数理解析研究所講究録
第564巻 1985年 42-58

分散型待ち行列網シミュレータD－SSQの処理能力解析

大阪大学 工学部　渡辺 尚 (Takashi Watanabe)
佐藤 圭 (Kei Sato)
中西 暉 (Hikaru Nakanishi)
真田 英彦(Hidehiko Sanada)
手塚 慶一(Yosikazu Tezuka)

1．まえがき

待ち行列網の解析手段の一つである待ち行列網シミュレーションは、主として大型計算機上での逐次処理によって行なわれている。しかし、モデルが複雑化したりサンプル数を増大させようとすると、処理速度、記憶容量、コスト等の面で制限をうける。

一方、待ち行列網シミュレーションを新たに1つの計算機ジョブとして見直すと画像処理等とは異なって処理の実行に確率的不確定要素を含むジョブであると言える。従って、処理の実行以前に処理計画をたてることはできない。以上の2点から本研究では待ち行列網シミュレーションの並列性に着目した分散型シミュレータを構成しシミュレーションに課せられた制限を緩和するとともに事前に処理計画をたてることのできないジヨブの分散処理について知見を得ることを目的とする。筆者らはすでに並行処理事象型待ち行列網シミュレータD－SSQを提案し、その処理能力はノード数に対して線形に増加すること、キャンセル、履歴保存等のオーバーヘッド分に要する処理時間が長いために処理能力は期待したほど大きくならないことを明らかにした。[3] 本稿では先行に規制をかけキャンセル量を減らすことによって処理能力を改善できることを示す。さらに、処理能力が最大となる最適規制値を予測するために規制先行制御方式の近似解析を行なう。

2．待ち行列網シミュレーションとその分散化

待ち行列網はキュー，サーバおよびそれらの接続関係を表わす構造と、キュー動作，客の到着，サービス動作等の動作の2面から構成されている。従って、待ち行列網シミュレーションを分散型システムで実行する場合どちらに注目するかによって2つの分散方式が考

えられる。

まず、構造に注目して待ち行列網シミュレーションを分散化する方式がノード分散であり、データ交換網の場合は、交換機に１つのプロセッサを割り当てることに相当する。一方動作に注目して分散化したものがプロセス分散である。データ交換網のシミュレータのプロセス分散では交換機の各機能にプロセッサを割り当てることになる。Ｄ－ＳＳＱでは扱いやすさからノード分散を採用している。

待ち行列網シミュレーションのもう１つの特徴は、画像処理等とは異なって確率的不確定要素を含むことである。これは以下に説明するunknown型メッセージパッシングを含むからである。メッセージパッシングとは複数の通信しあうプロセスがあった場合、それらの間に生じる通信の形を　Ｃ．Hewittらがpast型、now型、future型の３つに分類したものである。[1] これらは、図１(a)に示すようにpast型メッセージパッシングは、プロセス１はメッセージ送信後プロセス２からのメッセージを待つことなく処理を進める場合、now型メッセージパッシングは、プロセス１がメッセージを送信した後プロセス２からのメッセージを待つ場合、そしてfuture型は、プロセス１はメッセージ送信後いくつかの処理を実行した後プロセス２からメッセージを受信する場合を指している。しかし、この分類では、プロセス１はプロセス２からのメッセージ受信があるかどうか、ある場合はいつ受信するのかを予め知っていなければならない。分散化された待ち行列網シミュレーションではこれらすなわち、メッセージ受信の有無、およびメッセージ受信の時刻はプロセス２で確率的に決定される。例えば、図１(c)に示すようなプロセス２からプロセス１に送られるメッセージは確率的に分岐するモデルがあったとする。プロセス１は自分がメッセージを送ってから(プロセス１中の論理的な時刻で)いつメッセージが返ってくるかわからないため処理を中断せざるをえない。プロセス２からメッセージがリンク１をとおりプロセス１に送られた場合にはその時点で処理を再開できるが、プロセス２がメッセージをリンク２を通してモデル外に退去させた時にはプロセス１の処理は中断したままである。このような場合プロセス１の処理を再開するためにはプロセス２がメッセージをモデル外に送り出すと同時に制御メッセージをリンク１に送る必要がある。このような方法ではプロセス１はプロセス２からメッセージを受信するまで処理の実行を中断されることになる。従来のメッセージパッシングをknown型と呼べば、上記のようなメッセージの送受信をunknown型のメッセージパッシングと呼ぶのが適当である。以上の概念を表１にまとめて示す。

３．Ｄ－ＳＳＱの概要

unknown型メッセージパッシングを含む分散型待ち行列網シミュレーションを制御する方法としては電電公社のＮＥＷＴＳ、阪大基礎工のＨＡＳＳ－ＱＮ等の時刻駆動方式や慶

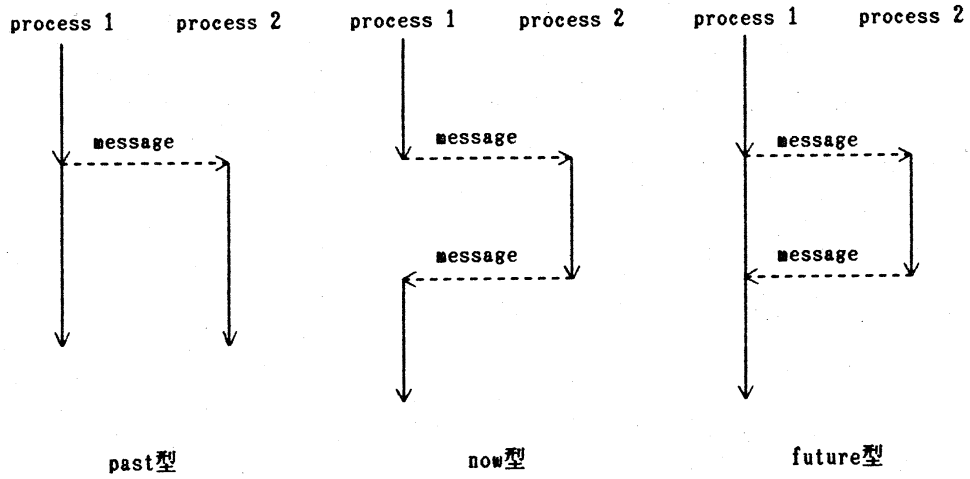


図 1(a) known型メッセージパッシング

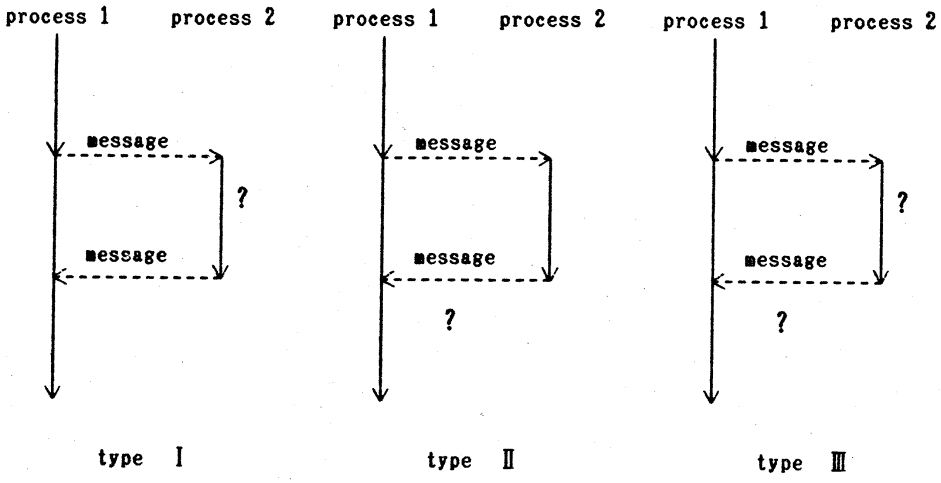


図 1(b) unknown型メッセージパッシング

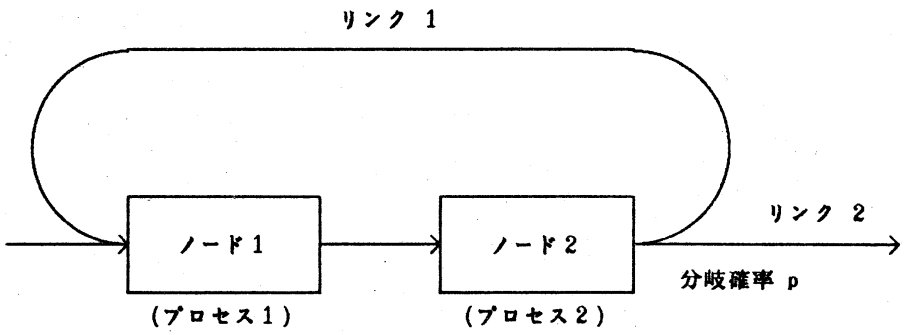
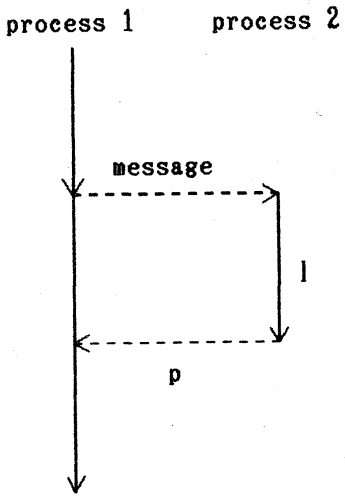


図 1(c) 2ノードフィードバックモデル

表 1　メッセージパッシングの分類



p	l	メッセージパッシングの型
0	−−	past型
1	確定	now型、future型
	?	unknown型 I
0~1	確定	unknown型 II
	?	unknown型 III

応大学のＫＤＳＳのような制御メッセージを送受する方法がある。Ｄ−ＳＳＱがこれらのシステムとは大きく異なる点は、Ｄ−ＳＳＱでは待ち行列網が本来持っている並列性を極限まで利用するため一時的なシミュレーション時刻の矛盾を許容する先行制御方式を用いている事である。先行制御方式とは、以下のアルゴリズムで示される。

(ｉ)各プロセッサは他のプロセッサとの時刻関係を無視して処理を行なう。

(ii)他のプロセッサからメッセージを受信しそのメッセージで表わされるパケットの到着時刻が現在のプロセッサの時刻から見て過去である場合には矛盾が生じる。これを時刻矛盾と呼ぶ。

(iii)時刻矛盾が生じた場合はプロセッサのシミュレーション時刻をパケットの到着すべき時刻に引き戻し、その間に実行した処理、すなわち実行しすぎた処理を未処理の状態にする。これをキャンセルと呼ぶ。

先行制御を行なうためには、以下の機能が必要となる。

・各プロセッサにおいてプロセッサの履歴を保存する。

・履歴をもとにキャンセル処理を行なう。

・システム全体の確定時刻を検出する。

ここで、確定時刻とは、Ｄ－ＳＳＱ中のすべてのプロセッサが到達したと保証できる時刻を言う。現在、Ｄ－ＳＳＱは図２に示すような６台のプロセッサ(１台のマスタプロセッサ[ＭＰ]と５台のコンピューティングプロセッサ[ＣＰ])で構成した実験システムで動作している。

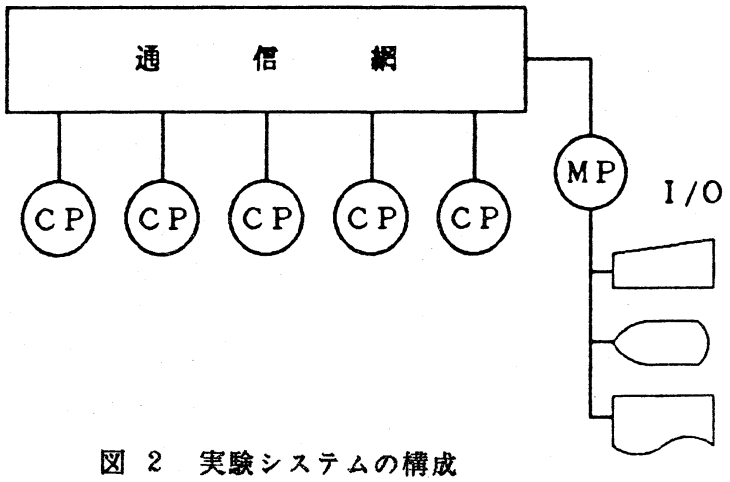


図 ２　実験システムの構成

先行制御方式を用いたＤ－ＳＳＱの処理能力(同一モデルのシミュレーションを行なった場合の１プロセッサシミュレータに対するＤ－ＳＳＱの処理速度比)はプロセッサ数に対して線形に増加することがわかっている。しかし、その値はそれほど大きくないこともわかっている。この原因はプロセッサが先行しすぎキャンセル量の多いことがその一因と考えられる。

４．規制先行制御方式

先行制御方式の処理能力改善案として先行に規制をかけキャンセル量を減らす規制先行制御方式を提案する。規制先行制御方式では、各プロセッサは、ある時刻(これを規制時刻と呼ぶ)以上に時刻を進めようとした時に進みすぎと判断し、シミュレーション時刻を更新しない２次的処理を行なうか単に時間を消費し、他のプロセッサの時刻が進むのを待つ。(この状態を以下遊休状態と呼ぶ)

規制時刻の設定は、システム全体に渡って行なうグローバル方式と隣接プロセッサからの情報を利用するローカル方式の２つがある。

(a)グローバル方式

システム全体の規制時刻を確定時刻に規制値を加えた値にする方式である。グローバル方式ではプロセッサのシミュレーション時刻のずれを規制値内に閉じ込める効果がある。

(b)ローカル方式

シミュレーションモデル上の通信回線を介して隣接するノードが割り当てられているプロセッサからの情報を利用し過渡的な時刻の進みすぎを押さえる効果が期待できる。

図3は規制先行制御方式を用いた場合のD－SSQの処理能力を実験システムで測定したものである。同図から明らかなようにグローバル方式を用いた場合約2割の改善効果が見られたもののローカル方式はアルゴリズム実装化の際のソフトウエアオーバヘッドが大きいために逆に処理能力は低下した。

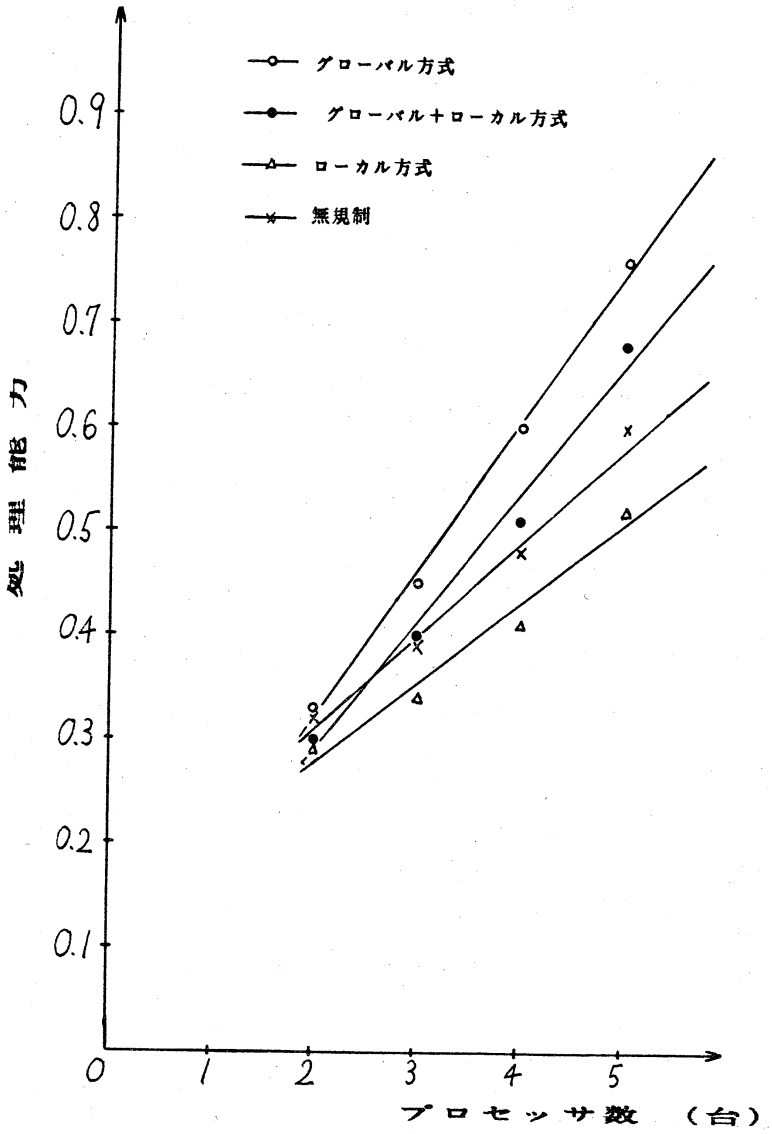


図 3 プロセッサ数に対する処理能力

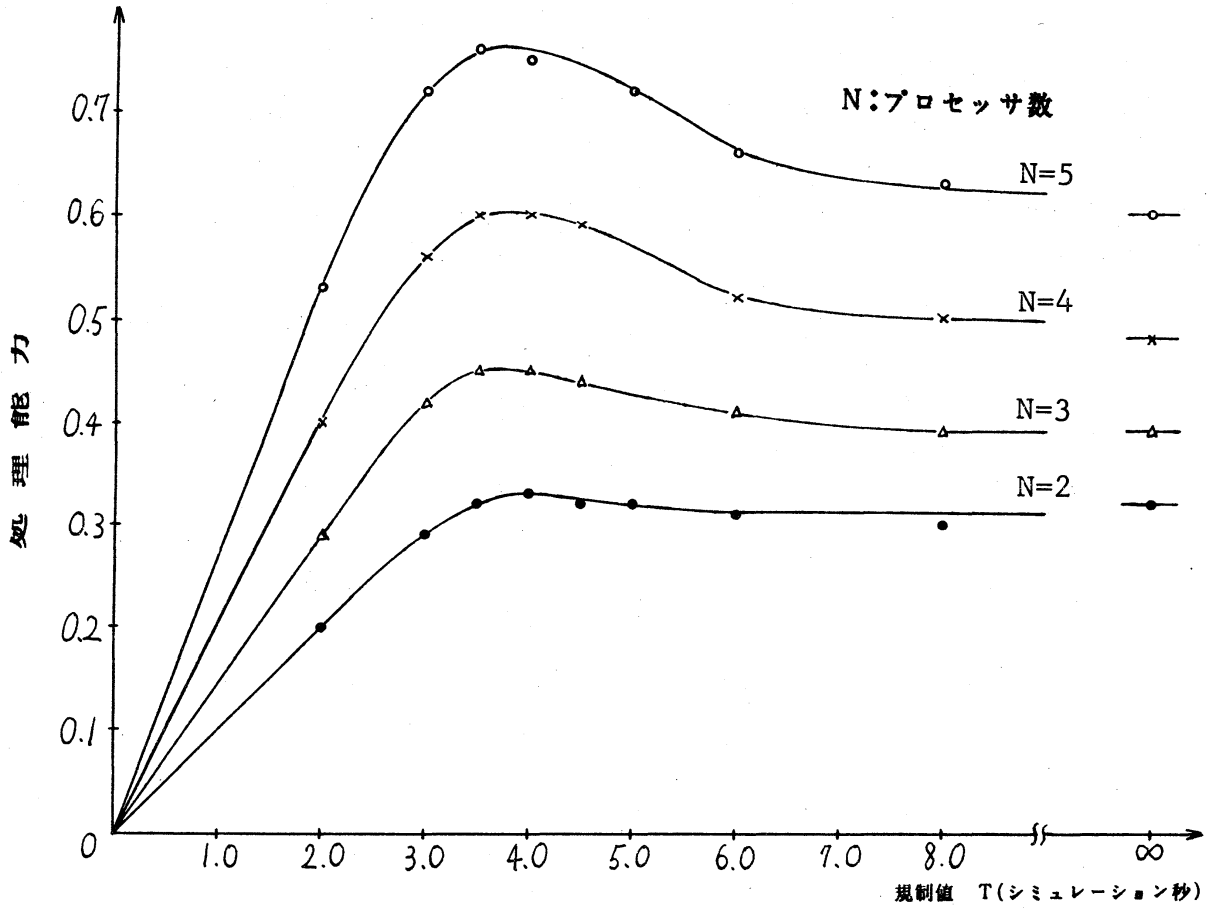


図 4 規制値に対するグローバル方式の処理能力

また、グローバル方式では処理能力を最大にする最適規制値が存在することが図4よりわかる。最適規制値が存在する理由は、規制値が小さく規制が強い場合ではプロセッサの遊休状態が多くなり処理能力は低下し、一方規制値が大きく無規制に近くなるとキャンセル量が増すためやはり処理能力は低下するためと考えられる。グローバル方式のもう1つの特徴はプロセッサ数が増加すると規制の効果がより大きく現われ、山は急峻になることである。また最適規制値はプロセッサ数やシミュレーションモデルによっても変動する。従って、グローバル方式を用いるためには予めできるだけ最適規制値に近い規制値を設定しなければならない。

5．規制先行制御方式の近似解析

グローバル方式を用いた規制先行制御方式はD－SSQの処理能力を改善できるが最大処理能力を与える最適規制値を予測しなければならない。そこで、本章では規制先行制御方式を近似的に解析し最適規制値を予測する。解析では実時刻とシミュレーション時刻の

２つの時刻を考慮しなければならない。以下実時刻に関する記号は小文字で、シミュレーション時刻に関する記号は大文字で表わす。また、対比用の１プロセッサシミュレータをＳ－systemと呼ぶ。各記号の添字１はＳ－systemを、ｍ はＤ－ＳＳＱを表わす。

待ち行列網シミュレーションにおけるシミュレーション時刻がどのように更新されていくかを示したものが図５と図６である。図５は、Ｓ－systemにおけるシミュレーション時刻の更新状況を表わしている。同図の小文字はノード１の内部で生じたイベント、大文字は他のノードからノード１への通信によって生じたイベントを表わす。このように実時刻が増加するに従ってシミュレーション時刻は階段状に増加する。一方Ｄ－ＳＳＱのプロセッサ１(ノード１が割り当てられているプロセッサ)におけるシミュレーション時刻の更新状況は図６のように凹凸のある時刻更新過程となる。これはＤ－ＳＳＱでは一度実行した処理を取り消してやり直しをする事があるからである。しかし、十分に長い実時間観測すれば全体としては実時刻に対しシミュレーション時刻は増加する。図６を用いてＤ－ＳＳＱの時刻更新過程を説明する。

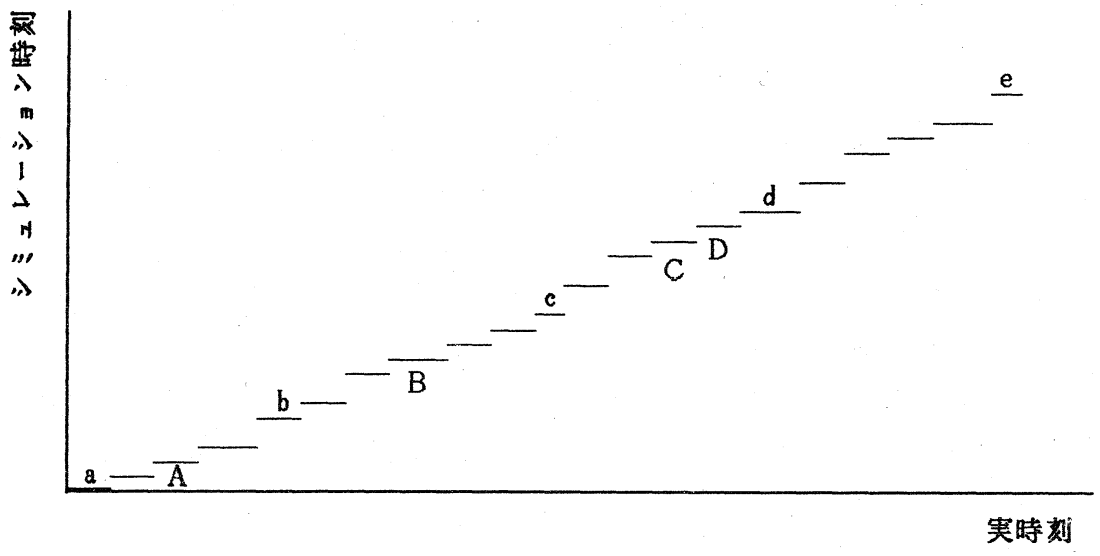


図 5　Ｓ－systemにおけるシミュレーション時刻更新過程

時刻０からシミュレーションを開始しa,bのイベント処理を実行した後他のプロセッサからイベントＡの処理要求が送信されたとする。この時このプロセッサの現在時刻はＴｂになっている。Ａの処理を実行すべき時刻T_A はＴｂよりも小さい、つまり過去であるため時刻矛盾が生じキャンセル処理を行なう。キャンセル処理終了後イベントＡの処理から

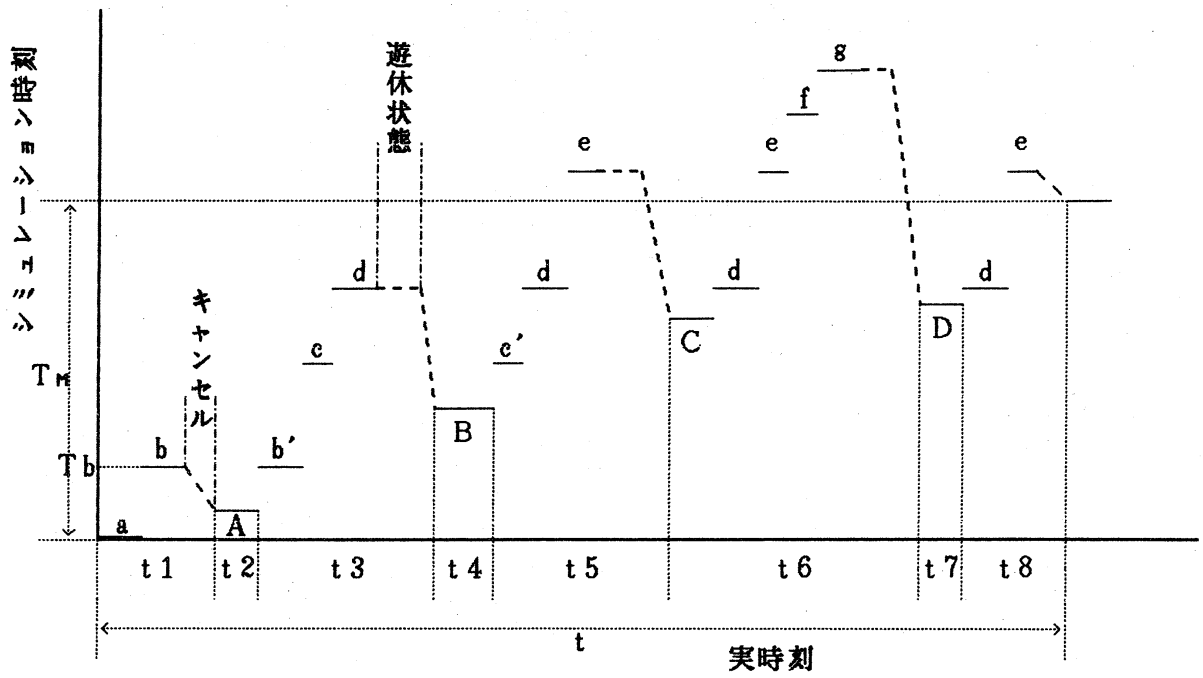


図 6　D－ＳＳＱにおけるシミュレーション時刻更新過程

実行を再開し、b′,c,dと進んだとする。このとき注意すべきことはb′がＡの影響を受けることがあるため必ずしもbとは処理内容が一致しないことである。dの処理実行後、遊休状態となっているのは次のイベントの処理を行なうと規制時刻を越えるためである。この遊休状態の間、プロセッサ１は他のプロセッサのシミュレーション時刻が進むのを待っている。

上記のようにして実時刻 t まで進んだ時には、シミュレーション時刻はT_M まで進んでいる。tまでに実行された処理の内有効な処理はa,b,c,d,Ａ,Ｂ,Ｃ,Ｄに対する処理である。e,f,gは無駄な処理を行なったわけであり、これらを無効処理と呼ぶ。またb,cの処理は２回、dは４回繰り返して実行している。

処理能力を解析するにあたってＳ－systemとＤ－ＳＳＱにおいて同じＮノードのモデルを実時間 t だけ実行した場合の記号を表２に定義する。Ｄ－ＳＳＱ中のプロセッサ数はＮとし、１プロセッサに１ノードが割り当てられているとする。

実時間 t 内に実効される処理数はＳ－systemでは

$$T_i / Ta_i \quad \text{または} \quad t / t_i \qquad (1)$$

Ｄ－ＳＳＱでは

$$T_M / Ta_M \quad \text{または} \quad t / t_M \qquad (2)$$

表 2　記号の定義

	S－system	D－SSQ
シミュレーションを行なう実時間	t	t
t時間に進むシミュレーション時間	T_1	T_M
1処理に要する平均実時間	t_1	t_M
1処理を行なって更新されるシミュレーション時間の平均	Ta_1	Ta_M

D－SSQにおける1処理は1有効処理を意味する。

D－SSQの処理能力の定義は、

$$A = T_M / T_1 = \frac{t_1 \cdot Ta_M}{t_M \cdot Ta_1} \qquad (3)$$

であるから、D－SSQ上に$Ta_M = N \cdot Ta_1$とジョブを展開できるとすれば、

$$A = N \cdot t1 / t_M \qquad (4)$$

となる。

図6に戻るとtは、

$$t = \sum_{j=1}^{8} tj = (ta+2tb+2tc+4td+3te+tf+tg)+(t_A+t_B+t_C+t_D)$$
$$+(\text{キャンセル時間})+(\text{遊休時間}) \qquad (5)$$

と表わされる。一般化すれば、次式となる。

$$t=\sum_{j=1}^{k} nj\times rj+\sum_{j=1}^{nc} rcj+\sum_{j=1}^{ni} rij \qquad (6)$$

ただし　k　：有効処理数
nj　：j番目のイベントの処理の繰り返し回数
rj　：j番目のイベントの処理に要した実時間
nc　：キャンセル回数
rcj　：j番目のキャンセル処理に要した実時間
ni　：遊休状態になった回数
rij　：j番目の遊休状態で費やした実時間

従って、1有効処理に要する平均実時間t_M は

$$t_M=\left(\sum_{j=1}^{k} tj\right)/k$$

$$=n\cdot t_{M0}+nc\cdot tc/k+ni\cdot ti/k \qquad (7)$$

ただし、$n=\overline{nj}$ ，$t_{M0}=\overline{rj}$ ，$tc=\overline{rcj}$ ，$ti=\overline{rij}$ （ ¯ はjについての平均）

これを式(4)に代入して

$$A=N\cdot t_1/(n\cdot t_{M0}+nc\cdot tc/k+ni\cdot ti/k) \qquad (8)$$

αをS－systemにおける1イベント処理時間に対するD－SSQにおける1イベント処理時間比($t_{M0}=\alpha\cdot t_1$)、ntを有効処理数と無効処理数の和、pcを1処理を行なった後に時刻矛盾が生じる確率、piを総処理数(nt,nc,niの和)とすれば、

$$A=N/[n\cdot\{\alpha+pc\cdot tc/t_1+pi\cdot ti\cdot(1+pc)/(1-pi)\cdot t_1\}] \qquad (9)$$

となる。

上式中のn,α,pc,tc/t_1,pi,ti/t_1について以下考察する。解析にあたっては次の仮定を設ける。

(i)確定時刻はすべてのプロセッサのシミュレーション時刻の最小値に等しく瞬時に求まる。

(ii)イベントを処理して更新されるシミュレーション時刻幅は、平均$1/\mu$シミュレーション秒の指数分布に従う。

(iii)規制値をTとする。

(a)pc:時刻矛盾の生じる確率は規制値Tが0に近づくとpc→0，T→∞においてpcはある一定確率に収束する。

$$pc = Pr \cdot \{1 - \exp(-b_0 \cdot \mu \cdot T)\} \qquad (10)$$

ただし　Pr：無規制時の時刻矛盾発生確率

b_0：定数

(b)ti:稼動中のプロセッサ数は注目しているプロセッサ以外の半分とすれば

$$N_w = (N-1)/2 \qquad (11)$$

確定時刻はN_wコのプロセッサによって$1/N_w \cdot 1/\mu$ずつ更新され、また1つの処理にはt_{M0}ずつかかるから注目したプロセッサが遊休状態を抜け出すまでの実時間は

$$ti = \frac{\frac{1}{\mu}}{\frac{1}{N_w} \quad \frac{1}{\mu}} \cdot \alpha \cdot t_1 \qquad (12)$$

となる。

(c)tc:キャンセル処理に必要な実時間は戻るシミュレーション時間内に存在するイベントの数に線形に増加すると考えられる。従って

$$tc = \{\eta_0 \cdot \mu \cdot E(T) + \eta_1\} \qquad (13)$$

η_0, η_1　：定数

E(T)　：規制値Tに対する戻り時間の期待値

E(T)は

$$\text{分子} = -T \cdot \exp(-l \cdot T) + \left(\frac{1}{l} - \frac{1}{m}\right) \cdot \{1 - \exp(-l \cdot T)\} + \frac{l}{m(m+l)} \cdot \{1 - \exp(-l \cdot T - m \cdot T)\}$$

$$\text{分母} = \frac{m}{l+m} + \frac{l}{l+m} \cdot \exp(-l \cdot T - m \cdot T) - \exp(-l \cdot T) \qquad (14)$$

ただし、βは無規制時の平均戻り処理数

$l = \mu / \beta$　　$m = l/3$

(付録参照)

(d)n:図6に戻ると0～tに進むシミュレーション時間T_Mは

$$T_M = \sum_{j=1}^{k} nj \times Tj - \sum_{j=1}^{nc} Tcj$$

Tj　：j番目のイベントによって更新されるシミュレーション時間幅

Tcj　：j番目のキャンセルで戻ったシミュレーション時間幅

jについての平均を考えれば、

$$T_M = k \cdot n \cdot T_0 - nc \cdot Tc \qquad (15)$$

ただし、$n=\overline{nj}$, $T_0=\overline{Tj}$, $Tc=\overline{Tcj}$

$T_M = k \cdot T_0$より

$$k \cdot T_0 = k \cdot n \cdot T_0 - nc \cdot Tc$$
$$= k \cdot n \cdot T_0 - k \cdot n \cdot pc \cdot Tc$$

よって $$n = T_0 / (T_0 - pc \cdot Tc) \qquad (16)$$

(e)pi:プロセッサが遊休状態となるのは、シミュレーション時刻更新幅が規制値Tを越える時であるから、

$$pi = \int_T^{\infty} pe(x)dx = \exp(-\mu \cdot T) \qquad (17)$$

以上を式(9)に代入すればD－ＳＳＱの処理能力Aは

$A = A(T, \mu; \alpha, \beta, \eta_0, \eta_1, Pr, b_0)$と表わされた。

ここで規制値Tはシミュレーションモデルのμに依存するためTの代わりに$K = \mu \cdot T$なる規制処理数Kを用いる。Kは規制値T内に平均して存在する処理の数を意味する。Kを用いると、

$A = A(K; \alpha, \beta, \eta_0, \eta_1, Pr, b_0)$となる。 (18)

上式より$(\partial A / \partial K) = 0$ $(K = Kopt)$となるKoptが最適規制処理数である。式(18)中のパラメータであるα、η_0、η_1はシステムのハードウエアに、またβ, Pr, b_0は先行制御のアルゴリズムに依存する。これらのパラメータは次のようなステップによって測定可能である。

step 1 S－systemとD－ＳＳＱのプログラムステップ数よりαを求める。

step 2 無規制$(K=\infty)$においてβ, Pr, η_0, η_1を測定する。

step 3 適当な規制処理数を設定しb_0を求める。

プロセッサ数5台の実験システムではこれらのパラメータは$\alpha=2.0$, $\beta=5.0$, $Pr=0.1$, $\eta_0=0.42$, $\eta_1=8.1$, $b_0=0.13$であった。

6．実験システムにおける実験値と近似解析値の比較

実験システムで直接測定した値と上記パラメータを代入して求めた近似解析値との比較を行なった。その結果、遊休状態確率以外はよい近似となった。遊休状態確率がずれた理由として確定時刻が求まった時点における規制時刻と現在時刻の差、つまり実質的な規制

値が約１／２になっているためと処理の内発生・伝送の処理は規制にかかりやすくなっている事からμの修正を施さなければならないことがあげられる。

図７～図１０に補正後の測定値と近似解析値との比較結果を示す。図７～図９は規制処理数に対する戻り処理数、キャンセル実時間、繰り返し回数を示している。いずれの図も・が測定値、実線が近似解析値を表わしている。これらの図から明らかなようにほぼ実験値に近い値が得られている。

図１０は規制処理数に対する処理能力を示している。処理能力を最大とする最適規制値はほぼ一致し最適規制値を予測する当初の目的は達せられた。しかし、処理能力の値は実験値の約1.5倍となっている。この理由は解析にあたって確定時刻が瞬時に求まると仮定したが実際には確定時刻はＭＰが一定実時間ごとにＣＰから現在時刻を集めその最小値をＣＰに知らせる方式を用いているため確定時刻更新には時間がかかる。また解析では統計量収集等の処理を無視していると考えられる。これらはすべてオーバヘッドαを小さく見積もったことになる。一例としてαを3.0にしたときの解析値は最大処理能力1.0、最適規制値7.8となった。

７．まとめ

先行制御方式の処理能力改善案としてグローバル方式とローカル方式の２つの規制先行制御方式を提案しグローバル方式では処理能力を約２割改善できることを示した。またノード数が増加すると規制の効果が大きくなるため最適規制値を予測する必要であることを示した。さらに、最適規制値を予測するためにグローバル方式の近似解析を行ない実験システムでの測定値と比較した結果よい近似であることを確認した。

本研究で予測された最適規制値を初期値として、実行時に規制値を補正していくダイナミック規制方式が今後の課題である。

[参考文献]

(1) Hewitt,C. and Baker,H.:"Laws for Parallel Communicating Processes",IFIP－77, Tronto,(1977)

(2) Yomezawa,A. and Matsuda,H.:"Towards Object Oriented Concurrent Programming", ソフトウエア科学・工学の数理的方法研究集会報告集(1984)

(3) 佐藤、中西、真田、手塚:"待ち行列網シミュレータＤ－ＳＳＱについて",待ち行列理論とその応用研究集会報告集(1983)

(4) 肥塚、渡辺、佐藤、中西、真田、手塚:"Ｄ－ＳＳＱにおける先行規制方式の検討",情報処理学会分散処理システム23－2(1984.7)

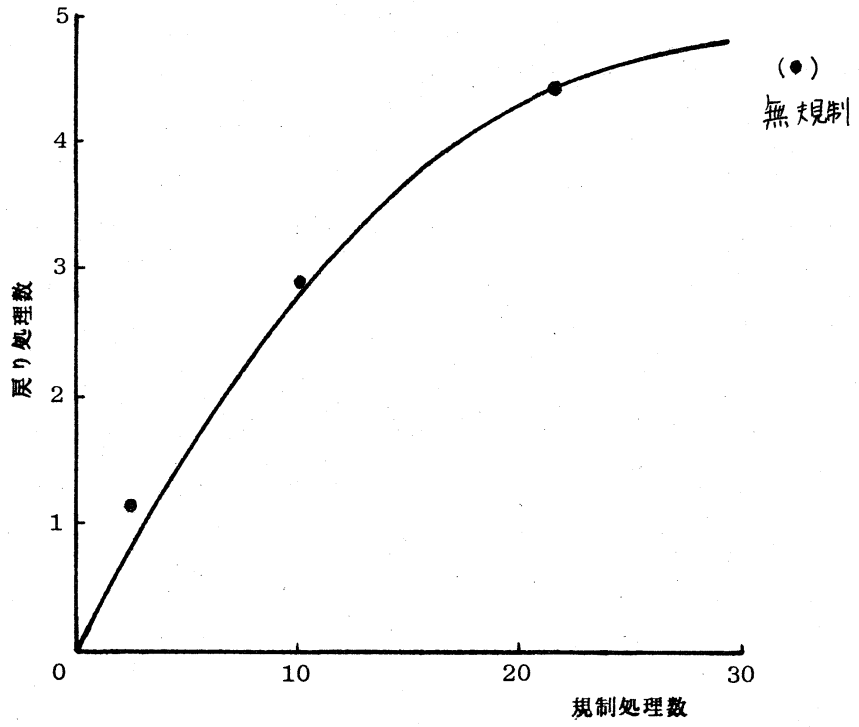


図 7　測定値と解析値の比較(戻り処理数)

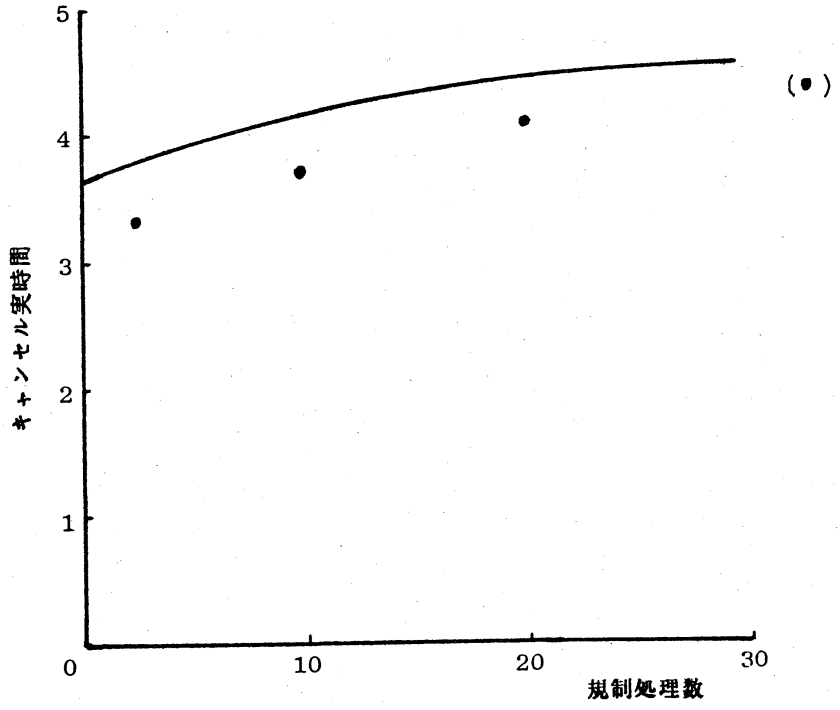


図 8　測定値と解析値の比較(キャンセル実時間)

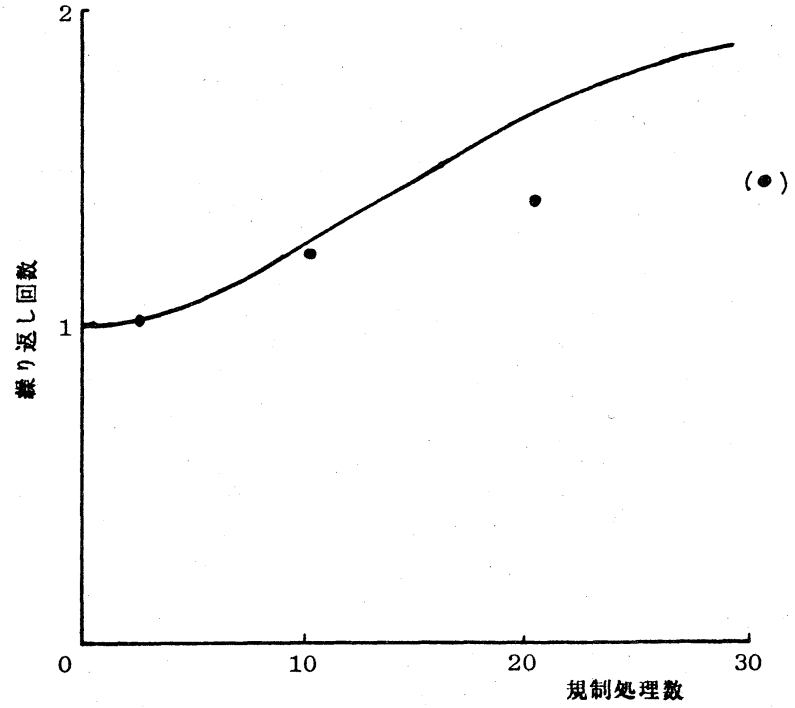


図 9　測定値と解析値の比較(繰り返し回数)

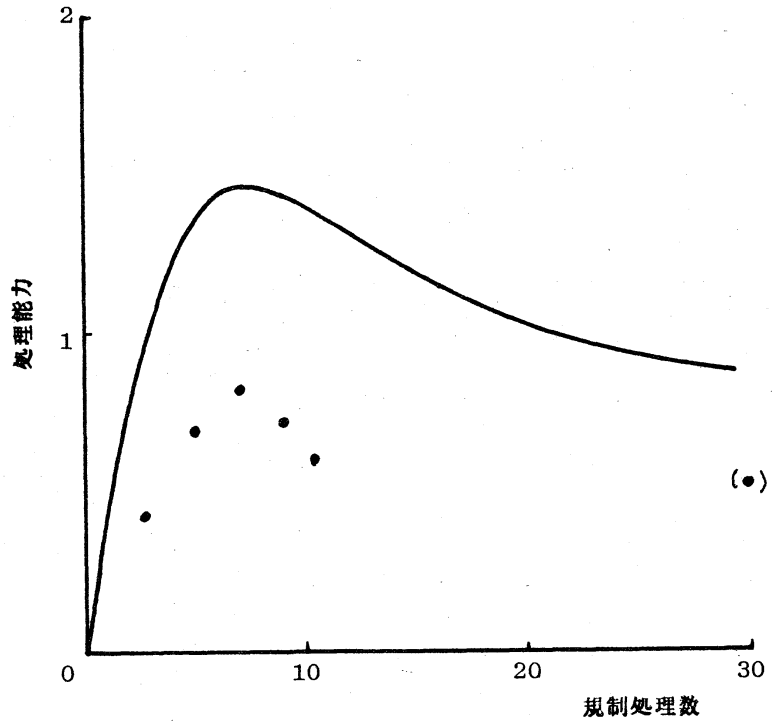


図 10　測定値と解析値の比較(処理能力)

[付録]

平均戻り時間の計算

キャンセルが生じるまでのプロセッサ１の時刻更新幅の分布を　$p(x)=l\cdot\exp(-l\cdot x)$
プロセッサ２の時刻更新間隔を　$p(y)=m\cdot\exp(-m\cdot y)$と仮定する。(本来ならば１／$\mu$の指数分布がいくつか重なったものであるがここでは単一の指数分布とする)プロセッサ２からプロセッサ１へメッセージの送信が起こり時刻矛盾が生じた時の戻り時間の期待値は

$$E(T)=\frac{\int_{x=0}^{T}\int_{y=0}^{x}(x-y)\,p(x,y)\,dy\,dx}{\int_{x=0}^{T}\int_{y=0}^{x}p(x,y)\,dy\,dx}$$

$p(x,y)=p(x)\cdot p(y)$と仮定する

で計算できる。

これを計算すれば

$$\text{分子}=-T\cdot\exp(-l\cdot T)+\left(\frac{1}{l}-\frac{1}{m}\right)\cdot\{1-\exp(-l\cdot T)\}+\frac{1}{m(m+l)}\cdot\{1-\exp(-l\cdot T-m\cdot T)\}$$

$$\text{分母}=\frac{m}{l+m}+\frac{l}{l+m}\cdot\exp(-l\cdot T-m\cdot T)-\exp(-l\cdot T)$$

(付１)

となる。

シミュレーションにおいては、発生,伝送,到着が主な処理内容であるからプロセッサ２からプロセッサ１へメッセージを送信する時刻更新幅１／mは、１／lの３倍すなわち$m=l/3$が成立する。

また$T\to\infty$において$E(T)\to 1/l$　すなわち無規制状態では１／lだけ戻ることになる。１／l間シミュレーションされる処理数はμ/lである。これをβとすれば$l=\mu/\beta$となる。