

プログラムの粗粒度分割による並列化手法

2D-8

中谷 洋司

濱田 喬

東京大学

学術情報センター

1 はじめに

近年、科学技術計算など膨大な計算量を必要とする分野では、単一のプロセッサで処理していたのでは能力的に処理が間に合わなくなっている。そこでマルチプロセッサシステムなどの並列計算機の開発が行われている。

並列プログラミングとは、複数のプロセッサによって同時に実行されるコンピュータプログラムを作成することである。並列計算機的能力を十分に引き出すためにはこのような並列プログラミングをうまく行うことが必要となる。しかしそれには経験が必要で、一般的には非常に難しいものである。

そこで、逐次的に書かれたプログラムを自動的に並列化することが考えられている。ここでは、C言語で書かれた逐次的なプログラムを依存関係をもとに解析し、粗粒度に分割して、疎結合マルチプロセッサ上で実行する方法について検討している。

実行の方法としては、管理プロセスにより、各プロセスの実行や、データの管理をすべて行わせることにより、通信を制限し、同期を取りやすくしている。

2 マルチプロセッサシステム

まずは、用いるマルチプロセッサシステムについてその特徴をまとめる。

- それぞれのプロセッサは独自に命令列を解釈し、実行する。
- 各プロセッサ間のデータ交換は、メッセージ通信方式で行われる。
- 予め分割されたプログラムを各プロセッサに割り当てて実行することにより並列実行を行う。
- プログラミング言語にはC言語を用いる。

A Method of Parallelizing Programs with Coarse Grained Division

Youji NAKATANI (University of Tokyo)
Takashi HAMADA (National Center for Science Information Systems)

3 プログラムの解析

3.1 依存解析

プログラムを並列実行するために、依存関係の解析を行いプログラムを分割する。依存関係には次のものがある。

制御依存 分岐命令等の制御構造による依存関係。実行結果により後の部分の実行の流れが変わる。

データ依存 同じ変数を違った場所で扱っている時に前後関係を保つための依存関係。フロー依存、逆依存、出力依存の三つがある。

この二つの依存関係をグラフに表したものがプログラム依存グラフ¹⁾である。これはプログラムを中間記述する方法として良く用いられる。

プログラム依存グラフの節点はプログラム中の命令文やその集まりを表し、枝は、各節点間のデータ依存や制御依存の関係を表す。プログラム依存グラフのうち枝として制御依存関係だけを表したものを制御依存グラフと呼び、データ依存だけを表したものをデータ依存グラフと呼ぶ。

3.2 プログラム分割

プログラム依存グラフを用いてプログラムを並列実行の単位であるプロセスに分割していく。そのために、以下の方針をとる。

- 各プロセス間のデータ交換を少なくする。
- プロセス間にできるだけ多くの並列性を持たせる。
- 各プロセスの大きさをできるだけ等しくし、均等にプロセッサに割り振れるようにする。

そのためには、まず各命令毎のプログラム依存グラフを作成し、以下の方法で命令をプロセスにまとめていく。

- 制御依存グラフをもとに、逐次的にしか実行できない部分の命令を一つにまとめる。

- データ依存グラフを用いて、データ依存により、実行順序が定まっている命令をその順序で一つにまとめる。
- 並列性が存在するプロセスでも、小さすぎて、立ち上げ時のオーバーヘッドの方が時間のかかるようなプロセスを一つにまとめる。

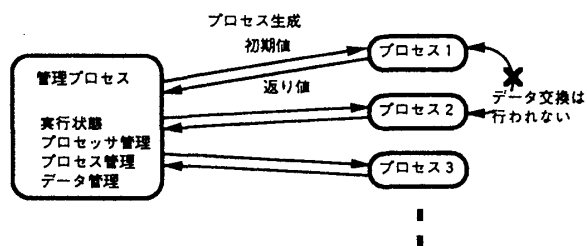


図 1: プロセスの実行

4 並列実行

次にプロセスに分割されたプログラムを並列実行する方法について述べる。

4.1 実行状態

各プロセスについて実行状態²⁾を設定する。実行状態というのは各プロセス毎に設定され、そのプロセスが実行可能となった時に真の値をとるものである。実行状態が真となるのは以下の二つの条件が満たされた時である。

- そのプロセスの実行を強いる制御状態が真となること。
- そのプロセスがデータ依存しているプロセスが既に実行されたか、この先実行されることがないと判明すること。

この実行状態を用いて、以下のようにして各プロセスが実行されていく。

1. 全てのプロセスについての実行状態を調べる。
2. プロセッサに実行可能なプロセスを割り当てて実行する。
3. 実行中のプロセスの終了を待つ。
4. プロセスの終了により、実行状態を更新し、新たに実行可能となったプロセスを捜し、それを実行する。

4.2 管理プロセス

並列実行の方法としては、特別のプロセスを一つ走らせて、それで全てのプロセスを管理することにする。その特別のプロセスを管理プロセスと呼ぶことにする。管理プロセスの行う作業は、以下の通りである。

- プログラム中で使われる全てのデータの値を管理する。

- 実行可能となったプロセスをプロセッサに割り当てて実行する。
- 終了したプロセスからデータを受け取り、新しく実行可能となるプロセスを捜す。

このようにデータ、プロセスの管理を一箇所に集めることにより、一度実行されたプロセスは、その後終了まで同期の必要がなく独自に実行される。

5 おわりに

以上、メッセージ通信方式のマルチプロセッサシステムにおいてプログラムを粗粒度分割することにより、並列実行する方法について述べた。

効率良くプログラムを並列実行させるためには、うまくプロセスに分割することが必要である。今後、どのようにすればうまく分割することができるかについて検討を進めていく。

参考文献

- [1] J. Ferrante, K. J. Ottenstein, and J. D. Warren, "The program dependence graph and its use in optimization", *ACM Trans. Programming Languages and Systems*, vol.9, no.3, pp. 319-349, July 1987.
- [2] Milind Girkar and Constantine D. Polychronopoulos, Member, IEEE, "Automatic extraction of functional parallelism from ordinary programs", *IEEE Trans. Comput.*, vol.3, pp166-178, Mar. 1992.