データ依存性を考慮した動的負荷分散法の検討

指導教官 市川 周一　　学籍番号 963734 山下 真史

1 分散計算システムと既存の研究

分散計算システムとはネットワークで接続された複数のプロセッサ間で計算を分担し実行するシステムである．既存のシステムとして Ninf や NetSolve などがある．Ninf[1] では，分割されたジョブを投入するとメタサーバと呼ばれるサーバが負荷分散を行う．メタサーバはシステム内のデータの依存関係に基づいてスケジューリングを行うが，現在のところ処理時間の最小化は行っていない．処理時間を最小化する負荷分散法の研究が急務である．

分散計算環境にキューイング理論を適用してさまざまな研究が行われている．例えば研究 [2] では，ノードの負荷が閾値よりも大きければ小さいノードに実行させることでシステム全体のスループットを最大化する．これらの手法ではジョブ相互は独立であるとして平均反応時間を評価しているが，分散計算システムでは依存関係のある複数のジョブ全てを処理しなければならない．図 1 のような依存関係のあるジョブを分散して処理する場合，$JS3$ の開始は $JP1$ ~ JPD全ての実行終了後となる．即ち $JP1$ ~ JPDのうち最も実行が遅れたものにより全体の実行時間が決定される．$JS1$ ~ $JS3$ の実行時間を最小化するには依存性を考慮したスケジューリングが必要になる．

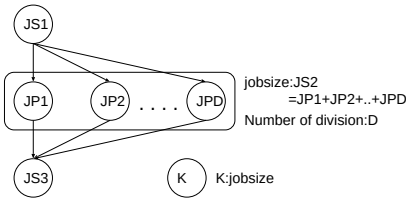


図 1: 依存関係のあるジョブ

2 実験方法

本研究では単純化のため以下のような理想的モデルを扱う．

- プロセッサ m 台が均一なネットワークで接続されている．
- 各プロセッサを $P_i(i = 1, ..., m)$ とあらわす．
- 各プロセッサの処理しているジョブ数は一定時間 $u[sec]$ 毎に入手できる．
- プロセッサのスケジューリングはタイムスライスを無限小としたラウンドロビンとし，プロセッサはジョブの処理のみを行う．
- 到着するジョブのサイズは平均$\overline{w} = 1[unit]$ の指数分布に従う．
- P_iの処理速度は $S_i[unit/sec]$ とする．
- P_iに投入されるジョブは到着率λ $_i[jobs/sec]$ のポアソン分布とする．
- P_iに投入されたジョブは P_iのジョブ数が全体の平均ジョブ数より多いときのみ最もジョブ数の少ないプロセッサに割り当てられる．
- P_iで実行が開始されたジョブは最後まで P_iで実行される．
- 1 つのジョブ転送先を決定するのに $0.05[sec]$ かかるものとする．
- 投入されたジョブはサイズにかかわらず $0.1[sec]$ の通信時間がかかるものとする．
- 実行結果の転送にも $0.1[sec]$ かかるものとする．
- システム内の各プロセッサのジョブ到着率の総和と処理速度の総和の比をシステム負荷と定義する．
  システム負荷Λ $= \overline{w} * \sum_{i=1}^{m} \lambda_i / \sum_{i=1}^{m} S_i$

このモデルに入力として図 1 のような依存関係のあるジョブを投入し，以下の条件でそれぞれ応答時間を測定する．

- システム負荷Λ を変える．(1)
- 並列処理可能部分の分割数 DをΛ ごとに変える．また各プロセッサのジョブ数の入手間隔 $u[sec]$ も変化させる．(2)

3 実験結果

(1) では$m = 10, S_i = 1.0(i = 1, ..., m), JS1 = JS3 = 1, JS2 = 8, D = 4, u = 0.01$ としてジョブを投入した．結果を図 2 に示す．Λ が大きくなるにつれ応答時間も指数的に大きくなっている．

(2) では$m = 10, S_i = 1.0(i = 1, ..., m), JS1 = JS3 = 1, JS2 = 100, u = 5.0$ とし，Λ ごとに Dを変化させ並列処理実行可能部分の分散の有効性を調べた．ここでは $JPi = JS2/D$で一定とした．図 3(a) の場合ではどのΛ に関しても $D = 9$ ぐらいで最小の応答時間が得られ，$D = 10$ では応答時間が増大している．$D = 9$ のときは $D = 1$ に対して 2.8 倍から 5.5 倍速度が向上している．その度合はΛ が小さい程大きい．Λ が小さいと投入したジョブ以外のジョブが少ないため，実行時間がその分短縮されるからである．

図 3(b) では (a) の $u = 5.0$ を $u = 10.0$ に変化させた結果を示している．こちらも (a) と同じような結果が出ているが，応答時間が最小になる Dが (a) の場合に比べわずかに小さい．$D = 8$ で応答時間が最小になっている．それ以上では応答時間が増大している．また全体的に応答時間が (a) に比べわずかに大きい．これは u が大きくなったことにより，負荷分散の効果が (a) よりも小さくなったためである．u が小さい場合は各プロセッサの負荷情報が短時間毎に更新されるため，大きい場合よりもよりよい負荷分散が行える．

この 2 つの結果より，最適な分割でシステムの性能が低負荷時で 5 倍程度，高負荷時でも 3 倍程度向上することが分かった．しかしやみくもに分割数を増やしても逆効果であることも分かった．原因としては分割数が多くなるとシステム内の高負荷なプロセッサにジョブを分配しなければならず，そのプロセッサの実行が遅れることにより全体の実行時間が長くなってしまうということが考えられる．

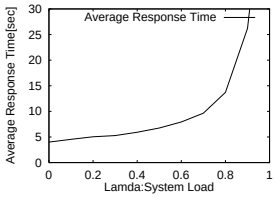
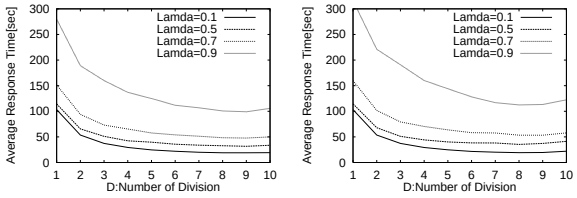


図 2: システム負荷と応答時間



(a)$u = 5.0[sec]$　　(b)$u = 10.0[sec]$

図 3: ジョブの分割数と応答時間

4 今後の課題

本研究により依存関係のあるジョブに対して既存の研究でも 3 倍から 5 倍の性能向上が得られることが分かった．しかし最適な分割数 Dや最適な分割法 (JPi の値) を予測しスケジューリングしなければこの効果は得られない．既存の研究ではこの点については考えていないので今後の課題として本研究より複雑なモデルで依存関係のあるジョブの最適な負荷分散法を考えたい．

参考文献

[1] 中田秀基, 高木浩光, 松岡 聡, 長嶋雲兵, 佐藤三久, 関口智嗣, “Ninf による広域分散並列計算”, 並列処理シンポジウム JSPP '97, pp.281-288(1997).

[2] 小林真也, 小川智之, 渡部 尚, “マルチコンピュータシステムにおける自律的負荷分散方式”, 電子情報通信学会論文誌 D-I, Vol.179-D-I, No.11, pp.903-915 (1996).