

3F-1

バッチ処理における ファイル引継ぎ処理の高速化方式

清井雅広¹ 長須賀弘文¹ 新井利明¹ 新村義章²

¹(株)日立製作所システム開発研究所 ²(株)日立製作所ソフトウェア開発本部

1. はじめに

銀行等のオンライン処理は年々取り引き量が増加し、それに伴いバッチ集計業務も長時間化している。一方オンライン処理のサービス時間は延長する傾向にあり、バッチ処理を行える時間帯はますます制限されつつある。そのため、バッチ処理時間の短縮が急務の課題である。

バッチ処理を高速化するための機能は、これまでもいくつかの手法が提案されている^[1]。これらの方式では、ファイル単位で行っていたデータ引継ぎ処理をレコード単位で行うことによりジョブを並列処理し、また一時ファイルへの入出力要求をメモリ上のデータ転送で置き替えることで実入出力時間を削減する。

本発表で提案するジョブ間高速ファイル引継ぎ方式（図1）は、先行ジョブの実行結果を複数のジョブが参照する処理や、保存が必要なデータをジョブ間で引継ぐ処理にも適用できるなど、既存の機能よりも広範囲に適用できることに特長がある。本発表では、その実現方式と評価結果を報告する。

2. 機能概要

提案方式は、ユーザプログラムを変更することなく、以下の機能を実現する。本方式の適用は、ジョブの実行方法を記すジョブ制御文で指定する。

(1) 入力ジョブ多重化機能

1つの先行ジョブが出力したデータを複数の後続ジョブが同時に参照しながらデータの引継ぎを行う機能である。これにより入力ジョブが

複数存在する場合でも、データ引継ぎを行いな

(2) バックアップ機能

引継ぎデータを外部記憶装置に保存しながら、ジョブ間でレコード単位のデータ引継ぎを実行する機能である。これにより中間データの保存を必要とするデータ引継ぎ処理を高速化できる。

3. 実現方式

(1) ジョブ待合せ方式

本方式は、限られたメモリしか利用できない場合でも大容量のデータ引継ぎを可能とするため、メモリ上に用意したバッファ領域をラップアラウンドに使用する。このとき、データ引継ぎを行うジョブは相手方のジョブを意識せず互いに独立に処理を進めていく。よって、ジョブ間でレコード単位のデータ引継ぎを保証するためには、ジョブ間の待合せ制御が必要である。

しかし、データを受渡すたびに待合せ処理を

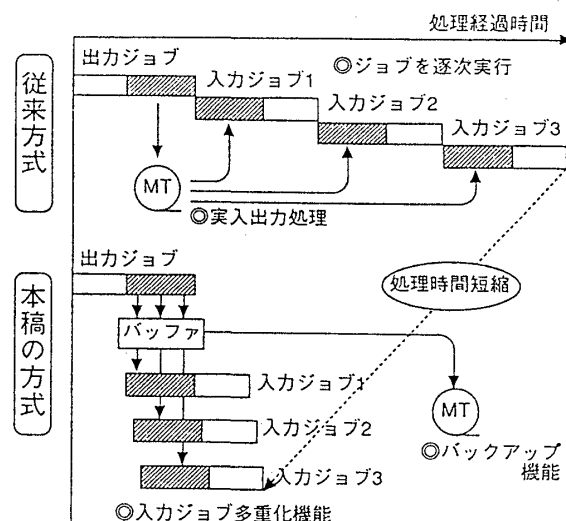


図1. 機能概要

High-speed file transfer method in batch processing

Masahiro KIYOI¹, Hirofumi NAGASUKA¹, Toshiaki ARAI¹, Yoshiaki SHINMURA²

¹Systems Development Laboratory, Hitachi, Ltd., ²Software Development Center, Hitachi, Ltd.

実行したのでは、CPUオーバーヘッドが膨大になる。特に入力ジョブが複数になった場合、出力ジョブが頻りに待ち状態/解除を繰り返す可能性がある。この状態を防ぐため、ある一定回数の入出力ごとに一括して待合せ制御を行う方式を提案した。

(2) バッファ管理方式

大容量のバッファであれば待合せ制御を行う回数は減少するが、入出力要求頻度の比が小さい場合は、小容量のバッファでも待合せ処理を行う回数は少ない。メモリの有効活用の観点から、本方式では小容量のバッファから処理を開始し、入出力の実行状態をモニタしながら、待合せ処理回数が一定の基準を満たすようにバッファを拡張する方式を採った。

バッファをラップアラウンドに利用する本方式では、処理の遅いジョブと早いジョブの処理データ数の差は最大でバッファ容量分となる。そこで処理の遅い入力ジョブが存在する場合、バッファを増分し、他のジョブが先に実行することを可能とすることで、遅い入力ジョブの影響を小さく抑えている。(図2)

(3) バックアップ方式

バックアップ機能は、入力ジョブ多重化機能を利用して実現する。すなわち、バックアップ機能が指定された場合、引継ぎデータを外部記憶装置へコピーするタスクを起動し、これを入力ジョブの1つとして入力ジョブ多重化機能を適用する。

このとき、上で示したバッファ管理方式により、大容量メモリを利用できればバックアップ

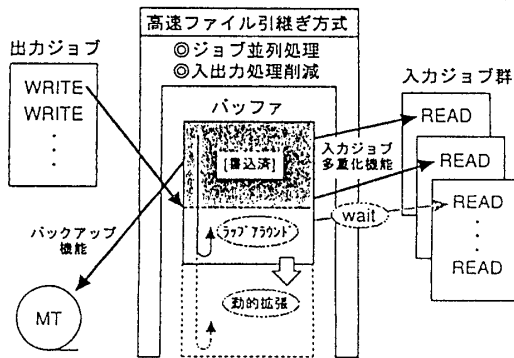


図2. 方式概要

処理の影響を他に及ぼすことなくジョブ間でのデータ引継ぎ処理を実現できる。

4. 性能評価

実際のバッチ業務を想定し、図3で示す構成のジョブ群に対して本方式を適用した。データの引継ぎを磁気ディスクを用いて行った場合と比較すると、処理経過時間を約6割短縮した。

(図4) 今回の測定例では、バックアップ機能を合わせて適用しても処理経過時間はほぼ同じであった。

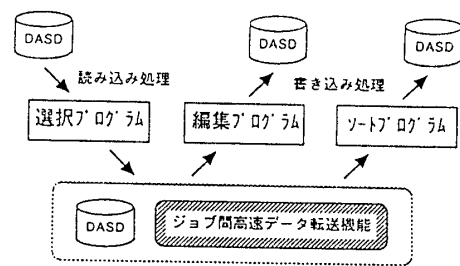


図3. 測定ジョブの構成

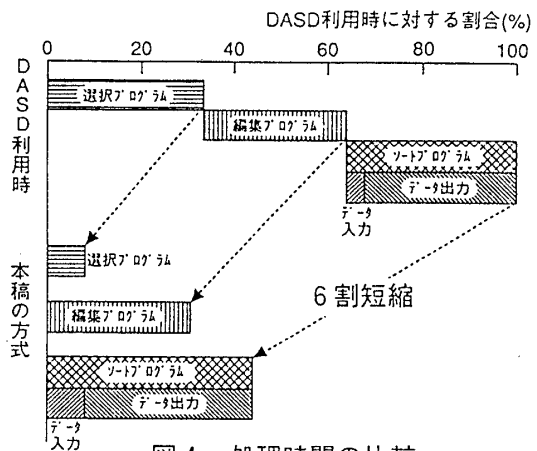


図4. 処理時間の比較

5. 結論

入力ジョブが複数存在する場合や、中間データの保存を必要とする場合にも適用可能な、ジョブ間高速ファイル引継ぎ方式を開発・評価した。今後はジョブの実行多重度を更に向上させる方式を検討する予定である。

参考文献

[1]長須賀, 他: ジョブ間並列同期転送機能の開発と評価-VOS3/AS:PREST-, 情報処理学会第41回全国大会講演論文集, 1990年9月。