1P-8

ソーティングにおける高速化の一手法（Ⅱ）

－ クイックソートの入出力並行処理方式 －

小永井豪*，米城範正**，大熊和明**，小端則夫**
* 株式会社富士通静岡エンジニアリング
** 富士通株式会社

1．はじめに

現在，最も平均性能が良いとされているクイックソートは，主記憶内に入力されたレコードをいかに早く並べるかを主体に考えられている。したがって，主記憶内にすべてのレコードが入りきらない大量データでは，あまり適用されていない。しかし，最も平均性能が良いという利点を持つので，本稿では大量データのソート処理（ストリング生成部）にクイックソートを適用できないか考察した。

2．クイックソートを適用した場合の問題点

ストリング生成部にクイックソートを適用すると，図1に示すように，①レコード格納領域への入力処理，②入力した全レコードに対するソート処理，③レコード格納領域内レコードの出力処理を繰り返すことになる。

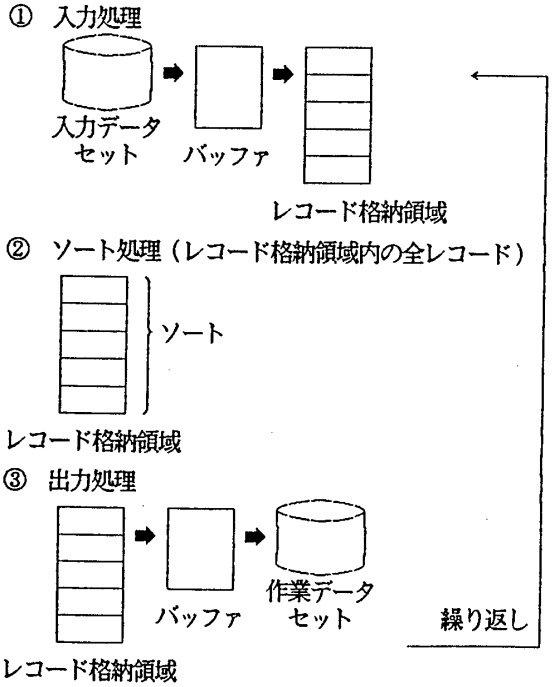


図1 クイックソートを単純に適用した処理

このため，図2に示すように，入力処理，ソート処理，出力処理が順次繰り返されるので，経過時間が増加することになる。

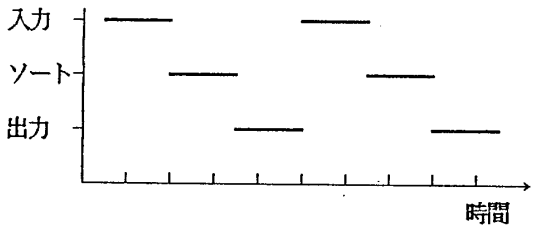


図2 クイックソートを適用した場合のタイムチャート

3．問題解決のための手法

経過時間の増加を解消するため，図3に示すように，入力処理，ソート処理，出力処理をすべて並行に処理させる工夫が必要である。

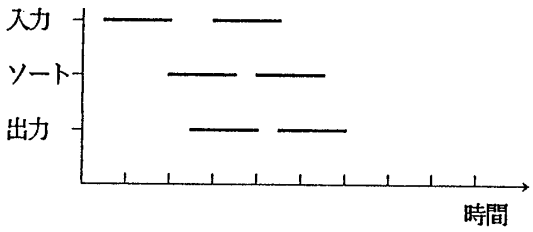


図3 並行処理した場合のタイムチャート

このため，図4に示すように，一部のレコードを昇順に並べるごとに（②），そのレコードを出力する（③）。次に，出力した結果空いたレコード格納領域にレコードを入力する（④）。これにより，並行処理が可能になる。

A Method for Acceleration of Sort Processing (2)
Takeshi KONAGAI, Norimasa YONEKI, Kazuaki OHKUMA, Norio OBATA
FUJITSU SHIZUOKA ENGINEERING Ltd., FUJITSU Ltd.

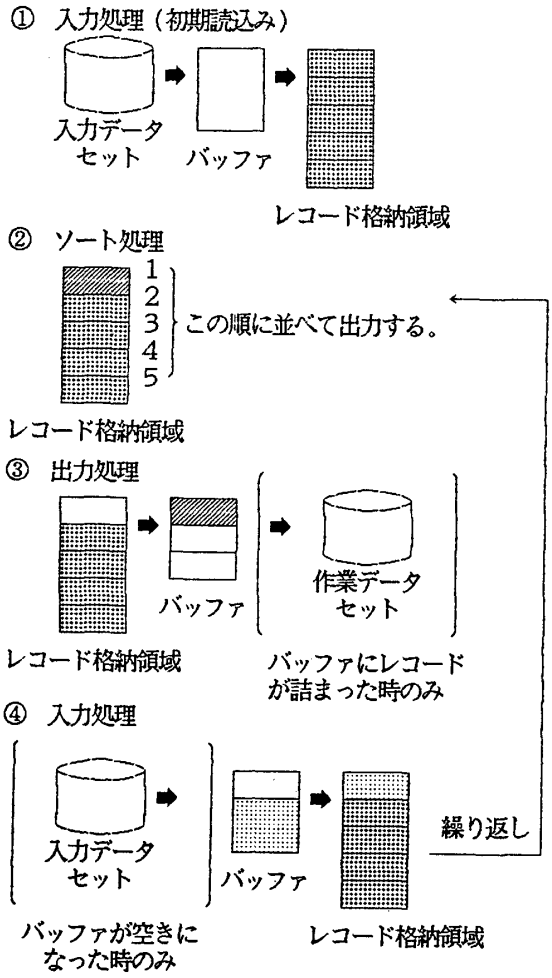
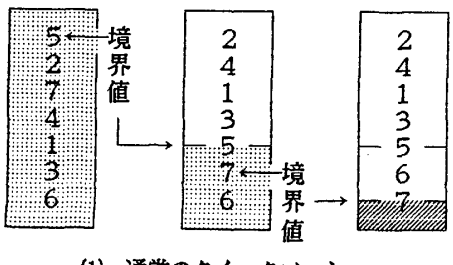
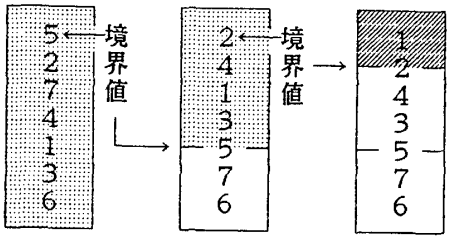


図4　並行処理させるための手法

この処理方式において，昇順にレコードを出力するためには，小さい値を持つレコードを先に並べる必要がある。しかし，通常のクイックソートはスタックを最小限にするため，レコード数の少ない群を先に分割する。このため，どの部分が先に並ぶかは不定である（図5－(1)参照）。これを，小さい値を持つレコード群を先に分割していくことにより，小さい値を持つレコードを先に並べることができる（図5－(2)参照）。



(1) 通常のクイックソート



(2) 変更したクイックソート

図5　処理の順による並び方の違い

4．効果

クイックソートを単純に用いた場合（①）と並行処理させた場合（②）のデータ量と経過時間のグラフを図6に示す。並行処理させた場合は，３０～４０%の削減となっている。データ量の増加に伴って，クイックソートの回数も増加するので，この差はさらに開く。すなわち，大量データほど経過時間の短縮効果が現れる。

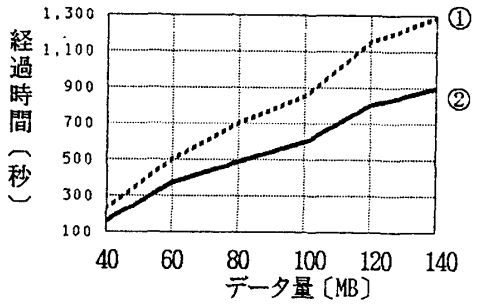


図6　経過時間の比較

5．おわりに

以上のように，必要な処理を極力並行に処理することにより，経過時間を短縮できることが分かった。しかし，大量データのソート処理にクイックソートを適用するという課題に対しては，ストリング数の増加等の問題があり，適用範囲に制限がある。

参考文献

(1) Donald E.Knuth : "THE ART OF COMPUTER PROGRAMMING Volume 3 / Sorting and Searching" ,ADDISON-WESLEY PUBLISHING COMPANY

(2) ALFRED V.AHO : "DATA STRUCTURES AND ALGORITHMS" ,ADDISON-WESLEY PUBLISHING COMPANY