マルチノード・マルチスレッドプログラム向け並列ランタイムシステムの設計

鈴木 悠一郎†‡　岩井田 匡俊†‡　緑川 博子†‡　甲斐 宗徳†‡

The Design of Parallel Runtime System for Multi-node Multi-thread Programs

YUICHIRO SUZUKI, MASATOSHI IWAIDA, HIROKO MIDORIKAWA and MUNENORI KAI†‡

1. はじめに

高性能な並列計算機は，複数の計算ノード群からなる分散メモリ環境であるクラスタ構成が一般的である．クラスタ上で並列処理をするプログラムは，共有メモリ環境である計算ノード内では OpenMP，分散メモリ環境である計算ノード間は MPI で記述するハイブリッドプログラミングがデファクトスタンダードとなっている．OpenMP は，逐次プログラムコードに pragma 指示文を挿入するだけで並列化を可能にするが，MPI は，ノード間で共有するデータは，変数名がノード間で一意にならず，アドレス領域も変わるため，煩雑で開発コスト(コーディング，デバッグ，保守等)が高く，生産性が低いことが問題になっている．このため，現状の並列プログラミングの生産性の難点を解決するために，多くの分散メモリ環境向け並列言語（Chapel[1], X10[2]など）が開発されてきている．

本稿では，既存の逐次言語を使用するユーザに対して，ノード内並列と同様にノード間並列でも共有できるアドレス領域を仮想的に提供し，特にプログラマビリティとポータビリティの点で生産性が高い並列言語を実現するための処理系として，MiMoSa システムを提案する．MiMoSa は，近年開発されている並列言語(Chapel,X10)などと異なり，以下のような設計指針で開発されている．

- ノード間共有データのアドレス領域すべてに，必要があればプログラムからアクセスできるような並列言語向けの処理系とする
- 既存の逐次言語から利用でき，ノード内の OpenMP や Pthreads の並列性記述も，ほぼそのまま生かすことができるようにする
- アルゴリズム上，頻繁に用いる限定的データの高速アクセスを実現するため，共有データの一貫性制御とは別に，ノード間共有データ領域からノード内ローカル変数へ直接コピーを行う高速アクセス API を提供する

2. MiMoSa システム

MiMoSa システムは，C 言語向けライブラリとして提供し，ポインタを利用したアドレス領域の共有を可能とする．ユーザレベル SDSM（Software Distributed Shared Memory）として実装しており，以下のような特徴がある．

- マルチスレッドユーザプログラムに対応
- フルマルチスレッドサポートレベルの MPI を用い，複数システムスレッドによる非同期通信・実行を生かしたページ送受信の並列化

2.1 マルチスレッドユーザプログラムのサポート

一般に，ページベース SDSM は，OS ページを基にしたサイズのページで，ノード間で共有するアドレス領域を管理し，アクセス検知にメモリ保護属性を使用する．このため，マルチスレッドユーザプログラムをそのまま動作させると，遠隔ページを受信する際に，ページの保護属性の変更から遠隔ページの反映までの間で，複数スレッド間で正しいデータを保証することができなかった．筆者らは，遠隔メモリページングシステムである DLM において，ページ受信時の全スレッドの停止をさせる単純な機構を導入し，実用レベルで一定の有効性を示した[3]．MiMoSa でも，同機構を導入して，マルチス

† 成蹊大学理工学研究科理工学専攻
Graduate school of Science and Technology, Seikei University
‡ 独立行政法人科学技術振興機構，CREST
Japan Science and Technology Agency, CREST

レッドユーザプログラムに対応できるようにしている．

2.2 マルチスレッド実装によるページ送受信の並列化

MiMoSa では，Master，Receiver，Merger の 3 システムスレッドがノード(プロセス)間の送受信，要求処理を行い，並列化を高めている．ノード間通信には，ポータビリティの高い MPI を採用している（図 1）．

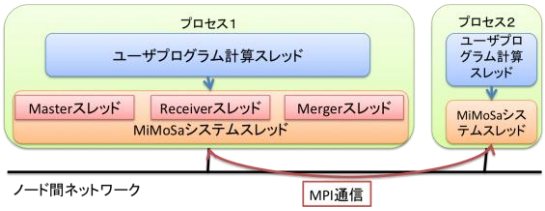

図 1　MiMoSa システム全体図

Master スレッドは，ユーザプログラム実行開始時に生成され，MiMoSa の他のシステムスレッドを生成・管理し，ノード内ユーザプログラムと他ノードからの要求の両方に対応して処理を行い，他ノードへのページ送信も行う．一方，Receiver スレッドは，Master スレッドから生成され，主に他ノードからの要求やページの受信を行う．Master スレッドは，Receiver スレッドが外部から受け取った外部メッセージ(自ノードへのページ要求やデータ割り付け，同期，データ更新情報など)のキューと，自ノードユーザプログラムからの内部メッセージ（データ割り付け，同期，他ノードへのページ要求など）のキューの両方を見て，キューから要求を取り出し，対応した処理を行う．Receiver スレッドは，ページ受信完了時は，コンシステンシを維持するためにユーザプログラムスレッドを一時停止させる処理も行う．図 2 に，ページの送受信の並列化の例を示す．

Merger スレッドは，ノード間で共有するデータの一貫性を同期する際に，Receiver スレッドから動的生成され，変更部分のページの受信とマージ処理を行う．マージ処理をする期間でのみ動的に生成され，必要以上に計算リソースを消費しないようにしている．図 3 に，Merger スレッドを使用した，ページマージ処理の並列化の例を示す．

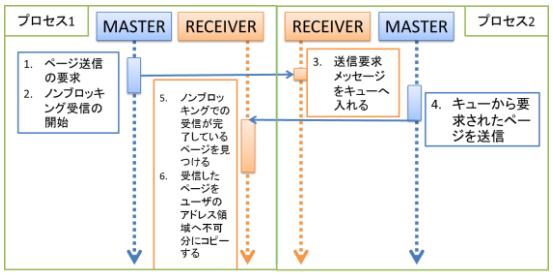

図 2　ページの送受信の並列化

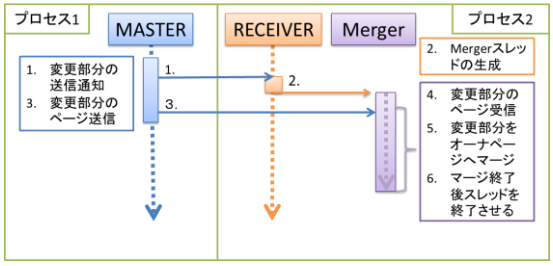

図 3　マージ処理の並列化

3. おわりに

本稿では，非同期実行されるマルチスレッドアプリケーションの動作を可能とし，マルチスレッド MPI で実装した SDSM として，MiMoSa システムを提案し，設計と実装を示した．今後，動作検証と各部の性能評価実験を行い，実装最適化を進める予定である．

また，既存の C プログラムや OpenMP プログラムにユーザが新たなディレクティブ指示文を加えるだけで，コア並列とノード並列機能をそれぞれ追加できるインクリメンタルプログラミングモデルと API を提案している[4]．このモデルのランタイムシステムとして，MiMoSa を利用できるような開発も進める予定である．

参　考　文　献

[1] Chapel：http://chapel.cray.com/

[2] X10：http://x10-lang.org/

[3] 鈴木，鷹見，緑川，“マルチスレッドプログラムのための遠隔メモリ利用による仮想大容量メモリシステムの設計と初期評価”，情処学会 HPC 研究会 Vol.2011-HPC-132, No.13，pp.1-6，November 2011.

[4] 直木，緑川,甲斐，“マルチコアプログラムにノード並列機能を加える API の提案”，第 11 回 FIT 論文集(第一分冊),B-003, pp169 -171 ，September 2012.