システムソフトウェアと
オペレーティング・システム 61－5
（１９９３．８．１８）

分散環境上で共有メモリ空間と メッセージパッシング型通信機構を提供する 並列記述言語処理系 ParaDisE の試作と評価

岡村耕二* 平原正樹** 縄田毅史*** 荒木啓二郎*
* 奈良先端科学技術大学院大学 情報科学研究科
** 九州大学 工学部
*** 松下電器産業（株）情報システム研究所

我々の研究の目的は、具体的な分散/並列アプリケーションの記述、実行を通じて、分散 OS および、並列記述言語で構成される分散/並列環境が提供すべき機能を追求することである。

本稿では、我々が設計、実装した並列記述言語処理系 ParaDisE の概要を述べ、ParaDisE 上で分散/並列アプリケーションの記述、実行を行なった結果により分散環境が分散/並列アプリケーションに対して提供すべき機能について議論する。結論では、分散環境が、分散/並列処理のために必要で十分である機能を示す。

Implementation and Study of ParaDisE on Distributed Environment

Koji OKAMURA* Masaki HIRABARU**
Takeshi NAWATA*** Keijiro ARAKI*
* Graduate School of Information Science
Advanced Institute of Science and Technology, Nara
** Department of Computer Science and Communication Engineering,
Kyushu University
*** Matsushita Electric Industrial Co., LTD.
Information Systems Research Laboratory

Recently there are many distributed operating systems and parallel programming languages. However, because of lacks of assumption of the concrete applications using these operating systems and programming languages , occasionally the functions for the distributed and parallel processing are unnecessary or insufficient.

In this paper, we designed and implemented the parallel and distributed environment: ParaDisE for parallel distributed processing. And we discussed the necessary functions for the parallel and distributed processing with programming and executing the distributed and parallel applications. Finally we suggest the necessary and sufficient functions of distributed and parallel environment for distributed and parallel processing.

1 はじめに

現在、多くの分散 OS や、並列記述言語が存在しているが、それらの言語の仕様や、OS の機能は、具体的なアプリケーションを想定されていない場合が多いため、必ずしも、分散/並列計算を行なうために適切に提供されているとは限らない。我々の研究の目的は、具体的な分散/並列アプリケーションの記述、実行を通じて、分散 OS および、並列記述言語で構成される分散/並列環境が提供すべき機能を追求することである。

そこで、疎結合・密結合が混在する分散システム上で、単一の言語を用いて分散/並列アプリケーションの記述が可能でありまた、単一の処理系上でそのアプリケーションを動作させることができる分散環境 ParaDisE の構築および、ParaDisE 上で実際の分散/並列処理を行なうことで、分散環境が提供すべき機能の見直しを行なうことを目指した [2]。

我々は、プログラム言語としては、並列プログラミング言語 Ada の持つ、タスク、ランデブといった並列処理機能を提供する Ada [1] 言語のサブセットを ParaDiseE の言語仕様として定め、ParaDisE を LAN で結合された UNIX 群で構成される疎結合分散環境上で実現した。

分散/並列アプリケーションとして、定型的な並列計算である数値計算を想定し、ParaDisE/言語 を用いて、実際に分散/並列ソーティングの記述を行ないプログラムの実行を ParaDisE 上で行なった。本稿では、分散/並列処理環境 ParaDisE の概要と、ParaDisE 上で実際にアプリケーションを実行したことで得られた分散/並列処理系に必要な機能について報告する。

2 分散/並列プログラミングの問題点

まず、対象とする分散/並列システムのモデルを示すと共に、分散システム上における分散/並列プログラミングの問題点を指摘する。

2.1 分散システムのモデル

我々は、対象とする分散環境として、疎結合分散環境と密結合分散環境が入り交じった分散環境を想定している。典型的な分散システムの例を図 2-1 に示す。

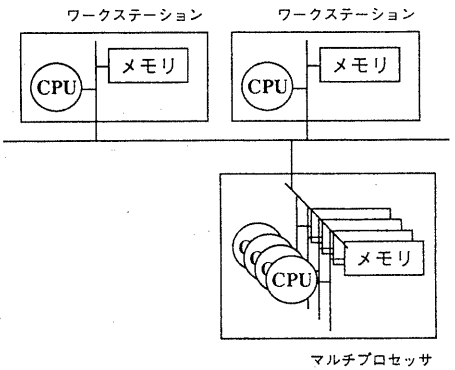

図 2-1: 分散システム

このような分散システムでは、前処理・後処理は、フロントエンドのワークステーションや、ワークステーション群で分散処理を行ない、主なる処理はマルチプロセッサ上で並列処理で高速に行なうというのが典型的な利用法である。

2.2 分散/並列プログラム

我々は、対象とするプログラムが MIMD 型の分散/並列プログラムであることを想定している。MIMD 型のプログラミングでは、各計算機毎にプログラムを作成するか、同じプログラムを全ノードにロードして、ノード毎に内部で振舞いを変えるというのが一般的である。前者は、クライアント/サーバ型のシステムで、また後者は、マルチプロセッサ型のシステムで用いられている方法である。しかし、これらの方法では、プログラム間のデータの授受の検査が困難であるため、信頼性を欠くという問題や、大域的な最適化が困難であるため、高効率の追求が困難になるという問題がある。

これらの問題を解決するために、全体を一つのプログラムとして作成し、ロードは機能ごとのモジュール単位に行なう方式を提案する。しかし、この方法ではあるモジュールが、他のノードから参照渡しをされた場合に問題が生じる。

3 分散/並列処理系 ParaDisE

2 章であげた問題を解決するため、我々は、分散/並列環境 ParaDisE (Prallel/Distributed Environment) を

提案し、ParaDisE を我々の研究のプラットフォームとした。我々は ParaDisE 上において単一アドレス空間モデルの提供および、メッセージパッシング通信機構の実現を目指した [5]。

3.1 単一アドレス空間モデル

単一アドレス空間モデルは、分散環境のアドレス空間を共有メモリのように、どこからでも見えるアドレス空間と考えるものである。この参照の一様性を保証するための単一のアドレスを分散アドレスと呼ぶ。図 3-1 に単一アドレス空間を図示する。

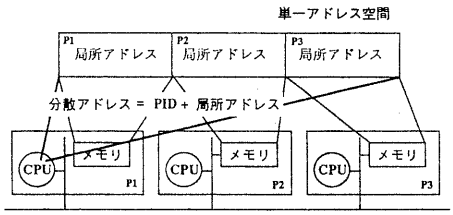

図 3-1: 単一アドレス空間

単一アドレス空間により提供される分散アドレスを用いることで、異なるノード間の参照渡しが可能になる。また、コンパイラは、局所参照、遠隔参照の区別を付けることで、無用なアドレス変換のオーバヘッドを避ける。コンパイラが参照の局所・遠隔の区別を付けることで、アドレス変換が省略できる利点を生み、分散キャッシングのためのヒントを出力することができる。

3.2 言語仕様

ParaDisE の言語、ParaDisE/言語の仕様は並列プログラミング言語 Ada のサブセットである [3] 。ParaDisE/言語は、Ada のサブセットであるため、既存の Ada 処理系をプログラムの開発、テスト、デバッグ環境として利用できる。

ParaDisE/言語には Ada から、基本的な言語仕様として、パッケージ、配列、多重定義、アクセス型を採用した。また、並列処理に関する言語仕様は、タスク (型) 宣言、タスク配列、動的タスク生成、エントリ呼びだし、アクセプト、選択的アクセプトである。また、タスクの従属関係および終了・消滅に関するセマンティクスの簡略化を行なった。削除・簡略化した機能の大部分は、基となった Ada の基準文法書に明確な定義がないか、分散環境上での実現が非現実的なものである。

3.3 コンパイラとローダ

ランデブの引数は、値渡しでも、アドレス渡しでも構わない。しかし、アドレス渡しの場合、副作用が問題となる。我々は、この問題を分散ディスプレイ方式を用いることで対処した。また、分散アドレス渡しの場合、その引数の内容に参照が起こる場合には、キャッシングを行なって、それ以後の参照のコストを低減させる。

ParaDisE では Ada におけるタスク、パッケージ、手続き、関数などのコンパイル単位をモジュール分割の単位としている。分割の際には、それぞれの単位に表 3-1 にあげた属性を付与する。

表 3-1: 属性

属性名	割り当て
all	すべてのプロセッサ
any	いずれか一つのプロセッサ
floating	必要 (参照) に応じて
specified	プロセッサ番号を指定

タスクは、どのプロセッサ上でも動的に生成される可能性が高いので all であることが望ましい。他の単位はいずれか一つのプロセッサ上にだけ割り当てられる any であることが望ましい。

これに対して並列性の向上のために、複数のプロセッサから呼び出される手続き・関数は、呼びだし側プロセッサで重複して、局所的に実行することができる。コンパイラはそのようなそれ自身が遠隔参照を行なわないようなパッケージ・手続き・関数を見つけ出すと、floating 属性を付与する。floating 属性を持つ分割単位への参照は全て局所参照になる。即ち、参照する各々のプロセッサ上に floating 属性を持つ分割単位のコピーが配置されることになる。プログラムライブラリとして用意するパッケージ・手続き・関数の大部分は floating 属性を持つ。

any を持った手続き・関数は遠隔呼びだし、floating を持った手続き・関数は局所呼び出しとなる。all, any, floating の付与はコンパイラが行ない、その属性に依存して、遠隔参照・局所参照の区別および、参照の最適化

を行なう。図 3-2 に付与された属性に従ってモジュールをロードした例を示す。

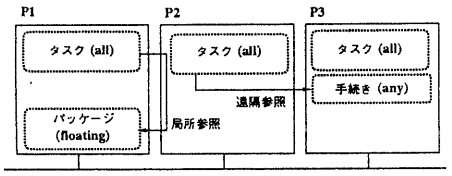

図 3-2: 属性に従ったロードの例

なお、specified は、プラグマ等でプロセッサ番号を指定する場合に使用する。

3.4 分散 OS

ParaDisE における分散 OS カーネルは ParaDisE/DaOS (Distributed Ada Operating System) である [4]。ParaDisE/DaOS は、ParaDisE/言語の仕様を実現するために必要となる OS の機能を重点的にサポートしている。

ParaDisE/DaOS は教育用の OS である XINU [6] を拡張した OS で、階層構造に設計、実装されている。以下に各階層の機能の説明を行なう。

- DaOS カーネル

  ParaDisE/DaOS のカーネルは XINU である。XINU がスレッド管理、スレッド通信、同期管理を行なっている。ただし、ParaDisE は、UNIX 上で動作するため、メモリ管理や、低レベルな入出力の階層は、UNIX ライブラリを用いている。

- ネットワークサポート

  DaOS カーネル自身には、ネットワーク機能がないので、ネットワークに対する汎用的なアクセスは、この階層で行なわれる。

- 分散アドレス管理

  単一アドレス空間サポート部は、分散アドレスに対するサービスを対応するノードの局所アドレスに対して行なう。サービスの種類には、分散アドレスへの読み出し、書き込みおよび、手続き呼び出しがある。手続き呼び出しは、ブロック型と、非ブロック型を用意している。

- タスク管理

- タスクの生成/消滅

  遠隔/局所ノードにタスクの生成/消滅を行なう。タスクの親子関係は、Ada におけるタスクの親子関係に準じている。

- ランデブ

  タスク間通信として、ランデブをサポートしている。他ノードにあるタスクへのエントリコール、エントリアクセプトまた、選択的なアクセプトをサポートしている。

3.5 言語と OS のインタフェース

ParaDisE/DaOS の階層と ParaDisE/言語仕様は図 3-3 に示すように、大域変数は ParaDise/DaOS の分散アドレス管理の階層が、またタスキングは、タスク管理の階層がサポートしている。このように、ParaDisE/DaOS は ParaDisE/言語の仕様を意識して階層化している。

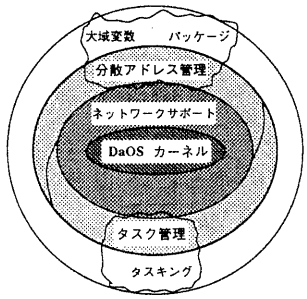

図 3-3: 言語と DaOS の階層

3.6 処理系概要

ParaDisE 上で、ParaDisE/言語で記述されたソースコードが実行形式になるまでの処理、およびそのファイルを実行する処理の流れを 図 3-4 に示す。

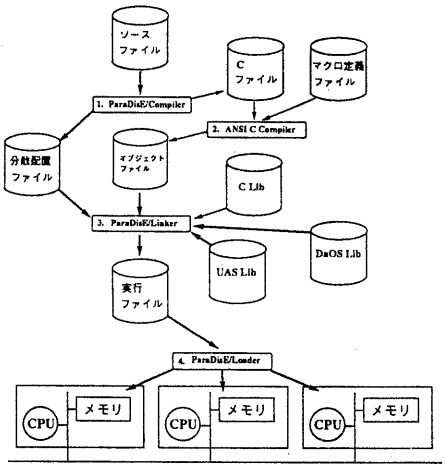

図 3-4: ParaDisE における処理のフローチャート

3.7 プログラムの実行

我々は、ParaDisE の実行処理系を TCP/IP で接続された複数の UNIX ワークステーションで構成される疎結合分散環境で実現した。

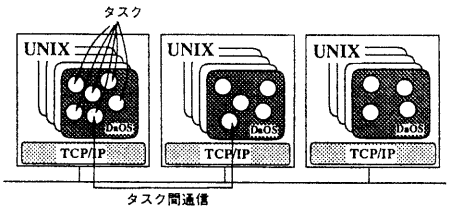

図 3-5: UNIX 上での DaOS の実行

ParaDisE/DaOS は、ハードウェア上で直接動作するネイティヴな OS ではなく、UNIX 上の一つのプロセスを仮想プロセッサとして想定し動作する。また、一台の物理プロセッサ上に複数の仮想プロセッサを割り当てることができる。ParDisE/ 言語のタスクは、UNIX 内のスレッドのように振舞う。図 3-5 に UNIX 上で DaOS が実行されている様子を示す。

4 アプリケーションの記述と実行

我々は、PraDisE/言語を用いて分散/並列アプリケーションの記述を行ない、そのアプリケーションを ParaDisE 上で分散/並列実行させた。分散/並列アプリケーションとしては、定型的な並列数値計算を想定した。並列数値計算は、分散/並列処理の記述、実行に対して必要な多くの機能を処理系に要求する。

4.1 分散/並列数値計算のモデル化

我々は以前の研究で疎結合分散環境における分散/並列数値計算のモデル化を行なった [7]。元データを保持しているタスクをメインタスク、分散して、ローカルに互いに並列に処理を行なうタスクをエージェントタスクと呼ぶ。エージェントタスクは、ノードに複数存在しても構わない。

我々は、疎結合分散環境における分散/並列数値計算を、以下の 3 つのフェーズに分けた。

1. データの分配

   メインタスクは、エージェントタスクにデータの分配を行なう。

2. ローカル処理

   メインタスクからデータを受けとったエージェントタスクはローカル処理を行なう。エージェント間で通信を行なうこともある。

3. 結果の収集

   ローカル処理が全て終了したエージェントタスクは、メインタスクに結果を返す。

しかし、このモデルは疎結合分散環境に依存しており、密結合分散環境における分散/並列計算に対しては適切なモデルであるとはいえない。そこで、モデルをより汎用的なモデルに拡張した。

1. 処理の分配

   メインタスクは、エージェントタスクに処理の分配を行なうと同時に、エージェントタスクにローカル処理を開始させる。

   疎結合分散環境であれば、処理の分配はメッセージパッシングを用いてデータも分配する。また、密結合であれば、データの分配は不必要であり、メインタスクとエージェントタスクは共有メモリに対する同期機構を用いて処理を開始する。図 4-1 にメインタスクからエージェントタスクに処理が分配される様子を示す。

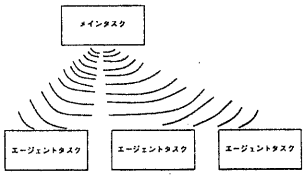

図 4-1: 処理の分配

2. ローカル処理

メインタスクから処理を受けとったエージェントタスクはローカル処理を行なう。エージェント間で通信を行なうこともある。エージェント間の通信は、分散環境の形態によって、メッセージパッシングで行なう場合もあれば、共有メモリで同期を取りながら行なう場合もある。図 4-2 はローカル処理の様子を示している。

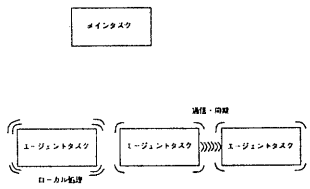

図 4-2: ローカル処理

3. 処理終了の通知

ローカル処理が全て終了したエージェントタスクは、メインタスクに処理が終了したことを通知する。

疎結合分散環境であれば、エージェントタスクは、処理結果をメインタスクに戻す必要がある。密結合では、共有メモリを用いているため、エージェントタスクはメインタスクに処理結果を戻す必要はないが、ローカル処理が終了したことを何かの手段で通知する必要がある。

メインタスクは、通知するエージェントタスクの順番はわからないので、非決定的に、エージェントと通信または、同期を行なうことができる必要がある。図 4-3 は、メインタスクが各エージェントタスクから処理終了の通知を受け取っている様子を示している。

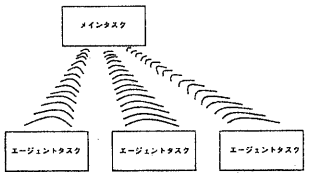

図 4-3 : 処理終了の通知

4.2 分散/並列数値計算の例

我々は、分散/並列アプリケーションとして、典型的な分散/並列数値計算であるソーティングを選択した。我々が選択した Odd-Even Sort によるソーティングアルゴリズムを、前節で定義した汎用的な分散/並列数値計算のモデルに合わせて説明する。

1. 処理の分配

メインタスクは、エージェントタスクにデータの分配を行なう。また、エージェントタスクは、自分のタスク ID をメインタスクから受けとる。

2. ローカル処理

各エージェントタスクはまず、分配されたデータのソーティングをローカルに行なう。ソートが終了すると、エージェントタスク間でマージを行なう。

タスク ID が、i 番であるエージェントタスクは、以下の操作をエージェントタスクの数の回数繰り返し、マージを行なう。

ループの回数が偶数 (奇数) 回目の時、

- 偶数 (奇数) 番目のエージェントタスク i は i+1 番目のエージェントタスクにデータを渡す。
- 奇数 (偶数) 番目のエージェントタスク i は受け取ったデータと自分のデータをマージし、分割して半分を i-1 番目のエージェントに返す。

3. 処理終了の通知

全ての処理が終了したエージェントタスクは、処理が終了したことをメインタスクに報告する。

4.3 プログラムの記述

Odd-Even Sort を、ParaDisE/言語を用いて記述した。現在の ParaDisE は、疎結合分散環境上で単一アドレス空間を提供しているが、環境には依存しないようにプログラミングを行なった。

1. 処理の分配

   メインタスクは、処理をエージェントタスクに分配する。この時、エージェントタスクには、自分の ID を知る必要がある。エージェントタスクが、自分の ID を取得する方法としては、以下の 2 つの方法をあげることができる。

   1. メインタスクから教えてもらう
   2. 大域的なテーブルを見る

   前者は、メッセージパッシングを用い、後者は、共有メモリを用いる。また、処理すべきデータも同様に、メインタスクから受けとる場合と、共有メモリから取ってくる場合がある。共有メモリの場合、値ではなく、アドレスだけ取っておいても良い。

   共有メモリを用いる場合、メインタスクと何らかの方法で同期を取る必要がある。同期は、例えば、OS が提供するセマフォを用いればよいが、他ノードのタスクと同期をとる必要がある場合は、ローカル用のセマフォは使えないので、ノード間で同期を取ることができるプリミティヴを用いる必要がある。ただし、参照の場合であれば、相互排除は不必要であるため同期は必要でない。ParaDisE では他ノードのタスクとランデブをすることができるので、この機能を用いた。

2. ローカル処理

   エージェント間の通信は、ランデブで行なわれる。データのやりとりは、共有メモリを用いても可能である。

3. 処理の終了通知

   ローカル処理の終了したエージェントは、メインタスクに通知する必要がある。メインタスクは、すべてのエージェントタスクから終了通知を受けとると、処理が終了したことを知る。

図 4-4 は、分散/数値計算モデルと ParaDisE/言語の依存関係を示している。

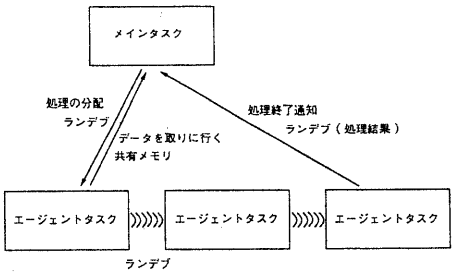

図 4-4: 数値計算モデルと言語仕様

5 評価と考察

ParaDisE/言語の仕様および ParaDisE/DaOS と、実際の数値計算の記述、実行に関する評価、考察を行ない、最後に仕様の拡張の提案を行なう。

5.1 ParaDisE/言語 仕様と数値計算

ParaDisE/言語の仕様のうち、タスクの親子関係とランデブを上げ、これらの仕様と数値計算の記述について考察を行なう

1. タスクの親子関係

   タスクの親子関係のセマンティクスを簡単に説明する。

   - 処理の開始

     親タスクの単位がタスク宣言の begin に到達した時に全ての子タスクは活動状態に入る。

   - 処理の終了

     親タスクはある単位は自分に従属しているタスクが全て終了してしまわない限り制御を自分の所から離すことができない。

   これらの仕様は、共有メモリも用いた時の「処理の分配」および、「処理の終了通知」のモデルとマッチしており、有効に利用することができた。

2. ランデブ

ランデブは、任意のタスクが、accept 状態で待っているタスクを指定して、エントリコールを行なうことで実現される。ParaDisE/DaOS では、遠隔なタスクとランデブをすることもできる。

・エントリコール

エントリコールを行なうタスクは、ランデブの相手を指定することができる。

・エントリアクセプト

エントリアクセプトでランデブを行なうタスクは、ランデブの相手タスクの指定をすることはできない。

選択的なアクセプトの仕様は、「処理の終了通知」のモデルとマッチしており、メインタスクとエージェントタスク間のの非決定的なランデブの実現に有効に利用することができた。

・優先度待ち行列

Ada のタスクのプライオリティは、プラグマで指定することにより変更可能である。しかし、この方法だけでは、一旦ランデブの待ち行列に入ってしまうと、優先順位の効果はなくなり、期待していたスケジューリングは保証されない[8]。本研究が対象とした数値計算では、この仕様は必要ではなかったため、検討は行なわなかったが、今後の課題の一つである。

5.2 言語の仕様、OS の機能の拡張について

本研究により得られた ParaDisE/言語で拡張すべき言語仕様や OS の機能をあげる。

・一斉通信/同期

処理の分配のフェーズでは、メッセージパッシングのブロードキャストや、バリア同期機構があれば、記述性や実行速度の向上が期待できる。しかも、ParaDisE では、タスクの親子関係をランデブで実現しているため、この機能は下位で実現されるとより多くの機能の効率を上げることが期待できる。

しかし、これらの、メッセージパッシングのブロードキャストや、バリア同期機構は、本来、ハードウェアがサポートすべき機能であるため、言語レベルでの提供方法が今後の課題となる。

・ネットワークを越えるタスク間の同期

共有メモリを使用する時の同期機構には一般にセマフォが用いられる。しかし、他ノードにあるタスク間の同期にはセマフォは使うことはできない。そこで、ノードをまたいで同期をとるようなプリミティブが必要である。

ParaDisE では、分散環境用に拡張したランデブを用いればよいが、メッセージパッシング機構で共有メモリを扱うのは冗長であるため、今後は別なプリミティヴで対応する必要がある。

6 まとめ

本稿では、共有メモリ空間とメッセージパッシング型通信機構を提供する並列記述言語処理系 ParaDisE の概要を述べ、ParaDisE を用いて、分散/並列アプリケーションの記述、実行を行ない、分散/並列アプリケーションに必要な機能に対する考察、評価を行なった。

参考文献

[1] "Reference Manual for the Ada Programming Language," ANSI/MIL-STD-1815A, United States Department of Defense , Jan., 1983. (情報処理振興事業協会編 "最新 Ada 基準文法書", bit 別冊 , 共立出版, 1984.)

[2] 岡村, 縄田, 平原, 荒木: "単一アドレス空間モデルに基づいた分散環境上での並列プログラミング言語処理系の実現", 信学技報, CPSY89-20, pp.33-38, 1989.

[3] 縄田, 岡村, 平原, 荒木: " 並列/分散環境上でのプログラミング言語処理系 ParaDisE", 情報処理学会 第 40 回 全国大会講演論文集, pp.754-755, 1990.

[4] 岡村, 縄田, 平原, 荒木: "分散環境上での並列プログラミング処理系用の OS DaOS の概要", 情報処理学会 第 40 回 全国大会講演論文集, pp.756-757, 1990.

[5] 平原, 岡村, 縄田, 荒木: "ランデブと共有変数を持つ並列型言語の実行支援系", 情報処理学会 計算機アーキテクチャ研究会資料, 90-ARC-82-4, 1990.

[6] Comer,D.: "Operarting System Design, the XINU Approach, " Prentice-Hall, 1984.

[7] 荒木, 岡村, 藤井, 平原: "LAN 上での分散計算における効率と粒度について", 情報処理学会マルチメディア通信と分散処理研究会資料、MDP-45-12, pp.85-92, 1990.

[8] Araki, Fujii, Hirabaru: "Real-Time Rendezvous for Distributed Ada Programming," Proc. Int'l Joint Workshop on Computer Communications, pp.9-16, 1991.