

並列計算機 (トランスピュータ・システム) における 6B-1 動作モニタの開発とその有効性

宮下 浩 作道 直樹 梅尾 博司

{miyasita, sakudo, umeo}@umeolab.osakac.ac.jp

大阪電気通信大学 工学部

1. はじめに

本稿では、並列計算機 (トランスピュータ・システム) におけるプロセッサの動作モニタの製作とその有効性について報告する。現在、トランスピュータのモニタやチューニングツールなど並列計算機のためのソフトウェアツールはほとんど整備されていないため、トランスピュータ上でのプログラム開発環境は快適であるとは言えない。そこで本研究ではトランスピュータの開発環境を改善とすることを目的とし、トランスピュータの状態とリンクを調べるモニタを製作した。本モニタシステムを使用することによりネットワーク上の各トランスピュータの効率、リンクの使用状況のモニタとしての使用が可能である。

2. モニタ・システム

2.1 モニタ

モニタはリンクモニタと状態モニタの2つからなり、それぞれ独立に動作する。トランスピュータ・システムの各プロセッサをモニタするにはユーザー・プログラムとモニタを並列に走らせる必要がある。

2.1.1 リンクモニタ

複数のトランスピュータを使ったシステムではシステム全体の実行効率上げることは困難である。トランスピュータは通信によって同期をとるため、各々のプロセッサのリンクの使用状況を把握することができれば、各プロセッサの通信待ち時間を知ることができる。ネットワークシステム全体の効率は各プロセッサの通信のオーバーヘッドが影響する。

リンクモニタは各プロセッサでユーザープログラムと並列に実行することにより、トランスピュータの4本のリンクの入出力をモニタする。

2.1.2 状態モニタ

プロセッサの状態はアイドル状態と実行状態の2つの状態がある。

アイドル状態とは、一般にプロセッサが通信待ちの状態のことをいい、通常プロセッサがアイドル状態であるとき、そのプロセッサは何も仕事をするとはできない。ネットワーク上の各プロセッサの効率はアイドル状態の時間に依存して悪くなり、結果としてシステム全体の効率低下をまねく。

実行状態とはアイドル状態以外の状態をいい、逐次型の計算機では基本的に常にこの状態である。

状態モニタはプロセッサの状態をモニタするプログラムであり、プロセッサ上で動作する全てのプロセスがアイドル状態であるか実行状態かをモニタする。

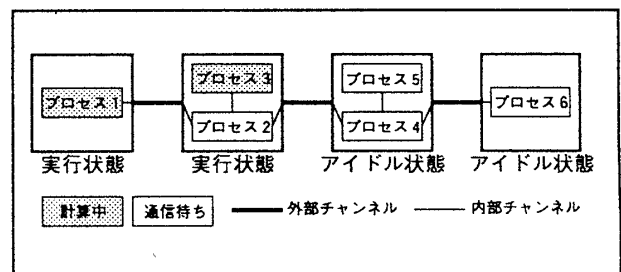


図1 実行状態とアイドル状態

2.2 モニタプログラム

本モニタを使用するには以下のプログラムを実行する必要がある。

2.2.1 付加プログラム

付加プログラムは、ユーザープログラムとモニタ部分を自動的にソースに付加するプログラムである。ユーザープログラムとモニタ部分は並列に実行するように付加される。ユーザーは付加プログラムを実行後、一度コンパイルを行ってから実行する必要がある。

A Development of Performance Monitor for Transputer Systems

Hiroshi Miyashita, Naoki Sakudo and Hiroshi Umeo
Osaka Electro-Communication University

2.2.2 回収プログラム

モニタを付加しているプログラムを実行すると、そのプログラムは各プロセッサのメモリの1部にモニタのデータを蓄える。そのため、回収プログラムによりデータを回収しファイル化する必要がある。

データは各プロセッサごとにメモリの上部100Kバイトの場所に蓄えられており、このデータをルートのプロセッサに転送することにより回収する。ルートのプロセッサはデータが送られてくると、ファイルに書き込み、次のデータを待つ。

2.2.3 表示プログラム

表示プログラムは、回収プログラムでファイル化

したデータを読み込み、解析し、画面に表示するプログラムである。リンクの状態とプロセッサの状態を表示する。

図2, 3はデータを左から右に送るプログラムを実行したときの画面図である(右端とその1段下の左端のプロセッサはつながっている)。四角がプロセッサの状態を、線がリンクの状態を表わしている。プロセッサの状態は実行状態であれば赤く表示する。リンクの状態は使用状態であるとき水色で表示する。

2.2.4 操作手順

操作手順は次のようになる。

付加プログラムの実行

コンパイル

実行

回収プログラムの実行

表示プログラムの実行

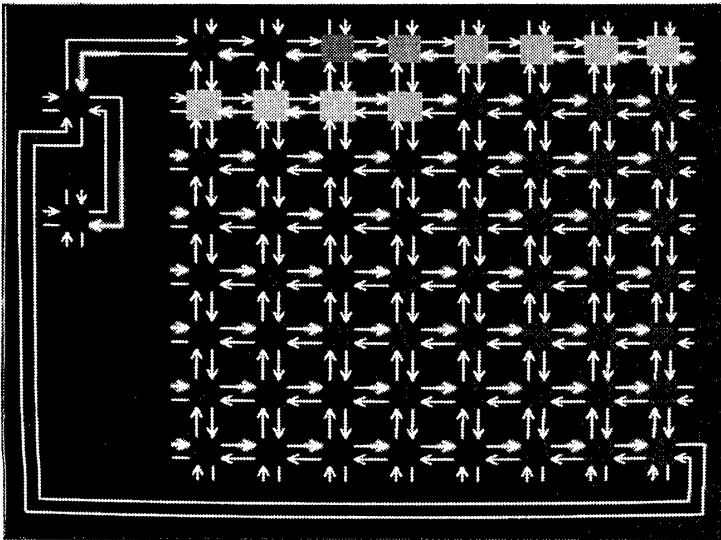


図2

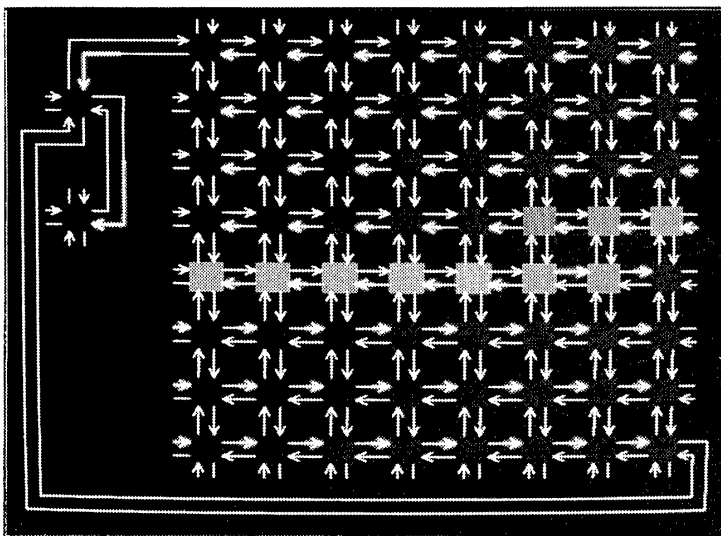


図3

3. おわりに

このモニタを使用することにより、各プロセッサの動作がかなり具体的にわかるようになり、その結果からチューニングやデバッグに役立てることが可能である。

参考文献

- [1] D.A.P.Mitchell, J.A.Thompson, G.A.Manson; "Inside The Transputer" Blackwell Scientific Publications, pp.1-80, pp.87-223, 1990
- [2] INMOS; "Transputer Development System Second Edition" INMOS, pp.229-239, 1988
- [3] コンカレントシステムズ; "TET (Transputer Efficiency Tester)" コンカレントシステムズ, pp.11-12, 1991
- [4] R.S.Cok; "Parallel Programs for the Transputer", Prentice Hall, (1991), (梅尾, 他(訳); "Transputer/ Occamによる並列プログラミング入門", 共立出版, (1993年, 10月予定))