シミュレーション言語GPSSの並列処理におけるモデル分割法*

4L－4

守屋 充雄　高井 峰生　成田 誠之助†
早稲田大学理工学部‡

1　はじめに

近年、大規模な通信システムや計算機システム等をシミュレートする必要が生じてきた。これらのシステムのシミュレーションは計算量が膨大になり、実用的な時間でシミュレートすることが困難である。そこで、シミュレータを並列化することで時間の短縮を図る研究がなされてきた。[1]

しかし、ユーザーが並列計算を行うプログラムを記述することは困難である。離散事象シミュレーションは幾つかのシミュレーション専用言語によってプログラミングされることが多いが、そこで、それらの言語を内部処理的に並列化すればユーザーが並列処理を意識すること無く処理の高速化が図れる。

本稿では、シミュレーション言語の中で代表的なGPSS(General Purpose Simulation System) を用いて、GPSS で記述されたモデルを並列処理するために、モデルを適切な構成要素に分割し、並列性を抽出する方法について述べる。

2　GPSS の内部処理

GPSS はトランザクション (以下 Xact と表す) 中心のモデル構成概念を用いている。Xact とは 時間の経過とともにシステムを動いていく「もの」である。GPSS で組まれたプログラムは、Xact がシステムの各要素 (facility, storage など) に影響を及ぼし、Xact の属性値自身を変化させながら、どのような経路を進行するかを記述したものである。

実際のプログラムは、Xact の流れる道をブロック命令を用いて記述する。ブロック命令（以下ブロックと表す）は、Xact の関連状態（設備の使用開始・終了、使用時間など）を指定する命令である。図1にプログラムの例を上げる。図1は、一定時間ごとに発生する Xact が設備 Fac1 もしくは Fac2 を経由して消滅するモデルを表している。

次に GPSS のプログラムがどのように内部処理されているかについて述べる。[2] GPSS では基本的に、GENERATE ブロックで発生した Xact を仮想時刻の経過と共に動かしていく。具体的には、任意の仮想時刻 T において動ける Xact を進行でき得る限り進行 (Xact が現在いるブロックの場所から処理をしていく）させる。その Xact が進行できなくなると、時刻 T で動ける他の Xact を探し、それが進行できなくなるまで処理することを繰り返す。

時刻 T において、どの Xact もモデル内で動けなければ仮想時刻の更新を行い、新しい仮想時刻で動き得る Xact が再び進行できなくなるまで処理する。これを繰り返してシミュレーションを行う。

	ブロック命令		動作説明
	GENERATE	2	Xact の発生 (2毎)
	TERMINATE	.5,Fac2	Xact の確率分岐
Fac1	QUEUE	Line1	待行列へ入る
	SEIZE	Line1	設備の使用開始
	DEPART	Line1	待行列を出る
	ADVANCE	3	時間消費 (3)
	RELEASE	Line1	設備の使用終了
	TERMINATE		Xact の消滅
Fac2	QUEUE	Line2	
	: （Fac1 と同様）		
	TERMINATE		

図1：GPSS のプログラム A

3　モデルの分割

モデル中に同一仮想時刻に複数個の Xact が同時に進行可能な場合、さまざまなブロックでそれぞれの Xact の処理が行われていることが考えられる。この事は、各ブロックで同時に Xact の処理が可能、つまり平行処理が行えることを示している。図1で言うと、発生・消滅などの各ブロックでの処理が同一仮想時刻で可能であるということである。

これらの各ブロックを並列計算機の処理要素（以下 PE）にマッピングすれば並列シミュレーションが行

*Model Decomposing Method for Parallelizing GPSS
†Takao Moriya, Mineo Takai, Seinosuke Narita
‡School of Science and Engineering, Waseda University

える。

ところが、すべてのブロックが独立して処理可能とは限らない。各ブロック間には依存関係があるものもあり、これを考慮しなくてはならない。

3.1 ブロック命令間の依存関係

GPSS にはモデルを構成する各要素（物理プロセス；以下 PP）に対応して、その性質を表す attribute（属性）が定義されている。モデルの状態の記述（モデルの処理）はこの独立した状態変数である attribute の値でなされる。attribute は、各ブロックで Xact を処理する過程において随時引用され、処理の対象値になる。attribute の具体例として、"facility（設備）の使用状態" などがあげられる。

ここで、同一の attribute に影響を及ぼすブロック群がある場合、これらは複数個集まって初めて処理が可能である。つまり、同一 attribute に何らかの関係をもつブロック群は分割不可能であり、依存関係があるといえる。

図 1 の例を用いると、"Line1 の使用・未使用" に対応する attribute に関係しているブロック群 {QUEUE, SEIZE, RELEASE} には依存関係があると言える。逆に、GENERATE ブロックの様に 1 つのブロックだけで処理が可能で、依存関係が無いブロックもある。

3.2 モデル分割の方針

依存関係のあるブロック群ごとに分割していくには、各ブロックが処理する際に引用する attribute を知る必要がある。

各ブロックはブロックの種類ごとに処理する内容が決まっており、必然的に引用される attribute の種類も決まっている。また、ブロックは普通いくつかのパラメータを有しており、このパラメータとブロックの種類からそのブロックに用いられる attribute を知ることができる。

この様にして各ブロックを解析することでそれぞれのブロックに必要な attribute を知り、依存関係の有無で分割の可否を判断する。

図 1 を用いて具体的な分割の方針を示す。SEIZE, RELEASE, QUEUE の各ブロックは、それぞれが処理時に設備 X の使用・未使用を表す種類の attributre (F-X とする) を必要とする。図 1 では 3 つのブロックとも "Line1" のパラメータを持っているので "F-Line1" の attribute を処理時に必要となり、3 つのブロックは分割不可能であることが判る。

同様に、QUEUE, DEPART の各ブロックは、それぞれが待行列 Y の長さを表す種類の attribute (Q-Y) を必要とし、図 1 の場合パラメータに "Line1" を持っているので "Q-Line1" を atttribute に持ち、この 2 つのブロックも分割不可能であることが判る。

ここで、ブロック QUEUE が "FAC-Line1" と "QUE-Line1" のどちらにも依存関係があるので、結局これら 4 つ {QUEUE, SEIZE, DEPART, RELEASE} のブロックは分割不可能であることが判る。

以上のように、ブロックを解析することによって、依存関係のあるブロック群ごとに分割、つまりモデルを PP ごとに分割する事ができた。これらの分割された PP は、それぞれで処理が可能であり並列処理が実現できる。

4 モデル分割の考察

上記の分割方針で自動的にプログラムを解析・分割するプログラムを作成した。これより、図 1 で用いたプログラムを解析した結果 7 つの PP に分割された。

すなわち、図 1 のモデルには 7 つの PE に分割できるだけの並列性を持っていると言える。このことは、単にモデルの並列性を抽出するだけでなく、モデルの持つ空間的な並列性の限界も算出できていることを意味する。

つまり、実際にモデルを並列処理する段階での最大限の効果が、並列処理前から予測できると言うことである。

5 おわりに

本発表では、モデルを解析し PP ごとに分割する手法を述べた。

この結果、実際にモデルを並列処理する段階での最大限の効果が、並列処理前からある程度予測できるようになった。 ただし、ブロック命令間の結合関係によっては分割できても並列効果が上がらないことも考えられる。

今後は、ブロック間の結合関係について並列効果が期待できる条件を究明していきたい。

参考文献

[1] R.M.Fujimoto. Parallel Discrete Event Simulation. Communications of ACM, vol.33, No.10, October 1990.

[2] 中西俊男 "コンピュータシミュレーション" 近代科学社 1977