並列ＲＤＢシステムにおける通信機能の実現方式

7W－3　藤原真二　長坂　充　鍵政　豊彦　正井　一夫　宮崎　光夫
（株）日立製作所中央研究所　同ソフトウェア開発本部

１．はじめに

近年、ＲＤＢシステムに対して要求される処理性能は、処理すべきデータ量の増大に伴い、ますます高くなってきている。この要求に答えるためにＲＤＢ処理を並列化するシステムが必要とされてきている。現在報告者らが開発中の並列ＲＤＢシステムでは、データを複数のプロセッサに分割することで並列に処理を行ったり、一つの処理を実行制御プロセスとデータアクセスプロセスとでパイプライン実行することにより処理時間の短縮を図っている。この場合に各プロセス間で通信することが必須であり、ＲＰＣ (Remote Procedure Call) 等の通常の通信機能の他に、新たな通信機能が必要となる。

本稿では、上記並列ＲＤＢシステムにおいて必要となる通信機能とその実現方式に関して報告する。

２．並列ＲＤＢシステムにおける通信機能の概要

本並列ＲＤＢシステムにおいて必要とされる通信機能は通信ライブラリとして提供している。

通信機能としてサポートしているものとしては、以下のようなものがある。

（１）RPCとsend/recv型通信機能

制御も含めて通信する場合に有効なRPCとデータベースのデータを転送する場合に有効なsend/recv型通信の両方を混在使用可能

（２）トランザクション制御関連

トランザクション制御情報を管理し、プロセス間で情報を転送する機能を提供。さらに、並列度を向上するためにサーバは複数プロセスから構成し、効率化のためにトランザクションごとにサーバ内では一つのプロセスのみと通信可能

（３）高速通信インタフェースサポート

ハードウェアによるメモリ間直接通信[1]を利用した高速インタフェースを利用可能

（４）非同期RPC

並列度を向上するために、プロセスがブロックするような同期型RPCだけでなく、要求の送信後に他の処理が可能で必要なときに応答を受信可能な非同期RPCを利用可能

（５）同一プロセッサ内通信最適化

通信相手が同一プロセッサか別プロセッサかを判定し、同一プロセッサの場合には効率の良い通信方式に自動的に切替

Implementation of Communication Function for Parallel RDB System
Shinji FUJIWARA, Mitsuru NAGASAKA, Toyohiko KAGIMASA, Kazuo MASAI and Mitsuo MIYAZAKI
HITACHI, Ltd.

３．RPCとsend/recv型通信機能

（１）RPCとsend/recv型通信混在時の課題

RDB処理の側から通信を利用する場合には用途に応じてRPCとsend/recv型通信を使い分けることが重要となる。しかしながら、到着した通信がRPCかsend/recv型を意識して処理する部分をRDB処理側で行うことは無駄が多い。そこで、RDB処理側としては通信ライブラリとのインタフェースレベルでRPCとsend/recv型通信を区別し、通信ライブラリ内で到着した通信を振り分けることとした。

（２）実現方式

RDB処理側では、必要に応じてRPCの受信またはrecvを発行する。通信ライブラリでは到着した通信がRPCなのかsend/recv型かを識別して振り分け処理を行う。

受信した通信がRPCの場合にRDB処理がRPC待ちの場合(1)にはそのままRDB処理に通信を渡して処理を続けさせ、RPC待ちでない場合(2)にはRPCキューにキューイングする。

受信した通信がsend/recv型の場合にも同様にしてRDB処理がrecv待ちの場合(3)にはそのままRDB処理に通信を渡して処理を続けさせ、recv待ちでない場合(4)にはsend/recvキューにキューイングする。

RDB処理がRPC待ちまたはrecv待ちにになる時にそれぞれ対応するキューに通信がキューイングされているかを参照し、キューイングされていれば取り出して処理を続ける。対応するキューにキューイングされていなければ、そのまま対応する通信が届くまで待ち状態になる。

説明の簡便のためRDB処理（コンテクスト）は一つとしているが、実際には複数のRDB処理が並行して動作しており、RPCキューおよびsend/recvキューもコンテクスト毎に設けている。

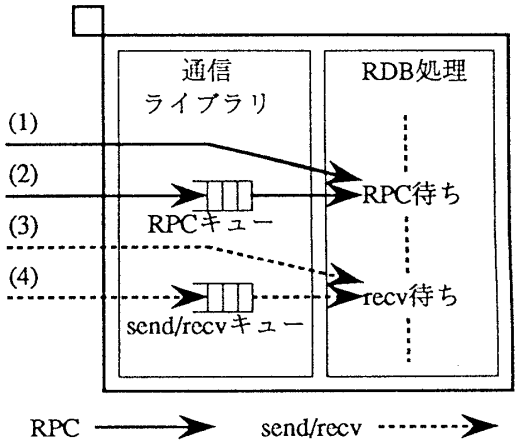

図１．　RPCとsend/recv型通信の概要

4．拡張チェインドRPC方式

（1）トランザクション関連の課題

並列ＲＤＢシステムでは、一つの問い合わせ処理を実行する際に、複数の異なるプロセスから同一サーバに対してRPCが実行されることがある。また、並列度を向上するためにサーバを同一の処理が可能な複数のプロセス群で実現している。

RPCをそれぞれ異なるサーバプロセスで実行すると、各プロセス毎にデータベースのオープン処理やアクセス権限チェック等が必要となる。このような処理の重複を避けるためには、一つのトランザクション内で発行される同一サーバに対するRPCを同一のプロセスで実行させる機能が必要となる。

（2）チェインドRPC方式

サーバから一つのプロセスを選んで通信するためにチェインドRPC方式を使用している。

各サーバはそれぞれ一つのスケジュールキューを持ち、クライアントは一回目の要求であると、要求をスケジュールキューにキューイングする。サーバの各プロセスはスケジュールキューから要求を一つずつ取り出す。RPCの応答としてサーバプロセスのハンドル情報を返し、これを用いて二回目以降の同一サーバに対する通信を行う。

（3）実現方式

機能を実現する場合にクライアント側で情報を管理する方式と、サーバ側で管理する方式が考えられるが、性能上の観点からサーバ側で管理する方式を選択した。

図２はトランザクション識別子T01を有するクライアントプロセスがサーバAおよびBにRPCを発行し、各サーバがさらにサーバCにRPCを発行する場合の処理の流れを示している。

まず、クライアントプロセスがサーバAおよびBに発行したRPCは各サーバのスケジュールキューにキューイングされ、プロセスaおよびbが取り出す。プロセスaが発行したRPC要求はサーバCのスケジュールキューにキューイングされる。この時サーバCの拡張チェインテーブルにトランザクション識別子T01のRPC要求がプロセスaから発行されたことを登録する。プロセスbがRPC要求を発行した時は、上記テーブルを参照することで、トランザクションT01に関するサーバCに対するRPCが既に発行済みであることがわかる。

プロセスaが発行したRPCが既にプロセスc1で実行されている時にはプロセスbのRPC要求を直接プロセスc1に送信する。プロセスaが発行したRPCがまだサーバCのスケジュールキューの中に存在する場合には、相手先が決まらないため、サーバCのT01用拡張チェインキューを作成し、プロセスbが発行したRPCをキューイングする。拡張チェインキューにキューイングされたRPCは、プロセスa が発行したRPCをプロセスc1が取り出した後に、プロセスc1によって順次取り出される。

5．おわりに

本稿で説明した通信機能は、一つの処理を複数のサーバプロセスで実行する並列ＲＤＢシステムを実現する場合に何らかの手段により必要となる機能である。各通信機能の実現方式としては他の実現方式も考えられるが、本稿で説明した実現方式が効率のよい実現方式であり、開発中の並列ＲＤＢシステムで採用した方式である。

参考文献

［1］小原　他：並列ＲＤＢシステムにおける通信高速化方式，情報処理学会第49回全国大会，7W-4，1994.

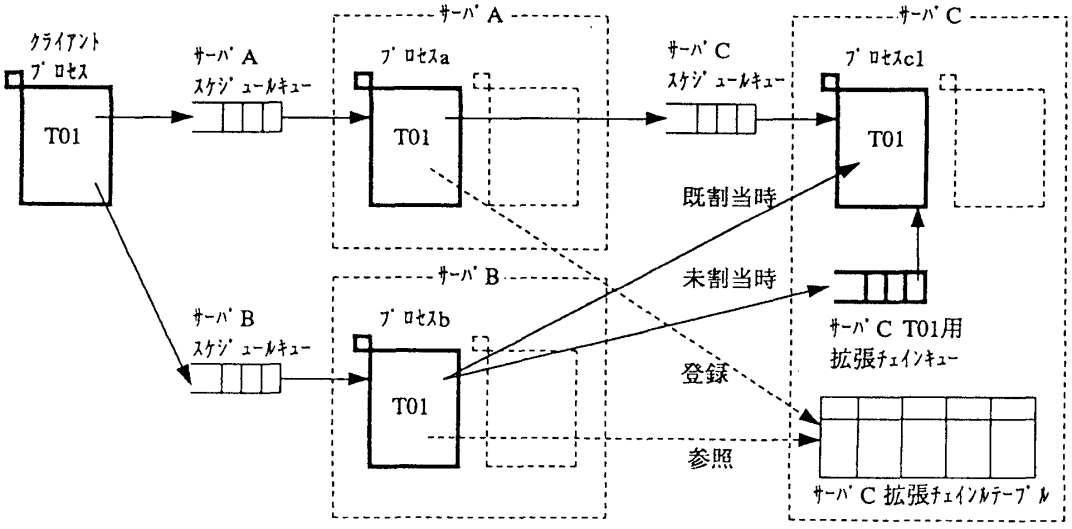

図２．　拡張チェインドRPCの概要