最適系列分割問題に対する効率的分枝限定法の構築と諸特性解析

加地太一[†] 大内東[††]

頂点が連続的な番号を保持し，始点が初期番号，終点が最終番号となり，両端点が異なるグラフを $G(V,E)$ とする．本論文における最適系列グラフ分割問題はグラフ G に対して，各頂点に与えられた重みの総和がブロックサイズ $P(>0)$ 以下であり，かつ，部分集合の頂点番号が連続的に保持される条件のもとで，カットされる辺のコストの和が最小となるよう分割する問題である．本問題の一つの応用例としては，プログラムを一定の大きさの単位で記憶領域に割当を行うページングの手法が考えられる．本論文では動的計画法による最適系列分割問題に対して，探索法および限定操作の観点から改善の余地があるものと考え，分枝限定法の手法を導入することによって効率的算法を構成する．さらに得られた算法の数値実験にもとづいて算法の特性と性能評価を行い，理論的計算量についても論じる．以上より，漸近的計算量は等しいが，細分化禁止則，反復回数の減少，同レベルの優越関係による削除，下界値による限定などの探索空間の実際の絞り込みによって計算量の負担を軽減することが可能であることを示す．

Development and Characteristic Analysis of Branch-and-Bound Algorithm for Optimal Sequential Partitions of Graphs

Taichi Kaji[†] and Azuma Ohuchi[††]

Optimal sequential partitions of graphs is to find a minimum cost partition of the nodes of a graph into subsets of a given size, subjecting to the constraint that the sequence of the nodes may not be changed, that the nodes in the subset must be of consecutive numbers. One possible application of this problem is in partitioning computer programs into pages for operation in a paging machine. The partitioning minimizes the number of transitions between pages. This paper shows how Branch-and-Bound methods can be used to reduce storage and, possibly, computational requirements in discrete dynamic program for optimal sequential partitions of graphs. For this problem, we develop the use of Branch-and-Bound methods for dynamic programming algorithm. We experimentally and theoretically examine the effectiveness of Branch-and-Bound methods and the performance of the algorithm. Our computational experience demonstrates that the hybrid approach yields dramatic savings in both computer storage and computational requirements.

1. はじめに

頂点に重み，辺にコストを付与する頂点番号順に順位付けされたグラフを G とする．最適系列分割問題とはあらかじめ分割数を固定せず，G を分割した各部分グラフの頂点の重みがブロック・サイズ以下であり，かつ部分グラフの頂点番号が連続性を保持する条件のもとで，切断される辺のコストの和が最小となる分割を求める問題である．

この問題に対して Kernighan[1] は分割をブレーク・ポイントを用いて表現し，標準的な動的計画法 (DP) の考えにもとづいて，計算過程で必要とされる増分コストの計算時間をも含めて，全体の計算量がグラフの辺数に比例する算法を提示した．また浅野[4] は区間の集合上で定義される問題を考察し，区間木によるデータ構造によって動的計画法を効率的に実行する手法を示した．その中の一例として本問題を扱い，同じ計算量の算法を開発した．

動的計画法の最適性の原理と，分枝限定法における優越関係には大きな共通点があることが知られている[6]．分枝限定法で下界値テストは行わず，探索法として順位を適当に選んだ横型探索法を用い，優越関係

† 北海道情報大学経営情報学部情報学科
Department of Information Science, Faculty of Business Administration and Information Science, Hokkaido Information University
†† 北海道大学工学部情報工学科
Department of Information Engineering, Faculty of Engineering, Hokkaido University

による限定操作のみを用いるとき，この分枝限定法は動的計画法の手順と等価になる．したがってこの立場では動的計画法と分枝限定法との間には本質的な差はないといってよく，動的計画法の計算手順は分枝限定法の一種とみなせる．しかし動的計画法をこのように分枝限定法とみなすと，必ずしも最良の探索法とはいえない横型探索法を用いていること，および下界値等による限定操作を全く用いていないという点からさらに改善の余地がある．この視点から標準的な動的計画法の上に組み立てられた Kernighan の算法も改善可能と考えられる．

本論文ではまず最適系列分割問題の動的計画法による構成を，探索法の選択や部分解の限定操作をより柔軟に行える分枝限定法によって再構成する．その構成には Kohler[2] の論文を参考にし，本問題の特殊性にあわせて分枝限定法の構成要素である探索法，分枝規則，優越関係，下界値関数，上界値，削除規則，停止則を考察し効率のよい算法の構成を目指す．探索法として，計算終了までに分枝される部分解の個数を最小にすることが期待できる最良下界探索法を用い，限定操作として優越関係テスト，下界値テストに加え，この問題に固有な特徴を利用した細分化禁止則を取り込んだ分枝限定法による構成を行う．この際，細分化禁止則は分枝規則の中に取り込み分枝数の絞り込みを行い，生成される部分解の大幅な減少と，それにともなう増分コスト計算の減少を図っている．これを最良下界探索構成法とする．また探索法を，順位としてレベル値を用いた横型探索法に換えた構成についても述べる．この構成は下界値テストと細分化禁止則の利用をやめたとき，Kernighan の動的計画法による構成と同等なものになる．さらに，離散的動的計画法の計算量の改善に分枝限定法の考えがどのように利用できるかについて述べた Morin, Marsten[5] の論文の考えにしたがって，これら二つの規則を直接 Kernighan の算法の動的計画法による構成部分に反映させることも可能である．これをレベル順横型探索構成法とする．

さらに，以上の構成による算法の数値実験にもとづいて，予期された算法特性の実証と性能評価を行う．また算法の理論的計算量の推定についても考察する．優越テストはプログラム上では，開リストまたは閉リスト中の同レベルな部分解との比較によって行われる．このとき閉リストによる限定操作が最も有効に働くのは最良下界探索構成法の場合であり，実験によればほぼ 30% の棄却率が得られている．一方，レベル順横形探索構成法ではこの閉リストによる限定操作は全く機能しない．細分化禁止則による削除効果は探索法の選択によらず，どちらの構成に対しても等しく有効であり，約 20% から 45% の棄却率を得ている．算法の性能評価のために，数値実験から決定できる指標として，計算の複雑度を表す総分枝数，子部分解の同レベル判定による平均棄却率，下界値テストによる棄却数，U カット数および開リスト長を表として提示し，プログラム挙動の理解と分析のためのデータとした．下界値テストはレベル順横形探索構成法では有効に働きプログラムの停止を早める．これに反して最良下界探索構成法ではその効果は期待できない．最良下界探索構成法で停止を早めるには確定性の利用が有効であり，停止までの反復回数が $n+1$ の約 60 から 90% に短縮できることが実証できた．プログラムが停止するまでの総分枝回数については最良下界探索構成法の方が有利である．最後に両構成法による算法の理論的計算量について考察する．

2. 最適系列分割問題に関する記法と諸定義

連続的に番号づけられた n 個の頂点からなる集合 $V=\{1,2,\cdots,n\}$，辺集合 E からなる無向グラフを $G(V,E)$ とする．便宜上，人為的に番号 $n+1$ の頂点を加え，$c_{n,n+1}=0$ とする．各無向辺を頂点対 $\{i,j\}$ で表すとき，辺 $\{i,j\}$ には非負のコスト c_{ij} を付与する．さらに各頂点は $W=\{w_1,w_2,\cdots,w_n\}$ の重みをもつ．ただし，$0<w_i\leqq P$, $1\leqq i\leqq n$ である．ここで P は**ブロックサイズ**と呼ばれる数である．本論文ではすべての重み w_i と P は正の整数とする．

次の二つの制約(Ⅰ)，(Ⅱ)のもとで G を k 個の部分グラフ $G_i(V_i,E_i)$, $i=1,2,\cdots,k$ に分割することを**系列分割**と呼ぶ．ただし k はあらかじめ与えていない数である．

(Ⅰ)　G_i の頂点の重みの総和 $\leqq P$

(Ⅱ)　任意の G_i の各頂点番号は連続的な番号をもつ．

ここで**最適系列分割問題**とはこの系列分割の中で，切断される辺のコスト，すなわち異なる部分グラフ中に両端点をもつ辺のコストの総和を最小にするように分割する問題である．

部分グラフ G_i の頂点集合 $V_i=[b_i,b_{i+1})=\{b_i,b_i+1,\cdots,b_{i+1}-1\}$ は**ブロック**と呼ぶ．V_i での一番小さな頂点番号 b_i を**ブレーク・ポイント**と呼ぶ．ブレーク・ポイント b_i は頂点 b_i-1 と頂点 b_i の間に切断が存

在することを意味する．集合 $\{b_1, b_2, \cdots, b_k\}$ は一意的に分割 $\{G_1, G_2, \cdots, G_k\}$ を表現する．有向グラフに対しては，(i, j) と (j, i) の辺は $i<j$ である単一辺 $\{i, j\}$ で置き換えると同時に，すべての有向辺を無向辺で置き換える．そのとき，コストは $c'_{ij}=c_{ij}+c_{ji}$ とする．

図1は連続的な頂点集合 $V=\{1, 2, \cdots, 10\}$ からなり，上部の数字のコストをもつ辺からなるグラフである．また各頂点の重みは1とする．図1上の破線で切断された分割がブロックサイズ $P=4$ での最適解を与える．このとき，各ブロックは頂点集合 $V_1=\{1, 2, 3, 4\}$，$V_2=\{5, 6\}$，$V_3=\{7, 8, 9, 10\}$ からなり，ブレーク・ポイントの集合は $\{1, 5, 7, 11\}$ となる．また最小となるコストの総和は83である．11は便宜上，加えた頂点である．

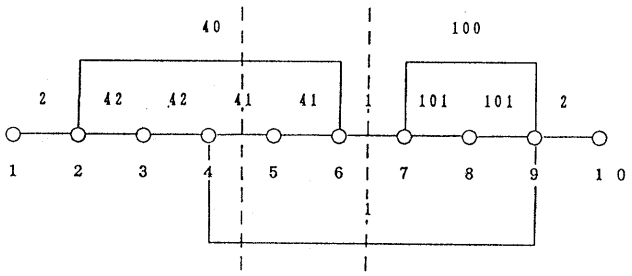

図1 系列分割グラフ
Fig. 1 Sequential Graph to be partitioned.

分割を表現するブレーク・ポイントの集合 $\{b_1, b_2, \cdots, b_k\}$，$b_1=1$，$b_k=n+1$ を与えられた問題の**完成解**，また各ブロックが系列分割の制約条件(I)，(II)を満たす完成解を**可能解**という．第 m 番目以降のブレーク・ポイントの値が不定である列 $\{b_1, b_2, \cdots, b_m, *, *, \cdots, *\}$ を**部分解**，その最後の確定要素である b_m をこの部分解の**最終ブレーク・ポイント**と呼ぶ．今後，部分解を単にその部分系列 $\pi=\{b_1, b_2, \cdots, b_m\}$，$b_m<n+1$ によって表し，最終ブレーク・ポイント b_m を **BP(π)** で表す．また BP(π) の値を部分解 π の**レベル値**といい，最終ブレーク・ポイントの等しい部分解同士を**同レベルな部分解**という．$\pi'=\{b_1, b_2, \cdots, b_m, b_{m+1}\}$ のことを π にブレーク・ポイント b_{m+1} を付加することにより，π より**分枝**して得られる部分解という．π を π' の親部分解，π' を π の子部分解という．

完成解の目的関数値および**部分解のコスト**はそれらのブレーク・ポイントにより切断されたグラフ G の辺のコストの総和である．部分解のコストは常にそれから分枝により派生されるすべての完成解の目的関数値の下界値を与えている．

任意のブレーク・ポイント x に対して次のブレーク・ポイントが y に置かれたときの**増分コスト**を

$$C(x, y)=\sum_{\substack{x\leq i<y \\ j\geq y}} c_{ij} \qquad (2.1)$$

によって定義する．上で定義した親子関係にある π と π' に対して次の関係が成立する．

$$\mathrm{Cost}(\pi')=\mathrm{Cost}(\pi)+C(b_m, b_{m+1}) \qquad (2.2)$$

また，ある時点において，分枝の対象となる部分解を**活性化**された部分解と呼び，活性化された部分解の集合を**開リスト**に格納する．同じ時点において，すでに分枝されたか，終端に達した部分解を**非活性化**された部分解と呼び，非活性化された部分解の集合を**閉リスト**に格納する．

3. 最適系列分割問題の分枝限定法による構成法

本章では標準的な動的計画法の考えのもとに組み立てられた Kernighan の算法に対して，分枝限定法の考え方を導入し改善を試みる．そのとき，最適系列分割問題を戦略の多様性と問題の特性に容易に対応できる柔軟性をもつ分枝限定法で構成する．その概要は，探索法として最良下界探索法，限定操作としては下界値テストと最適性の原理に対応する優越関係テスト，また分枝操作の中に本問題の特殊性を利用した細分化禁止則を取り込む構成であり，これを**最良下界探索構成法**という．さらに，探索法をレベル順による横型探索法に換えた構成も考える．これは動的計画法に対応するもので，**レベル順横型探索構成法**という．算法の構成にあたっては Kohler の構成法を参考に，探索法，分枝規則，優越関係，下界値関数，上界値，削除規則，停止則の各要素ごとに考察を進める．

なお，分枝限定法の一般的構成は**図2**に示しておく．

3.1 最良下界探索構成法

(1) 探索法

あらゆる探索法の中で計算終了までに分枝される部分解の個数を最小にする特徴をもつ最良下界探索法を採用する．最良下界探索法は優先順位付き待ち行列のキー値を下界値とすることによって構成される．これによって開リスト中の下界値最小の部分解を取り出

し，分枝を行った後，その部分解自身を閉リストに移す．

ある許容部分解が同レベルの部分解の中での最小コストのものになることを部分解が**確定**されたという．探索法に最良下界探索法を用いるこの構成のもとでは開リストの先頭から取り出された部分解は常に確定される性質があることが知られている[6]．さらに閉リストに移された部分解が再び開リストに戻らず，許容部分解はコスト値の大きさの順に一つずつ確定していくことによって，解を得るまでの反復回数の減少がもたらされる．

（2） 分 枝 規 則

分枝規則により選ばれた部分解の集合を**候補者集合**という．本問題の特性を利用し，分枝による絞り込みを行い，限定操作の一部を候補者集合の縮小で実現する．

本問題に対する分枝規則は各ブロックの頂点の重みの和がP以下であることを保証する**容量制約則**と，次に述べる**細分化禁止則**を用いて構成する．

ここで $d(y)=\max\{x \mid \sum_{i=y}^{x-1} w_i \leqq P\}$ とする．与えられた単調増大な任意のブレーク・ポイントの部分列

$$b_i, b_{i+1}, \cdots, b_j \tag{3.1}$$

に対して $[b_i, b_j)$ が容量制約を満たすとき，すなわち $b_j \leqq d(b_i)$ であるとき，細分点 $b_{i+1}, \cdots, b_{j-1}$ を設けないことによって，明らかに，目的関数の値を改善できるか，また少なくとも改悪することはない．よって最適解を求める立場では $b_j \leqq d(b_i)$ を満たすブレーク・ポイントの列 (3.1) による細分化された分割を考慮する必要はない．これを細分化禁止則という．

すでに細分化禁止則を満たしている部分解 $\pi_i=\{b_1, b_2, \cdots, b_i\}$ に b_{i+1} を付加し，部分解 π_{i+1} を派生するとき，容量制約則と細分化禁止則の性質を満たすように $b_{i+1}=\mathrm{BP}(\pi_{i+1})$ を決めるには次の不等式を分枝規則として採用すればよい．

$$\mathrm{BP}(\pi_{i+1}) \leqq d(\mathrm{BP}(\pi_i))$$

かつ

$$\mathrm{BP}(\pi_{i+1}) > d(\mathrm{BP}(\pi_{i-1})) \tag{3.2}$$

ここで，π_{i-1} は π_i の親部分解とする．

```
(1)   procedure Branch and Bound
(2)   begin
(3)     π_cur := 初期部分解;   U := Maxval;    { U は上界値 }
(4)     repeat
(5)       while ( "π_cur より分枝規則に従って次の子部分解π_son を生成する" )
(6)         begin
(7)           "π_son の下界値を次の式で求める．"
                lbd(π_son) = lbd(π_cur) + C(BP(π_cur),BP(π_son));
(8)           if ( "π_son と同レベルな部分解π_mk が開リストあるいは閉リスト
                      に含まれている" )
(9)             if ( lbd(π_mk) > lbd(π_son) )
(10)              if ( "π_mk が開リストに含まれている" )
(11)                "開リスト中のπ_mk を π_son で置き換える．";
(12)              else
(13)                "π_mk を閉リストから削除し、かわりに π_son を開リスト
                        に挿入する．";
(14)            else
(15)              "何もしないで、π_son を捨てる．";
(16)          else
(17)            "π_son を開リストへの挿入対象とする．";
(18)          if ( lbd(π_son) > U )
(19)            "挿入対象の π_son を捨てる．";
(20)          else if ( BP(π_son) == n+1 && lbd(π_son) < U )
(21)            begin
(22)              U := lbd(π_son); π_opt := π_son;
(23)              "開リストから lbd 値がU より大なるものをすべて捨てる．";
(24)            end;
(25)          "挿入対象である π_son を開リストに入れる．";
(26)        end;
(27)      "現在のπ_cur を閉リストに入れる．";
(28)      "開リストの key 値が最小の要素を次のπ_cur とし、その要素を開リストか
            ら取り除く．";
(29)    until (開リスト = 空);
(30)    if (U <> Maxval)
(31)      "π_opt が最適解である．";
(32)    else
(33)      "解は存在しない．";
(34)  end;
```

図 2 分枝限定法のアルゴリズム
Fig. 2 An algorithm of Branch-and-Bound.

とくに，境界条件として，$\mathrm{BP}(\pi_i)$ が1のときは

$$d(\mathrm{BP}(\pi_i)) \geqq \mathrm{BP}(\pi_{i+1}) \tag{3.3}$$

とし，$d(\mathrm{BP}(\pi_i)) \geqq n+1$ ならば

$$\mathrm{BP}(\pi_{i+1}) = n+1 \tag{3.4}$$

とする．

（3） 優 越 関 係

優越関係は部分解の集合で定義された2項関係で，二つの部分解の優劣を比較する．部分解 π と π' において，それぞれを親とする可能解の集合の中でコストが最小なものを比較し，二つの部分解の優劣性を決定する．π から派生するすべての可能解の中で最小のコスト値が π' から派生される同様なコスト値より低い値を示すなら，π は π' に対して優性であり，逆に後者は前者に対して劣性であるという．劣性の π' からは最適解は分枝されないので削除可能である．これを**優越テスト**と呼ぶ．

本問題では**同レベルにおける優越関係**が成立する．同レベルにおける優越関係とは，同レベルな任意の二つの部分解を π，ρ とした場合，$\mathrm{Cost}(\pi) < \mathrm{Cost}(\rho)$ ならば，どちらの同レベルな部分解 π，ρ においても以

後派生するブレーク・ポイントのパターンは等しいので，コストの単調増加性より π は ρ に対して優性であることが成り立つ．

この同レベルにおける優越関係により各レベルでの最良な値を決定できる．これは動的計画法における最適性の原理に対応するものである．

（4） 下界値関数と上界値

与えられた部分解 π に対して π から派生するすべての完成解からなる集合を S とする．任意の部分解 π に対して**下界値決定関数** $l(\pi)$ は

$$0 \leqq l(\pi) \leqq \mathrm{Min}\{\mathrm{cost}(\pi') - \mathrm{cost}(\pi) \mid \pi' \in S\} \tag{3.5}$$

を満たすものとする．

$$\mathrm{lbd}(\pi) = \mathrm{cost}(\pi) + l(\pi) \tag{3.6}$$

は親 π から派生されたすべての部分解のコストの下界値をあたえる．このとき**下界値テスト**は上界値 U を用いて

$$\mathrm{cost}(\pi) + l(\pi) \geqq U \tag{3.7}$$

を満たす π を削除するものである[5]．上界値 U はすべての可能解の目的関数の上界であり，現在知られている最良値すなわち暫定値をとる．

なお $\mathrm{cost}(\pi)$ の計算は増分コスト関数 $C(x, y)$ を用いて，(2.2) 式により再帰的に計算できる．(2.2) 式中の増分コスト計算は，その定義式である (2.1) 式を直接用いる方法に対して，以下の Kernighan の漸化式を採用することによって，本問題のコストの全計算量はグラフの辺の数に比例する[1]．

$$\begin{aligned} C(x, y) = {} & C(x, y-1) \\ & + (c_{y-1,y} + c_{y-1,y+1} + \cdots + c_{y-1,n}) \\ & - (c_{x,y-1} + c_{x+1,y-1} + \cdots + c_{y-2,y-1}) \end{aligned} \tag{3.8}$$

（5） 削 除 規 則

削除規則は優越関係や下界値テストなどを使って，候補者の削除，現在の活性化部分解，および非活性化部分解の中から不用のものを除去する操作を開リスト，閉リストに関する操作として記述したものである．

確定性を利用すると削除規則は次のように分枝限定法の一般的構成に対して簡略化でき，それによって，オーバヘッドの減少が見込まれる．

R1：π の同レベルな部分解が開リストまたは閉リスト中に見いだされないなら，π を新規要素として開リストに挿入する．

R2：π の同レベルな部分解が閉リスト中に見いだされるなら，π は棄却する．

R3：π の同レベルな部分解が開リスト中に見いだされ，$\mathrm{lbd}(\pi)$ が開リスト中の同レベルな部分解の下界値よりも改善されているとき，π による更新を行う．改善されていないとき，π は棄却する．

（6） 停 止 則

分枝限定法では通常開リストが空になったとき，算法を停止し，この時点で上界値が初期設定から改善されていれば最適解であると判明する．この方式による算法の停止を**正規停止則**という．これに対して最良下界探索構成法では正規停止則以前に最終ブレーク・ポイントが $n+1$ の可能完成解に達したときに停止する**特殊化された停止則**を採用し，停止過程を早める．

3.2 レベル順横型探索構成法

この構成での探索法は，次の分枝を行う対象として開リスト中の最小レベル値をもつ部分解を取り出すことによって，レベル順による横型探索法を用いる．

分枝規則，優越関係，下界値関数は前記の最良下界探索構成法で使用したのと同様な構成要素を用い，細分化禁止則を初めとして，同レベルにおける優越関係および増分コストの漸化計算式を算法構成にそのまま取り込む．探索法としてはレベル順に横型探索法を構築するために，優先順位付き待ち行列のキー値を部分解のレベル値に設定する．また下界値テストを積極的に利用することによって，正規停止則のもとで解を得るまでの反復回数が $n+1$ 以下にできる．削除規則を構成するにあたって，ある対象部分解 π から分枝された任意の部分解は $\mathrm{BP}(\pi)$ より大きなレベル値をもち，閉リストにより同レベル要素が見いだされることがない性質に着目すると，以下のような規則の使用が効果的である．対象とする子部分解 ρ に対して

RL1：ρ の同レベルな部分解が開リスト中に見いだされないなら，ρ を新規要素として開リストに挿入する．

RL2：ρ の同レベルな部分解が開リスト中に見いだされ，$\mathrm{lbd}(\rho)$ が開リスト中の同レベルな部分解の下界値よりも改善されているなら，ρ による更新を行う．改善されていないなら，ρ は棄却する．

RL3：部分解が可能解に到達したとき，下界値の値が上界値 U より改善されていれば，上界値を改善した値に変更し，この上界値より大なる下界値をもつ開リスト中の部分解をすべて削除す

る．これを**Uカット**という．Uカットとは別に通常の下界値テストによる限定操作は常に行う．

すでに知られているように，横型探索法のもとで削除規則として優越関係を用いれば，最適性の原理と同じ効果が実現され，動的計画法の手順と等価になる[5),6)]．ただし上に示した下界値テストと細分化禁止則による限定の効果によって，この構成ではさらに動的計画法の計算効率を高めている．

4. 算法の特性と性能評価

本章では以上の構成によって得られた算法の数値実験にもとづいて，算法の特性と性能評価を行う．また算法の理論的計算量についても論じる．なお，数値実験では上界値の初期値は十分大きな値とする．

優越テストによる限定操作は開リストおよび閉リストの同レベルな部分解との比較によって行われている．このとき，閉リストによる効果が著しいのは最良下界探索構成法の場合であり，優越テストが効果的に働く．また細分化禁止則の効果が両探索法に対して有効に働くことを示す．さらに算法の性能を評価するためにいくつかの特性数を導入し，これらを用いた数値実験の結果から，算法の挙動と特性を調べる．最後に系列分割問題に対する分枝限定法の計算量を評価し，それが優先順位付き待ち行列に依存することを示す．

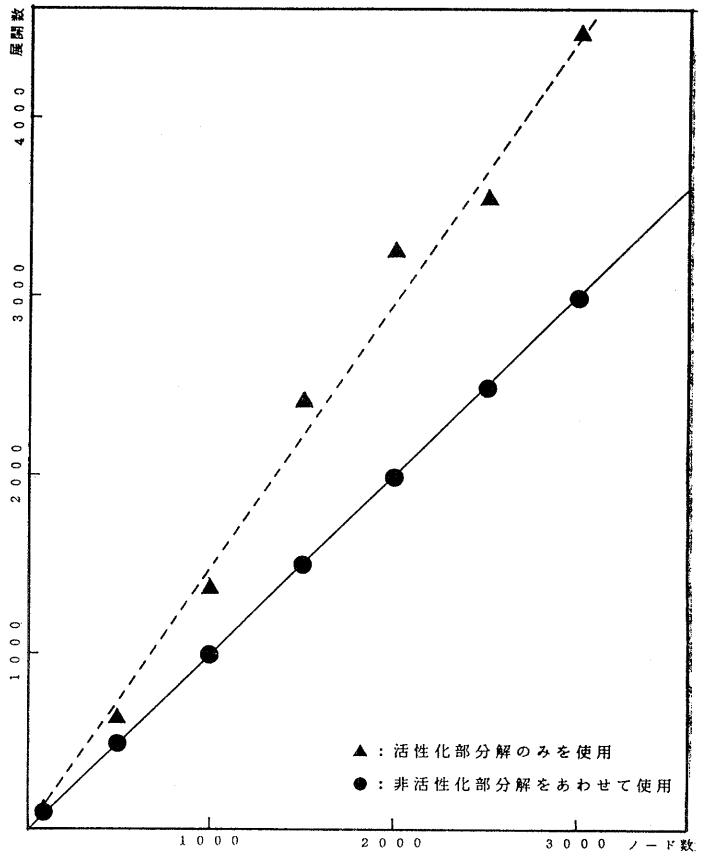

図 3 頂点数と展開数
Fig. 3 Number of nodes vs number of branching nodes.

4.1 探索法の選択による削除効果

レベル順横型探索構成法の場合，レベル値の大きさの順に解が順次決定していくため，閉リスト中に同レベル要素を見いだすことがないので，閉リストの使用が限定操作に影響を及ぼすことはない．したがって，この構成では閉リストを設定する必要はない．

それ以外の一般の構成法では閉リストを設定することによって，その情報を候補者の限定操作に有利に利用できる．その中で最もその情報を有効に利用できるのが確定性を利用する最良下界探索構成法である．その場合にも開リストのみを利用して最良下界探索構成法を実現することも可能であるが，閉リストの導入が効率の改善におよぼす効果は大きい．この効果を数値実験で確かめた結果を**図3**に示す．図3では開リストにより活性化部分解のみを使用する場合と，閉リストを導入して非活性化部分解をあわせて使用する場合を比較しており，前者に対し後者ではほぼ 1/3 の削除効果が認められる．

4.2 細分化禁止則導入による効果

開リスト中のある部分解 π を分枝して得られるすべての子部分解の集合を B，それに細分化禁止則を適用して決定される候補者集合を D とすると，D の生成過程で，$B-D$ に属する各部分解を根とするすべての部分木が探索木から削除され削除効率が改善される．この細分化禁止則による候補者の絞り込みが本問題の二つの算法構成に対しても非常に有効であることを数値実験で確かめた．それによれば細分化禁止則の適用は両探索法に対して約 20% から 45% の削除効果をも

たらし，しかもこの効果はブロックサイズの増加とともに向上することが示された．最良下界探索構成法による一例を示すと，**表1**はブロックサイズを10に固定し，頂点数を100から3,000に変化させた場合の発生した子部分解数とそのうち細分化禁止則により削除された子部分解数を示している．**表2**は頂点数100のグラフでブロックサイズを変化させた場合の同様な事項について表示している．

表1 頂点数に対しての細分化禁止則の評価（ブロックサイズ=10）
Table 1 Results of experiments for number of nodes by rule of prohibition of partitioning.

頂点数	100	500	1000	1500	2000	2500	3000
子部分解の発生数	858	4769	9750	14748	19678	24717	29596
細分化禁止則による削除数	258	1412	2644	4017	5326	6716	7830

表2 ブロックサイズに対しての細分化禁止則の評価（頂点数=100）
Table 2 Results of experiments for block size by rule of prohibition of partitioning.

ブロックサイズ	4	8	12	16	20	24	28
子部分解の発生数	385	693	1027	1333	1518	1581	1792
細分化禁止則による削除数	86	182	324	555	690	692	885

4.3 動作特性と評価

分枝限定法の枠組みの中で各算法特性を比較検討するための指標として以下の特性値を用いる．

計算の複雑度として解を得るまでの反復回数を用いる．すなわちこの値は算法の開始から開リストが空になるまでに開リストから取り出された分枝対象となる部分解の総数である．

子部分解棄却率を，算法の全過程で分枝規則により親部分解から分枝された子部分解の総数に対する開または閉リストによる削除規則（最良下界探索構成法では3.1節の規則R2，R3またレベル順横型探索構成法では3.2節のRL2）によって棄却された子部分解の総数の比率として定義する．

また下界値関数による削除効果を下界値テストによって棄却する回数とUカット数で表す．下界値テストによる棄却数は分枝された子部分解を暫定値である上界値によって棄却する回数である．Uカット数は探索木の葉に達したときに変更された上界値との比較によって開リスト中の部分解を削除する数である．これらは3.2節のRL3の規則に対応する．

これらの特性数を数値実験で計測した一例を**表3，表4**に示す．数値実験から判明した結果について以下にまとめる．なお，この数値実験では下限値決定関数はすべてのπに対して$l(\pi)=0$としている．

レベル順横型探索構成法では開リスト長は定常的にはブロックサイズと一致し，Uカットが起きた時点から急速な減少を示す．最初のUカットが生ずるのは第$(n+1)-P$段においてであり，それ以前に完成解に達することはない．Uの更新にともない，この段以後初めて下界値テストは有効となり，最適解への到達を早める．実際，正規停止則のもとで，下界値テストとその先取りであるUカットの作用で$n+1$段より早い停止が起こることを表3が示している．

表3 レベル順横型探索構成法による数値実験結果（頂点数=100，正規停止則使用）
Table 3 Numerical results by level order breadth-first search method.

ブロックサイズ	反復回数	子部分解棄却率（%）	下界値テストによる棄却数	Uカット数
4	99	37	1	2
8	95	60	3	2
12	95	63	3	6
16	88	65	5	3
20	86	65	15	4
24	91	76	0	4

表4 最良下界探索構成法による数値実験結果（頂点数=100，特殊化された停止則使用）
Table 4 Numerical results by best bound search method.

ブロックサイズ	待ち行列長			反復回数	子部分解棄却率（%）
	min	max	mean		
4	4	9	4.65	93	49
8	8	16	9.05	83	65
12	12	29	12.92	71	69
16	16	26	16.65	69	67
20	20	32	20.59	59	69
24	24	43	24.38	58	79

一方，最良下界探索構成法では開リスト長は反復回数ごとに変化するが，その傾向を理論的に推定するのは困難である．しかし多くの数値実験によれば，ほとんどの反復回数でブロックサイズの長さをとり，それよりきわだって長いものの出現は稀であり，平均長はブロックサイズ長をやや上まわる程度であることが判明した．

算法構成からわかるように，部分解は下界値の大きさの順に一つずつ確定していく．それゆえ，最悪でも$n+1$回の反復ですべての部分解の確定が行われる．

実際には最終ブレーク・ポイントが $n+1$ である部分解はそれ以前に確定されている．したがって正規停止則を最終ブレーク・ポイントが $n+1$ である部分解に達したときに停止するように変更すれば，より効果が改善できる．これを実験的に確かめた結果，停止するまでの反復回数は $n+1$ に対して 60〜90% の値であり，$(n+1)-P$ より小さな数となる．

最良下界探索構成法は反復回数は $(n+1)-P$ より小でレベル順横型探索構成法ではこれより大となるので，前者が有利である．また最良下界探索構成法の場合，優越関係による削除効率はレベル順横型探索構成法に対してやや上回った数値を与えている．この意味で分枝限定法の構成の枠組みの中では最良下界探索構成法はレベル順横型探索構成法より優位にあると考えられる．実際，数値実験においてもこの傾向は確かめられている．両者の反復回数が頂点数以下となるのに対して Kernighan の動的計画法では $n+1$ 回の反復回数が必要である．

また下界値による限定操作は最良下界探索構成法では最終ブレーク・ポイントが $n+1$ である部分解に達したとき，すなわちはじめて完成解に達したときに停止するため働かない．しかしレベル順横型探索構成法の数値実験では $l(\pi)$ の値を 0 としたにもかかわらず，下界値テスト，U カットによって効果的な限定操作が行われている．したがって $l(\pi)$ の値を緩和問題などを利用して合理的な値とすることによって，大きな P に対しては効果が期待できる．

4.4 分枝限定法の計算量

レベル順横型探索構成法と最良下界探索構成法について，漸近的計算量を求める．ただし本問題の応用上の立場から，疎（辺数$=O(n)$）であることが保証されている場合に計算量を導く．

両探索法共通の計算量に寄与する主な要因は増分コスト計算と優先順位付き待ち行列に関する計算量である．このうち増分コストの計算量は (2.2) 式を用いることによって，計算終了までに必要な総計算量は P が一定のとき $O(|E|)$ となり[1]，疎なグラフでは $O(n)$ となる．次に，優先順位付き待ち行列に関する総計算量は解が得られるまでに，この優先順位付き待ち行列に対する基本操作を反復する回数と，操作に要する計算量との積となる．反復回数は両構成法とも高々 $n+1$ 回である．一回の反復において，優先順位付き待ち行列から要素の取り出しは一回，また挿入，削除，更新の操作は高々ブロックサイズ分行われる．優先順位付き待ち行列にヒープ構造を用いることにすれば上記の各基本操作に要する計算量は $O(\log$（行列長）$)$ である．以上をまとめると優先順位付き待ち行列に関する総計算量は $O((n+1)\times(1+$ブロックサイズ$)\log n) \fallingdotseq O(n\log n)$ となる．したがって，総計算量は $O(n\log n)$ と考えられる．

この導出は優先順位付き待ち行列長が最大 n に達することを考慮して導いているが，レベル順横型探索構成法では優先順位付き待ち行列長はブロックサイズを越えない．これより P が一定のとき計算量は $O(n\log P)=O(n)$ となる．漸近的計算量は Kernighan と同じ結果になる．

一方，最良下界探索構成法では前に述べたように行列長は平均としてややブロックサイズを上回る程度で長いものの出現頻度はきわめて少ない．この数値実験の結果から，計算時間は実用上 $O(n)$ と考えてよい．また優先順位付き待ち行列に Van Emde-Boas priority queues[3] を用いると，最小値の取り出し，要素の値の挿入，書換操作は $O(\log\log N)$ で行える．ここで N はキー値の上界である．N として下界値集合の上界を取り，現問題でコスト値が有界であるとすると N は n に比例すると考えて良いので計算量は理論上 $O(n\log\log n)$ に改善できる．

5. おわりに

本論文では動的計画法による最適系列分割問題を分枝限定法の考え方のもとで，効率的算法を提案した[7),8)]．探索法に最良下界探索法，分枝規則に細分化禁止則を導入した最良下界探索構成法と，細分化禁止則と下界値による限定操作を導入した動的計画法に相応するレベル順横型探索構成法の特性と性能について詳細な検討を行い，その有効性を確かめ，その比較検討を行った．

特に，反復回数に関しては最良下界探索構成法では $(n+1)-P$ 以下であり，レベル順横形探索構成法ではそれ以上になる．下界値による限定操作は最良下界探索構成法では不要であり，レベル順横形探索構成法では $(n+1)-P$ 段以上で初めて有効に機能する．したがってブロック長 P が短いとき，下界値決定関数 $l(\pi)$ をどのように選んでもその効果には限りがある．一方，細分化禁止則による限定操作は両構成法ともに非常に有効であるが，最良下界探索構成法に対してより良い結果が得られている．

また，計算量に関しては疎（辺数$=O(n)$）なグラフ

に対して，レベル順横形探索構成法では $O(n)$ であり，最良下界探索構成法では開リスト上のデータの統計的性質から，実用上は $O(n)$ と考えてよく，理論上は開リストの優先順位付き待ち行列の構成法に強く依存する．これらの諸点を総合的に見れば最良下界探索構成法がより優れている．

プログラムの実装上から見れば，開リストの優先順位付き待ち行列構成はレベル順横形探索構成法に対してはレベル値をインデックスとする一次元配列を用いて容易に実現できるが，最良下界探索構成法では高機能の優先順位付き待ち行列の使用を検討しなければならない．

参考文献

1) Kernighan, B. W.: Optimal Sequential Partitions of Graphs, *J. ACM*, Vol. 18, No. 1, pp. 34-40 (1971).
2) Kohler, W. H.: Characterization and Theoretical Comparison of Branch-and-Bound Algorithms for Permutation Problems, *J. ACM*, Vol. 21, No. 1, pp. 140-156 (1974).
3) van Emde Boas, P., Kaas, R. and Zijlstra, E.: Design and Implementation of an Efficient Priority Queue, *Mathematical Systems Theory*, No. 10, pp. 99-127 (1977).
4) Asano, T.: Dynamic Programming on Intervals, Technical Report 91-AL-20, Information Processing Society of Japan, pp. 1-8 (1991).
5) Morin, T. L. and Marsten, R. E.: Branch-and-Bound Strategies for Dynamic Programming, *Operations Research*, Vol. 24, No. 4, pp. 611-627 (1976).
6) 茨木：組合せ最適化—分枝限定法を中心として—，産業図書 (1983).
7) 加地，大内：分枝限定法による系列分割問題の算法構成と効率，情報処理学会研究会報告，92-AL-28, pp. 25-32 (1992).
8) 加地，大内：分枝限定法による系列グラフ分割問題の高速化と拡張，情報処理学会研究会報告，91-AL-21, pp. 1-8 (1991).

(平成4年10月9日受付)
(平成4年12月9日採録)

加地　太一（正会員）

1960年生．1988年北海道大学大学院工学研究科情報工学専攻修士課程修了．同年(株)東芝入社．1989年北海道情報大学経営情報学部情報学科助手．システム工学，情報科学の研究に従事．日本 OR 学会会員．

大内　東（正会員）

昭和20年生．昭和49年北海道大学大学院工学研究科博士課程修了．工学博士．北海道大学工学部情報工学科教授．システム情報工学，応用人工知能システム，医療システムの研究に従事．人工知能学会，電子情報通信学会，計測自動制御学会，日本OR学会，医療情報学会，病院管理学会，IEEE-SMC 各会員．