1996年度日本オペレーションズ・リサーチ学会
春季研究発表会

資源要求のある待ち行列モデル — 2つの資源がある場合

01302440 東京工業大学 数理・計算科学専攻 *高橋幸雄 TAKAHASHI Yukio
東京工業大学 情報科学科 梅原元 UMEHARA Gen
(株) 日立製作所 システム開発研究所 木下俊之 KINOSHITA Toshiyuki

1. はじめに

計算機システムにおける一連の計算処理の中には、ファイルのデータ項目の更新のように、ひとつのジョブがそれをアクセスしている間は、ほかのジョブからのアクセスを禁止するケースが多く存在する。このような対象を、ここでは共有資源、または単に資源、と呼ぶ。この資源へのアクセスの競合は、計算機の性能に大きな影響を与える。しかし従来の待ち行列ネットワークでは、この資源を厳密に扱うことができなかった。

そこで [1] では、資源の概念を取り入れたセントラルサーバモデルを導入し、マルコフ連鎖の定常状態確率を数値的に求めることにより、資源の計算機性能に及ぼす影響を解析した。

ただし [1] における解析では、資源の種類は 1 つに限られていた。そこで、ここでは資源の種類を 2 種類に拡張したモデルを解析し、性能ボトルネックとなっている資源を 2 種類の資源に分割したときの効果などについて検討した。

2. モデル

システムは、通常のセントラルサーバモデルと同様に、1 つの CPU ノードと複数個の (ここでは 2 つの) Disk ノード、さらに各資源に対応した 2 本の資源待ち行列から成る (図 1)。ジョブ (客) の数は一定で N とする。

各ジョブは資源要求に関する状態を確率的に変化させながら、各ノードの間を確率的に動き回る。各ノードでのサービス時間は指数分布にしたがうものとし、そのサービス率は簡単のため、ノードごとに決まっているものとする。

資源要求に関して、各ジョブはつぎのような状態をもつ。

1. 資源を必要としない
2. 資源 A を必要とする
3. 資源 B を必要とする
4. 資源 A と B の両方を必要とする

この資源要求に関する状態は、CPU ノードでのサービス終了直後にマルコフ的に変化する。資源要求の状態が変化した場合は、どのような変化であろうとも、その時点で保持していた資源は、一旦、すべて解放する。なお、同じ資源要求状態が続く場合については、状態が変化せず継続する場合と、状態変化が起きたけれどもたまたま同じ状態になったという場合を区別する。前者の場合は資源は保持したままでつぎのノードへ進むが、後者の場合は保持している資源を一旦解放し、あらためて資源獲得の動作を行う。

CPU ノードでのサービス終了後、つぎにどのノードへ進むかは、そのときの新しい資源要求状態と資源保持状態に依存する。

1. 資源を必要としない場合、定められた確率で Disk ノード 1、Disk ノード 2 あるいは CPU ノードへ進む。CPU ノードへ進む場合は、この時点でそのジョブは終了してシステムの外へ退去し、新たなジョブが替わりに外から入ってくると解釈する。

2. 資源 A を必要とする場合、まず定められた確率で行き先ノードとして Disk 1 または Disk 2 を選ぶ。そのときすでに資源 A を保持しているれば、そのまま行き先ノードへ進む。資源 A を保持していないときは、一旦、資源-A 待ち行列に加わる。このとき他に資源 A を保持しているジョブがなければ、ただちに資源 A を獲得して行き先ノードへ進む。しかし他のジョブが資源 A を保持しているときは、資源 A が利用可能となるまで待ち行列に並んで待つ。

3. 資源 B を必要とする場合も上と同様である。

4. 資源 A と B の両方を必要とする場合、やはり定められた確率で Disk 1 と Disk 2 の中から行き先ノードを選択する。両方の資源をすでに保持しているならば、そのまま行き先ノードへ進む。保持していなければ (このときは資源要求状態変化のときの仮定から、どちらの資源も保持していない) まず資源-A 待ち行列へ進む。そこで資源 A を獲得した後、さらに資源-B 待ち行列へ進み、そこで資源 B も獲得できたならば、それから行き先ノードへと進む。(この手順は、資源獲得によるデッドロックを避けるためである。)

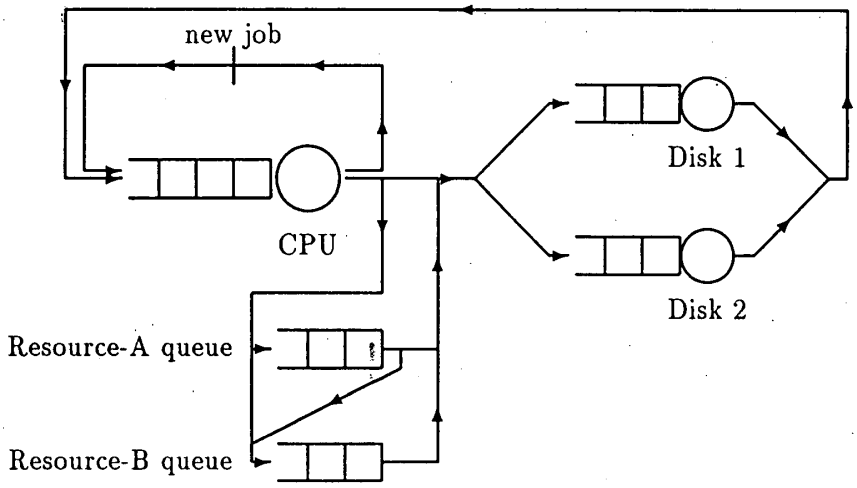

Figure 1: 資源要求のあるセントラルサーバモデル — 資源が 2 種類の場合

サービス時間、資源要求状態変化、ノード選択などの確率的事象は互いに独立であるものとする。また各待ち行列では先着順で処理される。

3. 数値解析例

ここではあるパラメータセットを用いて、ひとつの資源としていたファイルグループを 2 つの資源に分割したときの効果を数値的に解析した結果を紹介する。ここでは、ジョブのライフタイムの間に CPU ノードを通る平均回数が 5 回、資源要求状態の継続する平均回数が 2 回、資源 A と資源 B の要求が互いに独立で同じ確率で発生するようにパラメータを選んである。

解かなければならないマルコフ連鎖の状態数は、ジョブの多重度 (ジョブ数) N につれて増大する。

N	2	4	8	12
状態数	697	1,768	54,661	1,170,206

結果は図 2, 3 の通り。横軸は資源を取りにいく確率 (資源が 2 つの場合は少なくともひとつを取りにいく確率、ここではこれが前の状態に依存しないようにパラメータが選んである) であり、実線が資源がひとつの場合、点線が資源が 2 つの場合である。

これらのグラフから、CPU を効率的に使おうとすると、ジョブの多重度をあげなければならず、そのとき資源獲得確率が大きいと、資源獲得がボトルネックを発生させることが分かる。そして、資源獲得がボトルネックとなるときには、この例のように資源を分割することは効果的である。

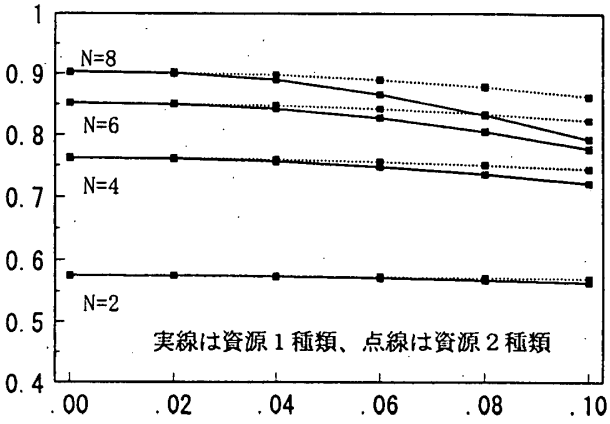

Figure 2: CPU 利用率

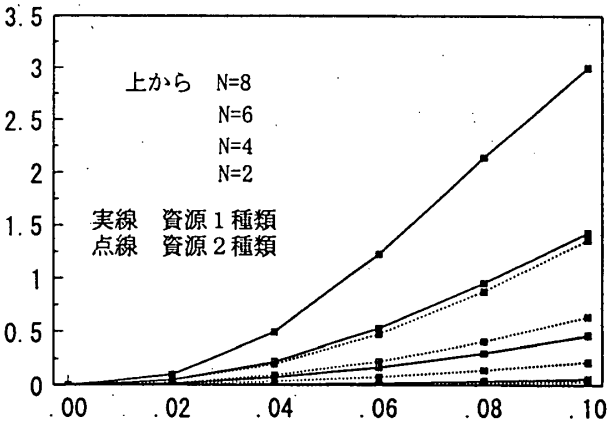

Figure 3: 平均資源待ちジョブ数

[1] 木下俊之、高橋幸雄「資源要求のある待ち行列網のモデル化の一提案」情報処理学会第 51 回全国大会 講演論文集 (4), pp.3–4, 5L–2 (1995.9).