

2Q-2 異種クラスタのためのタスクスケジューリング*

桑島 康二 渋谷 進†
茨城大学工学部情報工学科‡

1 はじめに

近年の計算機の普及やネットワーク技術の急速な進歩により、ネットワークで結ばれた多数の計算機を一つの並列計算機と見なして利用する方法が注目されている。クラスタ上で効率良く並列計算を行うためには、それぞれ計算機の処理能力や計算機間通信能力、各タスクの実行時間やタスク間の通信量などを考慮して、タスクを適切に割り当てる必要がある [1]。

不規則なタスクの到達パターンと各タスクサイズの違い、異なる処理性能を持つ計算機で構成される異種クラスタシステムにより、処理に偏りが生じ全体の処理に影響を及ぼすと考えられる。処理バランスを均衡化させる技術は、処理速度の遅い計算機より、処理速度の速い計算機にタスクを多く割り当てることによりシステム全体の性能を改善させることができる [2]。また、処理速度に大きな差が生じた場合、処理速度の速い計算機により多くのタスクを割り当て処理速度の遅い計算機にはタスクを割り当てないことによりシステム全体の性能向上を図ることができる。

本論文では、クラスタ内の異なる処理速度を持つ計算機にタスクを動的に割り当てる方法について提案する。異なる処理速度、タスクサイズを考慮し、各計算機にタスクを割り当てる方法求め、それにより全体の処理速度の向上を図る。

2 タスクスケジューリング

本研究ではスケジューリング方法として動的スケジューリングを用いる。通常、動的スケジューリングは高いシステムオーバーヘッドを伴うが静的スケジューリングより優れる。

タスクは不規則にシステムに到達するものとする。各タスク T_i はタスクサイズの値を持ち、このタスクサイズに基づき処理時間を求めることができるとする。

タスクがシステムに到達したとき、タスクはある計算機上にあるスケジューラに到達し、スケジューラによりスケジューリングされ適当な計算機に割り当てられる (図 1)。その際、スケジューラは各計算機から必要な情報を収集し、それを元に割り当てる計算機を決

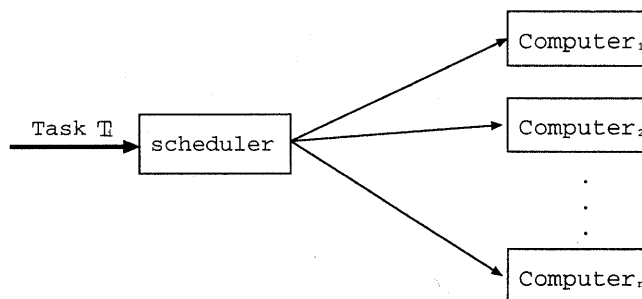


図 1: タスクスケジューリング

定する。タスクを割り当てられた計算機は逐次的にタスクを処理する。割り当てられた際にその計算機が処理中だった場合はそのタスクは待ち状態に入り前の処理が終了され次第処理される。

タスクは実際の処理の際、タスクサイズより予想される処理終了時間と異なる時間で終了することが予想される。これは、タスクコードにおけるデータ依存ループ、条件文の実行、システム構成が原因になる。

また、各タスク間におけるデータのやり取り、他のタスクとの処理順序などのタスク間通信も発生する場合もあるが、このような要因については本研究では考慮しない。

3 スケジューリング方法

スケジューラにタスクが到達したときスケジューラは各計算機における処理能力を考慮する。処理能力を考慮する方法として以下の二つを提案する。

一つは、各計算機の処理速度のみを用いスケジューリングする方法である。スケジューラは最初に各計算機の処理速度を取得し、そこから処理速度の比率を求めその比率に従いタスクを割り当てる。この際、タスクは処理速度の速い計算機から割り当てていく。この方法は静的スケジューリングといえる。

しかしながら、この方法を用いた場合タスクサイズを考慮していないのでタスクサイズに大きくばらつきがあるときに正しくスケジューリングされない。

そこで、二つめとしてタスクサイズを考慮する動的スケジューリング方法を提案する。タスク到達の際、スケジューラは現在の各計算機の処理速度、各計算機に既に割り当てられたタスク全ての合計サイズを得、それら値より割り当てる計算機を決定する。式として

*A Task Scheduling Method for Heterogeneous Cluster Systems

†Kouji Kuwajima, Susumu Shibusawa

‡Department of Computer and Information Sciences, Faculty of Engineering, Ibaraki University

表すと、ある計算機において

$$\frac{\text{合計タスクサイズ} + \text{到達したタスクサイズ}}{\text{計算機の処理速度}}$$

によりある値が求められる。この値を割り当て優先度と呼ぶとする。この値を全ての計算機に対して求め、最も小さい値を算出した計算機に対しタスクを割り当てる。これにより、タスクがスケジューラに到達した段階においてそのタスクを最も速く処理できる計算機にタスクが割り当てられることになる。

一つめの方法を用いた場合、タスクは各計算機に均一的に割り当てられ、各計算機の処理終了時間の均衡化を図る。二つめの方法の場合、タスクは処理速度の速い計算機に集中して割り当てられる。これらの方法により、システム全体の処理終了時間の向上を図る。

4 シミュレーション

以下のような制約のもと、前節で述べたスケジューリングについてシミュレーションを行う。

- タスク間の通信は無いものとする。各タスク間ではデータの依存が存在しないし、先行関係も無い。
- タスクは計算機の処理速度とタスクサイズに基づきスケジューリングされるが、実際の処理終了時間はスケジューリング時に予想された終了時間より速く終了する可能性がある。
- 各計算機は異なる処理速度を持つ。しかし、各計算機間の通信性能は一定のものとする。

計算機数を 32 台、タスク数を 100~100000 とし、二つのスケジューリング方法を用いスケジューリングする。計算機速度、タスクサイズ、タスク到達時間は乱数により与えるものとする。

シミュレーションの結果を図 2 に示す。Sch1、Sch2 はそれぞれ一つめ、二つめのスケジューリング法を表し、max は最大処理終了時間、すなわち全体の終了時間、ave は全計算機の処理終了時間の平均を表す。Sch1-max、Sch1-ave、Sch2-max はほぼ同様の直線を描いた。

5 考察

シミュレーション結果において二つの方法を比較したとき、全体の終了時間は二つめ方法を用いた場合の法が僅かに速く終了するが、ほぼ同等の結果であった。しかしながら、二つめの方法の場合、到達タスクに対しその時点で最も速く処理が可能な計算機にタスクを割り当てる。そのため、全計算機を使用せずに処理速度の速い計算機に対しタスクが多く割り当てられ、処理速度の遅い計算機ではタスクが割り当てられず処理が行われない。具体的には、タスク数 100 のとき、約 8 台の計算機でのみ処理が行われ、残り 24 台にはタスクは割り当てられない。同様にタスク数 1000 のとき

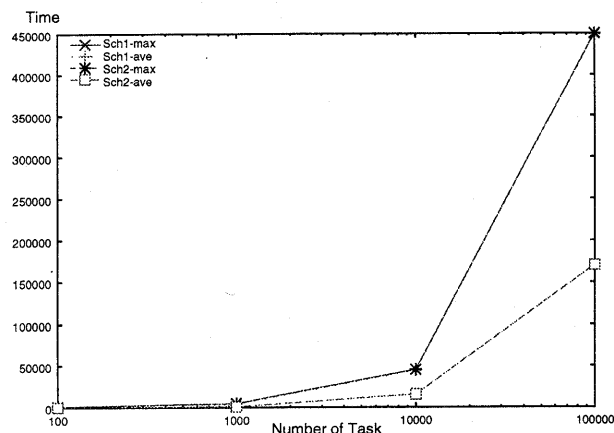


図 2: シミュレーション結果

は約 10 台、タスク数 10000 のときは約 12 台、タスク数 100000 のときは約 15 台の計算機のみでの処理が行われ、残りの計算機では処理が行われなかった。これにより、全計算機の処理終了時間の平均はひとつめの方法に比べ小さくなる。

この結果は、論文 [2] において提案されている方法の「あまりに処理速度の遅い計算機にタスクを割り当てない」という点に近い結果だと思われる。

6 おわりに

今回の提案した方法を用いてスケジューリングした場合、全ての計算機を用いるのではなく処理速度の速い計算機にタスク割り当てが集中する。これにより、例えば、過負荷がかかっている計算機の使用を避けることができる。

しかしながら、動的スケジューリングは高いシステムオーバーヘッドを伴うためスケジューリング時に静的スケジューリングに比べ処理時間がかかる。具体的には、タスク到達毎に割り当て優先度の算出が必要になるためである。

今後、このスケジューリング時の処理時間を考慮し、また、タスク間通信などのタスクモデルの拡張も視野にいれ、スケジューリング方法の発展、改良を行っていきたい。

参考文献

- [1] 菅谷 至寛, 阿曾 弘具, “非均一ネットワークにおけるタスク割り当て手法,” 情報処理学会論文誌, pp.2592-2602, Vol.41, No.9(Sep. 2000).
- [2] Xueyan Tang and Samuel T. Chanson, “Optimizing Static Job Scheduling in a Network of Heterogeneous Computers,” Proc. of 2000 Int'l Conf. on Parallel Processing, pp.373-382, Aug. 2000.