キューを利用した並列計算モデル

3G－2

北橋 洋三郎　山口 文彦　中西 正和
慶應義塾大学大学院 理工学研究科 計算機科学専攻

1. はじめに

計算機の歴史の初期から、スタックマシンと呼ばれるスタックを利用した計算機アーキテクチャが広く用いられてきた。しかしながら、スタックを用いた実行方式は計算の逐次的実行を強いる性質を持っており、パイプライン実行などの並列処理には向いていない。この問題を解決するために、キューマシンと呼ばれる新しい計算機アーキテクチャが考案された。キューマシンアーキテクチャでは、スタックの代わりにキューを利用することにより計算の並列実行を可能にしたとされる [1]。

本研究では、このキューマシンアーキテクチャをキュー機械モデルと呼ぶ計算モデルに抽象化し、キューを用いた計算の体系化を試みる。その上で、構成したモデルをもとに計算の効率に関する数学的な考察を行い、キューを使って効率良く並列計算を行うための普遍的な条件を導く。

2. キューマシン実行方式

キューマシンはキューの先頭からオペランドを取り出し、その計算結果をキューの末尾に入れる。この手順で評価できるように式をコンパイルするには、構文木を（右から左へ）幅優先で巡回し、訪れた順番と逆順で各ノードに対応する命令を出力すればよい（図1左）。キューマシンの実行をトレースしたものを図1右に示す。

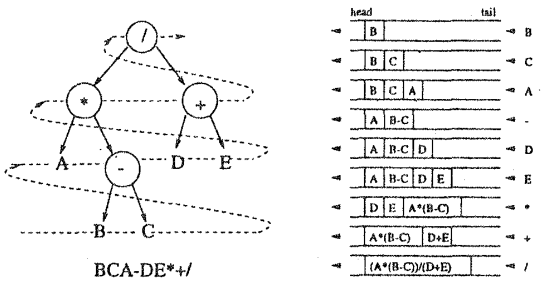

図 1: (左) キューマシンの機械語の生成
(右) キューマシンの実行トレース

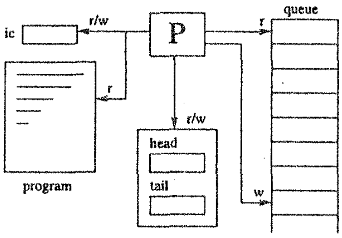

図 2: 逐次型キュー機械モデル $\mathcal{SM}$

3. 逐次型キュー機械モデル

逐次型キュー機械 $\mathcal{SM}$ とは、システム $(Q, I, \Sigma, \delta, q_0, q_F)$ をいう。Q は状態の有限集合で、q_0, q_1, q_2, q_3, q_F の5つの状態から成る。このうち q_0 を初期状態、q_F を停止状態と呼ぶ。I は命令の有限集合で、いくつかの単位実行命令と `load,jmp, jmpif, eval, return, halt` を要素とする。Σ はアルファベットで、機械で扱うプログラム及びデータは全て Σ^* の元とする。δ は動作関数を表す。

$\mathcal{SM}$ は1個の有限制御部（プロセッサ）と無限長のキュー領域を備えており、その他にプログラム領域と3個のレジスタ ic, $head$, $tail$ を持つ（図2）。プログラムを構成する命令にはそれぞれ番地が割り振られており、現在読んでいる命令の番地が ic (instruction counter) に保持される。また、キュー領域にも番地が付けられているが、プロセッサがランダムにアクセスすることはできず、$head$ の指す番地からの読み出しと $tail$ の指す番地への書き込みだけが許される。

$\mathcal{SM}$ は $ic = 0$、$head = tail = 0$ で初期状態 q_0 から計算を開始する。$\mathcal{SM}$ の動作関数 δ は、現在の状態、命令、及びキューから取り出すデータをもとにして、次状態、ic の更新値、$head$ の更新値、及びキューに格納するデータを与える。

Queue-Oriented Parallel Computational Model
Yozaburo KITAHASHI, Fumihiko YAMAGUCHI, Masakazu NAKANISHI
{ kita, yamagu, czl } @nak.ics.keio.ac.jp
Department of Computer Science, Faculty of Science and Technology, Keio University 3-14-1 Hiyoshi, Kohoku-ku, Yokohama, Kanagawa 223-8522, Japan

4. 並列型キュー機械モデル

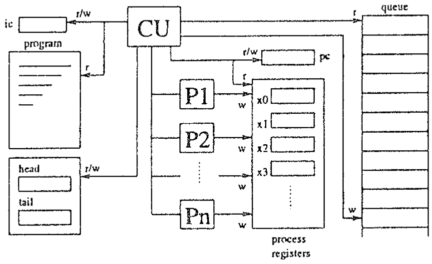

図 3: 並列型キュー機械モデル $\mathcal{PM}$

並列型キュー機械 $\mathcal{PM}$ とは、システム $(Q, I, n, \Sigma, \delta, q_0, q_F)$ をいう。I は、逐次型の命令に join 命令を追加したものである。n は CU を除くプロセッサの数を表し、2 以上の整数とする。

$\mathcal{PM}$ は処理装置として 1 個の中央制御部 CU と n 個のプロセッサ P1 〜 Pn を持ち、記憶領域として加算無限個のプロセスレジスタ $(x_0, x_1, \cdots)$ とレジスタ pc が追加されている（図 3）。pc（process counter）にはプロセスレジスタが現在何番まで使用されているかを示す値が保持される。

I に属する命令のうち、単位実行命令は各プロセッサに渡されて CU の動作とは独立に処理されるため、その間 CU は別の動作を続けることができるほか、複数のプロセッサが並行して処理を行うことが可能である。それ以外の命令については CU が単独で処理を行う。キューへの読み書きは CU のみが許可され、各プロセッサは CU から渡された仕事（プロセス）を処理して結果をプロセスレジスタに書き込むという動作をそれぞれ独立に繰り返す。

$\mathcal{PM}$ は $ic = 0$、$head = tail = 0$、初期状態 q_0 で全てのプロセッサが free の状態から計算を開始する。$\mathcal{PM}$ の動作関数 δ は、現状態、命令、busy なプロセッサの数、及びキューから取り出すデータをもとにして、次状態、ic の更新値、x_{pc} に書き込むデータ、pc の更新値、$head$ の更新値、及びキューに格納するデータを与える。

5. 時間量

プログラム P に対する様相の列の長さ −1、すなわち計算に要したステップ数を P の時間量と呼ぶ。ここでは議論をより一般化させ、単位実行命令の計算には j ステップ、それ以外の命令には 1 ステップを要するものとする。

いま、逐次型キュー機械 $\mathcal{SM}$ のプログラム P が与えられたとする。P は N 個の命令から成り、N_s 個の単位実行命令を含んでいる。このとき、P の $\mathcal{SM}$ における時間量は次式で与えられる。

$$N + (j-1)N_s \tag{1}$$

プログラム P を並列型キュー機械 $\mathcal{PM}$ のプログラム P' に書き換えると、P' は d 個の join 命令を含み、命令の数は全部で $N+d$ 個となる。ここで d は、P を表す構文木の深さである。

並列型機械では単位実行命令をプロセッサに渡すだけでよいので、これらは全て 1 ステップ相当と見なすことができる。しかし、全てのプロセッサが busy だったり、join 命令を実行したいときにプロセッサの計算がまだ終わっていない場合、CU は待ちを生じる。待ちを全く生じないとすれば P' の時間量は $N+d$ である。一方、考え得る最大の待ちを含む時間量は以下のように求まる。

$$N + dj \qquad (j \le n,\ N_s > d) \tag{2}$$

$$N + d + (j-1)N_s \qquad (j \le n,\ N_s \le d) \tag{3}$$

$$N + dj + (j-n)\cdot\frac{N_s}{n} \qquad (j > n,\ N_s > d) \tag{4}$$

$$N + d + \left(j + \frac{j}{n} - 2\right)N_s \qquad (j > n,\ N_s \le d) \tag{5}$$

以上より、(2) 〜 (5) で与えられた並列計算の（最大）時間量が、(1) で与えられた逐次計算の時間量で押えられるための条件は、

$$N_s > d \quad \text{かつ} \quad \frac{N_s}{N_s - d} < j \le n \tag{6}$$

$$\text{あるいは} \qquad N_s > \frac{n}{n-1}d \quad \text{かつ} \quad j > n \tag{7}$$

と結論される。この結論は、あるプログラムが逐次型モデルで計算するより並列型モデルで計算した方が時間量が小さくなるための充分条件である。

6. むすび

本研究では、キューマシン実行方式のモデルとして、FIFO 方式でデータを出し入れすることによって並列的な計算を可能とした機械モデルを考案した。本稿では割愛したが、このモデルは Turing 等価であることが証明されている。また、このモデルをもとにキューを使って効率良く並列計算を行なうための条件式を導出した。この式の解釈は、まさにキューマシン実行方式の計算効率を左右する様々な要因間のトレードオフを表現したものであった。

参考文献

[1] 前田敦司. キューマシン計算モデル 新計算モデルとその並列関数型言語への応用. 博士論文, 慶應義塾大学 理工学研究科, Mar. 1997.

[2] Douglas S. Bridges. *Computability A Mathematical Sketchbook*. Springer-Verlag, 1994.