組合せ問題の並列アルゴリズム

今井　正治

1. はじめに

整数計画法などによって定式化される組合せ問題の多くはNP完全であることが知られている．すなわち，この種の組合せ問題の最適解を求めるのに必要な計算量は問題の規模の増加に対して指数関数的に増加する．計算機技術の発展の結果，現在では，数 GFLOPS（数10億浮動小数点演算／秒）の演算速度をもつスーパー・コンピュータが製造されている．しかし，このような超高速計算機を用いても，大規模な組合せ問題を解くためにはばく大な時間が必要である場合が多い．

これまで，組合せ問題の効率のよい解法として，分枝限定法（Branch-and-Bound method），動的計画法（Dynamic Programming）などが用いられてきた［4］，［11］．これらの解法の共通の特徴の1つは，「与えられた問題を複数の部分問題に分割し，個々の部分問題の解を求めることによって，原問題の解を得る」という点である．原問題を分割して得られた部分問題は，多くの場合それぞれ独立に解くことができる．この場合，これらの部分問題を並列に処理することによって，原問題を解くのに必要な計算時間を短縮することが可能である．したがって，組合せ問題の並列処理は実用的な見地からも重要な問題である．

並列型分枝限定アルゴリズムのふるまいに関しては，興味深い現象が知られている．すなわち，p 台の処理装置をもつ並列計算機システム上並列型分枝限定アルゴリズムで組合せ問題を解くために必要な計算時間を $T(p)$ とする．このとき，ある条件のもとで，$T(1)/T(p)>p$ となる場合がある．この現象は，加速異常（acceleration anomaly）と呼ばれ，並列型分枝限定アルゴリズムに特有の現象である．他の並列アルゴリズムでは，$T(1)/T(p)\leq p$ となるのが普通であり，このような異常現象は報告されていない．

本論文では，組合せ問題の並列アルゴリズムについて考察する．まず，2.では，組合せ問題の代表的な解法の並列化の可能性と適合性について述べる．次に，3.では，組合せ問題の並列型アルゴリズムの実現に適した並列計算機システムを紹介する．4.では，並列型分枝限定アルゴリズムにおける異常現象について解説する．

いまい　まさはる　豊橋技術科学大学　情報工学系
〒441 豊橋市天伯町雲雀ケ丘 1—1

2. 組合せ問題の並列アルゴリズム

これまでに知られている組合せ最適化問題の代表的な解法には，分枝限定法，動的プログラミングなどがある．どちらの解法が与えられた問題を解くのに適しているかは，問題の性質および定式化による．ただし，これらの解法はいずれも，広い意味での分割統治法（divide-and-conquer method）にもとづいている．

分割統治法を適用して組合せ最適化問題を解く過程では，多数の未解決な部分問題が生成される．このとき，これらの部分問題が互いに独立である場合には，それぞれの部分問題を並行して解くことができる［3］，［13］．分枝限定法および動的プログラミングでは，多数の独立な部分問題を扱う場合が多いので，この2種類の解法はいずれも効率のよい並列化が可能である．

3. 並列計算機システムのモデル

3.1 並列計算機システムの分類

組合せ問題の並列処理の研究は，すでに1970年代に開始されている．しかし，当時に計算機の処理速度も遅く，大規模な並列計算機システムを実現することは非常に困難であった．しかし，近年のVLSI（超大規模集積回路）技術の発展の結果，多数の処理装置をもつ並列計算機システムを実現することが容易になり，数千～数万台の処理装置をもつ超並列計算機（Highly Parallel Compter System）も試作されている．

計算機システムの構成方法は，命令列およびデータ列の形態にもとづいて，次の4種類に大別できる［2］．

（1） SISD：Single-Instruction Stream, Single-Data Stream
（2） SIMD：Single-Instruction Stream, Multiple-Data Stream

（3） MISD：Multiple-Instruction Stream, Single-Data Stream

（4） MIMD：Multiple-Instruction Stream, Multiple-Data Stream

3.2 組合せ問題向き並列計算機システム

2.で述べたように，分枝限定法や動的計画法の並列アルゴリズムでは，部分問題の独立性に着目して，部分問題をそれぞれ異なる処理装置に独立したタスクとして割りつけて解くという方針が有望である．したがって，MIMD型のシステムは，組合せ問題の並列アルゴリズムの実行に適していると考えられる．組合せ問題の並列アルゴリズムの主要目的は，問題を多数のプロセッサで処理することによって問題を解くのに要する時間を短縮することである．したがって，多数のプロセッサを効率よく稼働させられるようにシステムを構成することが望ましい．

組合せ問題向きの並列計算機システムのアーキテクチャとしては，主に以下のような構成が提案されている．

（1） MIMD型密結合システム

（2） MIMD型疎結合システム

- 環状（ring）プロセッサ［18］
- 木状（tree）プロセッサ［1］，［6］，［8］，［13］
- 超立方体（hypercube）プロセッサ［14］

これまで，組合せ問題の並列アルゴリズムは，主としてMIMD型の並列計算機をモデルとした研究が中心であったが，最近SIMD型の計算機を対象にしたアルゴリズムの研究も行なわれている［7］．

4. 並列型分枝限定アルゴリズム

4.1 探索戦略

分枝限定法で，部分問題の処理の順序は探索戦略（search strategy）と呼ばれる．アルゴリズムを実行するのに必要な記憶空間の複雑さ（space complexity）およびアルゴリズムの計算時間の効率（efficiency）は，採用される探索戦略に大きく依存する．最もよく用いられる基本的な探索戦略は次の2種類である．

（1） 深さ優先探索（depth-first search）

（2） 最良優先探索（best-first search）

逐次的な深さ優先探索では，問題の規模に比例する記憶容量があれば，問題を解くことができる．一方，最良優先探索では，最適解を得るために処理される部分問題の個数が最小であるが，必要な記憶容量は，問題の規模に対して指数関数的に増大する［4］．

並列型分枝限定法でも，これらの探索戦略を用いることができる．この場合，理論的な興味は，次の2点であろう．

（1） プロセッサの台数および探索戦略と記憶容量の関係

（2） プロセッサの台数および探索戦略と計算時間の関係

4.2 プロセッサの台数と記憶容量複雑さの関係

深さ優先探索を用いる並列型分枝限定アルゴリズムで必要な記憶容量は，問題の規模（n）とプロセッサの台数（p）の積（$n \cdot p$）に比例する［5］．

また，並列型分枝限定アルゴリズムで最良優先探索を用いた場合に必要な記憶容量は，逐次的分枝限定法の場合と同様に，問題の規模（n）に対して指数関数的に増大する［5］．

4.3 プロセッサの台数と計算時間との関係

並列型分枝限定アルゴリズムでは，プロセッサの台数と計算時間との関係に関して，異常現象（anomalies）と呼ばれる興味深い現象が観察され，問題的解析も行なわれている［5］，［9］，［10］，［12］，［16］，［19］．

この現象について説明するために，分枝限定アルゴリズムの計算時間を，アルゴリズム中で処理される部分問題の個数で評価することにする．以下では，p台（$p \geqq 1$）のプロセッサをもつ並列計算機上での，並列型分枝限定アルゴリズムの計算時間を$T(p)$で表わす．また，アルゴリズムの加速率（speed-up）を$S(p) \equiv T(1)/T(p)$で表わす．

一般に，総計算量が同一であれば，逐次型アルゴリズムの計算時間$T(1)$と並列型アルゴリズムの計算時間$T(p)$の間には，明らかに，

$$T(1) \geqq T(p) \geqq T(1)/p \qquad (1)$$

という関係が成り立つ．したがって，加速率$S(p)$は

$$1 \leqq S(p) \leqq p \qquad (2)$$

となる．

分枝限定法の場合には，アルゴリズムを並列化することによって，部分問題の探索順序が変化し，総計算量自体が変化する．したがって，次の2通りの異常現象が起こりうる．

$$T(p) < T(1)/p \quad (S(p) > p) \qquad (3)$$

$$T(p) > T(1) \quad (S(p) < 1) \qquad (4)$$

式（3）が成り立つ場合には，プロセッサの台数分以上に処理速度が増加することになる．この現象を加速異

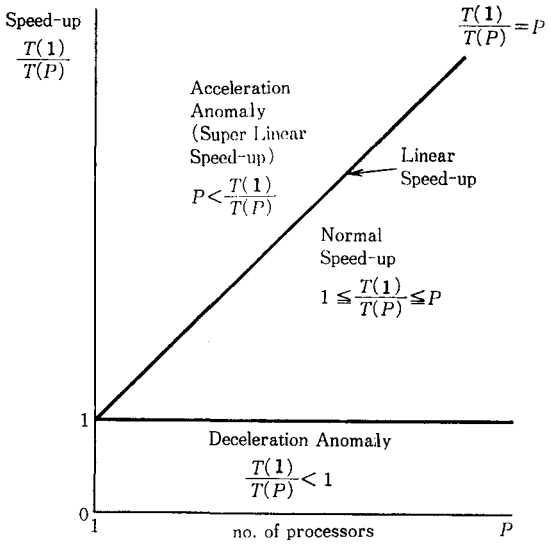

図 1 並列型分枝限定アルゴリズムにおける異状現象

常 (acceleration anomaly) または超線形加速 (super-linear speed-up) と呼ぶ．また，式(4)が成り立つ場合を減速異常 (deceleration anomaly) と呼ぶ (**図 1** 参照)．もちろん，「正常」な状態では，$S(p)$ の上界と下界は式 (2) で与えられる．

これらの異常現象は，探索戦略および目的関数の下界と深い関係がある．探索戦略によって，探索される部分問題の集合が異なるからである．以下の節では，最良優先探索および深さ優先探索の 2 種類の戦略で，異常現象が起きうる可能性について述べる．

4.3.1 最良優先探索

まず，最良優先探索を用いた場合には，異常現象は起こらない．これは，以下の理由による．臨界部分問題 (critical subproblem) の集合 C を次式で定義する．

$$C=\{s\mid \mathrm{low}(s)\leqq \mathrm{Copt}\} \qquad (5)$$

上式で，s は部分問題を表し，$\mathrm{low}(s)$ は s の最適解の下界 (lower bound) を表す．また，Copt は，原問題の最適解の目的関数の値を表わす．最良優先探索を用いる逐次型分枝限定アルゴリズムの計算時間 $T(1)$ は，$|C|$ と等しい．すなわち，$|C|$ は総計算量の下界を与える．並列型分枝限定アルゴリズムでも，すべての臨界部分問題の処理が終了するまでは，アルゴリズムは停止しない．したがって，

$$T(p)\geqq T(1)/p \quad (S(p)\leqq p) \qquad (6)$$

となる．また，$T(p)$ の上界は次式で与えられる[10]．

$$T(1)\geqq T(p) \quad (S(p)\geqq 1) \qquad (7)$$

したがって，最良優先探索を用いた場合には，異常現象は生じない．

4.3.2 深さ優先探索

次に，深さ優先探索を用いた場合について考える．深さ優先探索を用いた並列型アルゴリズムでは，異常現象が生じうる．これは，以下の理由による．

まず，深さ優先探索を用いる場合，逐次型分枝限定アルゴリズムにおいては，臨界部分問題はすべて処理する必要があるのは当然であるが，臨界部分問題に含まれない部分問題も処理する可能性がある．したがって，

$$T(1)\geqq|C| \qquad (8)$$

となる．並列型分枝限定アルゴリズムでも臨界部分問題は処理しなければならないので，

$$T(p)\geqq|C|/p \qquad (9)$$

となる．ところが，

$$T(p)\cdot p<T(q)\cdot q \qquad (10)$$

となるような探索木および $p,q(p>q\geqq 1)$ が存在することが知られている [5], [10]．$q=1$ の場合には，

$$S(p)>p \qquad (11)$$

となり，加速異常が生ずる．

また，式 (9) より，$T(p)\cdot p$ の下界は $|C|$ であるので，式 (4) を満足するような探索木も存在する．この場合には，

$$S(p)<1 \qquad (12)$$

となり減速異常が生ずる．

このように，深さ優先探索を用いる並列型分枝限定アルゴリズムでは，加速異常および減速異常が生じうる．

5. おわりに

本論文では，分枝限定法を中心として，組合せ問題の並列型アルゴリズムについて述べた．また，並列型分枝限定アルゴリズムに特有の，異常現象について簡単に紹介した．

頁数の制約もあり，分枝限定法以外の解法 (動的プログラミング法，分割統治法など) の並列化については，わずかしか述べられなかった．しかし，これらの解法も並列化に適しており，多数の研究成果が報告されている．

VLSI 技術の発展の結果，計算機の処理速度自体も高速化され，多数の処理装置をもつ超並列計算機の実現も容易になった．もちろん，並列化によって組合せ問題の本質的な困難さ自体が変化するわけではない．しかし，これらの超並列計算機を用いることにより，従来計算時間の制約のために解くことができなかった．大規模な組

合せ最適化問題のいくつかが実用的な時間内で解けるようになると期待される.

参考文献

[1] S. H. Fuller, J. K. Ousterhout, et al.: "Multi-Microprocessors: An Overview and working Example", Proc. of IEEE, Vol. 66, No. 2, pp. 216-228 (1978).

[2] M. J. Flynn: "Some Computer Organizations and their Effectiveness", IEEE, Trans. on Comp., Vol. C-21, No. 9, pp. 948-960 (1972).

[3] E. Horowitz and A. Zorat: "Divide-and-Conquer for Parallel Processing", IEEE, Trans. on Comp., Vol. C-32, No. 6, pp. 582-585 (1983).

[4] 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（講座数理計画法 8）(1983).

[5] 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」，電子通信学会論文誌，Vol. 62-D, No. 6, pp. 403-410 (1979).

[6] M. Imai: "A Double-Tree Structured Multicomputer System and its Application to Combinatorial Problems", 電子情報通信学会論文誌, Vol. E-69, No. 9, pp. 1002-1010 (1986).

[7] 岩本宙造，岩間一雄：「組合せ問題に対するRS型ベクトルアルゴリズム」，電子情報通信学会論文誌，Vol. J75-D-I, No. 3, pp. 143-151 (1992).

[8] A. K. Jones and P. Schwarz: "Experience using Multiprocessor Systems-A Status Report", Computing Surveys, Vol. 12, No. 2, pp. 121-165 (1980).

[9] V. Kumar and V. N. Rao: "Parallel Depth First Search. Part II. Analysis", Int'l Jour. Para. Prog., Vol. 16, No. 6, pp. 501-519 (1988).

[10] T. H. Lai and S. Shani: "Anomalies in Parallel Branch-and-Bound Algorithms", CACM, Vol. 27, No. 6, pp. 594-602 (1984).

[11] E. L. Lawler and D. Wood: "Branch-and-Bound Methods: a Survey", Oper. Res., Vol. 14, No. 4, pp. 699-719 (1966).

[12] G. J. Li and B. W. Wah: "Coping with Anomalies in Parallel Branch-and-Bound Algorithms", IEEE, Trans. on Comp., Vol. 35, No. 6, pp. 568-573 (1986).

[13] F. J. Peters: "The Tree Machines and Divide-and-Conquer algorithms", Lect. Note in Comp. Sci., Vol. 111, pp. 25-36 (1981).

[14] M. J. Quinn: "Analysis and Implementation of Branch-and-Bound Algorithms on a Hypercube Multicomputer", IEEE, Trans. on Comp., Vol. 39, No. 3, pp. 384-387 (1990).

[15] V. N. Rao V. Kumar: "Parallel Depth First Search. Part I. Implementation", Int'l Jour. of Para. Prog., Vol. 16, No. 6, pp. 479-499 (1987).

[16] E. A. Rpuul and G. L. Nemhauser: "Branch-and-Bound and Parallel Computation: A Historical Note", Oper. Res. Lett., Vol. 7, No. 2, pp. 65-69 (1988).

[17] 瀬口靖幸，田中正夫，中島利朗，他：「動的計画法の並列計算」，情報処理学会論文誌，Vol. 26, No. 6, pp. 824-830 (1985).

[18] B. W. Wah and Y. W. Ma: "MANIP-A Multicomputer Architecture for Solving Combinatorial Extrememum-Search Problems", IEEE, Trans. on Comp. Vol. C-33, No. 5, pp. 377-390 (1984).

[19] B. W. Weide: "Modeling Unusual Behavior of Parallel Algorithms", IEEE, Trans. on Comp. Vol. C-31, No. 11, pp. 1126-1130 (1982).