計算機制御システムにおける負荷評価法の開発

石田秀昭　福岡和彦　林利弘
（日立製作所　システム開発研究所，大みか工場）

1. まえがき.

オンライン計算機システムの一分野に、プロセス制御システムがある。鉄鋼・電力・化学など生産工場の工程管理や装置制御を行うシステム、交通関係の運行管理システムなど、プロセス制御の対象範囲は広く、年々、その数は増加している。

プロセス制御システムには、他のオンラインシステムにない顕著な特徴がある。それは、計算機に対して厳しい応答性を要求することである。つまり、計算機は、課せられた多くの仕事を、仕事毎に定まった応答時間内に処理しなければならず、その許容応答時間を越えて応答することは許されない。また、計算機を高い稼動率（高負荷）で有効に活用することが要求される。この相反する2つの要求を満足させなければならないところに、計算機制御システムの設計の難しさがある。

しかし、システム設計は、設計者の経験と勘により行われているのが現状である。このため、システムの実稼動前に、計算機負荷と、プログラムの応答時間の分析・評価を行い、定量データを得ることは、重要である。これを実現する手段として、CLS（Computer Load Simulator）と名付ける計算制御負荷評価シミュレータを既に、開発している。[1]

本文では、まず、CLSの計算機制御システム設計への適用について述べる。次いで、計算機制御システム負荷評価シミュレーション上の問題点を述べ、それを解決する手段として開発したシミュレーション方法の考え方と、具体的な手順について述べていく。

2. 負荷評価プログラムの現状.

応答性と負荷の両面から見て、適切な計算機制御システムとするために、既にCLSと名付ける計算機制御システム負荷評価シミュレータを開発している。このシミュレータは、制御用計算機のシステム・プログラムと等価な機能をもつモデルを内蔵している。そして、割込みレベルや、プログラムの優先レベル、コア配置等のアプリケーション・プログラムの情報と、コア容量、ドラム容量、データ転送速度、入出力機器の個数等のハードウェア情報を入力すれば、実際のオンラインの動きと等価な動きをシミュレータ上に実現する。しかも、実時間より、はるかに速く実行し、各プログラムの起動毎の応答時間と、計算機負荷の変動について、詳細なデータを提供する。

CLSでシミュレーションを行うに必要な入力情報は、次の通りである。

(i) 使用計算機種の指定.
(ii) ハードウェア構成の指定と、データ転送速度.
(iii) プロセスからの処理要求割込み時刻の指定.
(iv) タスク相互の関連を表わす情報と、タスク処理内容.
(v) シミュレーション時間範囲指定.
(vi) 出力要求フォーマット.

ここで、タスクとは、計算機制御システムを構成する複数個のアプリケーション・プログラムである。

また、ＣＬＳからの出力情報は、次の通りである。

(i) 入力リスト.
(ii) 計算機各要素の稼動率.
(iii) タスク個々の応答時間.
(iv) タスクの待時間.
(v) 制御の流れ.
(vi) 負荷の時間変動.

ところで、システム設計から実稼動に至る各段階において、シミュレータに与えることのできる情報、シミュレーションから得られる情報は、その精度、種類において異なる。ＣＬＳが、適用できる場面と効果は次の通りである。

(1) システム設計初期. ソフトウェア大まかな構成を与えて、計算機各要素の稼動率を得て、過不足のないハードウェア構成の決定を行う。

(2) システム設計後期. 詳細なソフトウェア構成、ハードウェア構成を与えて、各プログラムの応答時間を得、応答性を満足するシステム構成の決定を行う。

(3) システム稼動後. システム拡張、あるいは変更時のソフトウェア構成、ハードウェア構成を与えて、応答性と負荷変動を得、変更後のシステム構成の確認を行う。

このように、ＣＬＳは、ソフトウェア構成、ハードウェア構成について、粗い情報でも、詳細な情報でも入力できるため、上記の、どの段階にも適用可能である。更に、入力情報が詳細であればある程、応答性解析、負荷分析も現実に近い詳細さで行える点に特徴がある。図１に、ＣＬＳの構成を示す。

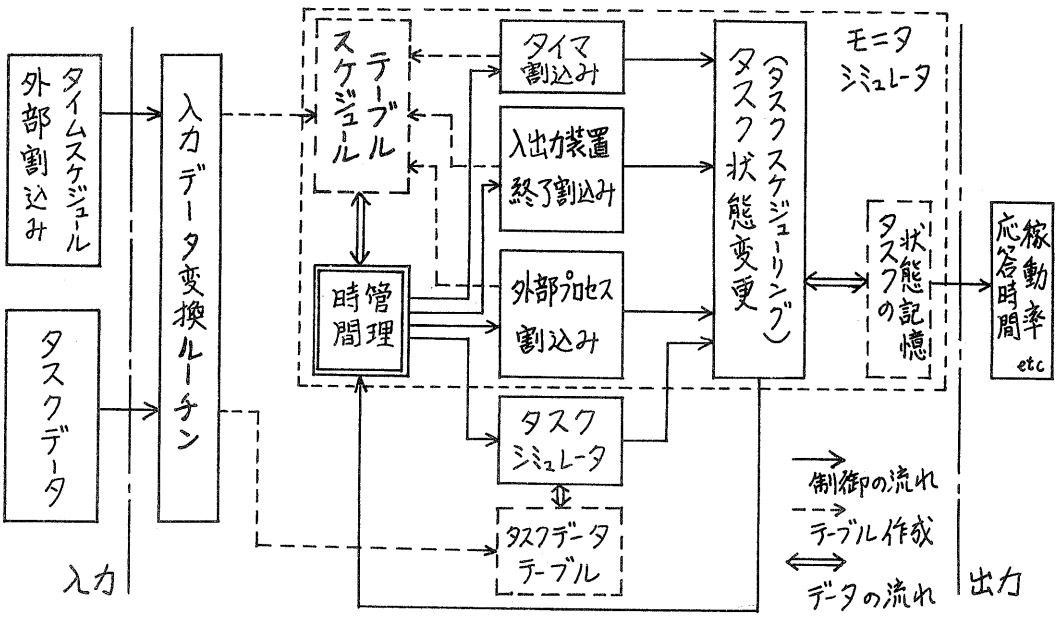


図１．ＣＬＳの構成.

3. 負荷評価プログラムの問題点.

CLSを用いて、シミュレーション、特に、システム設計後期、及び、システム稼動後の段階で適用した結果、次の問題点が明確となった。

(1) シミュレーション時間が長い。

(2) 1回のシミュレーションでは、特定条件下の応答時間、計算機負荷しか得られない。

以下、理由を述べる。

通常、1システムは、百数十個のタスク(アプリケーション・プログラム)よりなり、各タスクの内容は、起動頻度と処理時間において、多種多様である。また、一定周期毎に起動される周期タスクもあれば、ランダムに起動されるタスクもある。タスクiの応答時間R_iは、次式で表わされる。

$$R_i = f(Z_i, W_i) \qquad (1)$$

ここで、応答時間とは、要求が出されてから、その要求に応答する迄の時間である。Z_iは、タスクiの単体処理時間で、計算機内に、そのタスクしか存在しないとした時の、仕事を開始してから、終了する迄の時間である。W_iは、タスクiが他のタスクにより待たされる時間である。

ある1タスクの応答時間は、それが起動された時点の環境、つまり、その時、処理中である他のタスク、あるいは、その直後、起動される優先レベルの高いタスクに、大きく影響を受ける。タスクの起動周期は多様であるため、シミュレーション中のあらゆる時刻において、タスク起動下の環境が異なる。このため、シミュレーション時間を長くする必要が生じるが、いくら長くしたところで、応答時間の平均値さえ、満足に得られない。

また、起動頻度は非常に低い(例えば、1回/1日)が、処理時間が長いため、そのタスクより優先レベルの低いタスクの応答性に強い影響力を与えるタスクの存在がある。そして、それらのタスクと競合した時の応答時間を求めるため、初期起動タイミング等のパラメータを、いろいろ変化させる必要がある。しかし、1回のシミュレーションでは、特定条件下の設定しかできないため、シミュレーションを繰り返し行わなければならない。

4. 負荷評価方法.

上記の問題点を解決するためのシミュレーション方法について述べるに先立ち、次の前提と仮定を設ける。

(前提)

あるタスクの応答性解析を行うこととは、そのタスクの

(1) 平均応答時間.

(2) 最大応答時間.

を求めることであるとする。

応答性解析を行うタスクを対象タスクと呼ぶ。各タスクは、固有の許容応答時間を持ち、許容応答時間内に応答が完了すれば、応答条件が満たされたと考えることができる。対象タスクの応答条件が満たされているか否かの判定のために、平均応答時間と最大応答時間が必要である。

(仮定)

(1) 各タスクの単体処理時間は、タスク毎に一定とする。

(2) 各タスクは優先レベルを持ち、下位レベルのタスクが上位レベルのタスクを待たすことは、無いものとする。

(3) タスクには、周期起動とランダム起動の2種類がある。周期起動タスクについて、起動周期が与えられるように、ランダム起動タスクについても、平均起動周期が与えられるとする。

(4) タスクは、最小周期と呼ばれる、特定の時間幅の整数倍の時刻で起動され、その中間時刻では起動されないとする。また、各タスクの許容応答時間は、最小周期の整数倍で表現されるものとする。

各タスクの1回の仕事に要する処理時間(単体処理時間)は、起動毎に異なる。その理由は、タスクは、周囲のデータ内容を見て仕事を行なっているが、その周囲のデータは、時間と共に変化するためである。しかし、ここでは、各タスクは、平均的な処理時間で仕事をするとし、タスク毎の処理時間は一定とした。(仮定1)

同じコア・エリアを使用する2個のタスクのうち、下位レベルのタスクがコア・エリアを占有している時、上位レベルのタスクが起動され、退避条件が成立したならば、下位レベルのタスクは退避エリアに退避される。しかし、退避エリアが既に使用されている場合は、退避できず、上位レベルのタスクは、下位レベルのタスクに待たされることとなる。これは、退避エリアの増加等により、この不合理を回避できると考える。(仮定2)

ランダム起動タスクの起動頻度についても、大まかな起動周期値が与えられるものとした。(仮定3)

タスクの起動時刻を記述する単位として、最小時間間隔(最小周期)を設定する。これは、応答性解析に、どの程度迄の精度を必要とするかにより決定される。(仮定4)

4.1 平均応答時間.

各タスクの起動周期は、数百ミリ秒から、1日迄、多様である。従って、オンライン稼動中における対象タスクの平均応答時間を、シミュレーションで求めるためには、シミュレーション時間を、システムを構成する全タスクの最大周期以上にしなければならず、非常に長くかかる。このため、以下に述べる方法によるシミュレーション時間の短縮化を考えた。まず、平均応答時間を求める手順について述べ、そのあとで、その理由について逐次説明する。

(手順)

(1) 対象タスクと、その周期以下の周期を持つタスクのみに限定(このタスクの集合をタスク群Aとする)して、シミュレーションを行う。その結果得られる計算機負荷と、対象タスクの応答時間により、求まる点をPとする(図2)。なお、シミュレーションの設定は、次の通りとする。

a) シミュレーション時間:対象タスクの2周期分.
(P点の計算機負荷、応答時間は、2周期目の値を採る)

b) 各タスクの初期起動タイミング:
・周期起動タスク----計算機負荷を分散化するよう初期起動タイ

ミングを決定するアルゴリズムで決定.[2)]

・ランダム起動タスク----対象タスクの起動タイミングに一致させる.

(2) (1)と同じタスク構成のもとで、各タスクの処理時間のみを変化させて、シミュレーションを繰り返す。その結果得られる複数個の計算機負荷と、対象タスクの複数個の応答時間の値により、求まる応答曲線を l_0 とする(図2)。タスクの処理時間の変化方法は、各タスク一様に α 倍する(α は、0.5位から2.0位迄ふらせる)

(3) 対象タスクより周期が長く、対象タスクより優先レベルの高い、全てのタスク(このタスクの集合をタスク群IBとする)による単位時間当りの平均負荷 ΔL を式(2)より求める。T_b, Z_bは、それぞれ、タスク群IBに含まれるタスクbの起動周期、単体処理時間である。

$$\Delta L = \sum_{b \in IB} \frac{Z_b}{T_b} \qquad (2)$$

応答曲線 l_0 上で、計算機負荷 $L+\Delta L$ の点をQとする(図2)。

(4) 点Qの応答時間値を、対象タスクの平均応答時間とする。

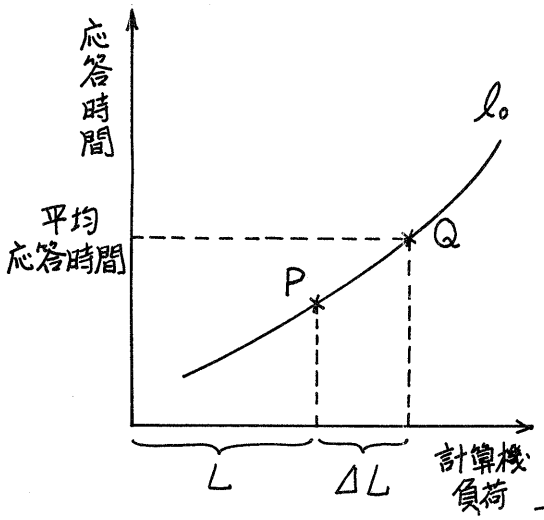


図2 平均応答時間.

点P ; 動作タスク IA
タスク単体処理時間 Z_a

曲線 l_0 ; 動作タスク IA
タスク単体処理時間
$Z_a \times \alpha$ $(\alpha > 0)$

$$L = \sum_{a \in IA} \frac{Z_a}{T_a}$$

$$\Delta L = \sum_{b \in IB} \frac{Z_b}{T_b}$$

T_0 ; 対象タスクの起動周期.
IA ; $T_a \leq T_0$ のタスク群.
IB ; $T_a > T_0$, 対象タスクより高優先レベルのタスク群.
T_a, T_b ; タスクa,bの起動周期.
Z_a, Z_b ; タスクa,bの単体処理時間.
a, b ; タスク群IA, IBに属するタスク.

(理由)

a. 手順(1)で、シミュレーション・タスクをタスク群IAに限定する理由.

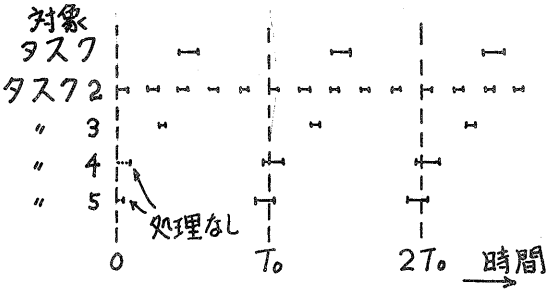
シミュレーションを行うタスクを、対象タスクの周期以下に限定すれば、計算機負荷も、対象タスクの周期と同一周期で変化するため、対象タスクの応答時間は、各周期で全く同じ値をとる。従って、シミュレーション時間は、対象タスクの一周期分でよい。[注1)] このシミュレーションにより得られた点Pは、対象タスクを最大周期とするシステムにおける、対象タスクの平均応答時間を示す点と考えられる。

注1) ただし、シミュレーション時間は、b.で2周期分に訂正される。

b. シミュレーション時間.

シミュレーション時間は、次の理由により、対象タスクの2周期分とする必要がある。

図3は、時間を横軸にとり、タスク毎の動きを示した一例である。5個のタスクの中で、タスク4と5は周期の整数倍の時刻前後で処理しているため、シミュレーションの最初では処理されず、その分だけ、計算機負荷は軽くなる。従って、一周期目の対象タスクの応答時間は、正しいと判断できない。このため、シミュレーション時間を対象タスクの2周期分とし、応答時間は、2周期目の値を採る。



T₀; 対象タスクの周期.

図3. タスクの動き

c. ランダム起動タスクの初期起動タイミングの決定.

ランダム起動タスクは、プロセス割込み等の外部条件によって起動されるため、初期起動タイミングを決定することができない。このため、ここでは、対象タスクの応答性に強く影響を与える最悪の状況を想定しなければならないと考え、ランダム起動タスクと対象タスクの初期起動タイミングを一致させることとした。

d. タスク群Bのタスクによる負荷の増加分が、ΔLとなる理由.

タスク群Bの各タスクは、対象タスクの起動周期以上の周期を持つため、タスク群Bを含めたシミュレーションを長時間行えば、対象タスクの一周期内の全ての時点で、等確率に起動され得ると考えることができる。このため、タスク群Bのタスクによる負荷の影響は、対象タスクの一周期内の全ての時点で、同程度の大きさと考えることができる。

e. 点Qが対象タスクの平均応答時間となる理由.

a.で述べたように、点Pの応答時間は、対象タスクを最大周期とするシステムにおける対象タスクの平均応答時間である。そして、応答曲線 l_0 は、P点の状態に、更に計算機負荷が加わっていった時の応答時間の推移をあらわしている。従って、P点の計算機負荷Lに、タスク群Bのタスクによる計算機負荷の増加分ΔLを加えたL+ΔLを計算機負荷とする点Qは、対象タスクの平均応答時間を与える点と考えることができる。

4.2 最大応答時間.

起動頻度は非常に小さいが、処理時間が長く、優先レベルも高いため、タスクの応答性解析上、影響力をもつタスクが存在する。対象タスクが、それらのタスクと競合した場合、対象タスクの方が優先レベルが低いと、その時の応答性は非常に悪くなる。各タスクには許容応答時間が与えられている。平均的には、許容応答時間内に応答していたとしても、このような応答性に影響力をもつタスクと競合した場合のように、突発的に応答時間が悪くなって、許容応答時間を越える

応答が起ったのでは、プロセス制御の意味をなさない。このため、最大応答時間を求める必要があるが、対象タスクと、影響力をもつタスクが競合するように、種々のケースを想定し、パラメータを設定し直して、繰り返しシミュレーションを行う必要が生じる。そこで、次の方法により、その思考錯誤的な方法の回避を図った。

(手順)

(1) 特定の処理時間を持ち、最上位の優先レベル、コア常駐の仮タスクを、処理時間を変えて複数個用意する。仮タスクの処理内容については、コア処理ステップ数とドラム転送処理ステップ数の構成比等を、次の要領で決定する。

a) タスク群IBのタスクのうち、処理時間の長いタスクの構成に類似したものとする。

b) 処理時間を簡単に算出可能とする。

(2) (1)で作成した仮タスク i ($i=1,2,\cdots$) とタスク群IAのタスクとにより、シミュレーションを行う。(シミュレーション時間、タスクの初期起動タイミングは、平均応答時間を求めた場合に同じ) 応答曲線 l_0 を作成した場合と同様の方法で、各タスクの処理時間を変化させて、仮タスクの個数分の応答曲線 l_i ($i=1,2,\cdots$) を作成する(図4)。また、l_i と仮タスクの処理時間 Σi の対応づけをしておく。

(3) タスク群IBの中から、最も長い処理時間 Σc をもつタスク(最も応答性に影響力をもつタスク) c をみつける。または、タスク群IBの複数個のタスクが、時間的に接近して起動される可能性があり、それらの処理時間の和 $\Sigma c'$ が Σc より大きい時は、$\Sigma c'$ を Σc とする。

(4) 処理時間 Σc に対応した応答曲線 l_c 上で、計算機負荷が $L+\Delta L$ となる点を R とする(図4)。ΔL は、式(2)で求めたのと同じである。この時、Σc に対応した応答曲線 l_c が無い時は、応答曲線間で補間する。

(5) 点 R の応答時間値を、対象タスクの最大応答時間とする。

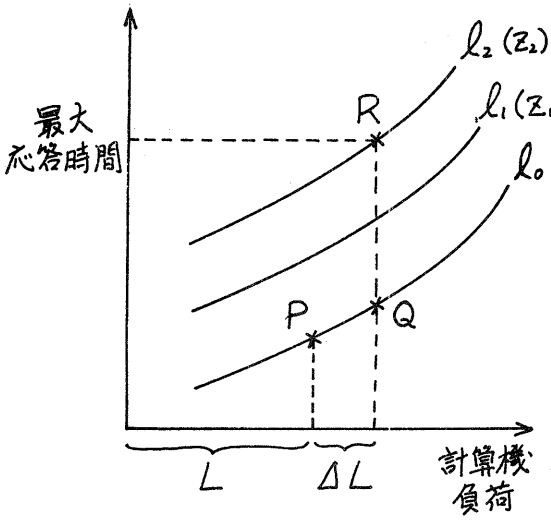


図4. 最大応答時間.

$l_2(\Sigma_2)$ 曲線 l_i; 動作タスク IA + i
タスク単体処理時間.
$\Sigma a \times \alpha$
$\Sigma i \times \alpha$ ($\alpha > 0$)

L
ΔL } 図2.に同じ.

IA; $Ta \leq To$ のタスク群.
To; 対象タスクの起動周期.
Ta; タスク a の起動周期.
a; タスク群IAに属するタスク.
i; 単体処理時間を Σi とし、対象タスクより高優先レベルのタスク. ($i \in IB$)
Σa; タスク a の単体処理時間.

(理由)

a. 手順(1)で仮タスクを用いた理由.

実タスクの代りに仮タスクを用いてシミュレーションを行なったのは、タスク群Bに属する単体タスクの影響だけでなく、複数個のタスクが時間的に接近して一時的に高負荷になった場合の影響についても、適用可能と考えたためである。

b. 手順(2)で応答曲線を作成しておく理由.

応答曲線を、タスク・データがそろった段階で作成しておけば、オンライン稼動時に起こりそうな、一時的に高負荷になる種々のケースがわかる毎に、応答曲線を見ただけで、応答性を把握できる。逐次、シミュレーションをやり直す必要はない。

以上の負荷評価方法の手順を示したのが、図5である。

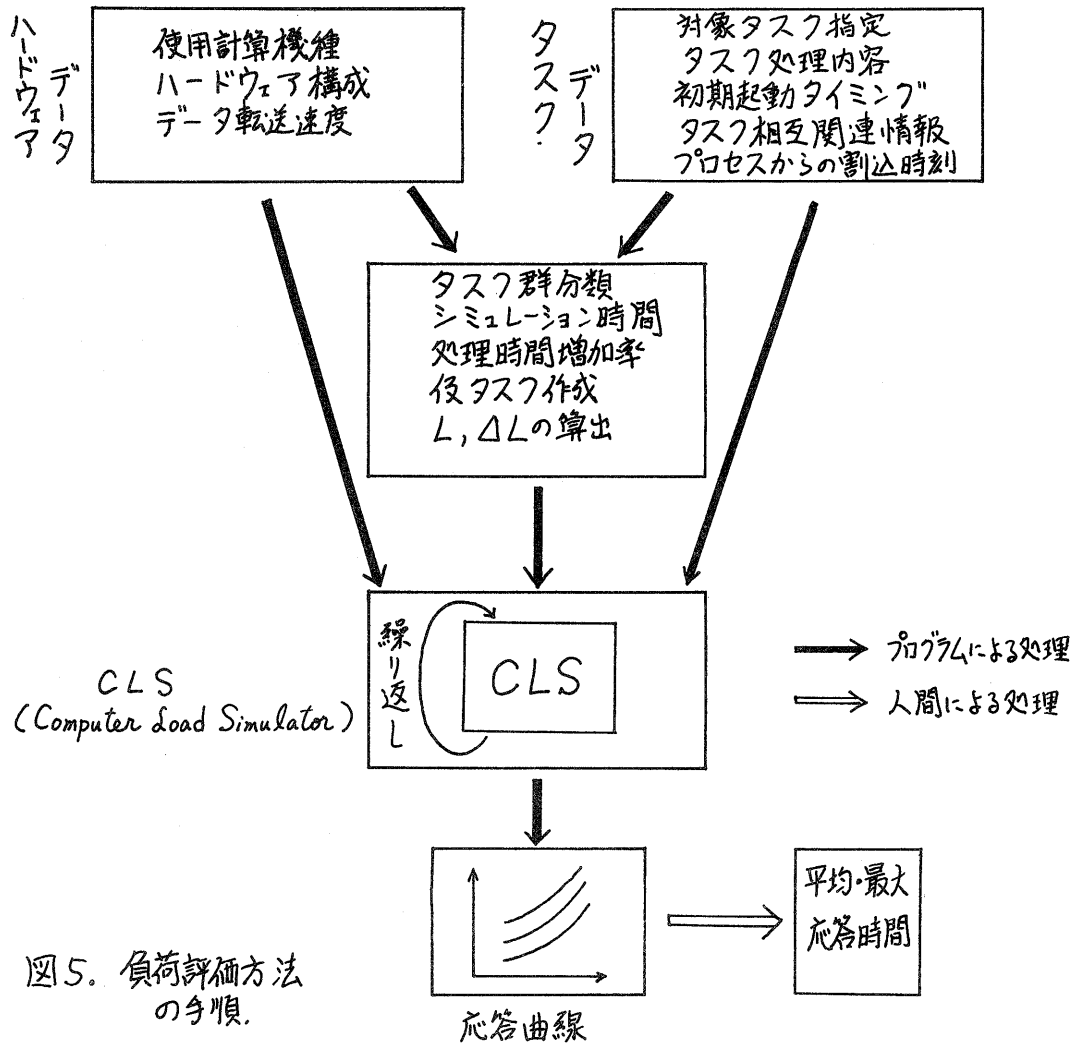


図5. 負荷評価方法の手順.

5. 適用例.

図6.1は、CRT応答曲線である。実測値3～4秒のところ、シミュレーションの結果、3.9秒と、平均応答時間において、実測値にほぼ近い値が得られている。図6.2は、最下位レベルのタスクの応答曲線である。

本方式によるシミュレーションにおいて、応答性解析を行うタスク、つまり、対象タスクとしては、CRT処理タスクと最下位レベルタスクの指定が多いと思われる。その理由は、次の通りである。

(CRT処理タスク)

1) プロセス制御におけるマン・マシン・コミュニケーションの設計項目として重要な位置を占めている。
2) オンライン稼動中にあっても、簡単に実測値を得ることができる。
3) 全ての計算機制御システムに存在し、システム毎の処理上の大きな違いもないため、システムの応答性解析の一つの尺度となり得る。

(最下位タスク)

1) システムの応答性解析上、重要である。

もちろん、各タスクの許容応答時間を満たしているか否かの解析を行うことは重要である。

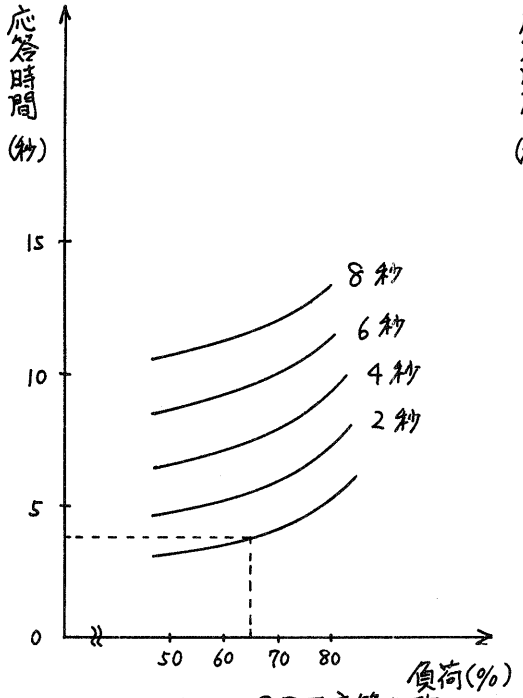


図 6.1 CRT応答曲線.

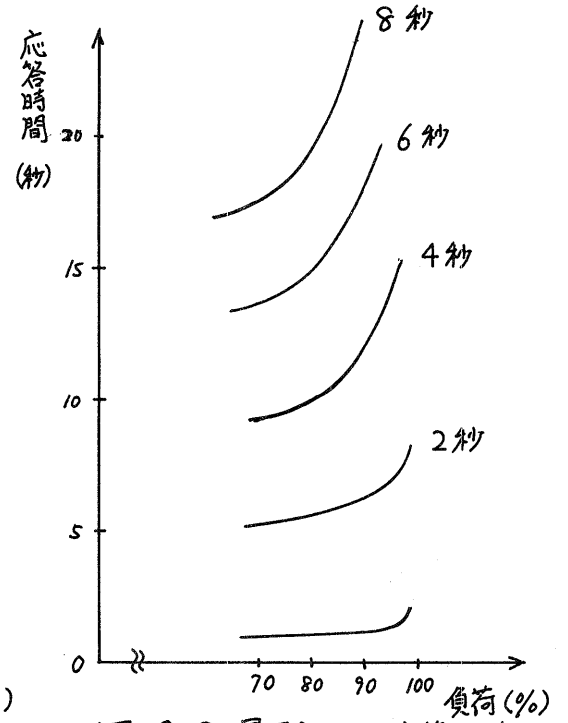


図 6.2 最下位タスク応答曲線.

6. むすび.

計算機制御システムの計算機負荷と応答性を評価する手法として、シミュレーションにより、平均応答時間と最大応答時間を求める手法を開発した。本方式の特徴は、(1) シミュレーション時間が短くて済むこと、(2) シミュレーションの結果を元に、パラメータを設定し直して、繰り返しシミュレーションを行う必要が無いことである。

〈参考文献〉
1) 福岡・三森・三巻　「制御用計算機負荷シミュレータ CLS」
電気学会誌 Vol 91, No.8 (1971)
2) 近藤・福岡・林　「計算機制御システムにおけるプログラム起動
タイミング決定手法の開発と適用」
昭和52年情報処理学会沖18回全国大会