バッチランの最適スケジュール作成

松田 芳雄，河岸 憲一

1．はじめに

大規模システムのバッチの運用において，終了時刻の予測や投入スケジュール作成のために，各ランの所要時間（開始から終了までの時間）の予測が必要である．所要時間はその日の処理量や同時に走行するランの組合せにより変化し，予測を困難にしている．本稿では，UNISYS2200/1100シリーズ計算機における，バッチランの所要時間の予測とスケジュール作成の方法について紹介する．

2．バッチランの運用支援システム

バッチランの効率的運用のために，スケジュールの予測と作成を行う以下の機能が必要である．

(1) スケジュール予測機能

あらかじめ決められた順序でバッチランを走行させた場合，各ランのスケジュール(開始時刻と終了時刻)を予測する機能である．バッチ運用の開始時や途中で，全体の終了時刻を予測して制限時間内に処理が終了するかの確認を行う．

(2) スケジュール作成機能

全体の終了時刻が最も早くなるように，バッチランのスケジュールを作成する機能である．作成したスケジュールどおり運転することにより，業務終了時刻を早めることができる．

スケジュールの予測，作成のどちらにおいても各バッチランの所要時間を予測することが必要となる．以下にバッチランの所要時間の予測モデルについて述べる．

3．所要時間予測モデル

3.1 予測モデル概要

バッチランの所要時間は，CPUおよびI/O装置の使用時間とその待ち時間，メモリー待ち時間から構成される（式3.1）．したがって，所要時間の予測のためには，CPU，I/O装置の使用時間と待ち時間の予測が必要である．

(所要時間)＝(CPU使用時間)＋(CPU待ち時間)
＋(I/O使用時間)＋(I/O待ち時間)
＋(メモリー待ち時間)　(3.1)

3.2 使用時間予測モデル

(1) SUP値

UNISYS2200/1100計算機では，CPU，I/O装置の使用時間はSUP（Standard Unit of Processing）値で表わされる．SUP値はCPUの使用時間であるCPU・SUP値とI/Oの使用時間であるI/O・SUP値に分れる．CPU・SUP値とI/O・SUP値の合計をTOTAL・SUP値と呼ぶ．

(2) SUP値予測の考え方

各バッチランのSUP値は実行のたびに一定ではなく，処理するデータ量により変動する．しかし，多くのランは月末や週末に処理量が集中したり，特定の日や曜日に集中したりして，規則性がある場合が多い．この規則性をモデル化し，SUP値を予測する．

(3) SUP値予測モデル

日次ランの場合，SUP値の変動要因として「月」，「日」，「曜日」を考え，i月j日k曜日のSUP値の予測値S_{ijk}を以下のように表わすことにする．

$$S_{ijk}=\mu+M_i+D_j+W_k \qquad (3.2)$$

ここで，M_i，D_j，W_kはそれぞれi月，j日，k曜日のウェイトでμは定数項である．各ウェイトおよび定数項は過去の実績データをもとに最小二乗法により推定する．

3.3 待ち時間予測モデル

(1) 待ち時間予測の考え方

バッチランの走行環境が異なれば待ち時間も異なる．

まつだ よしお，かわぎし けんいち
日本ユニシス株式会社　〒135 江東区豊洲1-1-1

したがって，待ち時間はSUP値のように過去の実績データから予測することはできない．しかし，走行環境と待ち時間の関係がわかれば待ち時間を予測することができる．ここでは，走行環境として同時走行ラン数とCPU比率（バッチラン全体のTOTAL・SUP値に対するCPU・SUP値の比率）を考える．それらと待ち時間との関係は，シミュレーションにより求める．

(2) シミュレーション・モデル

計算機システムはCPU，メモリー，I/O装置で構成されているとする．バッチランはこれらの装置に対し処理要求を出し，OSがこれを管理している．これらの装置やバッチラン，OSの動きをモデル化し，シミュレートすることにより，待ち時間を算出する．

システム内にNランのバッチランが存在するとする．NランのバッチランがCPUとI/O装置を使う比率をR：$1-R$（R：CPU比率，$0<R<1$）とする．このとき各装置での待ち時間を求めるために以下のようにモデル化する．

イ）各バッチランはR：$1-R$の比率でCPUとI/O装置を使う．

ロ）CPUまたはI/O装置が空いていれば一定の処理を受ける．

ハ）空いていないときは待ち行列に並び，空くまで待って処理を受ける．

ニ）処理の終わったバッチランはイ）に戻り，上記を繰り返す．

このモデルを計算機でシミュレーションすれば，CPUやI/O装置での待ち時間（処理要求1回当り待ち時間）を測定することができる．

(3) 走行ラン・モデル

前提として，バッチ専用機や夜間バッチ業務を対象とし，計算機システム内にはバッチランとOSだけが走行しているものとする．2200/1100計算機ではメモリー待ちはスワップアウト/インとして現われるので，このためのI/Oも走行ランとして考える．すなわち，システム内にはバッチラン，OS，スワップアウト/インの3種類のランが走行する．走行ランは以下のように挙動するものとする．

イ）バッチランはCPUとI/O装置を交互に使用する．また，CPUもI/O装置も使用しない（スワップアウトされている）状態がある．スワップアウトされている状態とされていない状態の比率は$1-A$：A（A：インコア比率，$0<A<1$）とする．

ロ）OSはCPUだけを使用し，その優先度はバッチランよりも高い．

ハ）スワップアウト/インはI/O装置だけを使用し，その優先度はバッチランよりも高い．CPU，I/O装置の1回当り使用時間，OSのCPU利用率，インコア比率などのモデルのパラメータは対象計算機のハードウェア性能や実測結果をもとに決定する．

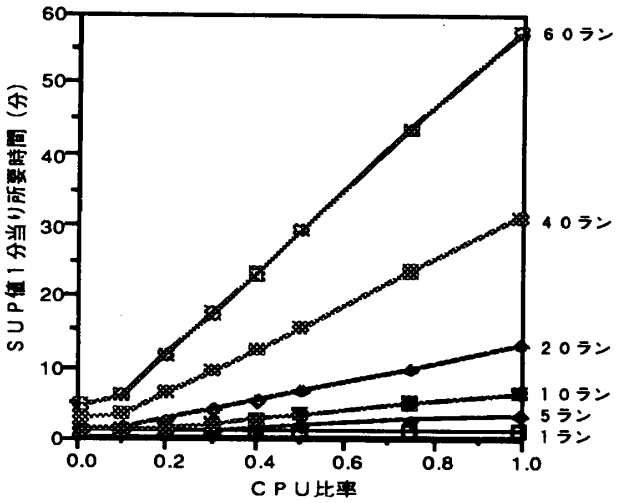

図1　TOTAL・SUP値1分当り所要時間

(4) シミュレーション結果

シミュレーションの実行により，特定の走行環境下でのCPU要求1回当り待ち時間，I/O要求1回当り待ち時間およびCPU利用率を求めることができる．計算の便宜のため，待ち時間をSUP値1分を使用するための所要時間（SUP値1分当り所要時間）に変換する．図1はTOTAL・SUP値1分当り所要時間である．この図よりバッチ処理計算機の同時走行ラン数とCPU比率から，SUP値を1分使用するための所要時間を求めることができる．

3.4 所要時間の計算方法

(1) 所要時間計算の考え方

バッチランの所要時間は各ランの走行順序（スケジュール）により異なるので，スケジュールを考慮しながら所要時間を求める必要がある．バッチランA，B，Cがあり，それぞれの開始時刻をt_{A1}，t_{B1}，t_{C1}とする．待ち時間なし（所要時間＝TOTAL・SUP値）で走行した場合の終了時刻をそれぞれt_{A2}，t_{B2}，t_{C3}とする（図2）．

図2は，待ち時間を考慮しないときの走行スケジュールであるが，区間L_2，L_3，L_4において複数のランが走行しているので，待ちが発生し各ランの所要時間は長くなる．各区間での同時走行ラン数とCPU比率は計算できるので，待ち時間予測モデルより各ランの所

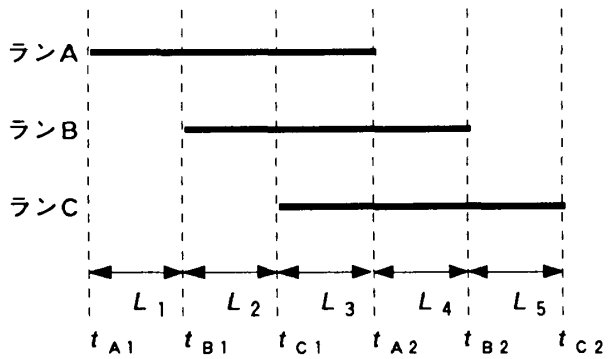

図2　バッチランA，B，Cの走行スケジュール

要時間を求めることができる．求められた所要時間で再度，図2のようなスケジュールを作成すると，複数ランが走行している区間L_2，L_3，L_4の長さが変わる．そこで，再度，所要時間を計算し，スケジュールを作成し直す．この計算を何度か繰り返し行い，最終的な所要時間を求める．実際には5回程度の反復で計算は収束する．

(2) **所要時間計算アルゴリズム**

所要時間計算のアルゴリズム（基本アルゴリズム）を以下に示す．このときバッチランのスケジュール(各ランの開始時刻と終了時刻）も同時に作成される．

【基本アルゴリズム】

1．SUP値予測モデルにより，各バッチランのCPU・SUP値，I/O・SUP値，TOTAL・SUP値を予測する．

2．TOTAL・SUP値を所要時間として，ランネットワーク（各バッチランの走行順序）に従いバッチランの初期スケジュールを作成する．バッチランのスケジュール作成にはPERT（Program Evaluation and Review Technique）の手法を用いる．

3．所要時間を再計算する．

イ．同時走行ラン数が変わる区間ごとに，待ち時間予測モデルを用いてCPU・SUP値1分当り所要時間を計算する．

ロ．イ.で求めた区間別CPU・SUP値1分当り所要時間を区間の長さを重みとして加重平均し，各ランのCPU・SUP値1分当り所要時間とする．

ハ．ロ.で求めたラン別のCPU・SUP値1分当り所要時間と1.のCPU・SUP値を掛けてCPUを使用するための所要時間を求める．

ニ．I/Oを使用するための所要時間をイ.からハ.と同様の方法で求める．

ホ．ハ．とニ．の所要時間を加えてバッチランの所要時間とする．

4．3．の所要時間とランネットワークをもとに，PERT手法によりスケジュールを作成し直す．

5．3．と4．を全ランの所要時間が変化しなくなるまで繰り返す．

4．バッチランのスケジュールモデル

4.1　バッチランのスケジュール作成方針

バッチランのスケジュール作成において重要なことは，個々のランの所要時間（ターンアラウンド）を短くすることではなく，全体の終了時刻を早くすることである．そのためには，単位時間当りの計算機の処理量（スループット）を上げる必要がある．

スループットを悪くする原因の1つはOSの過負荷である．同時走行ラン数が多くなるとスワップアウトが発生しOSが過負荷になりスループットは悪くなる．もう1つの原因は，計算機の稼働率が低く，能力を最大限に利用していない場合である．同時走行ラン数が少ないと計算機は稼働率が低くなるので，OSが過負荷にならない程度にラン数を増やす必要がある．使用する装置に偏りがあるときにもスループットは悪くなる．バッチランがCPUとI/Oの2つの装置を使う場合，どちらかの装置に使用頻度の偏りが生じた場合，片方の装置での待ちが多くなり，もう片方の装置が遊んでしまうことになる．バッチラン全体のCPU使用量とI/O使用量は一定であるので，システム内のCPU比率を一定に保って運転することにより回避することができる．

以上より，バッチラン全体の終了時刻を早くするために，以下の方針でスケジュールを作成する．

イ）バッチランはスワップアウトが発生しないラン数までは，できるだけ多く同時に走行させる．

ロ）システム内のCPU比率の管理限界を定め，管理限界内で運転するようにする．

4.2　スケジュール作成の方法

(1) **通常運転時のスケジュール**

スケジュール作成時の同時走行ラン数やCPU比率の制御は，PERTの山くずし手法を応用して行う．

まず，基本アルゴリズムによりバッチランのスケジュールを作成する．次に同時走行ラン数またはCPU比率が管理限界内にないランをそのランの余裕時間（最早開始時刻から最遅開始時刻の間）内で，全体の終了時刻が最も早くなる時刻に開始時刻を移動する．これを全ランについて繰り返し，すべてのランを管理限界内におくことによりスケジュールを作成する．

【山くずしアルゴリズム】

1．基本アルゴリズムによりバッチランのスケジュールを作成する．

2．同時走行ラン数の山くずしを行う．

イ．同時走行ラン数が上限数以上になっている時間帯において，その時間帯に走行しているすべてのランの開始時刻を余裕時間内で移動して，全体の終了時刻が最も早くなるランと開始時刻を探す．

ロ．イ．で見つけたランをその開始時刻に移動する．

ハ．全体の終了時刻を早くするランがなくなるまでイ．，ロ．を繰り返す．

3．2．と同様の方法でCPU比率の山くずしを行う．

4．全体の終了時刻を早くすることができなくなるまで2．，3．を繰り返す．

(2) デッドライン運転時のスケジュール

終了時刻を早くするためにバッチランに終了期限（デッドライン）を付ける方法がある．特定の業務や特定のランの終了時刻を早くするためにもこの方法は有効である．バッチラン全体の終了時刻を早めるためには，クリティカルパス上のラン（余裕時間が0のラン）にデッドラインを付ければ良い．ただし，あるランにデッドラインを付けて処理時間を短くしても，他のランの処理時間が長くなるので，全体の終了時刻が早くなるとは限らない．また，あるランにデッドラインを付けるとクリティカルパスが変わる場合がある．これらの点を考慮して，デッドラインを導入する必要がある．

【デッドライン・アルゴリズム】

1．基本アルゴリズムによりスケジュールを作成する．

2．クリティカルパス上の先頭のランから試してみて，全体の終了時刻が早くなる最初の1ランにデッドラインを付ける．

3．デッドラインを付けたラン以降を基本アルゴリズムでスケジュールし直す．

4．スケジュールし直したランに対し，2．の方法で1つのランにデッドラインを付ける．

5．デッドラインを付けるランがなくなるか，最後のランにデッドラインが付くまで3．，4．を繰り返す．

山くずしアルゴリズムとデッドライン・アルゴリズムを組み合わせることにより，最適なスケジュールを作成することができる．一般に，各ランの走行順序や走行時刻がランネットワークにより厳しく規定されて走行順序の自由度が小さい場合は，デッドライン・アルゴリズムの方が有効である．一方，各ランの走行順序や走行時刻が比較的自由に設定できる場合は，山くずしアルゴリズムが有効になる．

4.3 スケジュールの作成例

実際に走行した約800のバッチランに対して，山くずしアルゴリズムとデッドライン・アルゴリズムを適用して，処理時間をシミュレーションにより測定した．その結果，実際に6時間かかった処理が5時間程度で終了し，約20%の改善が行われた．

5．おわりに

以上の所要時間予測モデルは主に東京電力株式会社殿と，スケジュール作成アルゴリズムは中部電力株式会社殿と共同研究を行った成果である．掲載を許可していただいた関係者の方々に感謝します．

参考文献

松田芳雄，河岸憲一：バッチラン運用管理システムにおける所要時間の予測とスケジュール作成，日本ユニシス㈱技報41号，pp.107-126，1994年5月．