研究速報

特集 9

UDC 681.3：661.68

スーパーデータベースコンピュータSDC2における データ流制御とその評価

Evaluation of Flow Control Mechanism on the SDC2, the Super Database Computer Version 2

中村　稔*・田村孝之*・喜連川　優**・高木幹雄**
Minoru NAKAMURA, Takayuki TAMURA, Masaru KITSUREGAWA and Mikio TAKAGI

1．概　　要

大規模データベースに対する超高性能SQL処理を目的とし現在スーパーデータベースコンピュータ第二版（SDC 2）を開発中である．本システムはスーパーデータベースコンピュータ第一版（SDC 1）[3]に対して各モジュールの処理性能の向上を図るとともに，高機能オメガネットワークによる多モジュールの相互接続を行っており，現在その構築・評価を進めている．本稿では，SDC 2のアーキテクチャを示すとともに，そのシステムソフトウェアについて述べる．さらに，SDCでは大容量のデータがシステム内を循環するがデータ流の制御とデッドロックの回避法についても述べる．

試作したSDC 2[5]上での結合演算の実装と評価結果について述べ，実際の処理においてデータ流制御が適切に機能していることを示す．

2．SDC 2の構成

SDC 2[5]は最大でプロセッサ7台と磁気ディスク装置4台を密に結合したデータ処理モジュール，ならびに複数の処理モジュールを疎に結合する高機能オメガネットワーク[1),2)]からなるハイブリッドアーキテクチャをとる．図1にその構成を示す．この構成では密結合の利点である軽い通信コストによる高速性とモジュール数の増減によるスケーラビリティが同時に得られる．

SDC 2はSDC 1に対して以下に示す特徴を有する．

・データ処理モジュール（DPM）の性能向上と小型化
・ディスクコントローラ（DC）およびデータネットワークインターフェース（NI）にそれぞれ専用の制

*東京大学生産技術研究所　第3部
**東京大学生産技術研究所　付属機能エレクトロニクス研究センター

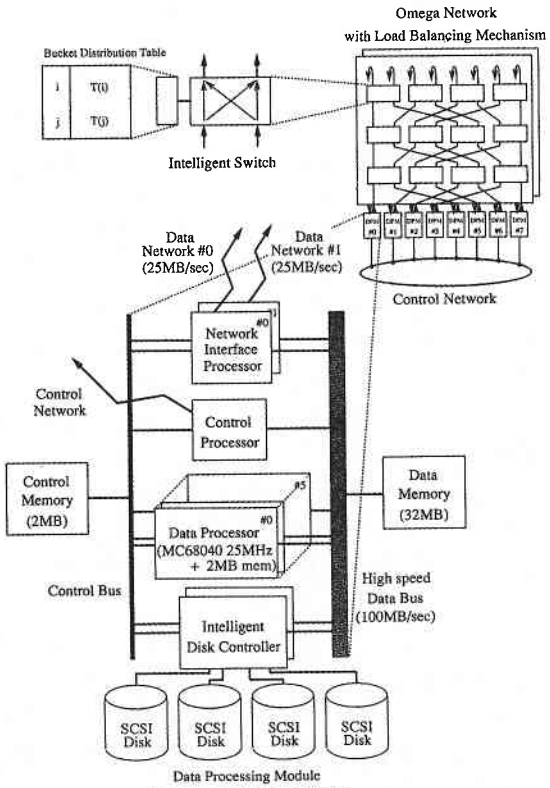

図1　SDC 2の構成

表1　SDC 1とSDC 2の構成比較

	SDC 1	SDC 2
要素プロセッサ	68020/20MHz	68040/25MHz
処理用プロセッサ数	4	7
データメモリ	8MB	32MB
データバスバンド幅	40MB/sec	100MB/sec
ディスク	2台/DPM	4台/DPM
データネットワーク リンクバンド幅	10MB/sec	25MB/sec
データ処理モジュール	2	8

御プロセッサを用い，コントロールプロセッサ（CP）の負荷を軽減
・データネットワークの二重化と性能向上ならびに，機能の追加
・ハードウェア/ソフトウェアの可変長タプルの支援

3．SDC 2のシステムソフトウェアの構成

SDC 2ではそれぞれのモジュール内において，次のようなプロセスが存在する．

・Executer：フロントエンドから渡されたコマンドをPMCに受け渡し，複数のモジュールにまたがった処理を行う際にモジュール間の同期をとる．
・Processing Module Controller（PMC）：Executerから渡されたコマンドに基づいてDiskman, Netman, DPPの制御と同期を行う．
・Diskman：Pageを単位としたDisk I/Oの制御とフローコントロールを行う．
・Netman：データネットワークの制御を行う．
・DPP：ディスクおよびデータネットワークからのデータを実際に処理し結果をディスクあるいはデータネットワークに送出する．
・Sampler：モジュール内の各プロセッサの稼働率やメモリの消費率に関するデータを収集し定期的にフロントエンドモジュールに報告する．このプロセスは実際のデータ処理には関与しないが，Schedulerに対するヒント情報としてフロントエンド上で利用される．

3.1　SDC 2におけるデータ流制御

ページによるデータの受渡し

SDC 2ではデータメモリを固定長のページに分割しモジュール内でのデータ交換はすべてこのページを受け渡すことで行われる（図2）．ページの受け渡しはコントロールメモリ上のバッファ構造体を通じて行われる．ページ内には複数のタプルが格納され，ページ長を越えなければタプルの長さに制限はない．また，データネットワークを用いたモジュール間のデータ交換はタプルを単位として行われ，ページからタプルへの変換およびその逆変換はデータネットワークインターフェース上で行われる．SDC 2上におけるページの受け渡しはすべては以下の4つのライブラリ関数のみによって行われる．

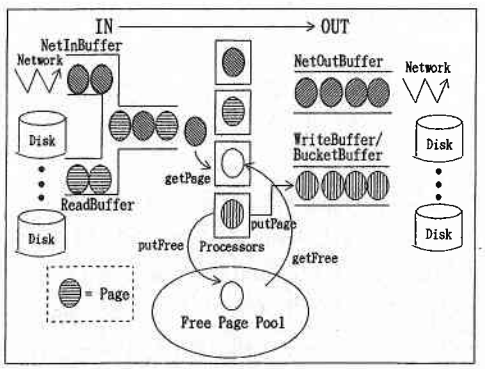

図2　モジュール内のデータの流れ

get Page()　バッファ構造体からタプルの入ったページを取り出す．
put Page()　バッファ構造体へタプルの入ったページを付け加える．
get Free()　フリーページプールから空のページを取り出す．
put Free()　フリーページプールへ空のページを返却する．

このようにページをデータ交換の単位とすることでディスク，プロセッサ，ネットワークにまたがるデータの受け渡しを単純化し統一的な扱いを可能にしている．

バッファ構造体

SDC 2で用いられるバッファ構造体はページが流れるパイプとみなすことができる．バッファの両端にはそれぞれ，ページの生成者，消費者となるプロセスが一つ以上存在する．バッファは別々のプロセッサ上のプロセスをつなぐ役割を果たし，同時にデータのフローコントロールとデッドロックの回避を行うポイントでもある．バッファへのページの供給と取り出しはそれぞれ，putPage()，getPage()ライブラリ関数によって行われる．それぞれのバッファには同時に格納できるページの上限数が指定でき，putPage()の際にページ数が上限値を越えていた場合は，putPage()を呼び出したプロセスはブロックされ，ページが消費されて上限値を下回った時点で処理が再開される．

SDC 2には以下のようなバッファが存在する．

readBuffer ディスクから読み出したデータをDPPに受け渡すためのバッファ．上限値が設定される．

writeBuffer 演算結果をディスクに書き出すためのバッファ．

bucketBuffer ハッシュ結合演算ではリレーションをハッシュ関数で分割し複数のバケットに分割する．分割されたバケットをディスクに中間ファイルとして書き出すためのバッファ．実際には分割数に応じて複数のbucketBufferが存在する．

netInBuffer データネットワークからのデータをDPPに受け渡すためのバッファ．上限値が設定される．

netOutBuffer DPPからのデータをデータネットワークに送り出すためのバッファ．

3.2 デッドロックの回避

SDC 2におけるデッドロックはフリーページが消費し尽くされることで発生する．SDC 2ではページの生成者は同時にフリーページの消費者であり逆にページの消費者はフリーページの生成者であるため，デッドロックを回避するためにはフリーページがなくなる前にページの生成を中断し消費者によってページを解放させればよい．

SDC 2ではディスクがデータの最終的な生成者であり消費者であるが，同時には生成者か消費者かの一方にしかなれない．結合演算処理などで作成されるハッシュテーブルは唯一データの消費者が存在しないが，これは処理を開始する前にその大きさが静的に決定できるのでデッドロックの発生原因にはなり得ない．

SDC 2におけるデッドロック回避の基本的戦略はディスクからのデータ読み出しを中断して出力結果または中間ファイルをディスクに書き出すことである．

フリーページプールにはデッドロック回避を開始するページ数（low Water Mark）と中断した処理を再開するページ数（high Water Mark）があらかじめ指定されている．Diskmanはデータの読み出しにおいてフリーページを獲得する際にフリーページの残量がlow Water Markを下回るとディスクからの読み出しを中断する．さらに，モジュール内の各バッファが保持しているページの量を調べ，その結果に応じて以下のようにしてデッドロック回避を行う．

ⅰ．**writeBufferにhighWaterMark以上のページがある場合**：writeBufferの内容をディスクの出力ファイルに書き出す．

ⅱ．**bucketBufferのページの合計がhighWaterMarkを越える場合**：bucketBufferの内容をディスクの中間ファイルに書き出す．

ⅲ．**ⅰ，ⅱ以外の場合**：これは他のモジュールのデータ消費が遅いためにnet-OutBuffer上のページが消費されないために発生する．この場合Diskmanはフリーページが highWaterMarkを上回るまでアイドル状態となる．

4．SDC 2におけるシステムソフトウェアの評価

これまでに述べたページによるデータ流制御がSDC 2上でどのように機能しているかを調べることを目的とし，実際にSDC 2上で結合演算を行いその振舞を調べた．

4.1 結合演算アルゴリズム

単純なハッシュジョインアルゴリズムであるGRACEハッシュジョインではスプリットフェーズで，それぞれのバケットを各モジュールに振りわけ，ジョインフェーズをモジュールごとに独立して実行する．ネットワークはスプリットフェーズでのみ使用する．

GRACEハッシュジョインアルゴリズム[3]では分割後のバケットの大きさが不均一であった場合，ジョインフェーズにおいてモジュール毎に処理するタプル数の差となって現れる．これはモジュール間での負荷の偏りとなり，処理速度の低下をもたらす．平坦化ハッシュジョインではモジュール間の均等な負荷分散を実現するためにリレーションのスプリットに際しバケット平坦化を行う．SDC2はバケット平坦化をハードウェアでサポートする高機能オメガネットワークを有している[2),4)]．その後ジョインフェーズが効率よく実行できるようにバケットサイズチューニングを施す．ジョインフェーズにおいてはバケットの収集と結合処理をオーバーラップして行うためにバケット分散によるプロトコルオーバーヘッドを最小に抑えることができる．

4.2 SDC上における結合演算処理

試作中のSDC 2システム上で結合演算処理を行った．最大4モジュールのオメガネットは現在，少々不安定なため，CNET用のEthernetを用いてシミュレートした．

- 拡張ウィスコンシンベンチマークに準じ，タプル長208バイト，タプル数100万件のリレーションに対し以下のような結合演算を行う．各モジュール毎に100万件のリレーションを用意する．

```
insert into result
select A.*,B.*
from relA A, relB B
where A.k1 = B.k1 and B.k1 < X;
```

- 結合演算はGRACEハッシュジョインアルゴリズムおよび平坦化ハッシュジョインアルゴリズムを用いて行う．
- モジュール間の負荷の偏りをシミュレートするためにそれぞれのアルゴリズムでバケット分割時に各バケットに含まれるタプルの数が以下に示すようなZipf分布に準ずるように振り分ける．

$$|Ri| = \frac{|R|}{i^z \cdot \sum_{j=1}^{Hs} \frac{1}{j^z}} \quad (1 \leq i \leq Hs)$$

図3に4モジュールのSDC 2においてサブバケットのタプル数を不均一分布した時の結果を示す．縦軸は処理時間（秒）横軸はZipf（z）におけるzの値である．図3から平坦化ハッシュジョインアルゴリズム（BSJoin）が負荷の偏りにかかわらずほぼ一定の時間で処理を完了していることがわかる．これは平坦化ハッシュジョインがバケットの

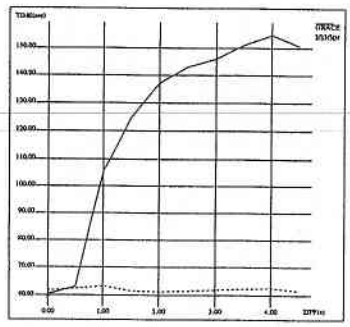

図3　結合演算実行時間-不均一分布データに対する処理性能

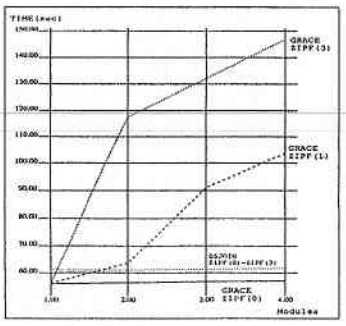

図4　結合演算実行時間
—駆動モジュール数への依在性—

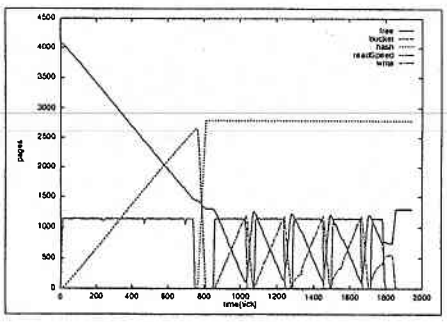

図5　均一負荷におけるGRACEハッシュジョイン実行時のメモリ使用量の推移

平坦化によるモジュール間の負荷の均一化を行うためで，このような処理を行わないGRACEハッシュアルゴリズムではデータの分布の偏りが大きくなるに従って処理時間が長くなる．

SDC 2では各モジュールが32MByteのステージングバッファを持つため，データの分布が均一であった場合はメモリ上で結合演算処理が実行できる（Zipf(0)の場合）．データの分布が不均一の場合GRACEハッシュアルゴリズムでは特定のモジュールにタプルが集中するため，これを中間ファイルとしてディスクに書き出す必要があるためZipf(1)以降で急速に性能が低下している．Zipf(0)での処理速度の違いは平坦化ハッシュジョインアルゴリズムのオーバーヘッドによるものである．また，Zipf(0.5)まではステージングバッファ上での処理が出来るにもかかわらずGRACEハッシュジョインの処理速度が低下している．これは結合演算結果の大きさに偏りが発生するために出力ファイルの書き出し時間にばらつきが生じるためである．図4にモジュール数を変えた時の性能を示す．平坦化ハッシュは高いスケーラビリティを有することがわかる．また図5にGRACEハッシュジョインをZipf(0)のデータ分布を用いて実行した場合のメモリ使用量のグラフを示す．データ分布がZipf(0)であるので各モジュールに負荷が均一にかかるため，どのモジュールでもほぼ同様の結果になる．

図5において縦軸がページ数，横軸が時間（tick＝1/32sec）である．ただしディスクの読み出し速度だけはpages/secである．また初期状態のフリーページ数は各モジュール毎に4096である．“free”はフリーページの合計，“bucket”は全バケット中に含まれるページ数の合計，“hashTable”はハッシュテーブルに使用されたページ数，“readSpeed”はディスクからの読み出し速度を示す．ディスクからの読み出し速度は1150ページ/秒程度であり，ほぼ最大性能が達成されていることがわかる．これはデータ処理がディスクからの読み出し速度に追従しているためであり，負荷変動の少ない場合SDC 2のデータ流制御は良好に機能していることがわかる．Zipf分布を有する場合にも適切な動作をする事を確認している．

5．ま　と　め

SDC 2のシステムソフトウェアの構成とデータ流制御ならびにデッドロック回避の方法について述べるとともにGRACEハッシュジョインアルゴリズムおよび平坦化ハッシュジョインアルゴリズムをSDC 2上で実行し，メモリ使用率を観測しシステムソフトウェアが適切に機能していることを示した．現在SDC 2はさらに詳細な評価を進めるとともに並列化コンパイラを試作中である．

（1993年12月10日受理）

参考文献

1) M. Kitsuregawa and Y. Ogawa. Bucket Spreading Parallel Hash: A New Parallel Hash Join Method with Robustness for Data Skew in Super Database Computer (SDC). Proc. of 16th Int. Conf. on VLDB, pages 210-221, 1990.
2) 喜連川，小川．バケット平坦化機能を有するオメガネットワーク．情報処理学会論文誌，30(11)：1494，1989.
3) 中村，平野，原田，相場，鈴木，喜連川，高木．スーパーデータベースコンピュータ(SDC)上での平坦化ハッシュジョインの評価．並列処理シンポジウムJSPP'92，1992.
4) 田村孝之，中村稔，喜連川優，高木幹雄．スーパーデータベースコンピュータにおけるデータネットワーク系の実装．電子情報通信学会技術報告，CPSY93-31，1993.
5) 中村稔，田村孝之，喜連川優，高木幹雄．スーパーデータベースコンピュータ第二版（SDC 2）におけるデータ流制御の評価．アドバンストデータベースシンポジウム'93，pp. 113-122，1993.