらず，経験による人手作業に頼っているのが現状であり，ここで初めて組合せ最適化問題として提起した.

(2) 近似解法として r-近傍を使用した2-OPT法と，さらに精度を向上させる改善2-OPT法を提案した.

(3) 計算機実験により，改善2-OPT法は計算時間，精度の面で，実問題に十分適用できることを確認した．この際，目的関数は装置使用率の最大値最小より分散最小の方が優れていることが分かった.

(4) 以上の結果をふまえて，改善2-OPT法を採用した，ファイル最適収容計画支援ツール：FALLOCを実用化し，その概要を紹介した.

参考文献

1) 大山：最適化モデル分析，1章，日科技連出版社 (1994).
2) Martello, S. and Toth, P.: An Algorithm for the Generalized Assignment Problem, *Operational Research*, North-Holland, Amsterdam, pp.589–603 (1981).
3) Goffman, E.G., Garey, M.R. and Johnson, D.S.: Bin Packing with Divisible Item Sizes, *Journal of Complexity*, Vol.3, pp.406–428 (1987).
4) 今野，鈴木：整数計画法と組合せ最適化，5章，ORライブラリー7，日科技連出版社 (1982).
5) Fisher, M.L., Jaikumar, R. and Van Wassenhove, L.N.: A Multiplier Adjustment Method for the Generalized Assignment Problem, *Management Science*, Vol.32, No.9, pp.1095–1103 (1986).
6) Jornsten, K. and Nasberg, M.: A New Lagrangean Relaxation Approach to the Generalized Assignment Problem, *European Journal of Operational Research*, Vol.27, pp.313–323 (1986).
7) Ross, G.T. and Soland, R.M.: A Branch and Bound Algorithm for the Generalized Assignment Problem, *Mathematical Programming*, Vol.8, pp.91–103 (1975).
8) Guignard, M. and Rosenwein, M.B.: An Im-

proved Dual Based Algorithm for the Generalized Assignment Problem, *Operations Research*, Vol.37, No.4, pp.658–663 (1989).
9) Barcia, P. and Jornsten, K.O.: Improved Lagrangian Decomposition: An Application to the Generalized Assignment Problem, *European Journal of Operational Research*, Vol.46, pp.84–92 (1990).
10) Guignard, M. and Rosenwein, M.B.: An Application-oriented Guide for Designing Lagrangian Dual Ascent Algorithms, *European Journal of Operational Research*, Vol.43, pp.197–205 (1989).
11) Jornsten, K.O. and Varbrand, P.: A Hybrid Algorithm for the Generalized Assignment Problem, *Optimization*, Vol.22, pp.272–282 (1991).
12) Graham, R.L.: Bounds on Multiprocessing Timing Anomalies, 17, *SIAM Journal of Appl. Math.* (1969).
13) 久米：統計解析への出発，シリーズ入門統計的方法，岩波書店 (1993).

付　　録

A ランダム問題の生成法

各 DK の容量およびアクセス時間がすべて同じであるような問題に限定する．そのうえで，DK 数 m，ファイル数 n，DK の容量 c，DK の使用率の平均 $\bar{b}$ が与えられたとき，m 台の DK の使用率がすべて $\bar{b}$ に一致する解が少なくとも 1 つ存在する問題をランダムに生成する手順をここで述べる．

仮に全 DK の使用率が同一になるような解が少なくとも 1 つ存在する問題が得られているとする．この問題で適切な順序にファイルを並べると，各ファイルの使用率の累積値が $\bar{b}$ の整数倍になるようなファイルが m 個存在する．このアイデアをもとに，次のような手続きで問題を生成した（**図 14** 参照）．

ステップ 1) m 個のファイルを選択し，各々の累積使用率を $\bar{b}$ の整数倍に設定する．

ステップ 2) 残り $(n-m)$ 個のファイルの累積使用率を 0 から $\bar{b}m$ までの一様乱数で設定する．

ステップ 3) 同一の累積使用率を持つファイルがあれば，片方の累積使用率を再度一様乱数で決定する．

ステップ 4) 累積使用率の小さいファイルから順に並べ，各ファイルの累積使用率と直前のファイルの累積使用率との差をそのファイルの使用率とする．

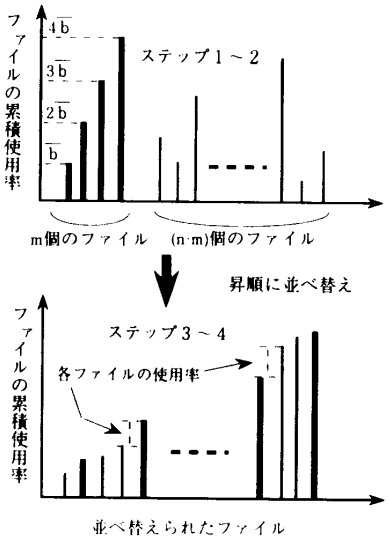

図 14　ランダム問題生成過程
Fig. 14　Process to generate a random problem.

ステップ 5) 上のステップまでで，n 個のファイルが m 群に分類されている．各群ごとに，群内のファイル容量の総和が c となるように，各ファイルの容量をランダムに決める．

(平成 7 年 9 月 4 日受付)
(平成 8 年 4 月 12 日採録)

野瀬　純郎（正会員）

1945 年生．1967 年神戸大学工学部電気工学科卒業．1969 年同大学院工学研究科電気工学専攻修士課程修了．同年，日本電信電話公社電気通信研究所入社．以来，論理回路の故障診断方式の研究実用化，大型情報処理システム，通信処理システムの開発，および，システムの性能評価に従事．現在，NTT ソフトウェア株式会社ネットワークサービス事業本部担当部長．

清田三紀雄

1954年生．1977年東京工業大学工学部経営工学科卒業．1979年同大学院総合理工学研究科システム科学専攻修了．同年，(株) 構造計画研究所入社．以来，輸送計画，原子力安全性評価，通信網信頼性評価，コンピュータ性能評価，組合せ最適化等のオペレーションズ・リサーチ関連プロジェクトに従事．現在，(株) 構造計画研究所数理技術部OR技術室室長．OR学会会員．

斎藤　努

1966年生．1988年東京工業大学理学部情報科学科卒業．1990年同大学院理工学研究科情報科学専攻修了．同年，(株) 構造計画研究所入社．以来，コンピュータ性能評価，組合せ最適化等のオペレーションズ・リサーチ関連プロジェクトに従事．現在，(株) 構造計画研究所数理技術部OR技術室勤務．OR学会会員．