Lites における並列分散処理機能の設計†

4Z－1 安田 絹子[1] 平川 泰之[1] 遠山 緑生[1] 斎藤 鉄也[1] 服部 隆志[2]
[1]慶應義塾大学 大学院政策・メディア研究科 [2]慶應義塾大学 環境情報学部

1 はじめに

複数マシンから成る並列分散システム上でアプリケーションを開発するためには，複数の計算機資源を有効に扱うための使い易いインターフェースが求められる．本稿では，複数マシンで動作する Mach マイクロカーネル上においてアプリケーションの開発・実行環境を提供する Lites (4.4BSD Lite Server)[1] の並列分散拡張について述べる．

これまでにも，並列システム上やワークステーションクラスタ上で動作する UNIX 系列のオペレーティングシステムやサーバは数多く研究されている [2, 3, 4]．また，殆んどのシステムはシステムコールのエミュレートや既存システムを拡張することで UNIX 環境との互換性を提供している．しかし，その上で動作する並列アプリケーションを開発するためには，新しい API (Application Program Interface) に従って，逐次プログラムの開発とは違うスタイルでプログラムを書かなければならないものが多かった．

我々が拡張を行なっている並列分散 Lites では，UNIX のプロセス，パイプ，ファイルなどのプログラムインターフェースを自然に並列分散拡張することで，従来のプログラミングスタイルを保ったまま並列分散プログラム開発環境を提供する．また，シェルなどの適切な wrapper を用意することで，従来の UNIX コマンドをそのまま並列システム上で動作させることも可能である．本サーバは，これらの基本的な機能の拡張に加え，共有記憶などの並列分散アプリケーションに不可欠なサービスの提供も行なう．

以下本稿では，2章で並列分散 Lites の設計と構成についてより詳細に述べ，3章で Lites が提供する並列分散プロセス管理機構について述べる．その後 5章でまとめと今後の予定を述べる．

2 並列分散 Lites の設計と構成

並列分散 Lites は，ネットワーク接続された同種の PC あるいはワークステーションから構成される並列分散コンピュータシステム上で動作し，複数のユーザによる並列分散処理応用ソフトウェアの開発・実行を支援するための一般的なサービスを提供する．並列分散処理のために拡張された機能には，並列分散プロセス管理機構および，並列分散プロセスに対してサービスを行なう共有記憶，データストリーム，ファイル入出力機構が含まれる (図 1参照)．

各機構は以下のようなサービスを提供する．

- **並列分散プロセス管理機構** 並列分散システムにおける複数ノード上でのプロセス (群) の生成，消滅，同期を制御する

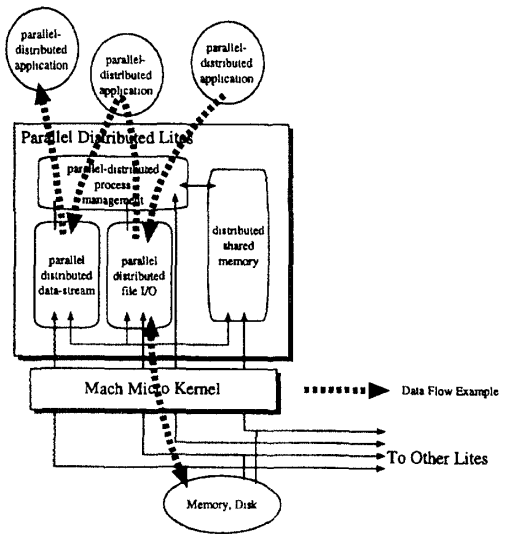

図 1: 並列分散 Lites の全体構造

- **分散共有記憶** システム上で記憶領域の共有を仮想的に行なうために，複数マシン間での記憶領域の一貫性を維持する
- **並列分散データストリーム** 協調動作する二つのプロセス群の間で並列なデータ転送を提供する
- **並列分散ファイル入出力** 複数のディスク上に分割して格納されているファイルに対し，協調動作するプロセス群による並列なデータの読み書きを実現する

複数の計算機資源を有効に活用し，かつ無駄なネットワークトラフィックを抑えるためには，これらの各機構が互いに協調して効率良く動作することが重要である．並列分散 Lites では，UNIX のパイプラインの概念を並列分散拡張し，並列分散プロセス群をパイプラインでつないだときにシステム全体が最も効率良く動作するよう設計を行なった．

並列分散 Lites の最大の特徴は，システム内に分散するデータの位置に従って並列分散プロセスを生成できるという点である．まず，ファイルの並列入出力を実現するために，ファイルは複数 (あるいは単一) のマシン上に分割して格納される．これらの並列分散ファイルに読み書きを行なう並列分散プロセス群は，ファイルのデータの各位置に従って生成することが可能である．また，二つの並列分散プロセス群を並列分散データストリームでつないだ場合，データを受けとる側のプロセス群をデータを出力する側のプロセス群の位置に従って生成することができる．また，パイプライン機構だけでは開発できるアプリケーションの種類が限定されるため，データ共有機能を補完する分散共有メモリを提供する．

3 並列分散プロセス管理機構

並列分散 Lites では，並列分散プロセスの生成は他の機構と密接に関連して行なわれる．ここでは，並列分散 Lites の各機構のうち並列分散プロセス管理機構について

Design of Parallel and Distributed Extensions for Lites
Kinuko Yasuda[1], Yasuyuki Hirakawa[1], Norio Touyama[1], Tetsuya Saitou[1], and Takashi Hattori[2]
[1]Graduate School of Media and Governance, Keio University
[2]Faculty of Environmental Information, Keio University
E-Mail: <kinuko@sfc.keio.ac.jp>

†本研究は情報処理振興事業協会 (IPA) の創造的ソフトウェア育成事業「並列・分散処理基盤ソフトウェアの開発」の一部として行なわれた．

述べる。

並列分散プロセス管理機構は，並列分散プロセスの生成，終了，管理を行なう．各機能は UNIX のプロセス管理機能とほぼ同等の操作を行なうが，それぞれ複数ノード上での複数プロセスの生成，管理を支援し，各ノード上で動作する複数の Lites が協調することでシステム全体の効率と一貫性を保つように機能する．

並列分散プロセスは，UNIX における fork(), vfork() を並列分散拡張したシステムコール pfork(), pvfork() によって生成される．pfork() によって作られたプロセス群は，アドレス空間を含むプロセス情報を親プロセスから継承する．アドレス空間は，分散共有メモリ機構を用いて継承され，必要に応じて共有あるいはコピーされる．pvfork() は並列分散シェルなどを効率良く実装するために提供されるもので，exec() が呼ばれるまでは子プロセス群は親プロセスのあるローカルノードで実行される．

並列分散プロセスの proc 構造体 (プロセス構造体) は，通常の proc 構造体と同様に親子関係に基づいて管理される．親子関係はノードを越えて管理される必要があるため，proc 構造体を親子，兄弟プロセスなどの必要なノード情報を含むよう拡張する．proc 構造体の管理は分散して行なわれ，シグナル通知などプロセスの親子関係を辿る必要がある場合以外はローカルに処理が行なわれる．

pfork() および pvfork() における並列分散プロセスの生成先は，引数として渡されるノードテーブルによって決定される．ノードテーブルは，ユーザが新たに生成することも可能だが，並列分散ファイル構造体 (並列分散ファイルあるいは並列分散パイプを指す) からも得ることができる．この機構によって，並列分散 Lites の特徴であるデータの位置に従ったプロセスの生成が可能となる．

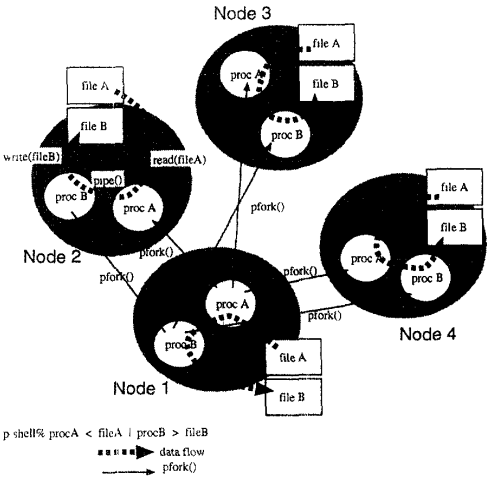

図 2: Lites における並列分散パイプライン

図 2：プロセス A が並列分散 fileA から入力を受け付け，プロセス B との間にパイプをはってデータを送り，プロセス B が file B に書き出す場合．各プロセス A, B は自分の分身を file A のデータ位置に基づいて Node 1, 2, 3, 4 上に作り，file A から file B までのリダイレクトおよびパイプによるデータ転送は各ノード上でローカルに行なわれる．

pfork() 後の子プロセス群は，プロセスをリモートノードに生成し，各データの継承，proc 構造体のメンテナンスなどを行なった後，オープンされている並列分散ファイルおよび並列分散パイプの入出力の再設定を行なう．並列分散ファイルの場合は，生成先のノード上にある断片ファイルに対して再オープンを行なう．並列分散ファイルは通常は仮想的に一つのファイルとして扱われるが，断片ファイルに対して入出力の再設定が行なわれた場合，ファイルに対する読み書きは断片ファイルに対してのみ行なわれる．並列分散パイプの場合，相手側のプロセス群のうち同じノード上にあるプロセスに対してパイプを接続し直す．もしも同じノード上に相手の接続先のプロセスが存在しなかった場合，ノードを越えてパイプを接続することになる．これらの作業が終ると，pfork() は終了する．

pfork() 中のファイルおよびパイプの再設定によって，pfork() 後の子プロセス群は仮想的に一つのファイルあるいはパイプを通し，協調して並列に入出力を行なうことができる．また，入力データの位置情報を子プロセスに伝播させていくことができるため，理想的な場合では各システムコール内で転送される内部情報以外のデータは全てローカルに処理が行なわれる (図 2 参照)．

4　関連研究

ネットワーク接続されたワークステーションのための分散システム NOW [2] では，既存の OS の上に並列分散環境を構築するための GLUnix と呼ばれるシステムレイヤを構築している．我々の Lites とは対象的に，GLUnix は並列プログラムの開発のために独自の API を提供している．また，NOW では並列プログラムのために Split-C などの言語拡張を行なっている．

UNIX とのセマンティクスの一貫性を保った分散ファイルシステムやプロセス管理を提供しているマルチコンピュータのための分散 OS の研究としては，Solaris MC[3] があげられる．Solaris MC と並列分散 Lites の違いは，Solaris MC がシングルシステムイメージを提供するという点である．また，プログラミングの観点からも，CORBA を採用し全てのシステムコンポーネントとの通信のために ORB を使うなどの新しい試みを行なっており，従来のスタイルの継承に重点をおいた我々の拡張とは方向性が異なっている．

5　まとめと今後の予定

本稿では，複数計算機資源を有効に扱うための一般的なインターフェースを提供する Lites における並列分散機能について述べた．我々が拡張した並列分散 Lites では，UNIX のパイプラインの概念を拡張することで，従来のプログラミングスタイルを保持したまま並列分散プログラムを開発・実行することが可能である．また，分散共有メモリ機構を提供することで，パイプラインだけでは実現できない応用アプリケーションの開発を支援している．

現在，並列分散 Lites は本稿で述べた設計に基づき各機構を実装中である．今後，これらの機構を利用する並列分散シェルおよび並列分散アプリケーションを実装し，性能評価を行なっていく予定である．

参考文献

[1] J. Helander, "Unix under Mach – The Lites Server", Master Thesis, Helsinki University, Dec 30, 1994.

[2] T.E.Anderson, D.E.Culler, D.A.Patterson, et.al., "The Case for Networks of Workstations: NOW", *IEEE Micro*, Feb, 1995.

[3] Y.A.Khalidi, J.M.Bernabeu, V.Matena, K.Shirriff, M.Thadani, "Solaris MC: A Multi-Computer OS" Sun Microsystems Laboratories, Technical Report SMLI TR-95-48, Nov 1995.

[4] T.Wilkinson, T.Stiemerling, P.E.Osmon, A.Saulsbury, P.Kelly, "Angel: A Proposed Multiprocessor Operating System Kernel" European Workshop on Parallel Computing, 1992.