B4 システム・シミュレーションの1解法とその応用例

原田紀夫，久保秀士，三上　徹　（日本電気中央研究所）

1. はじめに

コンピュータ・システムの性能評価において，ディジタル・シミュレーションが多く用いられており，システム評価の強力な武器となっている．しかし，シミュレーションにも問題はない訳ではなく，種々の問題がある．たとえばシミュレーション・モデルを精密にすればそれに伴ってモデル作成のための時間や労力が増大したり，シミュレーション実行のための計算機時間が増大して，評価に長時間を要したり，シミュレーション・モデルの妥当性やシミュレーション結果の信頼性（たとえばシミュレーションの立上り時間や測定時間等）について問題が山積されていると言ってよい．またこれらの問題のほかにシミュレーションをどのように実行したらよいかとかシミュレーション結果をどのように解析評価したらよいかという問題もある．特にシミュレーションで得られた出力結果をどのように分析してシステムの評価に役立たせるかあるいはどこをどのように改善したら，どの程度の性能向上が期待出来るか等を解析，評価し，これが最も重要なものの1つであるが改善のための最適政策を導びくことである．極論すれば，この最適政策を決定してはじめてシミュレーションの任務は終ると言えよう．

ここではこれらの問題点の中で特にシミュレーションで得られたデータを解析，評価し，最適政策の決定を行なう手法を考察するものである．1つ1つの手法はすべてが目新しいものではないが系統的に示すことはシステムの性能評価法に何らかの役に立つものと思われる．本論ではシステム要因をリニアグラフによって表わすこと，要因間の関係を分析することおよび，サブグラフによってシステムを解析，評価すること等を一般的に述べるとともに1つの適用例を示す．

2. システム要因のグラフ化とデータのグラフ化

コンピュータ・システムにおいてはしばしばハードウエア，ソフトウエアというような分類が用いられている．これらは特に製造，製作的な立場からなされているものと思われる．特にシステム性能の評価という観点に立てばシステム性能が十分に得られるかどうかが問題であって，システムの構成要素がハードウエアで出来ているかソフトウエアで出来ているかは極論すれば二次的な事柄であると言えよう．したがってここではハードウエア，ソフトウエアにこだわることなく，より抽象的にシステム性能に影響を及ぼす要因として考え，仮りに「システム要因」と名付けて以下にこの名称を用いることにする．これはたとえて言えばリソースの使用回数，使用時間，使用率等のものを含むものである．

（1）システムのグラフ表現

上において「システム要因」という概念を導入したが注目すべきことは1つのシステムにおいて，システム要因は幾つもあるけれどもシステム要因の間には全く関係がない場合が少なく，むしろ因果関係あるいは影響関係がある場合が多い．したがってシステム要因間の関係を有向グラフによって表わすのが妥当であろう．次のようにシステムを表わすリニアグラフを定義しよう．

定義1

システムにおいて，システム要因をA_1，A_2，……，A_nとする．このとき，次のように方向枝を付ける．

A_iがA_jに影響を与えるときあるいはA_jがA_iの従属変数であるとき：$A_i \rightarrow A_j$

で表わし，影響関係が存在しないとき，A_iとA_jは枝で結ばない．

このようにして表わされたグラフを$G(V, E, \triangle)$で表わす．

ここで，Vは頂点全体の集合，即ちシステム要因を表わし，E，△は影響関係を表わす．

上の定義1のようにグラフGを決定した場合，有向グラフGは次のような性質をもつ．

性質 (i) positive degree $\delta^+(A)=0$かつ
negative degree $\delta^-(A)>0$なる節点が存在する．

性質 (ii) positive degree $\delta^+(A)>0$かつ
negative degree $\delta^-(A)=0$なる節点Aが存在する．

上の性質(i)を満たす節点はすべて方向枝の始点となっていることを意味しており，シミュレーション等における設定値あるいはパラメータを意味する．また性質(ii)を満たす節点AはAに結びついている枝のすべての終点になっていることを意味しており，これはシステムの最終評価項目を表わしていると考えてよい．たとえば後の適用例おいては同時接続数，ユーザのプログラム数等が性質(i)を満たし，応答時間が性質(ii)を満たす．

上記によってシステムをリニアグラフで表現することが出来るがそれだけでは有効とは言えず，システム表現グラフが次のような条件を満たすとき，最も有効であるといえる．

要請　A

システムの表現グラフにおいて，影響関係はすべてシステム要因の絶対量で決まる．たとえば図1において，要因Cが要因Dに及ぼす影響はCの絶対量によって決まり，A，BがCにどのように影響を与えたかには依らない．即ち，A，Bがそれぞれ3，5ずつ影響を与えてもA，Bが5，3ずつ影響を与えてもCの結果がA，Bの和で影響する場合にはともに8となり，CのDへの影響はともに差がない．

要請Aの条件を満たすようなシステムの表現グラフにおいては，グラフ内のサブグラフをとり出して議論することが出来，これによってシミュレーションの設定パラメータ以外の事柄も評価する

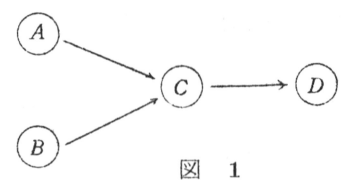


図　1

ことが出来る．これらの例については後に適用例において述べられるであろう．

（2）演算の表現

システム要因間には影響関係だけでなく更に強く演算関係がある場合がしばしばある．それらをシステムを表現するグラフによって表現しておくことは影響関係を明確に把握する手助けになろう．

したがってここでは要因間の加減乗除を次のように表示することにする．A，B，Cはそれぞれシステム要因とする．

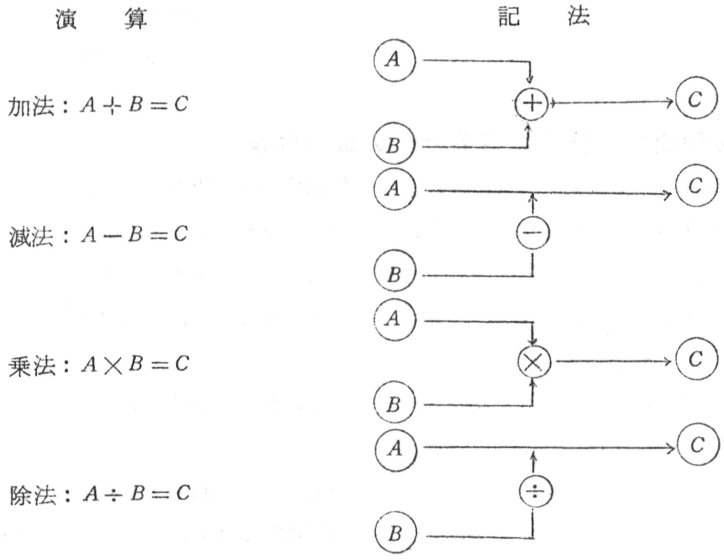


図2　加減乗除の表示

これらにおいて演算が2つ以上ある場合は始点に近い方の演算を先に行なうことと約束する．即ち，図3は$(A+B)/C=D$を表わしている．

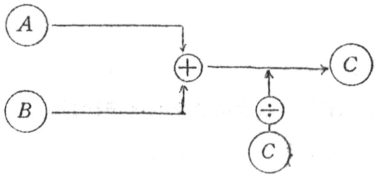


図3　$(A+B)/C=D$のグラフ

（3）データのグラフ化

システム要因は実測値あるいはシミュレーション結果等によって一連の値をとる．これらのデータをシステム表現グラフにおける各システム要因の実現値として対応する節点に値を入れ，そのグラフ全体がシステム表現グラフの一つの値とみなせる．これはベクトルの成分に値を入れることに似ているがシステム表現グラフでは成分間に影響関係が存在することが異なる．これらを例として示せば図4から図6のようになる．A，B，C，D，Eはシステム要因とする．各システム要因の値に対して単位が必ず必要であるが略してあり，数値が大きい程影響力が強いというものでもない

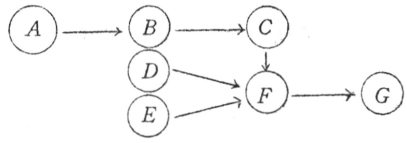


図4　システム表現グラフ

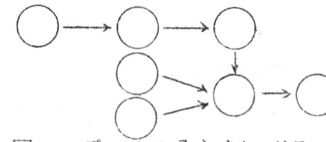


図5　データの入らないグラフ

$A=8$，$B=20$，$C=10$

$D=1000$，$E=50$，$F=100$，

$G=250$

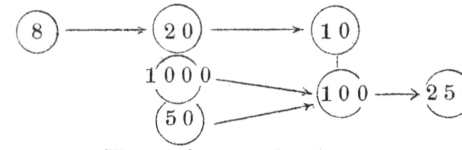


図6　データの入ったグラフ

3. システム要因間の分析と改善のための最適政策

システムの評価においては最終的に問題となるのは評価指標が期待どおりに満たされるかどうかである．たとえばＴＳＳでは応答時間が十分に得られるかどうかである．しかしシステムが常に動的である以上，ある時点，ある条件のもとにおいて所定の性能が得られることが判ったとしてもシステムの評価としては不十分であると言えよう．すなわち，次の２点はシステムの評価において欠かすことが出来ない．

(i)　どの要因がどのように他の要因に影響を与えるか．特に最終評価指標にどのように影響を与えるか．

(ii)　(i)によって影響の仕方が判明したならばどの要因をどのように改善したならば最終評価指標がどれだけよくなるか．特にどの要因をどのように改善したら，最も安くあるいは最も容易にあるいは最も早く，システムの最終評価指標を改善出来るか．

上記の(i)についてはここでは函数形の予測を直交多項式によって推定し，(ii)については簡単な最大原理の問題として処理することにする．

（１）直交多項式による函数形の推定

システム表現グラフをG，その実現値を$G^{(1)}, G^{(2)}, \cdots\cdots, G^{(s)}$とする．直交多項式による函数形の推定は独立変数が等間隔の場合は簡単に行なうことが出来る．しかしシステム要因の実現値の大部分は設定値ではなく，等間隔に値が散らばることはない．従って次のようにして実現値を等間隔で近似する．

(i)　クラス分け

実現値を適当な階級幅でクラス分けして，同一階級に入る実現値の平均をその階級の代表値として採用する．

(ii)　欠測値の補完

(i)において実現値の存在しない階級は線形性を仮定して埋め合わせ，直交多項式を用いた逐次法で欠測値を補うとともにその要因同士の関数関係を求める．

(i)(ii)によって，要因間の関係が直交多項式によって求められる．したがって，この操作を各

図８　シミュレーションモデルの表現グラフ

要因について行なえば隣り合う要因間でなくても関数関係を導くことが出来る．たとえば図4において$F=f(C,\ D,\ E)$，$G=g(F)$の様になり，$G=g\big(f(C,\ D,\ E)\big)$として$G$と$C$，$D$，$E$の関数関係を表わすことも出来る．この性質は，次の改善のための政策を決定する際に重要になってくる．

（2） 改善のための最適政策

前記(1)において各要因間の函数関係について述べたが，ここでは改善のための政策を考える．簡単のために図7のようなシステムを考え，更に要因間の函数はすべて1次式と仮定する．即ち

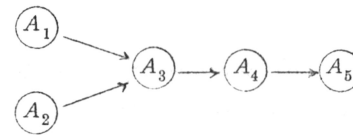


図7　システムのグラフ

$$\text{(a)}\quad \begin{cases} x_3 = a_1 x_1 + a_2 x_2 + b_1 \\ x_4 = a_3 x_3 + b_2 \\ x_5 = a_4 x_4 + b_3 \end{cases} \qquad (x_i \text{ は } A_i \text{ に対応する変数とする})$$

とする．このとき，ある改善によって，x_1，x_2が$x_1-\triangle x_1$，$x_2-\triangle x_2$となったとする．この改善によるA_5の改善幅を$\triangle x_5$とすれば簡単な計算によって

$$\text{(b)}\quad \triangle x_5 = a_1 a_3 a_4 \triangle x_1 + a_2 a_3 a_4 \triangle x_2$$

となる．要因A_1，A_2の改善の行ない難さをここでは仮りに「改善の困難度」と言うことにし，$C_1(\triangle x_1)$，$C_2(\triangle x_2)$で表わすことにしよう．このC_1，C_2は実際には改善に要するコスト，あるいは技術的な困難さを示してもよく，適当に正規化することによってそれらが混合していてもよい．したがって改善のための最適政策は$\triangle x_5$を先に与えたとすれば，次のように言うことができる．

システム要因A_5を$\triangle x_5$だけ改善するのにA_1，A_2を改善して実現するとした場合，最も容易に行なうには

$$\triangle x_5 = a_1 a_3 a_4 \triangle x_1 + a_2 a_3 a_4 \triangle x_2$$

のもとに$J=C_1(\triangle x_1)\triangle x_1 + C_2(\triangle x_2)\triangle x_2$が最小になるように$\triangle x_1$，$\triangle x_2$を決定することである．(注) 即ちこれは最適制御の問題である．

（3） サブグラフによるシステムの分析

1.において，システム表現グラフのサブグラフによる評価について触れたがシステムの表現グラフが要請Aを満足することによって各要因はcyc licの場合を除けば前歴を考慮する必要がない．このことから，システムの表現グラフの適当なサブグラフを取り出して，それを1つのシステム表現グラフとして考えて解析することが出来る．たとえば，のちに適用例で述べるように，コンピュータ・システムのシミュレーションでたとえばメモリ系がシミュレーション・パラメータとなっていなくてもシステム表現グラフ上のメモリ系に関係するシステム要因の

（注） 要因間の関数関係には通常，何らかの適用制限，たとえば，適用範囲等があって，上記のように簡単に1次式で書いた直線もある区間の範囲に限定されているが本稿ではそれらの制限は略して書かない．したがって，上記の2式がともに直線で解がないという事はない．しかし$\triangle x_5$が不適当であるときは解がない場合もあり得る．

みを含むサブグラフ上でメモリ系の評価を行なうことが出来る．これらの事は4.において詳しく述べられるであろう．

4. シミュレーションへの適用例

ここでは上記2.，3.で述べたシステムの性能解析法の具体的な適用例をＴＳＳのシミュレーションを例として示す．これによって一般的説明では不十分な点も具体的にはっきりすると思われる．

（1）シミュレーションモデルの表現グラフ

シミュレーションで対象としたモデルはＴＳＳであり，シミュレーション・パラメータは同時接続数，リモート・バッチ数およびユーザプログラム特性の3因子であり，各因子は1，2，3の3水準である．シミュレーション・モデルのシステム表現グラフは図8のようになる．図8においてシミュレーションの実行によって定まる節点は便宜上（　）で表わし，シミュレーションの設定値等最初から定められているものは［　］で表わす．

（2）欠測値の補完と関数関係

本シミュレーションは(1)で述べたように3^3型実験であるが，シミュレーション実施ケースは図9における黒印のケースであり，非実施ケースについては欠測値の推測を行なう．欠測値の補完は特に要因間の関数関係を推測するのに重要であるがここではシステムの最終評価指標である応答時間の欠測値についての推測を遂次法によって行なう．

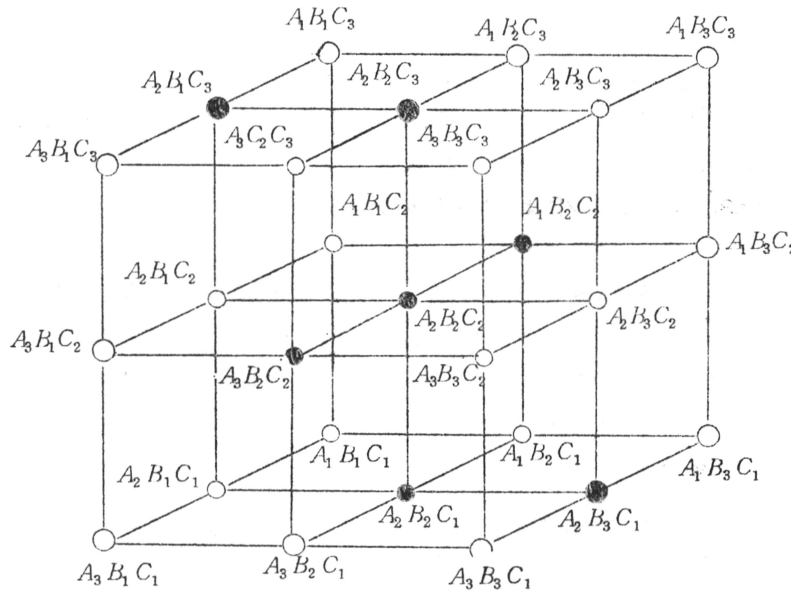
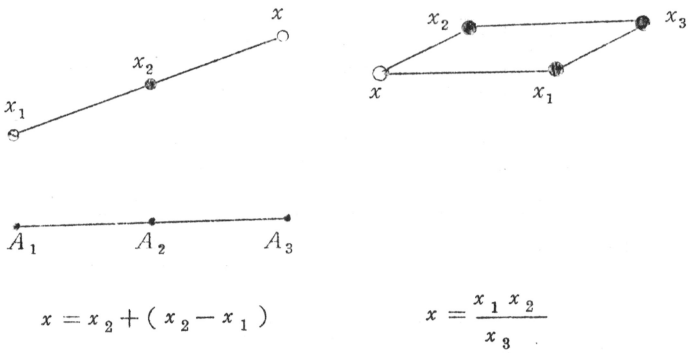


図9　シミュレーション実施ケース

(i) 逐次法の初期値の設定

初期値の設定は1パラメータに関しては線形性を仮定してそれぞれの成分について直線で求め，そのように求められた値のすべての平均をその点の初期値とする．即ち図１０のようにして求める．



$$x = x_2 + (x_2 - x_1) \qquad x = \frac{x_1 x_2}{x_3}$$

図１０　欠測値の初期値の計算

表１，２，３，４は第０次，１次，２次，３次の近似の表である．第３次近似における直交多項式は式(c)のようになる．

$$\begin{aligned}(c)\quad y &= m + a_1(A-\overline{A}) + b_1(B-\overline{B}) + c_1(C-\overline{C}) + d(A-\overline{A})(B-\overline{B}) \\ &\quad + e(A-\overline{A})(C-\overline{C}) + f(B-\overline{B})(C-\overline{C}) \\ &= 157 + 26(A-\overline{A}) + 8(B-\overline{B}) + 22(C-\overline{C}) - 10(A-\overline{A})(B-\overline{B}) \\ &\quad + 10(A-\overline{A})(C-\overline{C}) + 5(B-\overline{B})(C-\overline{C})\end{aligned}$$

ここでA，B，Cはそれぞれ，同時接続数，リモート・バッチ数およびプログラム特性であり，$\overline{A}$，$\overline{B}$，$\overline{C}$はそれぞれの平均値である．ここで$A_1, A_2, A_3, B_1, B_2, B_3, C_1, C_2, C_3$ は１，２，３の３水準である．

応答時間以外の要因についても同様にして求めることが出来る．即ちシステムの表現グラフのシステム要因のグラフ上の道を辿って上のような関数を求めて行けばどの要因が最終評価指標にどのように影響を与え，どの要因が重要であるかを判別することが出来るが，ここではサブグラフによるシステムの評価と最適政策について考察する．

（３）サブグラフによるシステムの解析

本シミュレーションではシミュレーションパラメータとしてメモリサイズやシステム・プログラムのステップ数等は入っていないがそれらに対する性能の変化についての解析を以下において行なう．システムプログラムのステップ数についても特にメモリ系に関係するタスク２，３について行なう．なおタスク２，３はロールイン，ロールアウトを司どるタスクである．

(i) メモリサイズの変化に対する推測

先に示したシステムの表現グラフの中でメモリ系が応答時間に影響を与える様子は図１１のようになる．図１１はシステム表現グラフ図８からメモリ系に関するものを抜き出し，後の解析が行い易いように多少の変形を加えたものである．

表1. 第0次近似

$A1$	$C1$	$C2$	$C3$	計
$B1$	− 39	−38	−37	−114
$B2$	− 36	−24	−12	− 72
$B3$	− 32	−15	− 2	− 45
計	−107	−77	−47	−231

$A2$	$C1$	$C2$	$C3$	計
$B1$	− 15	2	4	− 9
$B2$	− 14	11	28	25
$B3$	− 13	5	36	28
計	− 42	18	68	44

$A3$	$C1$	$C2$	$C3$	計
$B1$	− 1	37	75	111
$B2$	− 8	20	95	107
$B3$	− 15	3	59	47
計	− 24	60	229	265

$A1A2A3$	$C1$	$C2$	$C3$	計
$B1$	− 55	1	42	− 12
$B2$	− 58	7	111	60
$B3$	− 60	− 7	97	30
計	−173	1	250	78

表2. 第1次近似

$A1$	$C1$	$C2$	$C3$	計
$B1$	− 40	−41	− 34	−115
$B2$	− 32	−24	− 16	− 72
$B3$	− 32	−15	2	− 45
計	−104	−80	− 48	−232

$A2$	$C1$	$C2$	$C3$	計
$B1$	− 17	− 2	4	− 15
$B2$	− 14	11	28	25
$B3$	− 13	26	35	48
計	− 44	35	67	58

$A3$	$C1$	$C2$	$C3$	計
$B1$	6	37	75	118
$B2$	− 8	20	72	84
$B3$	− 12	19	68	75
計	− 14	76	215	277

$A1A2A3$	$C1$	$C2$	$C3$	計
$B1$	− 51	− 6	45	− 12
$B2$	− 54	7	84	37
$B3$	− 57	30	105	78
計	−162	31	234	103

表3. 第2次近似

$A1$	$C1$	$C2$	$C3$	計
$B1$	− 39	−38	− 37	−114
$B2$	− 31	−24	17	− 38
$B3$	− 32	10	3	− 19
計	−102	−52	− 17	−171

$A2$	$C1$	$C2$	$C3$	計
$B1$	− 16	− 1	4	− 13
$B2$	− 14	11	28	25
$B3$	− 13	30	35	52
計	− 43	40	67	64

$A3$	$C1$	$C2$	$C3$	計
$B1$	7	36	75	118
$B2$	− 3	20	67	84
$B3$	− 1	28	69	96
計	3	84	211	298

$A1A2A3$	$C1$	$C2$	$C3$	計
$B1$	− 48	− 3	42	− 9
$B2$	− 48	7	112	71
$B3$	− 46	68	107	129
計	−142	72	261	191

表4. 第3次近似

$A1$	$C1$	$C2$	$C3$	計
$B1$	− 44	−37	− 30	−111
$B2$	− 27	−24	− 7	− 58
$B3$	− 32	− 1	− 16	− 17
計	−103	−62	− 21	−186

$A2$	$C1$	$C2$	$C3$	計
$B1$	− 18	− 1	4	− 15
$B2$	− 14	11	28	25
$B3$	− 13	37	42	66
計	− 45	47	74	76

$A3$	$C1$	$C2$	$C3$	計
$B1$	8	30	75	113
$B2$	1	20	65	86
$B3$	4	31	68	103
計	13	81	208	302

$A1A2A3$	$C1$	$C2$	$C3$	計
$B1$	− 54	− 8	49	− 13
$B2$	− 40	7	86	53
$B3$	− 41	67	126	152
計	−135	66	261	192

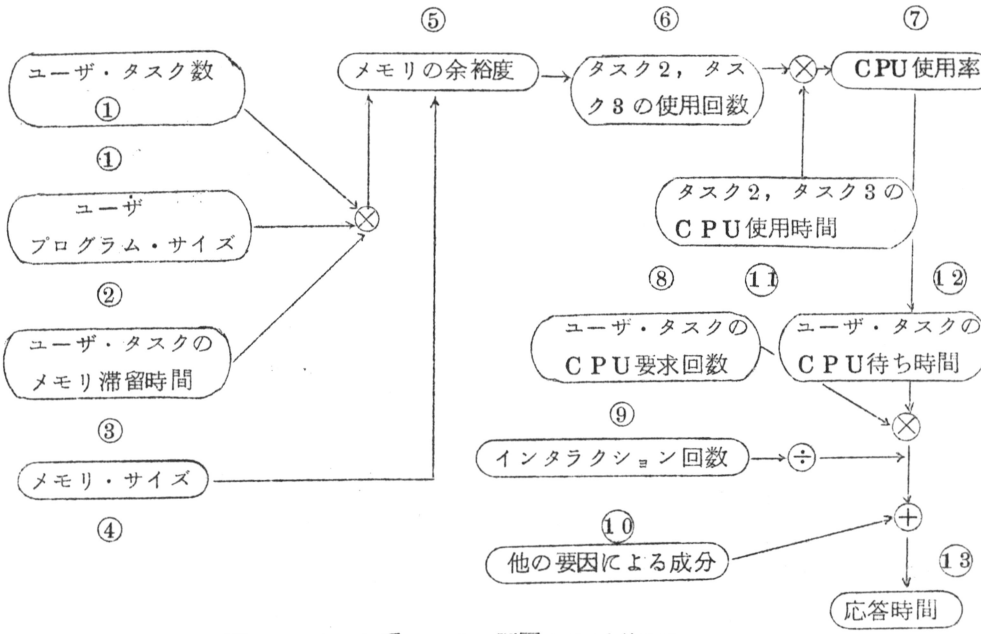


図11　メモリ系システム要因のサブグラフ

ここでメモリの余裕度とはメモリ上にある端末I／0中のユーザタスクの数とする．メモリではユーザタスクが現に動いている領域と端末I／0中で実際には動いていない領域とがあり，それぞれをここではアクティブ領域，インアクティブ領域と言うことにする．インアクティブ領域が大きいことはメモリに余裕が存在することであることから，その状態にあるユーザタスクの数を仮りに「余裕度」と名付ける．

図11におけるメモリ系のシステムの表現グラフでは要請Aが満足されており，各連続するシステム要因間の関数を逐次求めればよい．

今，メモリサイズを$\triangle M$だけ増加した場合，ユーザのプログラムサイズをuとして以下において解析する．

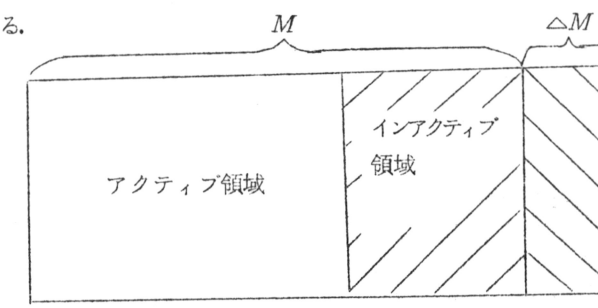


図12　メモリの状態

メモリ系のシステム表現グラフの余裕度をv_1とし順次v_2，v_3，v_4＝CPU待ち時間，v_5＝CPU使用時間とする．（図13参照）

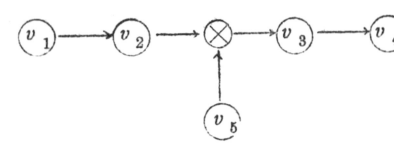


図13　メモリ系のシステム表現グラフ

v_{i-1}の変化に対するv_iの変化の様子は図14から図16までに示されており，各要因間を回帰直線で代表すれば

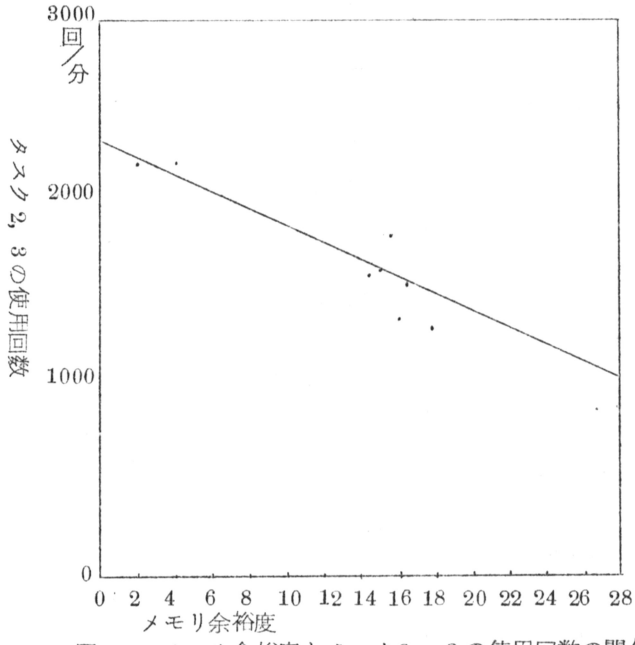


図14　メモリ余裕度とタスク2，3の使用回数の関係

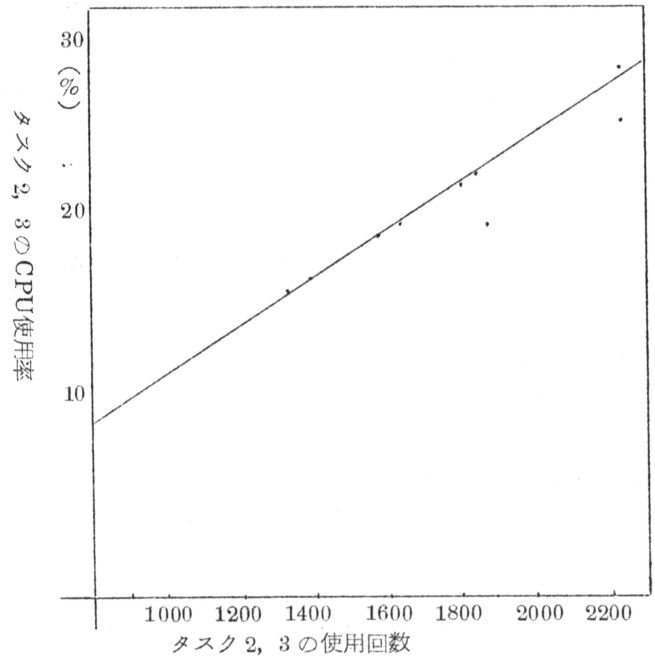


図15　タスク2，3の使用回数とCPU使用率の関係

$$v_i = a_{i-1} v_{i-1} + b_{i-1} \quad (i=1, 2, 3, 4)$$

となり，v_1と$\triangle v_1$だけ改善すれば，ユーザタスクのＣＰＵ待ち時間は

(d) $\triangle v_4 = a_1 a_2 a_3 \triangle v_1$

だけ改善される．

$\triangle R$は

(e) $\triangle R = \triangle v_1 \times$（ユーザタスクのCPU要求回数）

で与えられる．

例1 （数値例）

$\triangle v_1 = \dfrac{\triangle M}{u} = 10$，CPU要求回数$=15$ として数値例を示す．図14～16より．

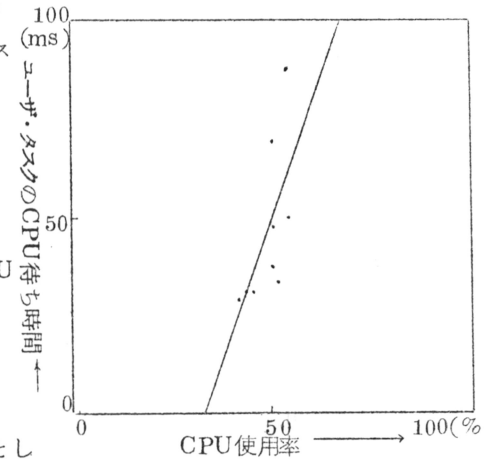


図16 CPU使用率とユーザ・タスクのCPU待ち時間の関係

$$a_1 = -46, \quad a_2 = \frac{1}{500} \times 6.4, \quad a_3 = 2.88$$

$$\triangle v_4 = -17.0\text{ms}$$

$$\triangle R = (-17.0) \times 15 = -255\text{ms}$$

即ち，メモリの余裕度を10だけ増すことにより，応答時間は255msだけよくなる．

(ii) システム・プログラムのステップ数の改善に対する予測

ここではシステム・プログラムの改善，特にプログラム・ステップ数を減らすことによって，システム性能がどれ程改善されるかをメモリ系の予測のところで現われたタスク2，3について行なう．

今，図13において

(f) $v_3 = \alpha v_2 v_5$

が成立する．ここで$a_2 = \alpha v_5$が成立する．αは命令実行時間によって決まる定数．このときv_4の変化分$\triangle v_4$は

$$\triangle v_4 = a_3 \alpha v_2 \triangle v_5$$

である．例2に数値例を示す．

例2

$\triangle v_5 = \dfrac{-v_5}{3}$，$v_2 = 1200$とする．即ち，プログラム・ステップの改善によって，ステップ数が$\dfrac{1}{3}$だけ減り，そのときのＣＰＵの使用回数が1200回／分とし，ユーザタスクのＣＰＵ要求回数を15とすると

$$\triangle v_4 = a_3 \alpha v_2 \triangle v_5 = \frac{-a_3 a_2}{3} v_2 = \frac{-1}{3} \cdot \frac{6.4}{500} \cdot 2.88 \cdot 1200 = -14.7\text{ms}$$

$$\triangle R = (-14.7) \times 15 = -220\text{ms}$$

となる．すなわち，応答時間は$\triangle R = -220$msだけ改善される．

（4）改善のための最適政策

(3)において，メモリサイズの増加，特にメモリの「余裕度」の増加による応答時間の変化，およびシステム・プログラムのステップ数の改善による効果について述べ，3.においては改善のための最適政策について述べた．ここではメモリの余裕度とプログラム・ステップ数の改善によって，応答時間の改善目標$\triangle R$を達成するのに最も容易に行なうにはどのようにすればよいかを考える．

$c_1(\triangle v_1)$，$c_2(\triangle v_5)$をメモリ余裕度，ステップ数を$\triangle v_1$，$\triangle v_5$だけ改善するのに要する困難度とする．このとき

(g) $$v_4 = a_1 a_2 a_3 v_1 + a_3 a_2 b_1 + a_3 b_2 + b_3$$
$$= a_1 a_3 \alpha v_5 v_1 + a_3 \alpha v_5 b_1 + a_3 b_2 + b_3 \qquad a_2 = \alpha v_5$$

従って，簡単な計算より，

(h) $$\triangle v_4 = a_1 a_3 \alpha v_5 \triangle v_1 + a_1 a_3 \alpha v_1 \triangle v_5 + a_1 a_3 \alpha \triangle v_5 \triangle v_1 + a_3 \alpha \triangle v_5 b_1$$
$$\triangle v_4 = \triangle R / (\text{ユーザ・タスクのCPU要求回数})$$

とすれば，(f)の制約のもとに，評価関数J

$$J = c_1(\triangle v_1)\triangle v_1 + c_2(\triangle v_5)\triangle v_5$$

が最小になるように$\triangle v_1$，$\triangle v_5$を決めればよく，これは最適制御の問題として，解くことが出来る．特に$c_1(\triangle v_1) = c_1$，$c_2(\triangle v_5) = c_2$と定数と仮定すれば簡単な最大問題となる．

5. おわりに

現在のシステム・シミュレーションでは得られたデータが十分に活用されていると言えない面がある．特にシミュレーションの出力データが多項目に亘っているために単に表としてまとめるだけでは，シミュレーションの出力データが「何を語っている」のか判然としない．したがって，ここではシステムの性能を規定する「システム要因」の間の構造をリニア・グラフで表現し，その間の定量的な函数関係を直交多項式で推定すること，重要な要因を抽出すること等を考察した．更に要請Aを満足するという仮定のもとにサブグラフによる，システムの評価を行い，これにより，シミュレーションのパラメータ以外の項目についても評価を行なうことが出来ることを示した．更に，改善のための予測を述べ，改善のための最適政策について問題を簡略化して述べた．これらの手法は更に定式化されるべきものであり，評価の信頼度についても問題がないわけではないが現時点でのシステムの性能評価法としてはかなり有効なものであると思われる．この手法は単にシミュレーション結果だけでなく，実測データの解析およびそれによる改善にもそのまま適用出来るものである．

おわりに日頃御指導頂いている研究部の方々に謝意を表する次第である．

参 考 文 献

1. Busacker and Saaty 「Finite Graphs and Networks」
2. 田口玄一 「実験計画法」上下
2. 宇野利雄 菊地豊彦 「最大原理入門」

本 PDF ファイルは 1972 年発行の「第 13 回プログラミング―シンポジウム報告集」をスキャンし、項目ごとに整理して、情報処理学会電子図書館「情報学広場」に掲載するものです。

この出版物は情報処理学会への著作権譲渡がなされていませんが、情報処理学会公式 Web サイトの https://www.ipsj.or.jp/topics/Past_reports.html に下記「過去のプログラミング・シンポジウム報告集の利用許諾について」を掲載して、権利者の捜索をおこないました。そのうえで同意をいただいたもの、お申し出のなかったものを掲載しています。

過去のプログラミング・シンポジウム報告集の利用許諾について

情報処理学会発行の出版物著作権は平成 12 年から情報処理学会著作権規程に従い、学会に帰属することになっています。

プログラミング・シンポジウムの報告集は、情報処理学会と設立の事情が異なるため、この改訂がシンポジウム内部で徹底しておらず、情報処理学会の他の出版物が情報学広場 (=情報処理学会電子図書館) で公開されているにも拘らず、古い報告集には公開されていないものが少からずありました。

プログラミング・シンポジウムは昭和 59 年に情報処理学会の一部門になりましたが、それ以前の報告集も含め、この度学会の他の出版物と同様の扱いにしたいと考えます。過去のすべての報告集の論文について、著作権者（論文を執筆された故人の相続人）を探し出して利用許諾に関する同意を頂くことは困難ですので、一定期間の権利者捜索の努力をしたうえで、著作権者が見つからない場合も論文を情報学広場に掲載させていただきたいと思います。その後、著作権者が発見され、情報学広場への掲載の継続に同意が得られなかった場合には、当該論文については、掲載を停止致します。

この措置にご意見のある方は、プログラミング・シンポジウムの辻尚史運営委員長 (tsuji@math.s.chiba-u.ac.jp) までお申し出ください。

加えて、著作権者について情報をお持ちの方は事務局まで情報をお寄せくださいますようお願い申し上げます。

期間：2020 年 12 月 18 日 〜 2021 年 3 月 19 日
掲載日：2020 年 12 月 18 日

プログラミング・シンポジウム委員会

情報処理学会著作権規程
https://www.ipsj.or.jp/copyright/ronbun/copyright.html