1M−8

文書画像理解のための分散処理方式

仙田修司　美濃導彦　池田克夫

京都大学工学部

1　はじめに

我々は、スキャナで入力された文書画像を理解し自動的に蓄積する文書画像データベースシステムを構築している。このためには、画像処理、文字認識処理、構造解析処理といったさまざまなレベルの処理を組み合わせて文書画像理解を行う必要がある。文書画像理解はそうした処理単位間での協調作業であると言える。

本稿では、このような協調作業を行うための環境として、処理単位(モジュール)を疎結合分散化して並列処理を行うモデルを提案する。本モデルの特徴は、モジュールインスタンスの動的バインディング、ストリーム中を流れるデータの明確な型付け、パイプライン結合による並列性である。また、モデルの実装による実験結果についても報告する。

2　分散処理モデル

2.1　モジュールとモジュールインスタンス

モジュールは入力データと出力データの仕様で定義される。これに対して、実際にプロセッサ上で処理を行う実体をモジュールインスタンスと呼ぶ。

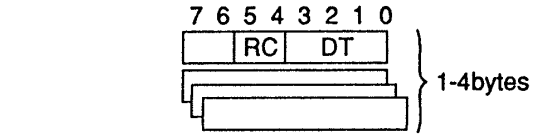
あるモジュールのモジュールインスタンスは同時に複数個存在し得る。このため、モジュールを呼び出す時に、モジュールとモジュールインスタンスとのバインディングが必要となる。このバインディングは動的なバインディングであり、呼び出し側のモジュールインスタンス(クライアント)が、呼び出したいモジュールのモジュールインスタンス(サーバ)の中から最適なものを選ぶことによって行われる。どのモジュールインスタンスが選ばれるかは非決定的であるので、本モデルのモジュールは局所状態を持たない関数的なものでなくてはならない。

現時点では、上記の動的バインディングを実装するためにブロードキャストパケットを使用している[1]。この手法では、クライアントはサーバに関する必要条件を記述したブロードキャストパケットをネットワーク上に送出し、条件を満たすサーバがそのクライアントに返事としてその時点でのCPU負荷や空きメモリ量などを返し、返事を受け取ったクライアントは最適なサーバを選択する。

ブロードキャストパケットを用いた動的バインディングの短所は、すべてのクライアントがその内部に複雑なバインディングの機構を有する必要があることと、ブロードキャストパケットが届かないサーバは利用できないことである。他の手法としては、モジュールインスタンスのバインディングを集中管理するサーバを設置する手法が挙げられる。バインディングの集中管理をすることで、ネットワーク全体を考慮した負荷分散を行うことが出来るが、バインディングサーバの処理が全体の処理のボトルネックとなる可能性がある。

2.2　ストリーム中のデータの型付け

ストリームはモジュール間でデータを転送するためのFIFOバッファである。本モデルでは、ストリーム中を流れるデータは明確に型付けされている。そのため、この型付きデータを監視することにより、モジュール間の結合に関する誤りが容易に発見可能である。データの型付けを行うことによってモジュールインスタンス間の通信量が増加するが、分散並列化された処理のデバッギングを容易にすることは大きな利益がある。



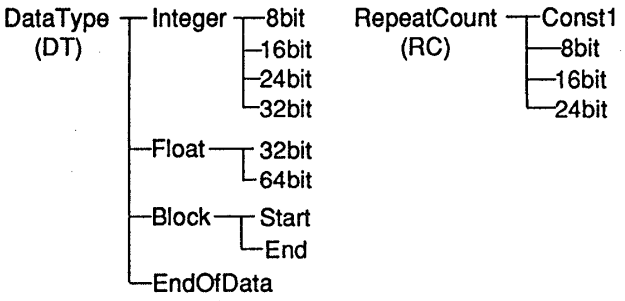


図1. DTS のタグ

本モデルでは、Data Transfer Standard(DTS)という明示的な型付きデータフォーマットを定義した。DTSは整数と浮動小数点数、およびそれらの配列や組み合せ(プロ

Design of a Distributed System for Document Image Understanding
Shuji SENDA, Michihiko MINOH, Katsuo IKEDA
Kyoto University

ック) をデータ型として持ち、データ型を表すタグをデータの前に付加する。図1にタグの構造を示す。DataTypeはデータの基本的な型を表し、RepeatCountが配列としての要素数を表している。

2.3 パイプライン結合

クライアントからサーバへ流れるデータは、ジョブリストとパイプラインデータに分けられる。ジョブリストはモジュール名とそのモジュールに与える引数の組を列挙したものであり、ストリームの結合先を示している。パイプラインデータは、先頭から徐々に処理を行うことが可能なデータである。このような、クライアントとサーバのFIFOバッファによる結合をパイプライン結合と呼ぶ。パイプライン結合されたモジュールは、データが全て揃っていなくても実行を進めることが出来る。モジュールのパイプライン結合の例を図2に示す。

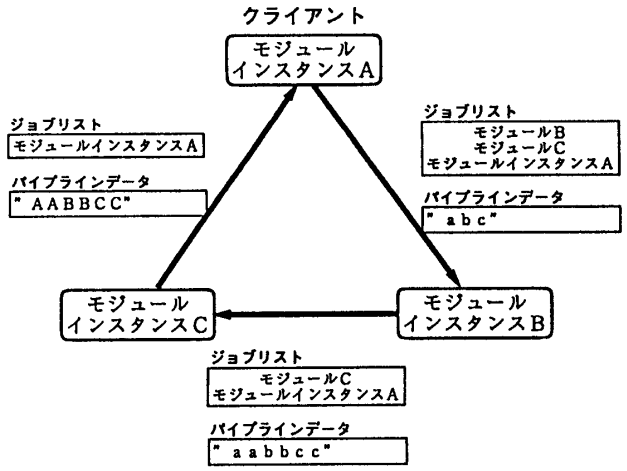


図2. モジュールのパイプライン結合

サーバは受け取ったジョブリストを保存しておき、与えられた引数とパイプラインデータを用いて要求された処理を行う。処理結果を出力する時には、ジョブリストの先頭の項目を取り除いたものを新たなジョブリストとし、一時的なクライアントとしてジョブリストの先頭のサーバを呼び出す。ジョブリストの効用は、サーバの処理結果の出力先の決定(バインディング)をサーバ自身が行うことにより、最適なサーバの選択が可能なことである。

処理結果の出力先は、動的バインディングによる不特定のモジュールインスタンスだけでなく、特定のモジュールインスタンスを指定することも出来る。モジュールインスタンスを直接指定したジョブリストを用いることにより、クライアントが一時的なサーバとなって、処理結果を受け取ることが出来る。

3 実験結果

イーサネットで接続された7台のSparcStationを用いて実験を行った。モデルを実装するために、DTSに準拠してメモリやソケットにアクセスするDTSライブラリ、動的バインディングによりパイプライン結合を行うクライアント・サーバライブラリをそれぞれ作成した。ブロードキャストパケットの送出とそれへの返答にはUDP/IP、ストリーム接続にはTCP/IPをそれぞれ用いた。

文字認識サーバを並列に用いて10文字の文字認識を行った場合の実行時間、10個のエッジ強調フィルタサーバをパイプライン結合した場合の実行時間を表1に示す。並列文字認識は、10個の文字認識要求を順次行い、その後、異なるポートからそれぞれの処理結果を受け取ることによって並列動作する。エッジ強調フィルタは、プロセスのフォークにより論理的に10個のフィルタが同時に動作する。「通常」の欄には参考として本システムを使用せずに実行した場合の実行時間を挙げた。

表1. 実行時間 (単位:秒)

	通常	1台	2台	4台	6台
並列文字認識	13.4	13.7	6.8	4.2	3.1
エッジ強調フィルタ	29.2	33.7	25.6	12.0	8.0

文書画像理解には、文字認識などの時間はかかるが容易に並列実行可能な処理と、画像処理のような通信量が非常に多い処理とが必須である。表1に示した結果より、本モデルでは、こうした相異なる性格を持つ処理のどちらに対しても有効であることが示された。

4 おわりに

疎結合分散処理モデルとして、モジュールインスタンスをパイプライン結合して並列実行を行うモデルを提案し、モデルの実装により実験し、有効性を確認した。

今後の課題として、分散環境における適切なエラー処理、モジュールインスタンスのバインディングをより効率的に行う手法の開発、ジョブリストの拡張による1対多および多対1のパイプライン結合などが挙げられる。

参考文献

[1] R.G.Smith: "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver," IEEE Trans. on Comp., Vol.C-29, No.12, pp. 1104-1113, 1980.

[2] 久世, 佐々, 中田: "ストリームによるプログラミングのための言語とその実現方式," 情報処理, Vol.31, No.5, pp. 673-685, 1990.

[3] W.R.Stevens: "UNIX NETWORK PROGRAMING," PRENTICE HALL, 1990.