ハイパフォーマンス
コンピューティング 73−12
（1998．10．9）

タスクスケジューリングにおける利用プロセッサ数の制限法に対する考察

木股洋　　小林真也

金沢大学 工学部 電気・情報工学科

マルチプロセッサシステムにおける従来のタスクスケジューリングアルゴリズムは，プロセッサが十分ある場合，必要以上にタスクを多くのプロセッサに分散させることで処理時間の増加をもたらすことがある．本稿ではまずこの問題に対する対処法として，実行可能なタスク数の期待値によってプロセッサ数を限定する方法について述べる．次にこの方法における実行可能なタスク数を求める手法としてタスクの処理される確率の変化を考慮した方式を提案し，その実験結果について述べる．最後にこの制限方法についての検討を加えその問題点について明らかにする．

Consideration in the Method for Limiting Number of Processors for Task Scheduling

Hiroshi KIMATA　　Shin-ya KOBAYASHI

Department of Electrical and Computer Engineering, Knazawa Univ.

Traditional task-scheduling method for multiprocessor systems tends to allocate tasks to too many processors if many processors are available, therefore in that case, communication overhead increases and then processing time takes a long time. In this paper, we describe the traditional solution for this problem, that limits the number of processors based on average number of tasks to be executed. We propose another method for calculating the average number of processing tasks and show the experimental results. Finally, we discussed an issue of these methods.

1　はじめに

マルチプロセッサシステムでジョブを高速に実行するためにはタスク間の依存関係や通信量を考慮して実行プロセッサや実行順序を決定してくタスケジューリングアルゴリズムが重要である．特にプロセッサ間通信が処理時間に及ぼす影響は大きく，通信を適切に扱うことがスケジューリングの性能を大きく左右することになる．

これまでにプロセッサ間の通信時間を考慮したスケジューリング法として，CP/DT/MISF法[1]，北川らの方法[2]，CP/RCO法[3][4]などが提案されている．しかし，これらのスケジューリングアルゴリズムはいずれも割当て時点における局所的な通信による影響を削除することしか考慮していないため、現在の割当てが将来の通信に及ぼす影響を考慮した割当てができない．そのため，タスク集合に対してプロセッサ数に余裕がある場合には通信オーバーヘッドが増えてしまうにもかかわらず，必要以上にタスクを多くのプロセッサに分散させてしまう．その結果，プロセッサ数の増加に対して始めは処理時間が減少するものの，ある時点からは増加することがある．

その一例として，文献[4]に取り上げられているルンゲクッタ法による常微方程式の解計算を行うジョブを，各スケジューリング法を用いてメッシュ結合マルチプロセッサシステムへ割当てたときの処理時間を図1に示す．

図1より，いずれのスケジューリング法で割り当てた場合でもプロセッサ数の増加に対してある台数までは処理時間は減少していくが，それを越えると逆に処理時間が増加していくのがわかる．これはプ

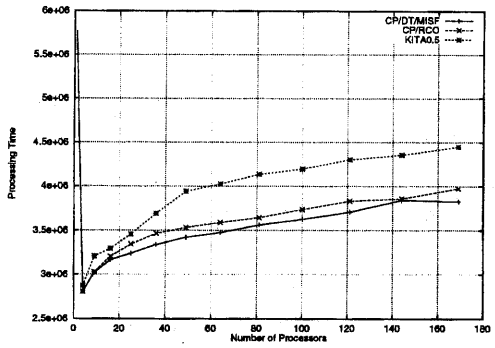


図 1: プロセッサ数 vs 処理時間

ロセッサ数が少ないとき時には，プロセッサ数の不足によりジョブの並列性を十分に利用した割当てができず，プロセッサ数が多い時には，タスクが割り当てられるプロセッサが分散されるため地理的に離れたプロセッサ間で通信が行われ，通信時間が増加してまうためと考えられる．

このことから，処理時間を少なくするためにはジョブの並列性を十分利用できる適切なプロセッサ数を求める必要がある．

この適切なプロセッサ数を求める方法として，我々は先の研究で実行可能なタスク数の期待値によってプロセッサ数を限定する方法についての提案を行っている [5]．本稿では，この方法に対して更に検討を加え，その問題点について明らかにする．

2 プロセッサ数の限定法 1

実際のスケジューリングにおいて処理時間が最小となるプロセッサ数は図 1 の例でも分かるようにスケジューリングアルゴリズムによる違いはあまりなく，タスク集合の性質によって決まってくると考えられる．

そこで，文献 [5] ではタスク間の通信による影響が無視できるような理想的な条件を仮定し，その条件下で実行可能なタスク数の期待値を求めることにより適切なプロセッサ数を決定する方法が提案されている．

タスク集合に対しプロセッサ数が十分あり，通信の影響が無視でき，最適な割当てが可能であれば，全処理時間はタスク集合の最大最長パス長となる．

全タスクの処理が最大最長パス長で終了するとき，タスク毎にそのタスクが最も早く処理を開始できる時刻である最早開始時刻と，最大最長パス長で全処理を終らせるために遅くともそのタスクが処理を終えていなければならない時刻である最遅終了時刻が求められる．

しかし，最早開始時刻から最遅終了時刻までの間にタスクが実際に処理されている時間はそのタスクのタスクサイズの時間だけである．従って最早開始時刻から最遅終了時刻までのある瞬間にタスクが処理されている確率は，$\frac{\text{タスクサイズ}}{\text{最遅終了時刻}-\text{最早開始時刻}}$ となる．更に，ある瞬間に処理されているタスク数の期待値は，その瞬間において各タスクが処理されている確率の和で表される．この瞬間毎に処理されているタスク数の期待値の最大値は，通信時間の影響が無視できるならタスク集合の並列性を十分に利用して処理時間を最小にするために必要なプロセッサ数の期待値になると考えられ，使用するプロセッサ数の限定には妥当な数であると考えられる．

以下にこの値を求めるアルゴリズムを示す．

(i) タスク k の最早開始時刻 T_{esk}・最遅終了時刻 T_{lck} を求める．

最早開始時刻 T_{esk} は式 (1) で定義され再帰的に求められる．ただし，S_j はタスク j のタスクサイズ，PT はタスク k の直前タスク集合である．

$$T_{esk} \equiv \begin{cases} \max\limits_{j \in PT}\{T_{esj} + S_j\} & for \quad PT \neq \emptyset \\ 0 & for \quad PT = \emptyset \end{cases} \tag{1}$$

また，最遅終了時刻 T_{lck} は式 (2) で定義される．ただし，lcp はタスク集合の最大最長パス長，cp_k はタスク k の最長パス長である．

$$T_{lck} \equiv lcp - cp_k + S_k \tag{2}$$

(ii) 任意の時刻 t におけるタスク k の処理される確率 $f(k,t)$ を求める．

$$f(k,t) \equiv \begin{cases} 0 & for \quad t < T_{esk}, T_{lck} \leq t \\ \dfrac{S_k}{T_{lck} - T_{esk}} & for \quad T_{esk} \leq t < T_{lck} \end{cases} \tag{3}$$

(iii) 任意の時刻 t における処理中タスク数の期待値 $F(t)$ を求める．

ただし，N はタスク集合のタスク数．

$$F(t) = \sum_{k=1}^{N} f(k,t) \tag{4}$$

(iv) 処理中タスク数の期待値の最大値 F を求める．

$$F = \max_{0 \leq t \leq lcp}(F(t)) \tag{5}$$

3 プロセッサ数の限定法2

先に述べたプロセッサ数の限定法ではタスクの最早開始時刻から最遅終了時刻の間に処理される確率が一定であるとしている(図2).

しかしタスクサイズが最早開始時から最遅終了時刻までの時間に占める割合が $T_{lck}-T_{esk} \leq 2S_k$ となるような場合には,タスクがどの時点で実行を開始するとしても $T_{lck}-S_k \leq t < T_{esk}+S_k$ の範囲ではでは必ず処理されるはずである.従って,開始時刻を段階的にずらしていくとタスクの処理される確率は図3のように変化すると考えられる.さらに開始時刻を細かくずらしていくことによって最終的には図4 (a)のように直線的に処理される確率が変化していくと考えられる.同様に考えると,$T_{lck}-T_{esk} > 2S_k$ の場合も図4 (b)のようになる.

従って,時刻 t におけるタスク k の処理される確率 $f(k,t)$ を求めると以下のようになる.

(a) $T_{lck}-T_{esk} \leq 2S_k$ の場合

図4 (a)より, $T_{esk} \leq t < T_{lck}-S_k$ では,

$$f(k,t) = \frac{t-T_{esk}}{T_{lck}-S_k-T_{esk}}.$$

$T_{esk}+S_k \leq t < T_{lck}$ では,

$$f(k,t) = 1-\frac{t-(T_{esk}+S_k)}{T_{lck}-(T_{esk}+S_k)} = \frac{T_{lck}-t}{T_{lck}-T_{esk}-S_k}.$$

従って, $f(k,t) \equiv$

$$\begin{cases} 0 & for \quad t < T_{esk}, T_{lck} \leq t \\ \dfrac{t-T_{esk}}{T_{lck}-T_{esk}-S_k} & for \quad T_{esk} \leq t < T_{lck}-S_k \\ 1 & for \quad T_{lck}-S_k \leq t < T_{esk}+S_k \\ \dfrac{T_{lck}-t}{T_{lck}-T_{esk}-S_k} & for \quad T_{esk}+S_k \leq t < T_{lck} \end{cases} \tag{6}$$

となる.

(b) $T_{lck}-T_{esk} > 2S_k$ の場合

図4 (b)の h は,

$$S_k \cdot h + \{T_{lck}-S_k-(T_{esk}+S_k)\} \cdot h = S_k$$

から,

$$h = \frac{S_k}{T_{lck}-T_{esk}-S_k}$$

である,よって,

$T_{esk} \leq t < T_{esk}+S_k$ では,

$$f(k,t) = \frac{h}{S_k}(t-T_{esk}) = \frac{t-T_{esk}}{T_{lck}-T_{esk}-S_k}.$$

$T_{lck}-S_k \leq t < T_{lck}$ では,

$$f(k,t) = h-\frac{h}{S_k}\{t-(T_{lck}-S_k)\} = \frac{T_{lck}-t}{T_{lck}-T_{esk}-S_k}.$$

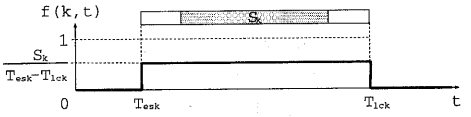


図 2: 限定法1

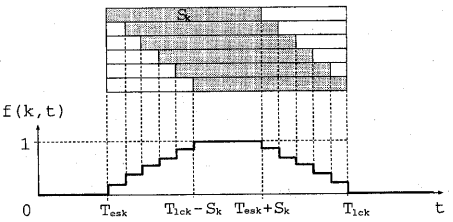
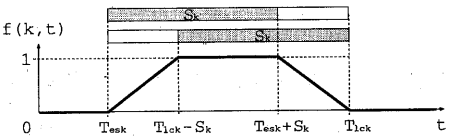


図 3: 処理確率の変化



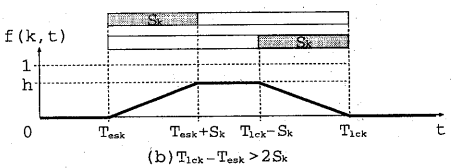


図 4: 限定法2

従って, $f(k,t) \equiv$

$$\begin{cases} 0 & for \quad t < T_{esk}, T_{lck} \leq t \\ \dfrac{t-T_{esk}}{T_{lck}-T_{esk}-S_k} & for \quad T_{esk} \leq t < T_{esk}+S_k \\ \dfrac{S_k}{T_{lck}-T_{esk}-S_k} & for \quad T_{esk}+S_k \leq t < T_{lck}-S_k \\ \dfrac{T_{lck}-t}{T_{lck}-T_{esk}-S_k} & for \quad T_{lck}-S_k \leq t < T_{lck} \end{cases} \tag{7}$$

となる.

式(6)(7)によって求められたタスクの処理される確率 $f(k,t)$ をもとに,限定法1と同様に任意の時刻 t における処理中タスク数の期待値 $F(t)$ を式(4)によって求め,処理中タスク数の期待値の最大値 F を求める.

この方法によって求められる処理中タスク数の最大値も,処理時間の増加を抑制するようなプロセッサ数を限定するには妥当な値であると考えられる.

4 評価と考察

プロセッサ数限定法1及び限定法2によって決定されるプロセッサ数と既存のスケジューリングアルゴリズムを用いて割り当てた場合に処理時間が最小となるプロセッサ数を比較する．

スケジューリング法は，CP/DT/MISF法，北川らの方法(通信係数0.5)，CP/RCO法とした．また，プロセッサの結合形態は完全網，ハイパーキューブ網，Illiacメッシュ網を対象とする．

限定法1及び限定法2によって決定されたプロセッサ数 F は整数値とはならないため，各結合形態に適用するため以下のようにプロセッサ数を補正する．

$$\text{完全網：}\ \lceil F \rceil$$
$$\text{キューブ網：}\ 2^{\lceil \log_2 F \rceil}$$
$$\text{メッシュ網：}\ \lceil \sqrt{F} \rceil^2$$

4.1 ランダムタスク集合による評価

乱数を用いて生成したランダムタスク集合による評価を行った．

ランダムタスク集合は次のような条件で作成した．タスク数Nのタスク集合の各タスクに1からNまでの通し番号を付け，タスクnがn+b番からn+b+a番までのタスクに対して0.3の確率で依存関係を持つものとした．ここでは、タスク数Nは200,400,600の3種類，aは5,10,20の3種類，bは1,5,10の3種類で計27種類とし，各種類毎に10組のタスク集合を生成した．このタスク集合ではaの値が大きいほど依存タスク数が大くなるためタスクどうしの関連は密となり，bの値が大きいものほど近隣タスクとの依存が少なくなるためタスクどうしの関連が疎となる．

表1に結果を示す．一致率はプロセッサ数限定法によって決定されたプロセッサ数(制限プロセッサ数)が各スケジューリング方式で最も処理時間が短くなったプロセッサ数(最適プロセッサ数)と完全に一致した割合，平均増加率は最適プロセッサ数での処理時間に対する制限プロセッサ数での処理時間の増加率を表す．

表1の一致率を見る限りではいずれの場合も限定法1と限定法2の顕著な相違はあまり見られない．CP/DT/MISF法・完全網の場合の最適割当て時を基準としたときの制限プロセッサ数の分布を図5，図6に，処理時間の分布を図7，図8に示す．分布に関しても両方式の明確な相違点はなく．全体としてはプロセッサ数を少なく見積もる傾向があるもの

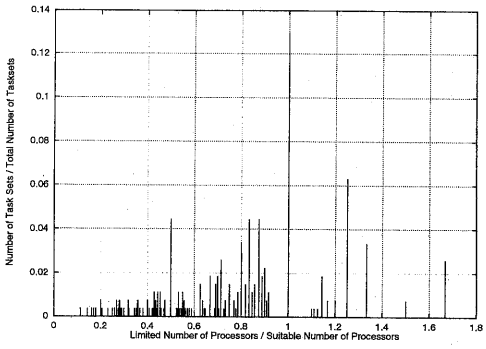


図 5: 制限プロセッサ数の分布 (限定法1)

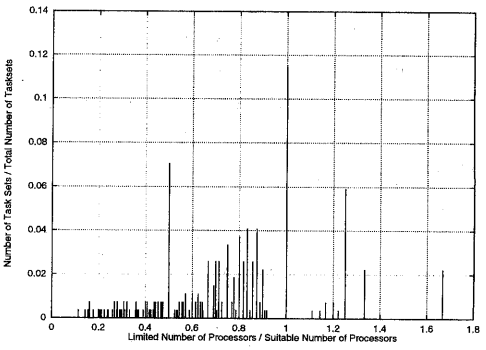


図 6: 制限プロセッサ数の分布 (限定法2)

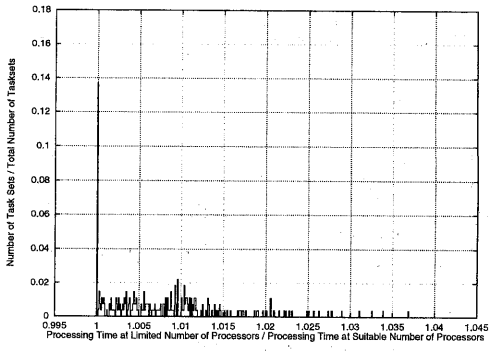


図 7: 処理時間の分布 (限定法1)

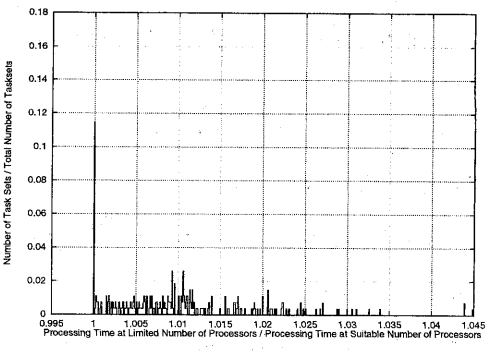


図 8: 処理時間の分布 (限定法2)

表 1: ランダムタスク集合による評価結果

スケジューリング法	結合形態	一致率		平均増加率	
		限定法 1	限定法 2	限定法	限定法 2
CP/DT/MISF	comp	0.130	0.115	1.008	1.010
	cube	0.463	0.485	1.007	1.008
	mesh	0.548	0.641	1.013	1.010
CP/RCO	comp	0.104	0.104	1.008	1.009
	cube	0.433	0.452	1.008	1.008
	mesh	0.530	0.622	1.014	1.010
KITA0.5	comp	0.189	0.159	1.021	1.027
	cube	0.478	0.478	1.008	1.010
	mesh	0.604	0.678	1.017	1.014

表 2: タスク集合の特徴の違いによる評価結果

依存特性	限定法 1			限定法 2		
(a, b)	N=200	N=400	N=600	N=200	N=400	N=600
5,10	0.581	0.462	0.362	0.565	0.460	0.360
5,5	0.542	0.329	0.433	0.541	0.326	0.391
5,1	0.788	0.449	0.398	0.775	0.438	0.410
10,10	0.831	0.696	0.733	0.801	0.641	0.676
10,5	0.932	0.860	0.777	0.880	0.811	0.711
10,1	1.080	0.776	1.084	1.024	0.714	1.039
20,10	0.872	0.876	0.881	0.769	0.775	0.815
20,5	0.984	0.915	0.930	0.854	0.871	0.892
20,1	1.143	1.288	1.245	1.098	1.295	1.220

の最適プロセッサ数と一致する割合が最も高いことが分かる.

また, 制限プロセッサ数の分布が同じような傾向を示していることから, 増加率に関しても同様に両方式における違いは見られなかったが. 共に効果的に処理時間の増加を抑えられていることが分かる.

4.2 タスク集合の特徴の違いによる評価結果の考察

タスク集合の特性との関係を調べるために, ランダムタスク集合による評価結果の詳細を表 3 に示す.

表 3 は CP/DT/MISF 法・完全網の場合の制限プロセッサ数／最適プロセッサ数の値の平均値をタスクの特性毎に分類したものである. 表 3 から, (a,b) = (5,10), (5,5) のような比較的タスク間の関連が疎なタスク集合に対しては, 最適プロセッサ数に対して少なく見積もり, 逆に (a,b) = (20,1) のようなタスク間の関連が密なタスク集合に対しては最適プロセッサ数に対して多く見積っていることが分かる.

これはどちらの方式も, 潜在的な並列性が高いと考えられるタスク間の関連が疎なタスク集合では, タスクサイズに対して (最遅終了時刻－最早開始時刻) が大きくなり, 瞬間におけるタスクの処理される確率が小さくなってしまうために任意の時刻における処理中タスク数の期待値が小さくなり, 結果的に限定プロセッサ数を小さく算出してしまうためだと考えられる.

逆に潜在的な並列性が低いと考えられるタスク間の関連が密なタスク集合では, タスクサイズに対

して(最遅終了時刻－最早開始時刻)が大きくなり，瞬間におけるタスクの処理される確率が大きくなってしまうために処理中タスク数の期待値が大きくなり，結果的に限定プロセッサ数を大きく算出してしまうためだと考えられる．

5 まとめ

本稿では，処理中タスク数の期待値に基づいたプロセッサ数の限定法の一手法を提案し，評価を行った．

限定法2では，最早開始時刻から最遅終了時刻までのタスクの処理確率を一定とした限定法1に対して，時間経過に伴うタスクの処理確率の変化を考慮することでより厳密に処理中タスク数の期待値を求めることを狙ったが，評価結果からは両方式における顕著な結果の相違は見られず，期待したような効果は得られなかった．しかし，両方式とも完全に適切なプロセッサ数を決定できるとは限らないものの効果的に処理時間の増加を抑制できることを示した．

また，ランダムタスクによる評価結果の詳細から，限定法1，2ともにタスク間の依存度合に対して同様の傾向を示すことが分かり，制限法における問題点を明らかにすることができた．

今後，この結果から導き出されたタスク間の依存度合による影響をアルゴリズムに反映させることでより厳密にタスク集合の並列性を求める方法を検討する予定である．

参考文献

[1] 笠原博徳, “並列処理技術”, コロナ社, (1991)

[2] 北川秀夫, 松田文夫, 内川嘉樹, 服部秀三, “通信時間を考慮にいれたマルチプロセッサスケジューリングアルゴリズム”, 信学論 D-I, Vol.J73-D-I, No.10, pp.812-817 (1990)

[3] 小林真也, 木村春彦, 武部幹, “マルチプロセッサシステムにおける通信時間を考慮したタスク割当て法”, 信学論 D-I, Vol.J76-D-I, No.12, pp.689-694 (1993)

[4] 小林真也, “不完全結合マルチプロセッサシステムに対するタスク割当て法の提案と評価”, 信学論 D-I, Vol.J79-D-I, No.2, pp.69-78 (1996)

[5] 小林真也, “タスクスケジューリングにおける利用プロセッサ数の制限法”, 信学論 D-I, Vol.J81-D-I, No.3, pp.353-355 (1998)