論　文

並列分枝限定法における分枝変数の選択に関する考察

大西　克実[†]　　榎原　博之[††]　　中野　秀男[†]

Effect on the Selection of Branch Variables in Parallel Branch and Bound Method

Katsumi ONISHI[†], Hiroyuki EBARA[††], and Hideo NAKANO[†]

あらまし　分枝限定法は，組合せ最適化問題の最適解を求めるために利用される解法であり，適用可能な問題の規模を拡大するために分枝限定法の並列化が考えられている．最近の PC 及びネットワーク技術の発達により従来の並列コンピュータとは異なったメタコンピューティング環境上での並列分枝限定法の研究も行われている．分枝限定法では，問題例ごとにプログラムの振舞いが変わるため有効な方策をあらかじめ決めることは困難である．本研究では，メタコンピューティング環境上で可能である問題の一部をはじめに複数の方策で解き有効な方策を見極め，以降の処理時間を短くする手法を想定し有効な方策の見極めが可能かを検討する．問題の具体例としては，巡回セールスマン問題を解く並列分枝限定法のプログラムを用意し，分枝変数の選択方法を変更し，最適解を一つ決定する評価と同じ最適値の解をすべて探索する評価の関係を計算時間・加速度の点から検討する．

キーワード　組合せ最適化問題，分枝限定法，並列処理

1. まえがき

交通機関における人員配置を考える問題や，生産工場でのスケジューリング・配送などの問題を一般化した問題として組合せ最適化問題がある．組合せ最適化問題の中でも特に難しい問題を解くために適用される解法として分枝限定法が有効な手法として知られている [4], [5]．分枝限定法はどのような組合せ最適化問題にも適用しやすいという特徴をもっているが，最適解を得るために必要な計算時間は問題の規模が大きくなるにつれて指数関数的に大きくなる傾向がある．このため，分枝限定法を並列化し問題を高速に解くことが提案されている [1], [13]．並列化の研究としては，理想的な並列計算機環境上で並列分枝限定法の動作を考察する研究や，共有メモリ型の実際の並列計算機上で実行する研究，1 台の計算機の中で仮想的に並列計算機を構成して行う研究などがあった [14], [15].

最近では，PC(パーソナルコンピュータ) の利用が進み，その上で動作する Linux に代表される PC-UNIX を利用したコンピューティング環境が比較的容易に構築できるようになっている．また，インターネット技術の発達によりネットワークを通じて複数の計算機を接続し，一つの計算機資源として利用する試み (メタコンピューティング) がよく利用されており，PVM や MPI のようなライブラリも提供されている．これらの分散環境におけるコンピューティング技法は今後も ITS での応用などますます必要になると思われる．

分枝限定法では，解こうとする問題ごとにその難易度や分枝の様子が異なり，ある特定の問題例に対して有効な方法が他の問題例では必ずしも有効ではない．そこで，メタコンピューティング環境では，比較的多くの計算機資源が利用できることを念頭に置くと，複数の方法で問題を解き始め，有効と思われる方法を推測できればそちらの方法で以降の問題を他の計算機を加えて解くようなことが考えられる．こうすることで最終的な解を平均的にはより早く得ることができる．並列分枝限定法では，分枝変数・探索・再割当てなどの組合せにより様々な方法が考えられる．本研究では，特に分枝木の構成に影響が大きい分枝変数の選択方法の違いが最適解を一つ決定する評価と同じ最適値をもつすべての解を探索する評価に与える結果を比較し，有効な選択方法を推測することが可能かどうかを調べることを目的とする．

本研究では，並列分枝限定法で解く具体的な組合せ

[†] 大阪市立大学学術総合情報センター，大阪市
Media Center, Osaka City University, 3–3–138 Sugimoto, Sumiyoshi-ku, Osaka-shi, 558-8585 Japan
[††] 関西大学工学部情報処理教室，吹田市
Faculty of Engineering, Kansai University, 3–3–35 Yamate-cho, Suita-shi Osaka, 564-8680 Japan

最適化問題として巡回セールスマン問題を取り上げる．巡回セールスマン問題 (TSP) は，組合せ最適化問題の中で，問題の記述の容易さ，応用の広さなどから特に広く研究されている問題であり [6]，最適解を求めるアルゴリズムや，近似解を求めるアルゴリズムなど様々な解法が考えられている．

まず，**2.** では，並列分枝限定法について述べる．**3.** では，本研究で対象とする巡回セールスマン問題について定義並びに分枝限定法による解法について述べ，実際に計算機実験を行った方法・環境についても述べる．**4.** では，計算機実験の結果及びそれに基づいて分枝変数の選択方法の影響について考察を行う．

2. 並列分枝限定法

2.1 分枝限定法

分枝限定法は，ほとんどすべての組合せ最適化問題に適用可能な計算手法である．分枝限定法では，直接解きがたい問題を一部の変数の値を固定することでより解きやすい部分問題に分解し，それらのすべてを解くことにより，間接的にもとの問題を解く．この部分問題に分解する操作のことを分枝操作と呼ぶ．この操作は生成された部分問題に対しても順次適用され，図1の探索木に示すように，もとの問題はより規模の小さい，すなわち簡単な問題に分解される．こうして簡単になった部分問題をすべて解くと，その結果としてもとの問題が解かれることになる．

可能な分解操作をすべて行うだけでは単純な列挙法と同じであり，大規模な問題に対して適用することはできない．これを避けるためには不必要な部分問題の生成を可能な限り抑えることが必要である．分枝限定法では，生成されたある部分問題の可能解が求まる場合（図の ⊙），あるいは問題の制約条件を満たさない場合や，最適解に対する上界値・下界値を利用するなどの方法により，その部分問題からもとの問題の最適解が求まらないことが何らかの理由によって結論できる場合（図の ●）には，それらの問題に対しては新たな分枝操作を適用しないようにする．これを限定操作と呼ぶ．分枝操作も限定操作もなされていない問題は活性部分問題（図の ◇）と呼ばれ，次に処理すべき活性部分問題を探索木の中から選び出すことを探索と呼ぶ．探索の方法としては，分枝木上の位置に着目した探索（深さ優先探索，幅優先探索），(対象とする問題が最小化問題の場合) 部分問題の下界値に着目した探索，それらを組み合わせた探索などが考えられる．

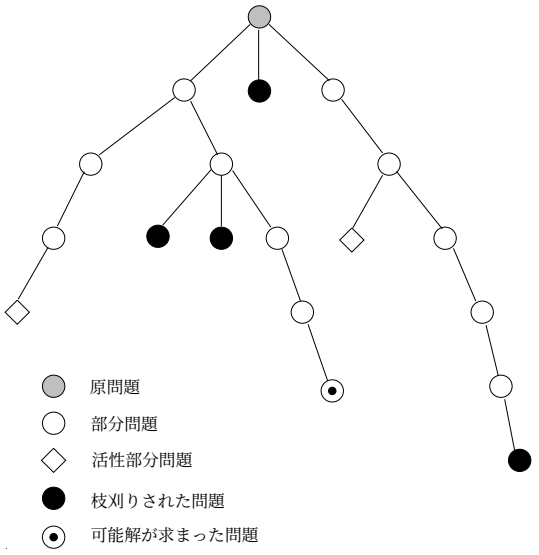

図 1　分枝限定法の探索木
Fig. 1　Search tree of branch and bound method.

分枝操作で活性部分問題を分解する時には，まず，値が固定されていない変数の中から，一つ若しくは複数個の変数を選ぶ．次に，分解する活性部分問題が表す解空間をすべて含むように整数，若しくは 0，1 に変数の値を固定した複数の部分問題を作る．このとき，値を固定する変数を選択する方法としては活性部分問題の探索方法と同様に様々な選択方法を考えることが可能であり，例えば分解した部分問題の下界値がより大きくなる変数を選択する方法などがある．変数の選択方法が異なると構成される分枝木の形が異なることになり，分枝木の一部を割り当てる並列分枝限定法では，探索に与える影響が大きいと考えられる．

2.2 仮想並列処理環境

本研究では，メタコンピューティング環境とも呼ばれるネットワーク上でワークステーションなどの計算機が複数接続されている分散処理環境を対象として並列分枝限定法のプログラムを作成する．このような分散処理環境を一つの仮想的な並列計算機とみなして並列処理プログラムの開発・実行を支援するためのツールとして PVM（Parallel Virtual Machine）と呼ばれるソフトウェアツール [10]〜[12] が提供されており，これを利用する．

PVM では，図 2に示したように，複数の異なる UNIX ワークステーションが接続された環境を仮想的に大きな一つの並列計算機とみなせるように様々な関数群を用意している．最新の PVM では，通常の

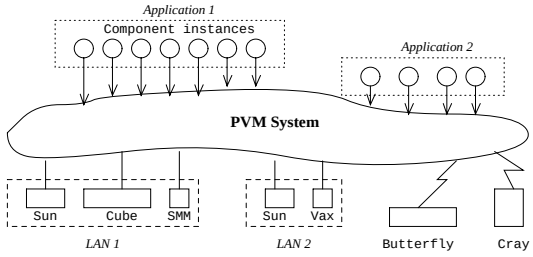

図 2　仮想並列計算機の例
Fig. 2　Virtual parallel computer.

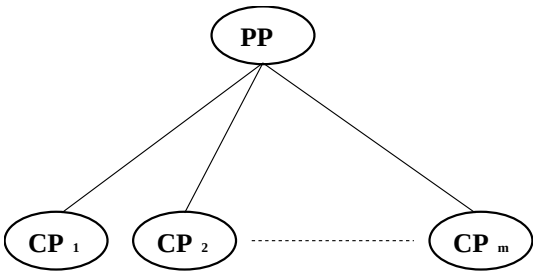

図 3　星状型マルチプロセッサシステム
Fig. 3　Star shaped multiprocessor system.

UNIX ワークステーションだけではなく，スーパコンピュータや超並列計算機をも仮想並列計算機の中に組み込んで使用することができる．

本研究では，並列処理システムとして図 3に示すような星状型マルチプロセッサシステムを，PVM ライブラリを利用し，分散処理環境上に構築する．PP は親プロセッサを示し，$\mathrm{CP}_i(i = 1, 2, \ldots, m)$ は子プロセッサを示す．共有メモリは用意せず，各プロセッサは固有のメモリだけを利用する． また，プロセッサ間のデータの伝送は PVM ライブラリに用意されている関数を呼び出汁メッセージとしてネットワークを通じて伝送する．

2.3　並列分枝限定法のアルゴリズム

2.2で述べたようなマルチプロセッサシステム上で並列分枝限定法を実現する方法として，本研究では次のような方法を採用している．

親プロセッサは与えられた問題のデータと解の管理，及び子プロセッサへの問題の割当てを行う．子プロセッサは親プロセッサから与えられた活性部分問題を初期問題とみなし，探索と分枝操作，限定操作を繰り返し行う．この方法では，親プロセッサに関しては記憶領域及び通信時間が節約でき負荷は軽くなる．しかし，子プロセッサに割り当てる部分問題は，すべてが同じ時間計算量（手間）で終了するわけではなく，その違いをあらかじめ知ることができないため，各子プロセッサで探索終了時間に差が生じる．活性部分問題がなくなった空き状態の子プロセッサをそのままにしておくことは，計算機の利用効率の点から考えても明らかによくないので，このような子プロセッサに対して，親プロセッサが再び部分問題を再割当てする必要がある．

そこで，本研究では，子プロセッサから活性部分問題がなくなったという報告が来た場合，親プロセッサに用意するリストに活性部分問題があれば，直ちにその中から問題を一つ選び再割当てを行う．割り当てるべき活性部分問題がリストにない場合，その他の子プロセッサから，部分問題を 1 問ずつ集め親プロセッサのリストに登録し，その中から問題を一つ選び再割当てを行う．

また，分枝木上で限定操作を効率良く行うためには，より最適解に近い値の暫定解を用いる必要がある．そこで本研究では，各子プロセッサで新たに暫定解が見つかると，ネットワークを通じてその値を他のプロセッサに PVM の機能により放送し，限定操作の効率を上げ，無駄な探索を行わないようにしている．

3.　計算機実験の方法と環境

3.1　巡回セールスマン問題

本研究では，組合せ最適化問題の中でも，よく取り扱われている巡回セールスマン問題を対象として計算機実験を行う．ここでは，この巡回セールスマン問題について述べる．巡回セールスマン問題としては，各節点間の枝の重みの扱いによって非対称巡回セールスマン問題と対称巡回セールスマン問題とに分けられる．非対称巡回セールスマン問題に対しては，割当て問題を緩和問題として用いた分枝限定法が有効に働くことから，50 万都市の問題まで解かれている [9]．本研究では，より困難であることが知られている対称巡回セールスマン問題を取り上げる．

対称巡回セールスマン問題とは無向グラフの最小コスト巡回路を求める問題である． n 節点 $1, 2, \ldots, n$ をもつグラフにおいて，各枝 (i, j) の重みを c_{ij} とする．ただし，$c_{ij} = c_{ji}$ である．各枝 (i, j) に変数 x_{ij} を付す（ただし一般性を失うことなく $i < j$ とする）と，以下のような式で示される．

$$\text{目的関数} \quad \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} x_{ij} \rightarrow \text{最小化}$$

制約条件 $\sum_{j=1}^{i-1} x_{ji} + \sum_{j=i+1}^{n} x_{ij} = 2\,; i = 1, 2, \ldots, n$

$x_{ij} = 0\,or\,1;$

$i = 1, 2, \ldots, n-1\,;$

$j = 2, 3, \ldots, n\,;\ i < j$

$x_{ij} = 1$ となる枝の集合は

長さ n の巡回路を作る．

3.2 分枝変数の選択

前節で述べたように巡回セールスマン問題は，グラフの最小コスト巡回路を求める問題として定義され，前節の定義からある枝を表す変数 x_{ij} を “0” 若しくは “1” にすることで部分問題に分割することができる．枝を選択する方法としては，既に述べたように下界値をより改善する枝を選ぶ方法などが考えられているが，本研究では，まず距離の短い枝は巡回路に含まれやすいと思われるので枝をコスト順に選択する方法を利用する．また，巡回セールスマン問題は，すべての都市を訪問する順序を並び換えその中からコスト最小の順列を求める問題とも考えられる．この場合，今までの訪問都市の中に含まれていない都市を次に選ぶことで部分問題を作ることになる．この次に訪れる都市への枝を選択する方法を利用する．前者の場合と比較して，枝刈りの影響を考える必要があるが，分枝の初期段階でより多くの部分問題が作られることが予想されるので，前者とは分枝木の形が異なったものになると思われる．本研究では，これら二つの分枝方法を取り上げ，分枝方法の違いが並列分枝限定法に与える影響を調べることを目的とする．

3.3 下界値と上界値

分枝限定法では，各部分問題の最適値に対する下界値と上界値の値が厳密に求められると，限定操作が効率良く行われ，分枝操作を必要とする活性部分問題の数が少なくなり，最適解を求めるために必要な時間を短くできる．このため，問題の性質を利用した精度のよい下界値・上界値を求めることが望まれる．

対称巡回セールスマン問題の下界値を求める方法として，最小 1-tree の性質を利用した方法を用いる [5], [7]. 1-tree とは，節点の数と同じ本数の枝からなり，無向グラフとして見た場合に連結しており，ある指定された一つの節点の次数が 2 であるようなグラフのことである．最小 1-tree は，次数 2 の節点 v_a を除く他の節点からなる最小木（minimum spanning tree）に，v_a に交わる枝のうちで重みの軽いものから順に選んだ 2 本の枝を加えることにより求められる．

表 1 Virtual machine の構成ホスト
Table 1 Virtual machine.

	A	B
機種	SONY QuarterL	SUN SS20
OS	NEXT STEP 3.3J	SUN OS4.1.4
メモリ	64MB	64MB
CPU	Pentium 100MHz	SuperSPARC 50MHz
台数	最大 51 台	1 台

対称巡回セールスマン問題では，各部分問題に対する上界値を求める良いアルゴリズムは提案されていない．したがって，限定操作の枝刈りに必要な暫定値は，なかなか更新されず初期値（一般に無限大）のままである．そのため，暫定解が見つかっていない分枝操作の初期の段階で下界値の大きい部分問題が多く生成され，活性部分問題の数が増大してしまう．これは分枝限定法の効率を悪化させる要因になる．

一般に，多くの組合せ最適化問題では最適解を得る必要がないならば，近似解法を用いて大規模な問題においても実用的な計算時間で近似解を求めることができる．この近似解法を用いて得られる初期問題の局所最適解を繰り返し複数個求め，それらの中で最も良い局所解を上界値として採用する方法を使用する．本研究では，近似解法として，近傍探索法の一つである B.W.Kernighan と S.Lin の提案した解法 [8] を取り上げる．この解法は α-optimal heuristic method を改良したものであり，本研究では，この解法 “KL-method” を用いて求めた近似解を上界値として用いる．

巡回セールスマン問題の例としては，TSPLIB(注 1) と呼ばれる一般に公開されている問題例の中からいくつかを選んで実行する． TSPLIB に含まれている巡回セールスマン問題の例は，この分野においてアルゴリズムの比較を行うために広く使われている．TSPLIB には，数都市程度の問題から数千都市程度の問題まで幅広い問題例が含まれているが，本研究では計算機を占有して利用できる時間の関係から，48 都市 (GR48), 51 都市 (EIL51)，70 都市 (ST70)，99 都市 (RAT99) の 4 題を取り上げ計算機実験を行った．

3.4 計算機環境

本研究において， PVM の仮想並列計算機を構成する UNIX ワークステーションとして，表 1に示すような計算機群を用いる．A の計算機の中から 1 台を選び，親プロセッサ用のプログラムを実行する．親プロセッ

(注 1)：ftp://elib.zib-berlin.de/pub/mp-testdata/tsp/

サ用を除く他の A の計算機上では子プロセッサ用のプログラムを実行する．

上界値を得るための近似解法を実行するプログラムは別の計算機 B 用として用意し，他の親プロセッサ・子プロセッサとは独立に実行する．このプロセッサで求めた近似解は，限定操作の効率を良くするための暫定解として，他のプロセッサに放送される．

子プロセッサが次に分解するべき活性部分問題を選ぶ方法には，**2. 1**で述べたように複数の方法が考えられ，最良下界値優先探索，深さ優先探索，幅優先探索及びこれらの複合型などが提案されている．本研究では，子プロセッサでの探索方法は，一般的に良い暫定解が早く見つかり，枝刈りの効率が上がるとされている最良下界値優先探索を各実験において共通の探索方法として採用する．

2. 3で述べたように，子プロセッサは，親プロセッサからの要求に応じて他の子プロセッサに再割当てされる部分問題を選ぶために子プロセッサ自身がもっている活性部分問題のリストの中を探索し，親プロセッサに部分問題を送る．ここで親プロセッサに送る部分問題を探索する方法も負荷の分散を考えるときには影響がある．本研究では，この場合の探索方法については，幅優先探索を共通の探索方法とする．幅優先探索を行う場合，親プロセッサに送られる部分問題は，他の探索法の場合と比べて，分枝木上でより根に近い位置にある問題が送られるようになる．根に近い部分問題の方が，変数の固定が進んでいないため，その部分問題の評価に要する時間が長くなることを期待できる．これにより，再割当ての回数が不必要に多くならないようにする．

4. 計算機実験の結果

計算機実験の結果を TSPLIB での問題ごとに図 5(GR48)，図 6(EIL51)，図 7(ST70)，図 8(RAT99) に示す．各グラフにおいて横軸は，子プロセッサの台数を表している．計算機を利用する時間の制限からすべての例題について子プロセッサの台数が 2 台の場合から実験を始め，最大 50 台の子プロセッサを利用して計算機実験を行った．ただし，ST70 の場合の一部の実験結果は実行時間の都合により子プロセッサ数が 5 台の場合から実験を行った．グラフ中の左縦軸は，計算が終了するまでの経過時間を秒単位で表している．また，グラフ中の右縦軸は，子プロセッサが 2 台の場合 (ST70 の一部は 5 台) を基準にして利用した台数に見合う加速度が得られるかを表示している．n 台の子プロセッサを利用すれば理想的には n 倍の加速度が期待されるが，グラフからもわかるとおり通信のオーバヘッドや負荷の不均一な分散などの原因により n を下回る加速になっている．

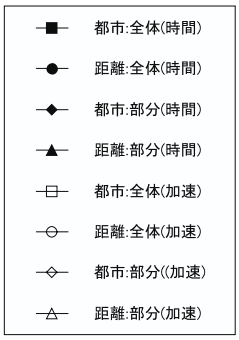

図 4 凡　　例
Fig. 4 Legends.

グラフ中には，経過時間 (時間)・加速度 (加速) でそれぞれ四つの結果がある (図 4)．それぞれの設定の違いは表にも示したとおり，分枝変数の選択方法と下界値の評価の違いによって異なる．分枝変数の選択方法は，**3. 2**で述べたように枝をコスト順に選択するか (距離)，訪問都市をもとに選択するか (都市) である．下界値の評価の違いは，枝刈りの評価を行う時点ですでに得られている暫定解の値と評価する部分問題の下界値が等しい場合，その部分問題を枝刈りするか (部分)，しないか (全体) に依存している．本研究においては近似解法を用いて分枝限定法とは独立に，近似解として暫定解を見つける方法を併用している．近似解法や，探索の過程から得られた暫定解と等しい下界値をもつ部分問題を枝刈りすると評価する部分問題の数は少なくなり，実行時間も短くなる．けれども，同じ最小巡回コストをもつ経路が複数存在する場合，それらを見つけることができない．今回の実験ではこの枝刈り条件を変更することで探索範囲の大きさを変更し 2 種類の分枝方法の優劣を問題ごとに検討した．

TSP に対する良い近似解法を利用すると分枝限定法だけの場合と比較して早期に十分精度の良い暫定解 (近似解) を利用できる．本研究で取り上げた例題の規模では最終的に最適解となる近似解を見つけるが，特に最適解であることを前提とせず最適性の確認を行っている．より規模が大きい問題を扱い暫定解として得られる近似値が最適値より悪い場合でも，探索範囲が広くなるだけであり解空間は同じであるから今回の実験結果と同様の予測は可能であると考える．

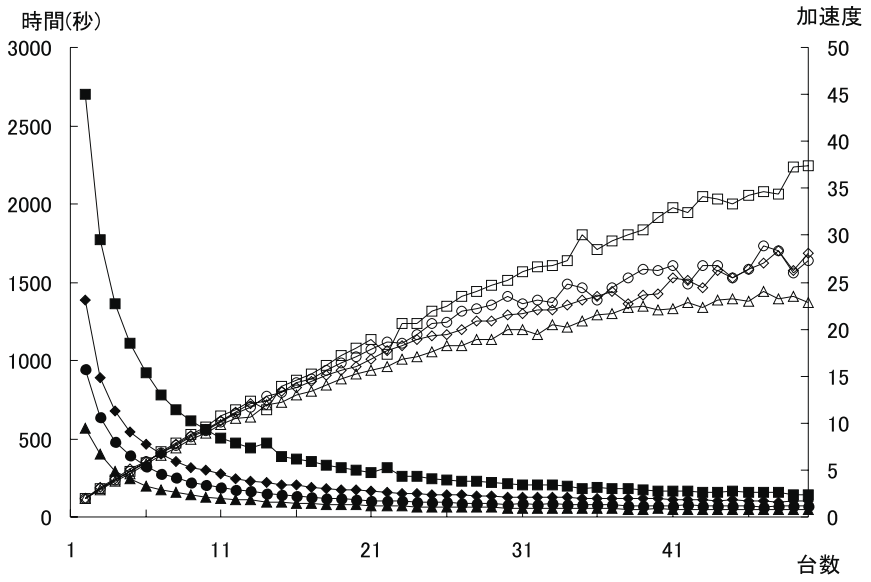

図 5　計算時間と加速度 (GR48)
Fig. 5　Computing time and acceleration.(GR48)

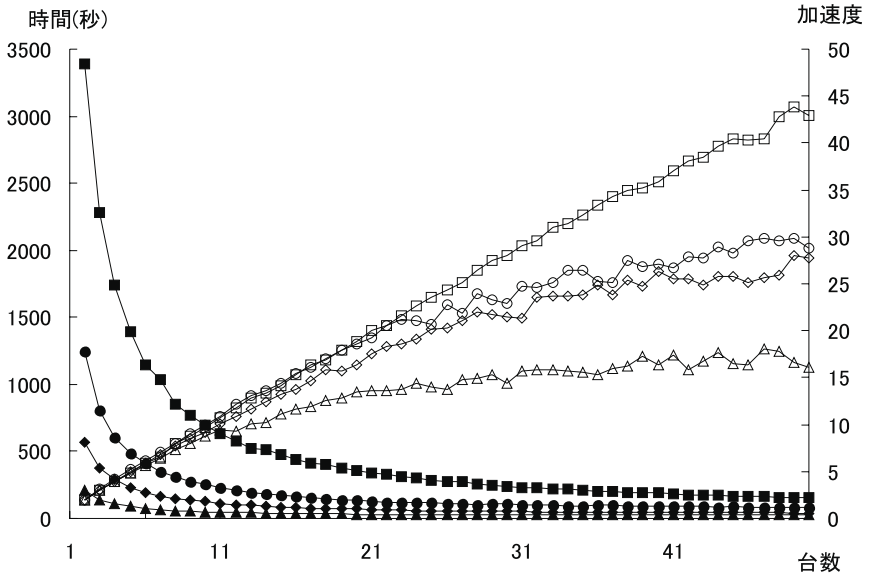

図 6　計算時間と加速度 (EIL51)
Fig. 6　Computing time and acceleration.(EIL51)

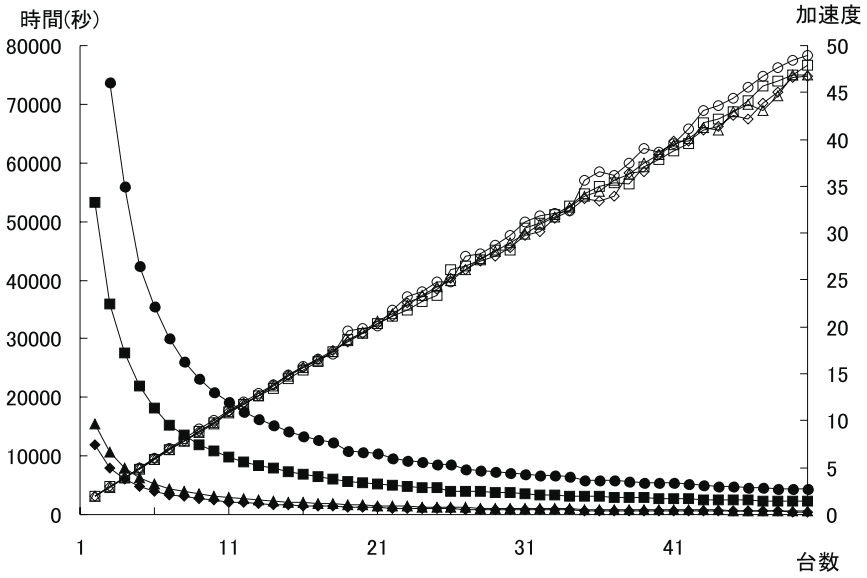

図 7　計算時間と加速度 (ST70)
Fig. 7　Computing time and acceleration.(ST70)

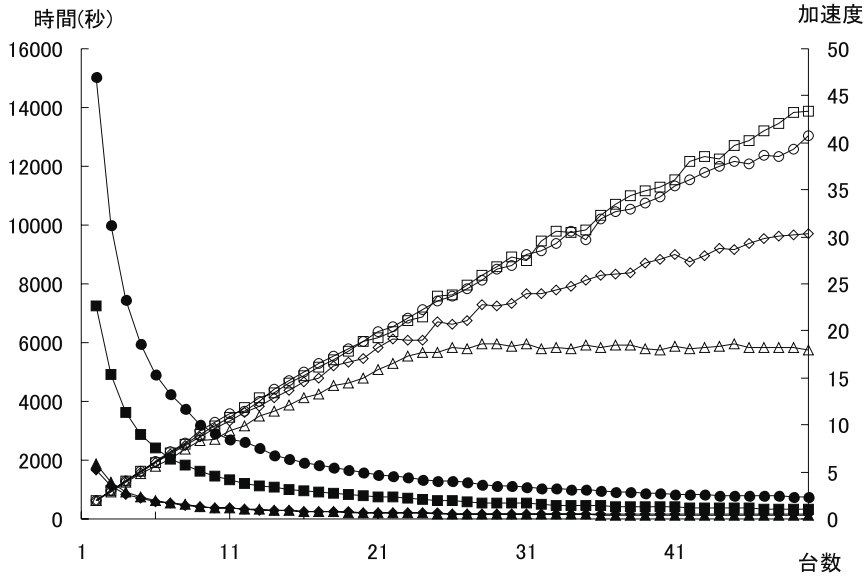

図 8　計算時間と加速度 (RAT99)
Fig. 8　Computing time and acceleration.(RAT99)

表 2 実験結果
Table 2 Summary of results.

総数 10 台

問題名	選択範囲台数	距離部分 5 台	都市部分 5 台	距離全体 10 台	都市全体 10 台
GR48		**245** 秒	540 秒	**205** 秒	560 秒
EIL51		**87** 秒	230 秒	**252** 秒	696 秒
ST70		6270 秒	**4805** 秒	20955 秒	**10857** 秒
RAT99		755 秒	**670** 秒	2936 秒	**1470** 秒

総数 40 台

問題名	選択範囲台数	距離部分 20 台	都市部分 20 台	距離全体 40 台	都市全体 40 台
GR48		**75** 秒	173 秒	**72** 秒	169 秒
EIL51		**31** 秒	69 秒	**92** 秒	189 秒
ST70		1597 秒	**1239** 秒	5478 秒	**2815** 秒
RAT99		**220 秒**	223 秒	879 秒	**411** 秒

また，精度の良い近似解が近似解法により早期に利用できるので，この暫定解が最適解にならない場合でも下界値優先探索を利用することにより，以降の探索中に見つかる次の最適性が保証されない (実際は最適解になる) 暫定解を見つけるためにかかる時間は，その解の最適性を保証するための探索時間と比較すると十分短くその影響は少ないとも考えられる．

グラフから，部分的な解空間の評価に必要な計算時間や子プロセッサ台数を増加させる効果は，同じ分枝方法であればより広い解空間を対象にする場合でも同様の傾向になっていると考えられる．具体的には経過時間について，初期に部分的な評価を実行するときは 5(20) 台，その後全体を評価する場合に 10(40) 台の計算機を利用するような状況の例を表 2に示した．太字で示したように 5(20) 台の計算機を利用し部分的な評価で有効であった分枝変数選択方法が，10(40) 台の計算機を利用しその後に行う全体としての評価でも有効であることがわかる (RAT99，総数 40 台を除く)．

問題例ごとに見ると，GR48 と EIL51 の場合については実験を行った計算機台数の範囲では，同じ分枝方法であればより広い解空間を対象にする場合でも経過時間・加速度について狭い解空間の場合と同様の傾向になっている．ST70 の場合の加速度は，グラフからも読み取れるように，どの設定であっても得られる加速度には大きな差が見受けられなかった．RAT99 の場合は，子プロセッサの数が 20 台を超えるようになると，都市順の部分評価の場合で加速度がほぼ一定になっており，それ以上のプロセッサを利用しても効果が少ない．一方，枝距離の部分評価の場合は加速度が向上している．経過時間についても台数が 20 台前後のところで逆転が起きており，表 2の最後のような逆転現象はこのことが原因であると思われる．このように効果が少ない方法と効果がある方法とを部分的な解空間で比較し，その結果をより広い解空間の場合で利用しても同様の傾向が得られないということがわかる．

今回想定しているメタコンピューティング環境では，多数の計算機を有効に利用して問題を解くことが求められる．そこで，まず問題の特徴を見るために利用できる計算機群を分割し，複数の解法で部分的な問題を解き，有効な方法を見極めた上で問題全体をあらためてすべての計算機を利用して解く方法が考えられる．並列分枝限定法の場合，従来は探索方法や負荷の分散方法を変更することが考えられているが，今回の結果から分枝する変数を選ぶ方法も実行に影響があることがわかる．そこで，これらの方法を組み合わせて問題ごとにより有効な方法を見つければ更に計算機群の能力を有効に利用できると思われる．今回の計算機実験では，経過時間・加速度を利用可能な計算機・占有時間の範囲で測定するため最適解を一つ，若しくはすべて決定する設定・限られた規模の例題を採用している．一般的な分枝限定法では最適解を一つ決定すればよい場合が多いが，その場合は暫定解と下解値の枝刈り条件を緩和することで部分的な評価を行うことなどが考えられる．この場合部分的な評価を行った後も同じ構造の解空間を評価することから条件を適切に選べば有効な方法の見極めは可能であると思われる．

5. むすび

本研究では，メタコンピューティング環境とも呼ばれる複数の計算機がネットワークを介して接続された分散処理環境を想定し，具体的な問題として巡回セールスマン問題を並列分枝限定法を用いて解くことを取り上げた．巡回セールスマン問題の場合，問題例ごとに難易度やプログラムの振舞いが変わるため有効な方策をあらかじめ決めることは難しくすべての問題例に対して有効なプログラムをあらかじめ用意することは難しい．本研究においても分枝限定法において重要な操作の一つである分枝する変数を選択する方法を変更するだけでも問題例ごとに違いが見受けられた．

また本研究では，暫定解と下界値との関係による活性部分問題の枝刈り条件を変えることで問題例を評価する範囲を制限し，解法の初期に当たる範囲が狭い結果と広い結果の比較を行った．計算機実験の結果から評価範囲が狭い場合に有効であった分枝変数選択方法は評価範囲が広い場合でも同等若しくはより有効であ

ることを確認した．

このことから，多数の計算機から構成されるメタコンピューティング環境では，初期に複数の方策で問題を解きはじめ処理の過程から有効な方策を見極めて以降は有効な方策のみに集中することであらかじめ有効な方策がわからないような場合に平均的な処理時間を短くすることが可能である．

今後の課題としては，子プロセッサを構成する計算機群は近似解法の子プロセッサを除けば今回の計算機実験では均一であったが，計算機群に性能差がある場合や，今回評価を行った経過時間・加速度以外に有効な初期暫定解としての近似解法が利用できない場合での暫定解の改善に対する影響，TSP 以外の組合せ最適化問題への適用等が挙げられる．

文　献

[1] 今井正治，吉田雄二，福村晃夫，“分枝限定アルゴリズムの並列化とその評価，”信学論 (D)， vol.J62-D,no.6, pp.403-410,June 1979.

[2] 池上敦子，青柳雄大，飯塚肇，“分散記憶型マシンにおける並列整数計画法，”信学論 (D-I)， vol.J75-D-I,no.9, pp.801-808,Sept. 1992.

[3] 品野勇治，檜垣正浩，平林隆一，“並列分枝限定法における解の探索規則，” 計測自動制御学会論文誌,vol.32,no.9,pp.1379-1387,1996.

[4] 茨木俊英, 組合せ最適化－分枝限定法を中心として，講座・数理計画法第８巻　産業図書，1983.

[5] 今野浩，鈴木久敏，“整数計画法と組合せ最適化，” ＯＲライブラリー第７巻　日科技連，1982.

[6] E.L.Lawler, J.K.Lenstra, A.H.G.Rinnooy Kan etc., The travelling salesman problem, John Wiley and Sons, 1985.

[7] M.Held and R.M.Karp,“The traveling salesman problem and minimum spanning trees:Part II,” Math.Program., vol.1, pp. 6-25, 1971.

[8] S.Lin and B.W. Kernighan, “An efficient heuristic algorithm for the traveling salesman problem,”Operations Research, vol.21, pp. 498–516, 1973.

[9] J.F.Pekny and D.L.Miller, “A parallel branch and bound algorrithm for solving large asymmetirc traveling salesman problems,” Math.Program., vol.55, pp. 17-33, 1992.

[10] V.S.Sunderam, “PVM: A framework for parallel distributed computing,”J.Concurrency: Practice and Experiecnce, vol.2,no.4, pp.315-339, Dec. 1990.

[11] G.A. Geist and V.S.Sunderam, “Network based concurrent computing on the PVM system,” J. Concurrency: Practice and Experiecnce, vol.4,no.4, pp. 293-311, June 1992.

[12] G.A.Geist, A.L.Beguelim, et al, “PVM3.0 user's guide and reference manual,” Technical Report ORNL/TM-12187, Oak Ridge National Laboratory, Feb. 1993.

[13] Catherine Roucariro, “Parallel branch and bound algorithms － An overview,” In M.Cosnard et al., Parallel and Distributed Alogorithm, pp. 153–16, North-Holland, 1989.

[14] A.Taudes and T.Netouseok, “Implementing branch-and-bound algorithms on a cluster of workstations － A survey, some new results and open problems,”Computing and Mathemataical Optimization, Lecture Note in Economics and Mathematical Systems, Springer-Verlag, vol.367, pp. 79-102, 1991.

[15] M.J.Quin: “Analysis and implementation of branch-and-bound algorithms on a hypercube multicomputer”, IEEE Trans. on Comput., vol.39,no.3, pp.384-387,1990.

[16] S.Tschoke, R.Lueling, and B.Monien, “Solving the traveling salesman problem with a distributed branch-and-bound algorithm on a 1024 processor network,” Proc. of 9th Int. Parallel Processing Symposium, pp.182-189, 1995.

（平成 12 年 11 月 30 日受付，13 年 2 月 19 日再受付）

大西　克実　（正員）

1992 阪大・工・通信卒．1994 同大大学院修士（通信）課程了．1996 大阪市立大学助手となり，現在に至る．組合せ最適化問題の解法に関する研究に従事．情報処理学会，日本 OR 学会各会員．

榎原　博之　（正員）

1982 阪大・工・通信卒．1987 同大大学院博士（通信）課程了．同年阪大工学部助手．1994 年関西大工学部専任講師となり，現在，助教授．組合せ最適化問題，計算幾何学，並列アルゴリズム等の研究に従事．工博．情報処理学会，IEEE，ACM 各会員．

中野　秀男　（正員）

1970 阪大・工・通信卒．197 年同大大学院博士（通信）課程了．同年阪大通信工学科助手．1991 阪大助教授．1995 大阪市立大学教授となり，現在に至る．離散最適化問題の近似解法の評価，情報セキュリティ，ソフトウェア工学等の研究に従事．工博．情報処理学会，日本 OR 学会，IEEE，ACM 各会員．