シミュレーションによるシステム評価法*

川　合　英　俊**

シミュレーションとは，見かけ上実質的な効果を惹き起すように真似をする意で，構造と動作の複雑なシステムの特性を解析するのに用いられる，一般的な手法である．システムのモデルは，プログラムの形でコンピュータの中に作成され，これを実行させることによりコンピュータにシステムモデルの動作を真似させることができる．シミュレーションは，真似たシステムの性能を測定して，これを評価するという重要な手法である．その用途は，**2.** でのべるシステム選択と，**3, 4, 5** でのべるシステム設計とに大別される．

1.　システム評価とシミュレーション

システム評価に用いられるデータは，そのシステムの性能を測定することによって得られる．せまい意味では，システムの外部性能を比較するときに**評価**といい，内部性能の抽出を**測定**ということもあるが，ここでは広い意味でシステムの良否を判断することをシステム評価という．

1.1　評価目的とシミュレーション手法

システム評価の目的は，(1) システムの選択，(2) 設計妥当性の確認，(3) システムの動作解析の3種に分けるのが一般的である．シミュレーションによる評価手法は，Fig. 1 に示したように，適用範囲が広いので，最も強力な手段であるといわれている[1]．その反面，複雑なモデルの作成は簡単でなく，シミュレーションの結果が出る頃には，もう用がなくなっていたという笑えぬ例[2]も少なくない．

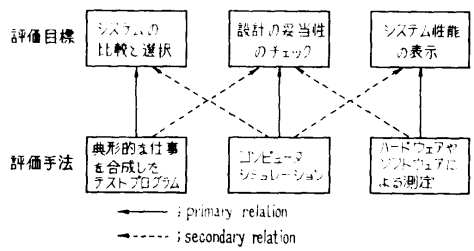

Fig. 1　Computer simulation and evaluation objectives

シミュレーションによる評価手法は，モデルのこまかい変更が容易なことから，上述の（2）に最も適合している．しかし，大幅の変更には耐えられない上に，詳細なモデルのシミュレーションは高価であり，大雑把なシミュレーションではシステムの特徴を抽出しにくいという使用上の困難さがある．

1.2　モデル化とその言語

システムのモデルをプログラムの形でコンピュータ内部で表現するには，汎用のプログラム言語かまたは**シミュレーション言語**が使われる．後者の代表例は，GPSS (General Purpose Simulation System)[3] と SIMSCRIPT[4] である．

GPSS では各ステートメントが動作を表わすブロックに対応し，ブロック網を組む形でシステムのモデルが作成される．システムの動作は，トランザクションがブロック網を移動することによって，シミュレートされる．もともと，システムを待ち行列とみて分析評価するための言語なので，コンピュータシステムのような高度に複雑なものを詳細にモデル化するのには向かない．**SIMSCRIPT** は，システムを状態と事象でモデル化し，システムの動作を事象の生起に伴なう動作を指定する形で記述する．GPSS より柔軟であるが，プログラムのしやすさは劣る．

ほかに提唱されているシミュレーション言語の多くは，システムの詳細なモデル化を容易にしようとするものであるが，部分モデルのファイル化などについてはまだ満足すべきものがない．モデルをインタラクティブにブロック化するなどの便法がとられている[5]にとどまる．

2.　システム評価用のシステム

シミュレーションによる評価手法は，事象が望ましい確率で分布するように生起させることができる点と，システムの構成とか仕事負荷の性格について実測値を用いることができる点で，有用で柔軟な方法である．シミュレーション手法が，システムの**選択**に先出

* System Evaluation by Simulation, by Hidetoshi Kawai (Electrotechnical Laboratory)

** 電子技術総合研究所　電子計算機部

つ比較に用いられるときには，多様な仕事負荷を処理するときのシステムの動作をシミュレートして，その能率を求める．

2.1 システム評価プログラム

評価プログラムシステムの例として，**SCERT** (System and Computers Evaluation and Review Technique)[6] という FORTRAN で書かれたアプリケーションプログラムの商品* がある．システム設置やアプリケーション開発の計画立案にも使用できるもので，Fig. 2 のように5個のフェーズからなる．

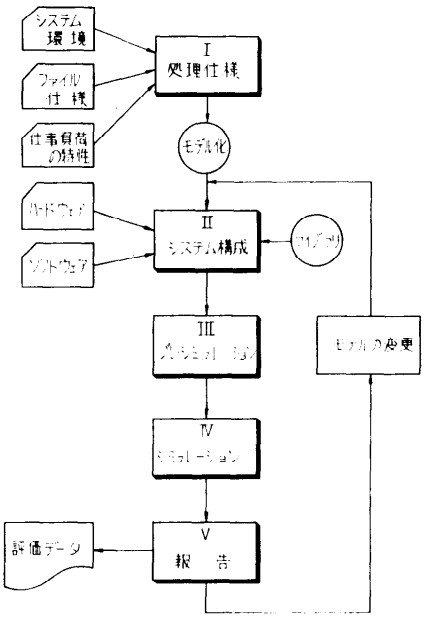

Fig. 2 SCERT Program

(1) 第1フェーズ **処理仕様**；仕事負荷の性格とその処理仕様を決める．価格の構造因子として，プログラマやオペレータの給料とか，プログラマの経験程度などを入力できる．

(2) 第2フェーズ **システム構成**；ハードウェアとソフトウェアの要素と構成を決める．参照するライブラリィには，政府に登録してある型番と単体のカタログ性能が蓄積されている．ソフトウェアは5種類に分類して，それぞれに指数を与えるだけなので，ターミナルインターフェースの特性と並んで，入力が可能であるとはいえ，システムの特徴を表現できるだけの詳しさでモデル化することは困難であるという欠点がある．

システムの環境条件として，床面積，空調，電力容量も記述でき，エラー率などについての長年の経験も蓄積されている．ほかに，数多くの評価項目を計算する実験式もそなえている．

(3) 第3フェーズ **プレシミュレーション**；システム構成の変形を求めて，評価項目の計算に適用する実験式を決める．システムの動作に伴なうタイミングやデータのレコード形式も決まる．スループットの計算にあたって，Fig. 3 のような機能的レベルのものや，それらの並列的な組合わせの表現も可能である．

ファイルの更新
○トランザクションファイル
　▽3個の数値フィールド，それぞれ 15 字
○レコードごとの処理
　▽3個の加算，5桁ずつ
　▽2個のデータ移動，10桁ずつ
　▽3個のデータ比較，7桁ずっ
＊オプション（半数のレコードについて）
　▽3個のデータ変換，5字ずつ

Fig. 3 Function level for computation time

(4) 第4フェーズ **シミュレーション**；まず，単一の仕事負荷を追試してスループットを求める．次に，第1フェーズで指定されたランダム事象にもとづいてモデルの動作を並列に重ね，待ち行列の統計値を得る．

(5) 第5フェーズ 評価データの**報告**；Table 1 に示した評価項目について結果が製表される．

このような大規模の評価プログラムを維持する力は，システム使用者の経験*の蓄積であろう．

2.2 システムを評価するコンピュータ

特にオンラインシステムは構造，動作ともにより複雑なので，システム評価にコンピュータを使うことが望ましく，ターミナル使用者の行動のシミュレーションをしようという試み[7] がある．評価系と被評価系との関係は Fig. 4 の通りで，**評価系**はシナリオ[7] またはスクリプト[8] と呼ばれる人工的仕事負荷を発生し

* アメリカの人口調査局 (CENSUS) が COMET (Computer Operated Machine Evaluation Technique) と称して発注したもの (Univac 1108 のアセンブラで書かれている) に由来するもので，調達庁 (GSA) は政府機関が購入するデータ処理システムをこれで評価して選択している．

* アメリカの政府機関の使用者の技術レベルは非常に高く，アセンブラを自営で作成する例も多い．その裏には非常（非情？）な計算高さがある．手のこんだシステム評価を行なっているために，政府機関内における IBM のシェア (25%) は小さいともいわれている．

Table 1 Evaluation criteria of SCERT

A	バッチとマルチプログラムのシステム性能	
1.	事象（ラン）単位の CPU 使用率	
	●ラン時間	●全消費時間
	●セットアップ時間	●メモリ容量
2.	日，週，月単位の CPU 使用率	
	●1. に同じ	
3.	システム解析	
	●入出力ごとのチャネル，CPU 待ち時間	●ブロック長さ，チャネル接続，レコード長さなどの最適値
	●計算時間	
	●巻戻しなどの空費時間	●実行命令数
	●プログラム容量	
4.	マルチプログラムスケジュール	
	●一緒にできるプログラム名のリスト	はその理由
		●メモリ容量
	●マルチプログラム可能なとき	●遊んでいる周辺装置の数
5.	スループット	
	●アイドル百分率	●マルチプロセッサのときの時間帯別負荷率
	●メモリ容量の荷重平均	
B	オンラインリヤルタイムのシステム性能	
1.	事象処理時間	
	●入出力時間	●全時間
	●処理計算時間	
2.	機器構成ごとのハードウェア使用率	
	●各周辺装置，CPU，チャネルの使用時間	時間比率
		●待ち行列の期待値と最悪状態
	●同上についての不用/使用の	
3.	実時間型ターミナルのシステム応答	
	●入力の一字目から出力の最終字までの応答時間	●50%，96%，99%の3点での応答時間分布
4.	メモリ容量	
C	価格と実現性	
1.	コンピュータ	
	●機器構成ごとの構成要素リスト	●電力容量，空調，床面積のリスト
	●その価格と月レンタル	
2.	価格の構成	
3.	プログラミング	
	●プログラムごとの開発に要する人月数	ップ数
		●提供者がプログラムするサブルーチンごとのステップ数
	●使用者がプログラムするステ	
4.	アプリケーション	
	●ラン時間	●セットアップ時間
	●プログラムに要する人月数	●プログラムの価格
D	システム仕様	
	●シミュレーションに使ったアプリケーションの入出力仕様	●同上の処理特性

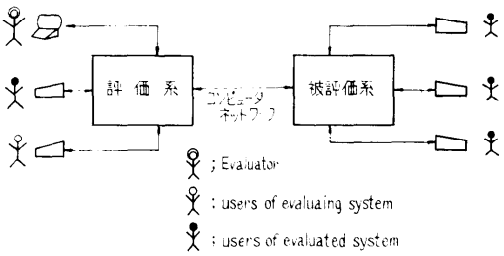

Fig. 4 Evaluated system and evaluating system

て，被評価系の反応を検出し分析するのである．

Fig. 4 から，シミュレーションに用いる実測値を評価系が集約できることが知れよう．MULTICS における評価系は**刺載系**[8]と**検出系**[9]に分れており，後者は CRT ディスプレイによるマンマシンインタラクションが可能である．スクリプト，検出表示のいずれについても設計用の言語が用意されている．検出系による科学的分析がなかったならば，オペレーティングシステムの改良の指針が得られなかったといわれ[10]，実際に直観と手作業による改良のさらに約 10 倍の改良が可能になったルーチンも多い．

評価系と被評価系が，コンピュータネットワークで接続される場合には，人工的仕事負荷の性格を変更することにより，使用者側のいろいろの要求に適合したシステムをきめこまかく選択することができる．また特定の被評価系に一見不自然な仕事負荷を与えて，システム飽和状態を人工的に惹起し，被評価系の動作を解析することもできよう．

評価項目は，外部性能に関するものが主になり，スループットと応答時間の2つに集約できる．しかし，両者の厳密な定義はもちろん，測定誤差にも問題が残っている．評価系がオンラインシステムの場合には，これを経由する被評価系の使用者からは評価系が見えないので，評価系は“透明な傍聴人”[7]などと呼ばれている．このような評価手法を用いても，システムの使い勝手の良さなどは，定量化することが難しい．

3. 仕事負荷のシミュレーション

システム開発にあたって，**設計**の良否を調べたり，**改良**の指針を得るために，システム評価が行なわれるときには，システムの各部の動作を時々刻々と追いながらシミュレーションが行なわれる．この方向で，IBM が 360/195 のトータルシステムのシミュレーションを 360/65* で行なって成功した例[11]がある．このときの中心的な課題は，設計中のコンピュータシステムが完成後に処理すべき仕事負荷を，**混合仕事** (mixed jobstreams) のモデルに構成することであった．

3.1 シミュレーションの必要性

結果から説明すると，Table 2 からわかるように，システムの処理能力は処理装置のサイクルタイムに逆比例するというほど単純には決まらない．360/85 にはキャッシュ**があるとか，360/195 の演算制御は並列化されているなどの，ハードウェアの方式上の差異のほかに，システム規模の違いにもとづく周辺装備の違いも大きいからである．これらの複雑な差異が全体的

* 言語としてはアセンブラが使われた．

** プログラマに見えないバッファのことで，似たものも含めてバッファメモリとかバッファストレージと総称する．

Table 2 Throughput ratio

システム 360 モデル	65 J	85 J	195 K
サイクルタイム (ns)	200	80	54
スループット			
(コンピュートバウンド)	1	4.4	12.9
(入出力バウンド)	1	4.1	12.2
(コマーシャル)	1	4.5	5.9

なスループットにどう効くのかを，数学的に解析することは不可能に近い．なぜならこれらの関係は仕事負荷の性格にも強く依存するからである．

そこで，360/195 の設計条件として，1970 年代の潜在需要に対して既存機種より高い処理能力を要求するという，抽象的な戦略から出発してシミュレーション手法を用いる．

3.2 仕事負荷の特性とシミュレーション結果

仕事負荷を巧みに組合わせて入出力待ちをへらし，メモリを有効に使うようになってきているという技術的な傾向を直観的に認め，オペレーティングシステムの主メモリに占める容量やファイル機器の性能の目安をたてる．360/195 が処理しなければならない仕事負荷に多様性を認め，**標準的仕事負荷**を次のように規定する．

数百種類の仕事負荷の例を追跡，解析する．解析は，システム各部のタイミングチャートとトレーサの結果から，実行した命令，処理装置時間，バッファメモリと処理装置間のやりとりなどを分析するという方法で行なう*．その結果，**仕事負荷の特性**は，命令混合比，アドレスパタン，命令の順序，入出力事象などの因子で表現できることがわかった．Table 2 に示したように，それぞれ 10 個前後の代表的なプログラムを組合わせて作った，3 種類の標準的な混合仕事負荷を，3 機種で実行またはシミュレートして，360/195 の設計の妥当性を調べたのである．実測値との誤差はわずかに 3% であった．

シミュレーションの結果わかったことに，偏微分方程式を解くという種類の仕事負荷に対して，360/195 は 360/65 の 19 倍の性能を発揮すること，バッファメモリの**ヒット率****が 98% 以上であることなどがある．ほかに，スループットの因子は，仕事負荷，OS***，I/O**** の 3 つであること，システム性能の因子は，CPU*****，主メモリ，チャネル，I/O，OS，ソフトウェアパッケージ，運用条件など 7 つの古典的なものに集約できることも明らかになった．つまり，性能向上の方針は，新奇なものである必要はなく，むしろ常識的なものでよいことがわかったのである．

* 分析データは磁気テープにして 1300 巻以上に登ったという．
** アクセスされたものがバッファメモリの中で発見される率．
*** オペレーティングシステム．
**** 入出力装置．
***** 中央処理装置

4. オペレーティングシステムのシミュレーション

OS のシミュレーションは必ずといってよいほど，**システム隘路**の発見と，改良の指針および達成度を見積るために行なわれる．特にオンラインシステムは，設計者の経験や直観が当らないことが指摘されており，シミュレーションはほぼ不可欠の技術になりつつある．性能を測定しただけでは，隘路を局所化することはできても，改良の程度を見積ることができないからである．

4.1 OS シミュレーションの特徴

OS のモデルを GPSS で作成した例[12]もあるが，GPSS やなまの機械語で書かれた部分モデルを，GPSS の構造に組み込むのが不便な点で，GPSS は欠点が多い．OS シミュレーションを直接目標にした言語には，割込み処理ルーチンやターミナル事象の扱いを容易にした **PACSS**[13] とか，アセンブラマクロのレベルでモデルを作成しようとする IBM の **CSS**[14] などがある．

OS のシミュレーションは，設計変更に伴なう効果の見積りと，システム設置時における機器構成の最適化との 2 つの用途を兼ねている．評価項目は，スループット，応答時間，装置使用率の 3 つに集約される．モデルの変更の幅は一般にせまく，誤差 10% 以下の精度が要求されるのが常である．

既存の OS の一部，例えばスーパバイザやデータマネージメントなど，と直接交渉できるように適当なマクロが用意されたシミュレーションシステムが好ましい．そのかわり，OS をよく知らない人はこのレベルのシミュレーションはしにくいという壁となる．また高級言語の通弊として，マクロの中味のデバッグには使用できないという欠点は残る．

4.2 CSS によるモデル作成

システムモデルは，システムの構成，OS の構成，仕事負荷の特性との 3 つから決まる．例えば Fig. 5 のような機器構成は Table 3 のように表わされ，シミュレートされるプログラムは Table 4 のように表わされる．I/O や OS 各部には，アセンブラマクロの他に，特に用意してライブラリに登録してある，18 個

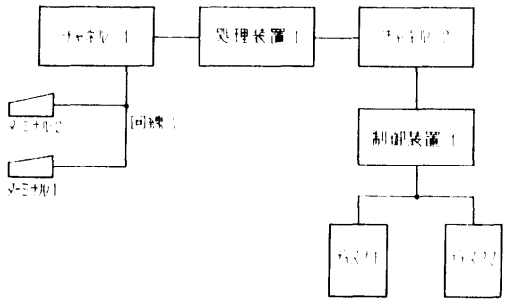

Fig. 5 Example of system configuration

Table 3 Equipments configuration statements
( ); Explanations

装置番号	装置の型	装置の特性
1	処理装置	PR 1 (優先度)
1	制御装置	
1	回線	14.7 (字/秒)
1-2	ターミナル	1
	転送速度	1/70, 2/90 (メッセージ/時間)
1-2	チャネル	1
1-2	ディスク	1, 156, 25 (キロバイト/秒)
	接続路	1/1-1, 2/2-1

Table 4 Actual and simulated programs

現実のプログラム

	LA	6, 1000	カウンタにセット
LOOP	GET	INPUT	レコードを読む
	BAL	14, PROCESS	処理する
	PUT	OUTPUT	レコードを書く
	BCD	6, LOOP	分岐
INPUT	DCB		ファイル定義
OUTPUT	DCB		ファイル定義

シミュレートされるプログラム

1	REFTL		ファイル定義
2	REFTL		ファイル定義
	MOVE	1000, SV6	カウンタにセット
LOOP	GET	$INPUT	レコードを読む
	PROCESS	1000	千分の一秒単位
	PUT	$OUTPUT	レコードを書く
	BRAD	SV6, 1, LOOP	分岐
INPUT	DCB	1	参照表1をみよ
OUTPUT	DCB	2	参照表2をみよ

の基本的モデルをも用いることができる.

プロセスの制御には, フローチャート表現に対応して特に用意した 40 個の CSS 命令が用いられる. 経験値をパラメタとして用いることができる上に, 使用者は, つくりつけのスケジューラや割込み処理ルーチンを取り替えることもできる.

ある OS のモデルは, CSS 言語で数百ステップのモジュール約 20 個からなり, 全部で一万ステップで あったという. このうちの約 85% は, ライブラリの部分モデルがそのまま利用されている. モデルの容量は わずかに 15 kB で, シミュレーションは 256 kB あれば十分可能である.

ここで問題にしなければならないことは, 現実の OS の開発に見合った経費とターンアラウンドタイムで, CSS システムが使いこなされるかという実用的価値である. 精度を 3% とすると, シミュレーション時間は, Fig. 6 に示したように, 現実のプログラム実行時間の約 10 倍もかかる. 精度を 10% にしても時間は半分にしか縮められないのである. したがって, 数日間, 数週間のシミュレーションにはふさわしくない.

シミュレーション時間比
=マクロ実行時間×事象当りのマクロ数×モデルの事象密度
=0.003(秒/マクロ)×2000(CSS 文/事象)×1000(メッセージ/時間)
=10(シミュレーション実行に要した時間/シミュレーション上でモデルが経過した時間)
ただし
マクロ実行時間
=(30~1000 命令)×15 μs(機種は 360/40)
=200×15 μs=3 ms

Fig. 6 Simulation time ratio

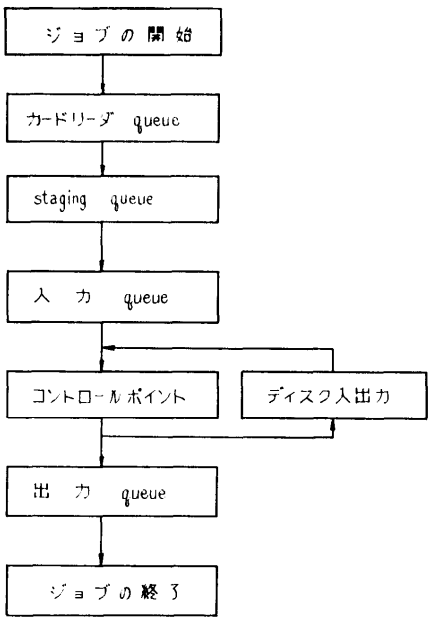

Fig. 7 Rough sketch of jobs Petri Net model

長期運用形態でのシステム評価にあたっては，もっと精度の悪い大雑把なシミュレーションで満足しなければならない．Fig. 7 のような大雑把なモデルを FORTRAN ユーザプログラムで書き，アカウンティングルーチンの結果を用いてシミュレーションした例[15]では，シミュレーション速度が実際の 50 倍も早かった．

この例でも，磁気テープをかける時間が3分であるのに比べて，その標準偏差は2分と非常に大きいなど，システム性能より人間の行為の方が大きい因子として，システム性能に効いている．この手法は，OS のこまかい改良よりも，むしろ運用形態の改善対策を調べるのに向いている．

5. 多重記憶系のシミュレーション

ハードウェアの**設計妥当性**を調べるためのシステム評価では，全面的にシミュレーションの評価手法を用いたもの[16]もあるが，システム規模が大きすぎるので，一般にはシミュレーション手法を補助的に用いる．例えば，インタラクティブな設計システムに組み込むもの[17]などである．

5.1 ハードウェアシミュレーションの特徴

階層をもった記憶系をシステム評価するのに，シミュレーションを用いた例[18]は，ハードウェアシミュレーションの特徴をよく表わしている．バッファメモリ*の効果を調べるときのシステム評価因子の数は多いので，多重記憶系を詳細にモデル化するには経費がかかりすぎる．

もし，設計因子が 10 あって，それぞれについて2通りの場合をすべて試みるとすると，1024 回ものシミュレーションを行なわなければならないことになるからである．多重記憶系の場合には，スタックプロセスという解析的手法とうまく組合わせて，これをほぼ千分の一に小さくすることができる．

組合わせ方の妥当性を高めるために，典型的な仕事負荷を数多く追跡して，品質のよいアドレスパタンを抽出する．さらに，ハードウェア的にみて，無理のない合理的な解析モデルを想定する．こうして，多重記憶系の特徴をよく表わすモデルの作成が可能になる．

5.2 スタックプロセスとシミュレーション

多重記憶系について，各種の構成，各種の動作アルゴリズムを比較するときの，システム評価関数にヒット率を用いる．ある特定の構成で，特定のページ置換アレゴリズムで動作するときの，ヒット率の計算にだけシミュレーションを用いる．このとき，アドレスパタンが変数となる．

リニヤな多重記憶系*は解析的に扱うことができ，Fig. 8 に示したような**スタック**に見たてても，バッファメモリの容量やページ置換アルゴリズムの変形に対して，大差のないヒット率の関数形が得られる．さらに Fig. 9 のように，記憶系をたてわりにして，斜めのページ転送を禁止したモデルについても，シミュレ

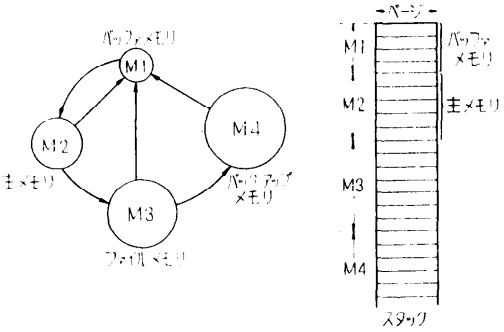

Fig. 8 Linear multilevel memories and stack

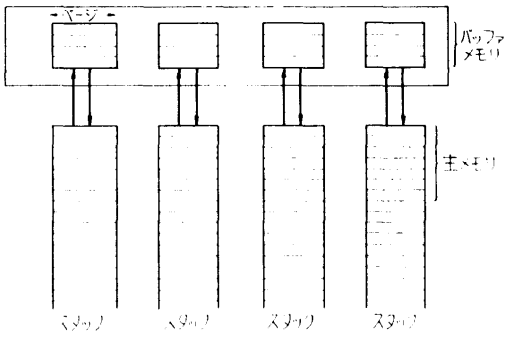

Fig. 9 Classification of multilevel memories

Table 5 Technology selection and system desgin of multilevel memories

諸　元	第1例	第2例
ページの大きさ（バイト）	16	32
たてわりの数	256	256
バッファメモリのページ数/たてわりの数	1	2
バッファメモリのアクセスタイム (ns)	60	60
主メモリのサイクルタイム (μs)	1	8
主メモリのビット当りの価格 (¢)	3	0.5
アドレス変換を含む実効的サイクルタイム (ns)	224	224
バッファメモリの価格 (k$)	16	66
主メモリの価格 (k$)	63	10

* 下層のどの記憶系からも直接バッファメモリへページを転送でき，下層のものへは直上のものからしか転送できない記憶系をいう．

* IBM 360/85 ではキャッシュといっている．

ーションによるヒット率の計算から，あまり性能の劣らないほぼ最適な設計を導びくことができる．Table 5 に，価格やタイミングに現実的な予想値を代入して設計した例を示しておく．

6. おわりに

システム評価でのシミュレーション手法は，複雑なシステム動作のシミュレーションから，変化に富んだ仕事負荷のシミュレーションへと，その役割を変えつつあるといえよう．

大規模のシステムのモデル化は，それを容易ならしめようと設計されたシミュレーション言語をもってしても依然として困難である．しかし，システム変更の効果を見積るための手法としてシミュレーションは，確固とした地位を占めている．

シミュレーションというシステム評価手法の当面の問題として，次の諸点が指摘されよう．

（1） 結果の質と比べて，徒らに大きなモデルを作成しなくてすむように，上手に解析的な手法と組合わせる．

（2） モデル作成作業，およびそのデバッグにあたって，モデルの作成や修正を部分的に独立して行なうなどの，プログラミングサポートを充実する．

（3） モデルの部分に，シミュレーションでない，実物の部分を組込めるようにする．

（4） システムの開発にあたって，シミュレーション言語と開発用のシステム記述用言語との関係を密着させる．

参考文献

1) Lucas, H. C., Jr.; Performance Evaluation and Monitoring. Computing Surveys, Vol. 3, No. 3, ACM, pp. 79-91 (1971).
2) Campbell, D. J. and Heffner, W. J.; Measurement and analysis of large operating systems during system development. FJCC 1968, pp. 903-914.
3) IBM Application Program; General Purpose Simulation System/360 User's Manual, H20-0326-2.
4) Markowitz, H. M., Hausner, B. and Karr, H. W.; SIMSCRIPT A Simulation Programming Language, Prentice-Hall, Inc., (1963).
5) 金田；対話形・グラフィック・シミュレーション・システム．情報処理，Vol. 13, No.10, pp. 665-672, (1972).
6) Ihrer, F. C.; A Technical description of the SCERT program. COMRESS, Inc., 5th Edition, (1967).
7) Kawai, H., Abrams, M. D. and Pyke, T. N., Jr.; Performance Measurement of Remote Access Computer Systems. 電総研い報，Vol. 34, No. 10, pp. 779-801, (1970).
8) Greenbaum, H. J.; A Simulator of Multiple Interactive Users to Drive a Time-Shared Computer System. MAC TR-58, MIT, (1969).
9) Grochow, J. M.; The Graphic Display as an Aide in the Monitoring of a Time-shared Computer System. MAC TR-54, MIT, (1968).
10) Saltzer, J. H. and Gintell, J. W.; The Instrumentation of Multics. Comm. ACM, Vol. 13, No. 8, pp. 495-500, (1970).
11) Marphy, J. O. and Wade, R. W.; The IBM 360/195 in a world of mixed jobstreames. Datamation, April 1970, pp. 72-79.
12) 渕，田中，真子，弓場，古川；ETSS の解析（その 2）——GPSS/360 による計算機操作システムの記述と解析について——電試い報，Vol. 34, No. 2, pp. 151-168, (1970).
13) 三上，久保，高橋，有福，北浦；コンピュータシステム性能評価シミュレータ PACSS. 情報処理，Vol. 12, No. 1, pp. 14-25, (1971).
14) Seaman, P. H. and Soucy, R. C.; Simulating operating systems. IBM Sys. J., Vol. 8, No. 4, pp. 265-279, (1969).
15) Noe, J. D. and Nutt, G. J.; Validation of a trace-driven CDC 6400 simulation. SJCC 1972, pp. 749-757.
16) 相磯；計算機システム・シミュレーションに関する研究．電試研究報告，第 703 号，(1969).
17) 加藤，川合，大石，古川，山崎；非同期論理回路のグラフィック・シミュレーション・システム．電子計算機研究会資料 EC 72-15 (1972-電子通信学会，(1972).
18) Mattson, R. L.; Evaluation of Multilevel Memories. IEEE Trans. Magnetics, Vol. MAG-7, No. 4, pp. 814-819, (1971).

（昭和 47 年 8 月 4 日 受付）