

4H-3

データ転送を考慮した最適化マルチプロセッサ
スケジューリング・アルゴリズム

伊藤敬介 宮川 尚 笠原博徳

早稲田大学理工学部電気工学科

1. はじめに

マルチプロセッサシステム上で単一プログラムを効率良く並列処理するには、処理すべきタスク集合をプロセッサにどのように割り当て、どのような順序で実行すべきか、という問題を解決しなければならない。しかし、この問題は、実行時間最小マルチプロセッサ・スケジューリング問題[1][2][3]と呼ばれ、特に、タスクプリエンプションが許されず、タスク間の先行制約が任意形状であり、タスク処理時間及びプロセッサ数が任意というような一般的な場合には、強NP困難となることが知られている[1][6]。これに対し著者らは、データ通信時間を考慮しない場合、実用的な最適化アルゴリズムとして、タスク数数百の大規模問題の多くを汎用計算機上で数分程度で解くことができるDF/IHS(Depth First/Implicit Heuristic Search)法を提案している[2]。しかし、特にタスク間のデータ転送時間が比較的大きい場合、データ転送を考慮したスケジューリングが必要である。このような場合に対しても著者らは、CP/DT/MISF法を提案し、その実用性に関して実マルチプロセッサシステム上で評価している。しかし、CP/DT/MISFはヒューリスティック手法であるため、必ずしも最適解を与えるものではない。そこで、本稿では、データ転送を考慮した実用的な最適化マルチプロセッサ・スケジューリング・アルゴリズム、DF/IHS[DT](considering Data Transfer)を提案する。

2. 問題の定義

本稿で扱うマルチプロセッサ・スケジューリング問題は、処理能力の等しい m 台のプロセッサにそれぞれの処理時間が異なる n 個のタスクからなるタスク集合 T をノンプリエンティブで並列処理するとき、その実行時間(スケジュール長)を、プロセッサ間のデータ転送時間を含めて最小にすることを目的とする。ここでのタスク集合 T は、タスクグラフという無サイクル有向グラフで記述されており、先行制約を表すエッジには、可変のデータ転送時間が重みとして付けられている。各エッジのテールとヘッドのノードが同じプロセッサに割り当てられたときはデータ転送をゼロとし、異なるプロセッサに割り当てられたときは、エッジの重み分のプロセッサ間データ転送が起こるものとする。

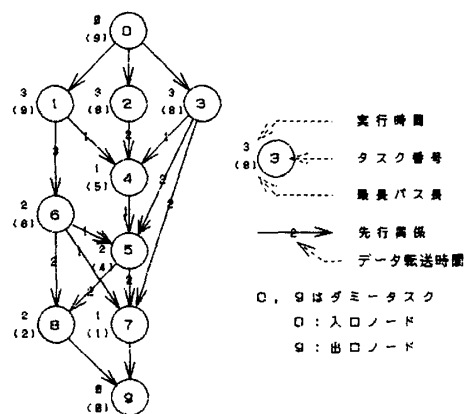


図1 タスクグラフの例

3. DF/IHS[DT] (Depth First/Implicit

Heuristic First[considering Data Transfer])

DF/IHS[DT]は、強NP困難である実行時間最小マルチプロセッサスケジューリング問題に対して提案された実用的な最適化アルゴリズム DF/IHSを、データ転送を考慮するように拡張したものである。よって、DF/IHSと同様に以下の特徴を持つ。

①効率の良い探索

前処理でCP/MISFのプライオリティを用いて、ヒューリスティックに良い解を探索木の左側に集めるため、探索部では単に左側優先でDF/FIFO形状の単純な探索を行えばよく、効率の良い探索が実現できる。

②少ないメモリ容量

DF/FIFO形状の探索を行なっているため、探索時に要するメモリ容量は、 $O(mn + n^2)$ (ただし m はプロセッサ数、 n はタスク数)で抑えられ、大規模な問題に対しても少ないメモリ容量で探索を行うことができる。

また、タスク間データ転送時間を考慮するため、以下の変更点が加わる。

③セクション規則の変更

従来DF/IHSでは、プロセッサ間データ転送時間を考慮しなかったため、実行可能なタスクを辞書式的の組合わせで選

びアイドルプロセッサに割り当てただけで良かったが、DF/IHS[DT]ではタスクとプロセッサの組み合わせにより発生するデータ転送時間が異なるので、組合わせて選んだものに対し、さらに順列をとる必要がある。また、DF/IHSでは、レディタスクテーブルRからのタスクのセレクションを行う際にSP値というポインタを用いている。DF/IHS[DT]におけるセレクション過程をRとSPを用いて示すと、次のようになる。レディタスク $\{1, 4, 5, \phi\}$ 、 $m_{av}=2$ のとき、

$[1, 4] [4, 1] [1, 5] [5, 1] [1, \phi] [\phi, 1]$

$[4, 5] [5, 4] [4, \phi] [\phi, 4] [5, \phi] [\phi, 5]$

の順に割りてる(但し、 m_{av} はその時点で使用可能なプロセッサ台数、 ϕ はアイドルタスク)。また、これに対応するSP値は次のようになる。

$[1, 2] [2, 1] [1, 3] [3, 1] [1, 4] [4, 1]$

$[2, 3] [3, 2] [2, 4] [4, 2] [3, 4] [4, 3]$

さらに、以下の拡張を行った。

④上限値の設定による探索の効率化

分枝限定法を用いて上限値と下限値の間に存在する解を探索する際に、上限値を任意に設定することによりその探索領域を分割し、効率的な探索を行う。まず、初期の上限値として与えられるCP/DT/MISFの解と全体の下限値の中間の値を“仮の上限値”として探索を始める。もし暫定解が見つかればその値と全体の下限値の中間の値を“仮の上限値”として新たに探索をはじめ、1つも解が見つからなければ“仮の上限値+1”を全体の下限値とし、それまでの暫定解と下限値の中間値を新たに“仮の上限値”として探索を続ける。このように、上限値と下限値の間を二分法を用いて探索することにより、段階的に全体の下限値を上げることが可能となり、従来ワークステーション上で3時間の探索でも解けなかった問題が20分で解けるなど、大幅な処理時間の短縮が実現された。

⑤DF/IHS[DT]自身の並列化

探索部を並列化することにより、探索の高速化を図る。PE間のデータ転送を最小とし、記憶領域も低く抑えるため、各PEに独立に深さ優先探索を行わせる。また、探索木がヒューリスティック的に左側優先で重み付けされていることを考慮し、 PE_1 は探索木の左から右へ探索を行い、他の PE_i ($i=2, \dots, m$)は PE_1 の探索パス上のノードを根ノードとして持つ部分探索木を右から左へと探索することにより、最適解の早期発見の可能性を上げている。この階層的な挟み撃ち探索の様子は図2に示される。同様の手法を用いた並列化DF/IHSでは、4台のPEで平均で1台の10倍以上の処理速度を得るなど、既にその有効性が示されている。並列

化DF/IHS[DT]の場合にもこのような加速効果を有効に引き出すことができ、処理の高速化が達成できる。

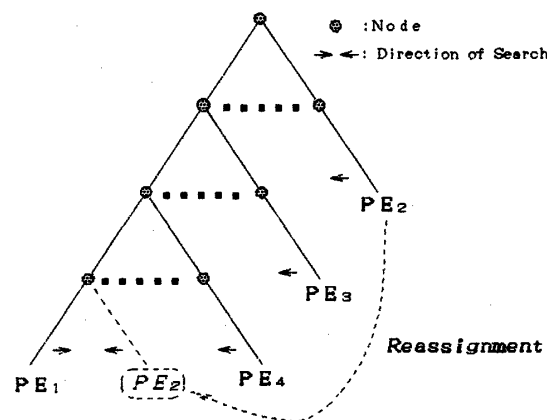


図2 階層的挟み撃ち探索

4. むすび

本稿では、データ転送時間を考慮した最適化マルチプロセッサ・スケジューリング・アルゴリズム、DF/IHS[DT]を提案した。今後の課題としては、探索途中で使用する下限値の改良が挙げられる。

参考文献

- [1]Coffman, E.G. (Ed.): "Computer and Job-shop Scheduling Theory", John Wiley & Sons (1976).
- [2]H. Kasahara and S. Narita, "Practical multi-processor scheduling algorithms for efficient parallel processing", IEEE Trans. Comput., C-33, 11, pp. 1023-1029 (Nov. 1984)
- [3]Ramamoorthy, Chandy and Gonzalez, "Optimal scheduling strategies in a multiprocessor system", IEEE Trans. Comp. C-21, 2, pp. 137-146 (Feb. 1972)
- [4]ラトナ, 笠原: "データ転送を考慮した最適化マルチプロセッサ・スケジューリング・アルゴリズム", 電子情報通信学会春季全国大会, D-99 (Mar. 1990)
- [5]伊藤, 田中, 笠原: "並列化マルチプロセッサ・スケジューリング・アルゴリズム", 情報処理学会研究報告 90-ARC-83, pp. 91-96 (Jul. 1990)
- [6]茨木俊秀: "組み合わせ最適化", 産業図書 (1983)