LCMPシステムにおける自動負荷均衡化機能

山路英一　大野美恵　宮入勉
（(株)日立製作所 中央研究所）（同 ソフトウェア工場）

1. はじめに

1台のプロセサから構成されるシングル・プロセサ・システムの処理能力面でのあい路を解消するため、または、信頼性向上を図るために、今後、LCMP（Loosely Coupled Multi Processor）システムをはじめとする複合計算機システムが増えていくことが予想される。複合計算機システムを構成し十分な性能を引き出すためには、システムにかかる負荷を各プロセサにバランスよく分配し、各々のプロセサにその能力を最大限に発揮させることが重要な問題となる。

これに対し、自動負荷均衡化（Enhanced Workload Balancing：EWB）機能は、LCMPシステムに焦点を絞り、バッチあるいはバッチ／TSS（Time Sharing System）混在システムにおいて、システム内の各プロセサにかかる負荷を自動的に均衡化することにより、システム全体のスループットおよびTAT（Turnaround Time）の向上を図るとともに、プロセサ間のTATのばらつきを抑えることを目的としている。

本報告では、LCMPシステムをサポートするジョブ入出力サブシステムJSS4（Job Spooling Subsystem 4）の現状のジョブ・スケジューム方式を述べた後、JSS4のもとでEWB機能を実現した場合の処理方式とその効果について述べる。

2. JSS4システムにおける現状のジョブ・スケジュール方式

2.1 JSS4システムの構成

JSS4は、HITAC Mシリーズの大型プロセサ（M－200H／180／170／160II）でVOS3（Virtual Storage Operating System 3）のもとで稼動し、8台までのプロセサでスプール・ボリューム（ジョブ制御文，SYSIN，SYSOUTデータを格納するスプール・データ・セットが置かれる）を共用して、ジョブ処理を分配することができる。

この場合、1台のグローバル・プロセサがジョブの入出力とローカル・プロセサへのジョブの分配を行なう。ジョブの実行はグローバル・プロセサとローカル・プロセサのいずれでも行なうことができる。

図2.1に3台のプロセサからなるJSS4システムの構成例を示す。グローバル・プロセサと各ローカル・プロセサはSYSIN，SYSOUTデータの受け渡しのためチャネル・スイッチを経由してスプール・ボリュームを共用する他に、制御情報（ジョブの選択要求，ジョブの終了通知，その他のサブシステム・イン

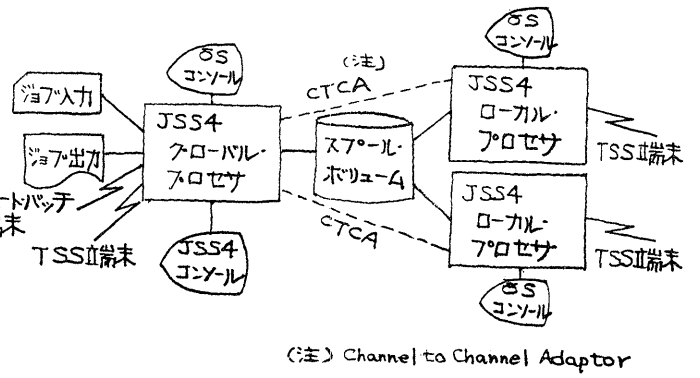


図2.1 JSS4システムの構成例

タフェイス用の制御情報）の受け渡しのためにＣＴＣＡ（Channel to Channel Adaptor）を介して接続されている。

ジョブの入出力はグローバル・プロセサで行なうが、ＴＳＳ端末はローカル・プロセサにも接続可能であり、この端末から会話型リモートバッチ機能を用いてジョブを登録することができる。また、ローカル・プロセサ中で実行中のプログラムから内部リーダ機能を用いてジョブを登録することもできる。

コンソールについては、各プロセサに接続される従来のＯＳコンソール（各プロセサのＩＰＬ（Initial Program Load）は、このＯＳコンソールを用いて行なう。また、必要なら当該コンソールからＯＳコマンドを投入できる）の他、グローバル・プロセサにはＪＳＳ４コンソールが接続され、このコンソールからＪＳＳ４コマンドあるいは各プロセサに対するＯＳコマンドを投入することによりシステム全体を１システム・イメージで制御することができる。

２．２　ＪＳＳ４におけるジョブの処理手順

図２．２にＪＳＳ４におけるジョブの処理手順を示す。ジョブの処理手順は以下に示すように大きく３つのフェーズに分けられる。

（１）Pre-Execution フェーズ――ジョブ入力，ジョブ制御文解析，デバイス・スケジューリング，ジョブ・スケジューリング

（２）Execution フェーズ――ジョブ実行

（３）Post-Execution フェーズ――ジョブ出力，ジョブ・パージ

上記３つのフェーズのうち、Pre-Execution フェーズおよびPost-Execution フェーズはグローバル・プロセサのみで処理され、Execution フェーズはいずれのプロセサでも処理される。

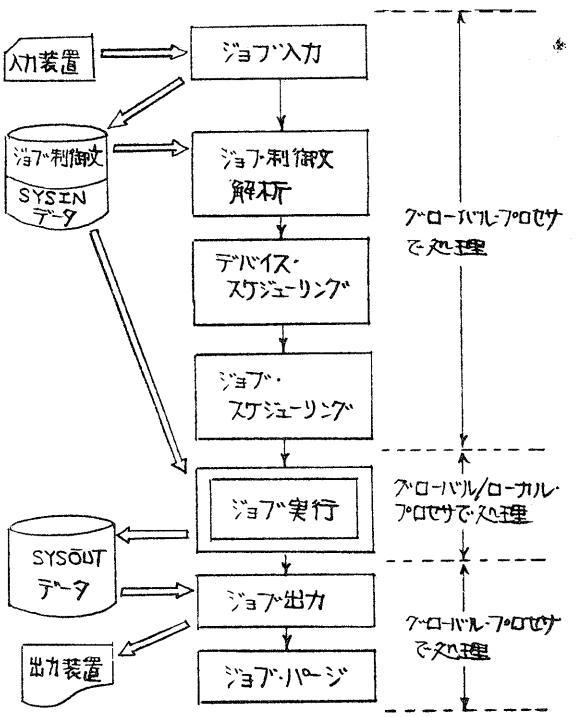


図2.2　JSS4におけるジョブの処理手順

２．３　現状のジョブ・スケジュール方式

２．３．１　ジョブ・クラス，ジョブ・クラス・グループとイニシエータの割当て

ジョブ・クラスはジョブをいろいろな属性に従ってグループ分けするために使用するもので、グループ間のシステム資源の競合を防止したり、ジョブに対する一定のサービスを保証することを目的として設定される。

ジョブ・クラス・グループは複数個のジョブ・クラスを集めたものであり、ジョブの実行に必要なイニシエータの割り当ては、ジョブ・クラス・グループ対応にプロセサごとのイニシエータの本数を指定することにより行なわれる。

２．３．２　現状のジョブ・スケジュール方式

ジョブ実行のスケジュールはＪＳＳ４のＪＳＰ（ＪＳＳ４ Support Pro-

gram：ＪＳＳ４の機能を実行するプログラム）の１つであるＧＭＳ（Generalized Main Service）により行なわれる。

ＧＭＳは、各プロセサのイニシエータからジョブの選択要求があると、そのイニシエータに対応するジョブ・クラス・グループに属するジョブの中から、各ジョブのジョブ優先順位，必要な論理リージョン・サイズ，入出力割合の大小等をもとに最適なジョブを選択するが、当該プロセサにおけるＴＳＳジョブを含めた実負荷とは無関係に、予め指定されたイニシエータ数に応じてジョブがスケジュールされる。

従って、各プロセサにおける負荷の変動（例えば、アクティブなＴＳＳ端末の増減）に対処するため、オペレータがその都度イニシエータの数を変更する必要がある。

３．ＥＷＢ（自動負荷均衡化）方式

ＥＷＢ方式は、前章で述べたオペレータ操作を自動化するためのものであり、各プロセサにかかる負荷を定期的に把握してバッチ・ジョブの処理多重度を動的に制御することにより、システム全体のスループットおよびＴＡＴの向上を図るとともに、プロセサ間のＴＡＴのばらつきを抑えるものである。

３．１　動作環境の設定

ＥＷＢは、システムが使用される環境に応じて、ユーザ（センタ管理者）がＥＷＢの動作環境（ＥＷＢ制御の対象となるプロセサ，多重度制御を行なうジョブ・クラス・グループ，多重度制御の方法等）を設定できるようにＥＷＢ定義パラメタを用意している。

ＥＷＢ定義パラメタは表３．１に示すように３種類あり、それらが集まってパラメタ・ライブラリ（ＳＹＳ１．ＰＡＲＭＬＩＢ）の１メンバＪＳＳ４ＷＢｘｘを形成する。

ＪＳＳ４ＷＢｘｘは複数個登録でき、どのメンバを使用するかはＥＷＢの起動時ならびにＥＷＢの動作環境の変更時に指定できる。

表3.1 EWB定義パラメタの仕様

パラメタ	サブパラメタ
PROC	NAME＝プロセサ名 ,GMIX＝グループ・ミクス定義名 [,MPL＝(多重度下限値，多重度上限値)]
GROUPMIX	NAME＝グループ・ミクス定義名 ,GROUP＝(ジョブ・クラス・グループ名[,ジョブ・クラス・グループ名,…]) ,MIX＝(イニシエータ比率[,イニシエータ比率,…])
OPT	[TRACE＝{OFF / ON}]

以下、ＥＷＢ定義パラメタについて簡単に説明する。

（１）ＰＲＯＣパラメタ

ＥＷＢ制御の対象となるプロセサ毎に、多重度制御を行なうジョブ・クラス・グループ，多重度制御の方法を指定する。

（ａ）ＮＡＭＥサブパラメタ――ＥＷＢ制御の対象となるプロセサ名を指定する。

（ｂ）ＧＭＩＸサブパラメタ――多重度制御を行なうジョブ・クラス・グループについて定義するＧＲＯＵＰＭＩＸパラメタの定義名を指定する。

（Ｃ）ＭＰＬサブパラメタ――多重度を変える範囲を指定する。

（２）ＧＲＯＵＰＭＩＸパラメタ

多重度制御を行なうジョブ・クラス・グループを定義するとともに、アクティブなイニシエータの数に応じたジョブ・クラス・グループ・ミクス（ジョブ・クラス・グループごとのイニシエータ数の相対的比率）を定義する。

（a）NAMEサブパラメタ――グループ・ミクス定義名を指定する。

（b）GROUPサブパラメタ――多重度制御を行なうジョブ・クラス・グループ名を指定する。

（C）MIXサブパラメタ――アクティブなイニシエータの数に応じて、ジョブ・クラス・グループごとのイニシエータ数の相対的比率を指定する（GROUPサブパラメタで指定された順に）。

（3）OPTパラメタ

EWB制御オプションを指定する。

（a）TRACEサブパラメタ――本サブパラメタでONを指定するとEWBの動作状況がトレースされる。

3．2　EWBの起動／終了と動作環境の変更

3．2．1　EWBの起動

EWBは、¥CALLコマンド（JSPを起動するJSS4コマンド。表3．2参照）により起動される。

¥CALLコマンドが投入されるとEWB　JSP（新設）が起動され、指定されたJSS4WBxxメンバのEWB定義パラメタに従って動作環境の初期設定を行なうとともに、STARTコマンドを内部的に発行して各プロセサの負荷計測モジュールを起動し、システム負荷の計測を開始する。

表3.2 EWBに関連したコマンドの仕様

コマンド	オペランド
¥CALL	EWB [,MBR = {OO / JSS4WBxx メンバの下2桁}]
¥CANCEL	EWB
¥START	EWB ,MBR = JSS4WBxxメンバの下2桁

3．2．2　EWBの終了

EWBは、¥CANCELコマンド（JSPを終了させるJSS4コマンド。表3．2参照）により終了される。

¥CANCELコマンドが投入されるとEWB　JSPはSTOPコマンドを内部的に発行して各プロセサの負荷計測モジュールを終了させた後、終了する。

3．2．3　動作環境の変更

EWBは、¥STARTコマンド（JSPを再開始させるJSS4コマンド。表3．2参照）により動作環境を変更できる。これを用いて、一日の時間帯により動作環境を変える運用を可能にしている。

3．3　システム負荷の計測

3．3．1　概要

プロセサごとに、定期的に、次の負荷情報を収集する。

（1）ESR（Effective　Service　Rate）関連情報

（2）CPU負荷情報

（3）ページング情報

これらの負荷情報の収集は、負荷計測モジュールが、RCM（資源集中管理）およびスーパバイザで収集しているデータをもとに行なう。

収集した負荷情報は、サブシステム・インタフェイスを通してグローバル・プロセサのJSS4に送信される。

3.3.2 ESR関連情報

あるインタバルにおけるジョブのESRとは次の式で定義されるものである。

$$ESR \triangleq \frac{\text{当該インタバルで使用したCPU時間} + \text{当該インタバルで使用した入出力時間}}{\text{当該インタバルにおけるジョブの経過時間}^{(注1)}}$$

$$\cong \frac{\text{CPU時間} + \text{入出力1回当りの平均入出力時間} \times \text{入出力発行回数}}{\text{経過時間}} \quad (\text{EWBで使用})$$

(注1) 経過時間は、ジョブが当該インタバルの途中で開始/終了していない限りインタバルの長さに等しい。但し、端末思考時間，ロング・ウェイト時間（WAITマクロでLONG=YESを指定した時）は経過時間に入れない。

上記の定義によれば、ジョブのESRは、ジョブが単独で実行されている時のジョブの処理速度を1とした場合の処理速度の比率（処理進捗度と呼ぶ）を表わしていると考えられる（但し、同一ジョブ内のCPU処理と入出力処理のオーバラップは無視できるものとしている）。

一方、上記の式の分母をインタバル長としたものをESR'とすると、同時に実行中のジョブのESR'の合計は、ジョブが端末入出力待ち/ロング・ウェイト状態にならずに単独で実行されている場合のスループットを1とした場合のスループットの比率（スループット係数と呼ぶ）を表わしていると考えられる。

ESR関連情報として、ジョブの処理進捗度とスループット係数を収集する。

3.3.3 CPU負荷情報

CPU負荷情報としてCPUのウェイト時間を収集する。TCMP（Tightly Coupled Multi Processor）構成になっている場合は、複数台分のCPU負荷情報が収集される。

3.3.4 ページング情報

ページング情報とは、ページングに関する種々の情報であり、以下のものが収集される。

(1) ページ・イン/ページ・アウト回数
(2) ページ・リクラメーション回数
(3) スワップ・イン回数
(4) 主記憶ページ・フレーム数
(5) 補助記憶ページ・スロット数

3.4 システム負荷の制御

3.4.1 概要

各プロセサから定期的に送られてくる負荷情報をもとにプロセサにかかる負荷の適否を判断し、バッチ・ジョブの適正多重度を決定する。そして、決定された適正多重度に従ってジョブの選択を制御することにより、プロセサにかかる負荷の適正化を図る。

3.4.2 適正多重度の決定

適正多重度の決定方法を以下に示す。

(1) スループット係数が予め定められた許容値に達している場合は現在の多重度が適正であると判断する。

(2) スループット係数が許容値に達していない場合は、各ジョブの処理進捗度の平均(各ジョブの経過時間による重みつき平均)を見て、
(a) 処理進捗度が予め定められた許容範囲を下回っている場合は、現在の状態は過剰に負荷がかかっていると見做して、多重度を減らした方がよいと判断する。
(b) 多理進捗度が許容範囲を上回っている場合は、現在の状態は負荷が不足していて、まだ負荷をかけてもよいと見做して、多重度を増やした方がよいと判断する。
(c) 多理進捗度が許容範囲に入っている場合は現在の多重度が適正であると判断する。
(3) 適正多重度を決定する際の多重度増減の範囲は高々1とするとともに(判断ミスにより多重度が大きく変動するのを避けるため)、ユーザの指定した多重度変更の範囲(PROCパラメタのMPLサブパラメタ)内で行なう。
(4) 上記でプロセサの適正多重度を決定した後、ユーザの指定したジョブ・クラス・グループ・ミクス(GROUPMIXパラメタ)に従って、各ジョブ・クラス・グループの適正多重度が決定される。

3.4.3 システム負荷の制御

決定された適正多重度に従ってジョブの選択を制御することにより、プロセサにかかる負荷の適正化を図る。

具体的には、
(1) イニシエータからジョブの選択要求があった時、当該イニシエータに対応するジョブ・クラス・グループの適正多重度に達しているならば、ジョブ選択要求を一時的に抑止する(ジョブ選択要求に対する応答を一時的に保留する)。

一方、
(2) 適正多重度の増加等により上記状態が解除されたならば、ジョブ選択要求の抑止を解除する。

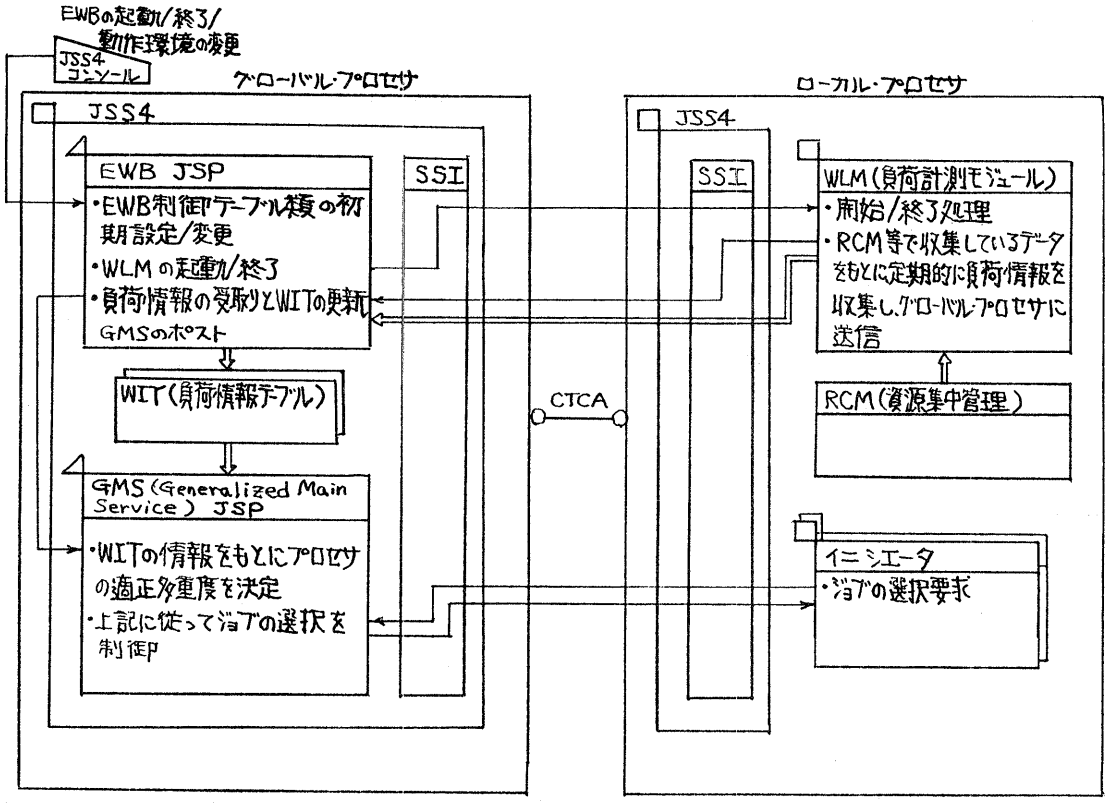
4. EWBのプログラム構造と処理の概要

図4.1にEWBのプログラム構造と処理の概要を示す。本章ではEWBのサポートに関連したモジュールのうちの主なものについて説明する。

4.1 JSS4のモジュールの構成と処理の概要

4.1.1 EWB JSP

EWBのサポートに伴う新設JSPであり、EWBを起動する¥CALLコマンドで起動される。EWB JSPの処理は次の2つに分けられる。
(1) EWBの開始/終了と動作環境の変更処理
(a) ¥CALLコマンドで起動されると指定されたEWB定義パラメタに従ってEWB制御テーブル類を初期設定するとともに、各プロセサの負荷計測モジュールを起動する。
(b) ¥CANCELコマンドが投入されると各プロセサの負荷計測モジュールを終了させた後、終了する。
(c) ¥STARTコマンドが投入されると指定されたEWB定義パラメタに従ってEWB制御テーブル類を変更する。



(注1) →は制御フロー, ⇨はデータフロー
(注2) CTCA: Channel to Channel Adaptor
(注3) SSI: Subsystem Interface
(注4) JSP: JSS4 Support Program

図4.1 EWBのプログラム構造と処理概要

(2) 負荷情報受信処理

各プロセサの負荷計測モジュールからサブシステム・インタフェイスを通して送られてくる負荷情報を受け取り、それにもとづいて負荷情報テーブル(プロセサ対応に作成される)を更新しGMSをポストする。

4.1.2 GMS JSP

ジョブ実行のスケジュールを行なうGMSに対して次の2つの処理を追加する。

(1) 適正多重度決定処理

EWB JSPからポストされた時、負荷情報テーブルの情報をもとにプロセサの適正多重度を決定する(詳細は3.4.2参照)。

(2) システム負荷の制御処理

決定された適正多重度に従ってジョブの選択を制御(イニシエータからのジョブ選択要求の一時的抑止/解除)することにより、プロセサにかかる負荷の適正化を図る(詳細は3.4.3参照)。

4.2 負荷計測モジュールの構成と処理の概要

負荷計測モジュールはJSS4のEWB JSPにより起動される新設モジュールであり、JSS4とは独立した空間で走る。

負荷計測モジュールは、負荷計測メインルーチンと、それからコールされるデー

タ収集ルーチン，データ転送ルーチンから構成される。

（1）負荷計測メインルーチン

（a）ＳＴＡＲＴコマンドで起動されると負荷計測のための用始処理（内部データ・エリアのイニシャライズ，タイム・インタバルのセット等）を行なう。

（b）一定時間間隔でデータ収集ルーチンをコールして負荷情報を収集し、データ転送ルーチンをコールして収集した負荷情報をグローバル・プロセサのＪＳＳ４に送る。

（c）ＥＷＢ ＪＳＰからＳＴＯＰコマンドが発行されると必要な終了処理（内部データ・エリアの解放等）を行なった後、終了する。

（2）データ収集ルーチン

ＥＳＲデータ，ＣＰＵデータ，ページング・データ収集ルーチンの３つがあり、それぞれ、ＥＳＲ関連情報，ＣＰＵのウェイト時間，ページングに関する種々の情報を収集する（収集される負荷情報の詳細は３．３参照）。これらの負荷情報の収集はＲＣＭおよびスーパバイザで収集しているデータをもとに行なう。

（3）データ転送ルーチン

収集した負荷情報をサブシステム・インタフェイスを通してグローバル・プロセサのＪＳＳ４（ＥＷＢ ＪＳＰ）に送信する。

5．ＥＷＢ方式の効果の予測

これまで述べてきたように、ＥＷＢ方式はＬＣＭＰシステムを構成する各プロセサの負荷状況を常時モニタリングし、過負荷の状況にあるプロセサについてはバッチの多重度を下げ，低負荷の場合にはバッチの多重度を上げて、各プロセサにかかる負荷を均衡化することにより、システム全体のスループットおよびＴＡＴの向上を図るとともに、プロセサ間のＴＡＴのばらつきを抑えることをその狙いとしている。

従って、ＥＷＢ方式を採用することにより、オペレータの手を煩わすことなく、

（1）システム用始時の各プロセサの負荷状況に応じた適正なバッチ多重度の設定

（2）システム用始後の各プロセサの負荷変動（例えば、アクティブなＴＳＳ端末の増減）に応じたバッチ多重度の動的な制御

を行なうことができる。

２５題のベンチマーク・ジョブを用いたシングル・プロセサ（Ｍ－１８０，主記憶は３ＭＢ）でのＪＳＳ４の性能測定結果にもとづいて、２台のＬＣＭＰの１台（ここではローカル・プロセサ）に負荷変動（増加）があった場合についてＥＷＢ方式の適用効果を予測した結果を図５．１に示す。

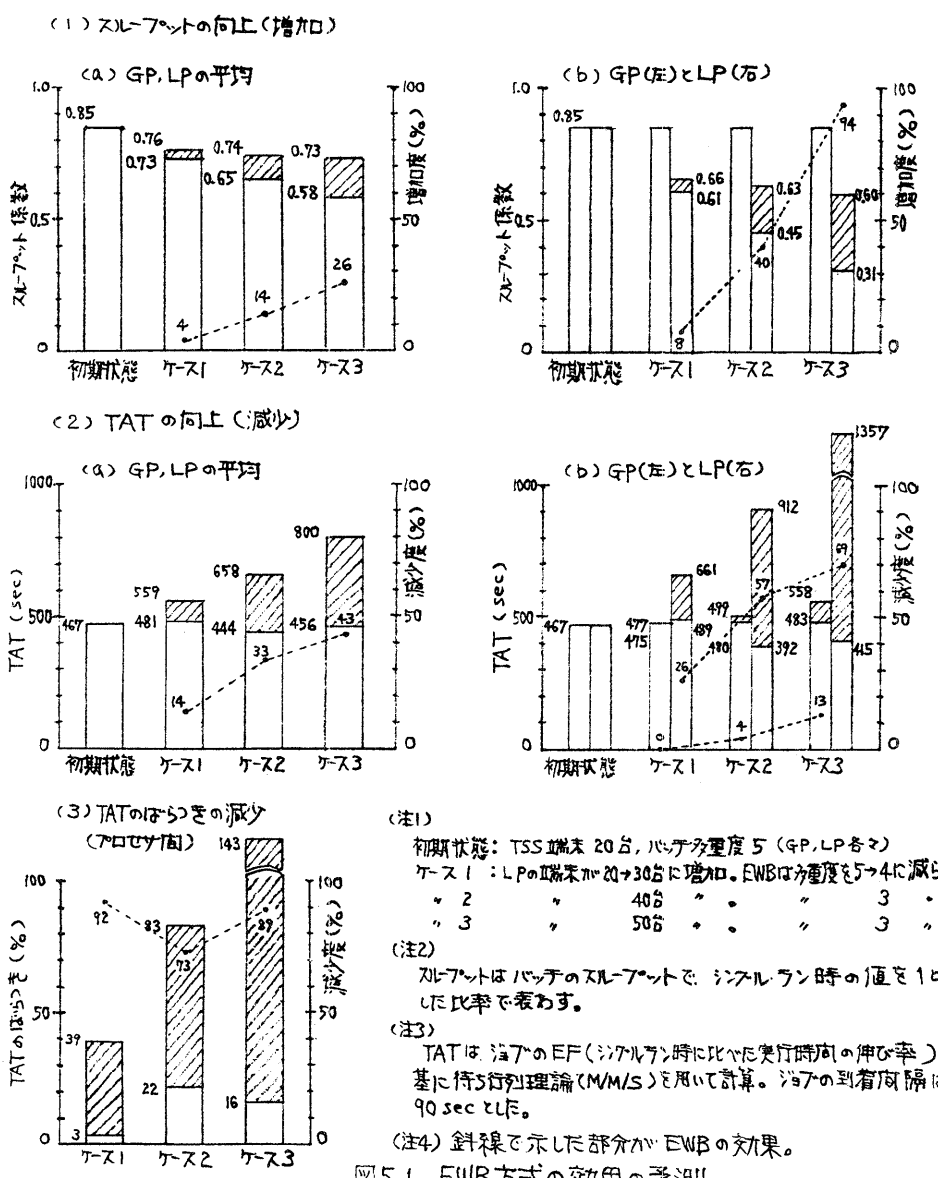
ＥＷＢ方式の効果は負荷変動の大きさと関係しており、一口には言えないが、ここで想定した３ケースについて見ると，

（1）スループットの向上　4～26％　（ローカルプロセサについて見ると　8～94％）

（2）ＴＡＴの向上　14～43％　（　〃　26～69％）

（3）ＴＡＴのばらつきの減少　73～92％

が期待される。



(注1)
初期状態: TSS端末 20台, バッチ多重度 5 (GP,LP各々)
ケース1 : LPの端末が20→30台に増加。EWBは多重度を5→4に減らす。
〃 2 〃 40台 〃 〃 〃 3 〃
〃 3 〃 50台 〃 〃 〃 3 〃

(注2)
スループットはバッチのスループットで、シングルラン時の値を1とした比率で表わす。

(注3)
TATは、ジョブのEF(シングルラン時に比べた実行時間の伸び率)を基に待ち行列理論(M/M/S)を用いて計算。ジョブの到着間隔は90 secとした。

(注4) 斜線で示した部分がEWBの効果。

図5.1 EWB方式の効果の予測

6. おわりに

LCMPシステムにおける各プロセサにかかる負荷を自動的に均衡化することにより、システム性能の向上を図ることを目的とした負荷制御方式について述べた。本方式は、中研で現在、実験システムを作成しており、今後はベンチマーク・ジョブを用いた評価のみならず、実運用環境での評価を行ない、よりよい方式へと発展させていきたい。

最後に、本方式の検討に際し貴重な御意見、御討論をいただいたソフトウェア工場の松永伍生氏、野口健一郎氏、森啓倫氏、原田晃氏、ならびに中研の堀越彌氏、本林繁氏、吉住誠一氏その他の諸氏に厚く感謝の意を表します。