並列オブジェクト指向言語 A-NETL の Windows NT マシンクラスタへの実装

4 E − 7

月川 淳　大津 金光　吉永 努　馬場 敬信
宇都宮大学工学部*

1 はじめに

A-NETL は、中粒度の並列処理記述を目的とする並列オブジェクト指向言語である[1,2]。これまでにメッセージパッシングライブラリ PVM と標準 C 言語へのトランスレータを用いてワークステーションクラスタで動作する処理系の実装を行った[3]。

本研究は A-NETL 処理系を Windows NT に特化した実装を行うことにより、Win32 API[4] の機能を十分に活用し高性能な処理系を実現すると共に、A-NETL の使用環境を拡大することを目的とする。

2 処理系の構成要素

本処理系の構成要素は次の通りである (図 1) 。

(i) メッセージパッシングレイヤ
A-NET デーモン(anetd) 及び、オブジェクトプロセスと動的リンクされるメッセージングライブラリ(anetdlib.dll) から構成され、コンピュータ間の通信とメッセージのオブジェクトへの配送を行う。

(ii)トランスレータ
A-NETL プログラムを Win32 API を用いた C++プログラムへ変換する。

(iii)ランタイムライブラリ
A-NETL 型なし変数の処理と、コンテキスト管理を行う。

(iv) ホストプログラム
メッセージパッシングレイヤの制御、プロセスの起動などの処理系全体のユーザーインターフェースを担う。

3 メッセージパッシングレイヤの実装

高性能な通信機能を実現するためにオブジェクト間でのバイト列送受信の機能のみを実装することとした。またマシンクラスタを使用する複数のユーザーのセキュリティを保護するために、メッセージパッシングレイヤプロセス自身や Named Pipe などのカーネルオブジェクトはすべて利用者のユーザーコンテキストで作成・実行する。

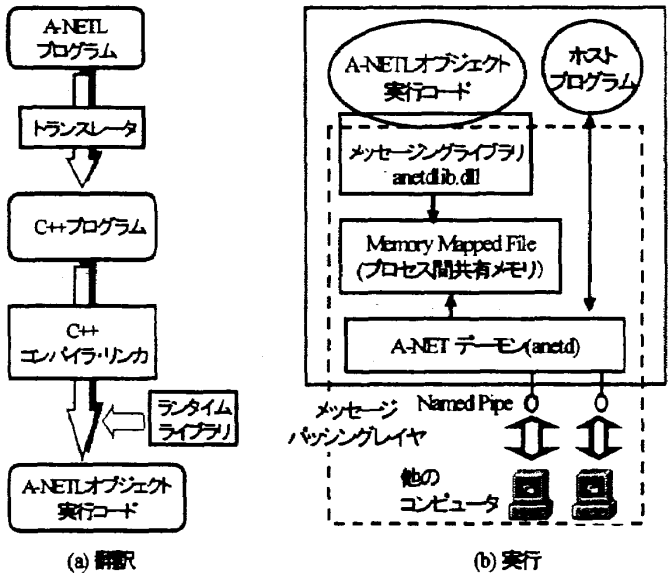

図 1 処理系の構成

(i) A-NET デーモン間の通信

コンピュータ間通信はメッセージパッシングレイヤの本体である A-NET デーモン(anetd)により行われる。各コンピュータの anetd は Named Pipe の複数のインスタンスで接続され、マルチスレッドでメッセージの受信処理を行う。また、効率的に通信を行うために ReadFileEx() などの非同期 I/O API を用いる。Named Pipe はネットワーク構成のトランスポート層に依存しないため、TCP/IP 以外にも NetBEUI などの軽量なプロトコルを使用することが可能となる。

(ii)デーモンと A-NETL オブジェクト間の通信

anetd と A-NETL オブジェクトプロセスは、不要なメモリコピーのオーバーヘッドを無くし、他プロセスのポインタ直接参照を可能とするために、各プロセスで同一アドレスにマッピングした Memory Mapped File (共有メモリ)を用いて接続する。メッセージを送信するオブジェクトは Memory Mapped File 上にバッファを確保し、そのアドレスを指定して anetd に対して送信要求をする。送信要求はウインドウメッセージを通じて行われる。

---

* Implementation of a Parallel Object-Oriented Language A-NETL for Windows NT Machine Clusters, Atsushi TSUKIKAWA, Kanemitu OHTSU, Tsutomu YOSHINAGA, Takanobu BABA, Utsunomiya University.

(iii)メッセージパッシングライブラリのAPI

A-NETL オブジェクトプロセスは anetdlib.dll が公開している関数群(API)を呼び出すことにより通信を行う。anetdlib.dll がオブジェクトプロセスに公開するAPI一覧を表 1に示す。

4　トランスレータ及びランタイムライブラリ

表 1 ANETDLIB.dll の公開関数一覧（一部）

API	機能
AnetdInitialize()	初期化
AnetdAllocBuffer()	送信バッファ確保
AnetdAllocPersistentBuffer()	送信バッファ確保
AnetdFreePersistentBuffer()	送信バッファ開放
AnetdSendMessage()	メッセージ送信
AnetdAddComputer()	ホストの追加
AnetdEnumComputers()	ホストの一覧取得
AnetdRecieveMessage()	メッセージ受信
AnetdCreateObject()	オブジェクトの作成
AnetdDeleteObject()	オブジェクトの破棄
AnetdHalt()	プログラム終了
AnetdSetPipeMode()	パイプモードの設定
AnetdGetPipeMode()	パイプモードの取得

トランスレータはA-NETL プログラムをWin32 API を用いたC++プログラムに変換する。これを、ランタイムライブラリとリンクしA-NETL オブジェクトの実行コードが生成する。この際特にコンテキストチェンジ、フューチャートラップ,型なし変数の処理等の機能を効率良く処理することが要点となる。

(i) コンテキストチェンジ

A-NETL オブジェクトではメソッドの終了や現在型メッセージ通信が行われるとコンテキストチェンジが発生する。本処理系ではコンテキストのCPU レジスタやスタックの保存・復元を高速かつ簡潔に処理するため、Win32 の Fiber 実行単位で管理する。Fiber は Thread と同様のコードの実行単位であるが、その切り替えはSwitchToFiber API を用いてプログラムが明示的に行うことができる。

(ii)フューチャートラップ

A-NETL では未来型メッセージを用いた場合、その返値が必要となるまでメソッドはサスペンドせずに実行を続けることができる。PVM を用いた処理系では、この機能を実現するために返値が参照されるたびにその検査を行う必要があった。本処理系ではこのオーバーヘッドを無くすために、Win32 構造化例外処理機能(Structured Exception Handler)を用いてフューチャートラップを実現する。

フューチャーメッセージの返値となる変数はポインタによる間接参照とし、フューチャーメッセージを送信時にこのポインタを無効な値(NULL)に設定する。この NULL ポインタにアクセス違反が発生した時点であらかじめ設定した例外ハンドラに OS より制御が渡るので、ここでコンテキストチェンジを実行する。リターンメッセージを受信時にはポインタを有効な値へと戻す。返値が格納されている場合にはポインタの間接参照によるオーバーヘッドのみでアクセス可能であり、特に返値がループ内で参照されるような場合には有効である。

(iii)型なし変数の処理

A-NETL で扱う型なし変数の処理は、トランスレータの生成するコードの可読性の向上とそれによるデバッグ性の向上、及び安定な実装とするために C++クラスライブラリを作成し対処する。また共有メモリに転送された変数を効率良く利用するために、変数の変更が行われるまでは不要なコピーは実行しない。

5　おわりに

並列オブジェクト指向言語 A-NETL 処理系をWindows NT マシンクラスタに実現するための実装上の工夫について述べた。

現在、メッセージパッシングレイヤの実装を終え性能評価及びパフォーマンスチューニングを行っている。

謝辞

本研究は、一部文部省科学研究費基盤研究(C)9680324、及び並列・分散処理研究推進機構の援助による。

参考文献

[1] T.Baba, T.Yoshinaga, T.Furuta, "Programming and Debugging for Massive Parallelism: The Case for a Parallel Object-Oriented Language A-NETL", Proc. Workshop on Object-Based Parallel and Distributed Computation (OBPDC'95), LNCS 1107, pp.38-58, 1995.

[2] T.Baba, T.Yoshinaga, "A-NETL: A Language for Massively Parallel Object-Oriented Computing", Proc. Programming Models for Massively Parallel Computers(PMMP'95), pp.98-105, 1995.

[3] 斎藤宣人,古田貴寛,月川淳,馬場敬信,吉永努,並列オブジェクト指向トータルアーキテクチャ A-NET －WS クラスタへの A-NETL の実装－,情報処理学会第 52 回全国大会予稿集 2L-6,1996

[4] Microsoft Corporation, "Platform, SDK and DDK Documentation", Microsoft Developer Network Library CD, Microsoft Corporation, Oct 1997.