様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成　２９　年　　５　月　３０　日現在

機関番号：　８２４０１
研究種目：　若手研究(B)
研究期間：　2014～2016
課題番号：　２６７３００６４

研究課題名（和文）マルチSPMDプログラム実行環境における耐障害アプリケーションの実現

研究課題名（英文）Fault tolerant computing based on a multi-SPMD programming/execution environment

研究代表者

辻　美和子（Tsuji, Miwako）

国立研究開発法人理化学研究所・計算科学研究機構・研究員

研究者番号：８０４６６４６６

交付決定額（研究期間全体）：（直接経費）　　1,100,000円

研究成果の概要（和文）：本研究においては，アプリケーションプログラマの負担を最小限に抑えた耐故障性実現のために，マルチSPMDプログラミング開発実行環境において耐故障性をサポートした．この開発実行環境では，ワークフローにおけるタスクを分散並列／共有メモリモデルとすることで，ワークフローモデルと分散並列／共有メモリモデルを適切に組み合わせて利用し，大規模システムにおいても高いスケーラビリティを実現した．さらに，障害が起こったタスクをハードビートにより検出して自動的に再実行することで，障害発生下でもアプリケーションを完遂可能な仕組みを実装した．

研究成果の概要（英文）：In this research, we have supported fault tolerance features in a multi-SPMD programming/execution environment, where tasks in a workflow are executed in distributed parallel. The programming environment adopts multi-programming methodologies across multi-architectural levels, such as Numa-core groups in a node, nodes in a cluster, a cluster of clusters, to realize scalability. To achieve a fault tolerance and resilience mechanism without any modification of the application’s source code, we have developed middleware to detect errors in remote programs and extended workflow scheduler to realize fault resilience.

研究分野：　高性能計算

キーワード：　耐故障性　プログラミングモデル

１．研究開始当初の背景

計算機の高性能化にともない，構成要素数の増加と複雑化により，システム全体の平均故障間隔（MTBF）は，短くなる傾向にある．そのため，大規模なアプリケーションを長時間にわたって実行する場合には，計算の中間結果を一定間隔で一時ファイルに書き出すなどの，障害を考慮したソースコードの書き換えが必要になる．しかし，そのような追加の変更は，アプリケーション開発者にとってはしばしば大きな負担になる．そこで，本研究では，アプリケーションのソースコードの変更なしに，耐障害性を実現する仕組みを開発した．

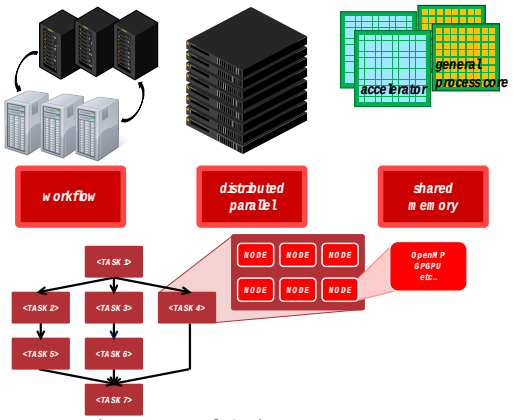
２．研究の目的

アプリケーションプログラマの負担を最小限に抑えた耐故障性実現のために，マルチ SPMD プログラミング開発実行環境において耐故障性をサポートする．マルチ SPMD プログラミング開発実行環境では，ワークフローにおけるタスクを分散並列／共有メモリモデルとすることで，ワークフローモデルと分散並列／共有メモリモデルを適切に組み合わせて利用し，大規模システムにおいても高いスケーラビリティを実現することができる．この環境を拡張し，耐故障性をサポートすることで，大規模システムのユーザの課題である，スケーラビリティと耐障害性という困難さを解決することが本研究の目的である．

３．研究の方法

(1) マルチ SPMD プログラミングモデルの概要

次世代システムは，Numa 構造からなるメニーコアやアクセラレータを持つ多数のノードから構成されると想定される．このようなシステムを効率的に利用するために，われわれはマルチ SPMD プログラミングモデルを提案し，このモデルにもとづいてアプリケーションを開発・実行する環境を実装した．マルチ SPMD プログラミングモデルは，ワークフローおよび分散並列／共有メモリモデルから構成され，ワークフローにおける各タスクが分散並列／共有メモリモデルにより記述され，実行される．われわれの開発・実行環境では，タスクの記述には，分散並列言語である MPI のほか，より平易にデータや処理の並列化を記述することが可能な並列言語である XcalableMP（XMP）をサポートする．ワークフローは，ワークフロー開発実行環境である YML によってサポートされる．YML では，YvetteML と呼ばれる記述言語によりタスクどうしの依存関係を記述することで，ワークフローを定義する．図にマルチ SPMD プログラミングモデルの概要を示す．



ワークフロースケジューラは，タスクの依存関係にしたがって，タスクの実行をミドルウエアに依頼する．本開発・実行環境のために構築されたミドルウエア OmniRPC-MPI は，マルチ SPMD プログラミングモデルにおける拡張マスタワーカ型プログラムをサポートする RPC ミドルウエアであり，タスクおよびノードの資源を管理し，ノードにリモートプログラムを起動し，リモートプログラムに特定のタスクの実行を依頼する．

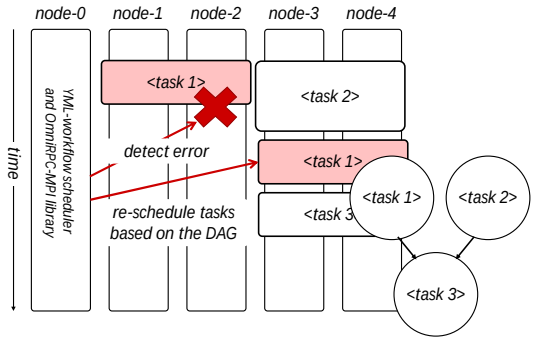
(2) 拡張の概要

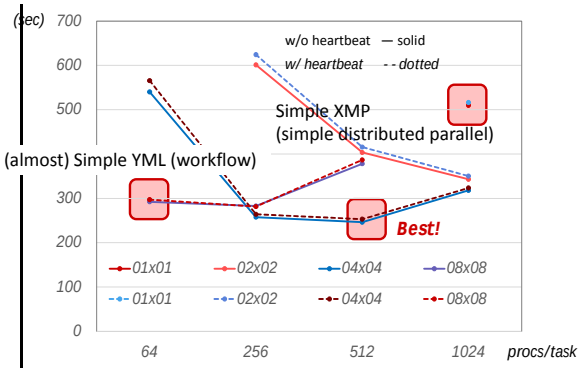
本研究では，マルチ SPMD プログラミング開発実行環境における耐故障性の実現のために，以下を行った．

- 障害検知のためのミドルウエアの拡張
- 失敗したタスクの自動再実行のためのワークフロースケジューラの拡張

障害検知のためのミドルウエアの拡張においては，マスタがワーカの故障を検知する仕組みとして，ワーカからのハートビートの送信と，マスタ側の受信およびリモートプログラム死亡判定などの実装を行った．ワーカプログラムは起動時に，ハートビート送信用のスレッドを起動し，スレッドは一定間隔ごとにソケット通信によってマスタに信号を送信する．マスタは，もともとワーカからのタスク終了信号を受信し，処理するためのスレッドを持つ．このスレッドがワーカからのハートビート信号も受信し，また受信管理する．ワーカからのハートビートが一定時間以上途切れた場合は，その事実をワークフロースケジューラに通知する．

失敗したタスクの自動再実行のためのワークフロースケジューラの拡張においては，ワークフロースケジューラが，ミドルウエアからの失敗の信号を受けて，失敗したタスクの状態を，実行中から実行準備完了に変更し，再びミドルウエアに実行を依頼するように拡張した．概要を下図に示す：





4．研究成果

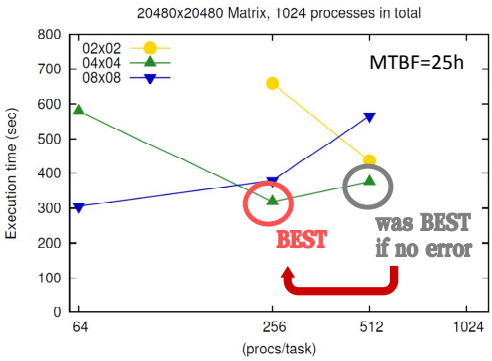
(1) ミドルウエアの拡張と性能評価

マルチ SPMD プログラミング開発実行環境における耐障害性の実現のために，3-(2) で述べたように，ミドルウエア OmniRPC-MPI を拡張した．マスタが，特定のワーカの状態を確認するための API および，ワーカが別のワーカの状態をマスタに問い合わせるための API を提供した．また，京コンピュータおよび FX10 で提供される，安全なリモートプログラム間の通信のために，通信にさきだって相手の生存を確認する API である FJMPI_Mswk_connect/accept 組をミドルウエアに組み込むことを検討した．
OmniRPC-MPI は，本開発実行環境のみならず，並列のワーカプログラムを用いる拡張型のマスタワーカ型プログラミングに利用することが可能である．よって，この拡張したミドルウエアをマルチ SPMD プログラミング開発実行環境に組込む前の予備実験として，フラグメント分子軌道法のアプリケーションである OpenFMO を OmniRPC-MPI で実装し，性能評価を行った．実験の結果，ハートビートを用いた場合のオーバーヘッドは非常に低いことがわかった．しかし，FJMPI_Mswk 関数は，ブロッキング通信を含むためにオーバーヘッドが大きく，本実装では採用を見送った．

(2) マルチ SPMD プログラミング開発実行環境の実装と性能評価

拡張した OmniRPC-MPI をマルチ SPMD プログラミング開発実行環境に組込むとともに，3-(2) で述べたワークフロースケジューラの拡張を行い，耐故障性を考慮したマルチ SPMD プログラミング開発実行環境を実装し，評価した．
96 ノードからなる FX10 クラスタにおいて，マルチ SPMD プログラミングモデルにおけるさまざまな並列性の下の性能を評価するために，タスク毎のプロセス数をさまざまに変化させ，実験を行った．実験の結果，故障検知のためのオーバーヘッドは，次の表に示すように，平均で 2.3 % 程度であった．

また，適当な MTBF を設定してランダムに故障を模してワーカプログラムを停止させる実験を行った．疑似故障発生下の実行時間の増加は最大で 19%だったが，タスクに対するプロセス数割り当てが適切に選択されているときには，疑似故障発生下においてもより少ない時間の増加で，アプリケーションを完遂できた．また，ワークフロー全体として用いるプロセス数が一定のとき，各タスクにわりあてる最適なプロセス数が，故障を想定しない場合と故障発生下で異なることを明らかにした（下図）．



(3) マルチ SPMD プログラミング開発実行環境の大規模システムにおける拡張と性能評価

より大規模なシステムにおける耐故障性マルチ SPMD プログラミング開発実行環境の性能を調査した．大規模システムとして京コンピュータを用いた．
本実装におけるタスクへのデータ入出力は MPI-IO により行われているため，まず予備実験として，京コンピュータにおいて，ワーカから MPI-IO 時と計算時にそれぞれ一定間隔でハートビートをマスタへ送信する単純なマスタワーカ型プログラムを実行し，MPI-IO 性能を評価した．評価の結果，MPI-IO とオーバーラップするハートビートはしばしば間隔が不規則になることがわかった．そこで，ミドルウエアをさらに拡張し，マスタが動作中のワーカを故障と誤判定することを防ぐ

ために，タスクへのデータ入出の前のハートビートに「出力中」の状態を示すフラグを追加し，この誤判定を防ぐ実装を行った.

５．主な発表論文等
（研究代表者、研究分担者及び連携研究者には下線）

〔学会発表〕（計　7　件）

1. 辻美和子,佐藤三久, "大規模システムにおける耐故障マルチ SPMD プログラミング開発実行環境の応用と評価",情報処理学会研究報告,2017-HPC-158,On Line,2017.03.08-10,情報処理学会,2015, 大月ホテル（熱海市・静岡県）

2. Miwako Tsuji, "Multi-SPMD Programming Paradigm for Extreme Computing",The 6th AICS International Symposium,2016.2.22-23,RIKEN/AICS,2016, 理化学研究所計算科学研究機構（神戸市・兵庫県）

3. Miwako Tsuji, Serge Petiton and Mitsuhisa Sato, "Fault tolerance features of a new multi-SPMD programming/execution environment", Proceedings of the First International Workshop on Extreme Scale Programming Models and Middleware SC15,pp.20--27 doi:10.1145/2832241.2832243,2015.11.14-20, ACM,2015,Austin (USA)

4. Miwako Tsuji, "Fault Tolerance features of YML-XMP", "Workshop on Langage and Programming Paradigm for Exascale Applications",2015.03.12-13, 2015, Houston (USA)

5. 辻美和子,佐藤三久, "マルチ SPMD"プログラミング開発実行環境における耐故障性実現に向けたワークフロースケジューリングの検討", 情報処理学会研究報告,2015-HPC-148,On Line,2015.03.02-03,情報処理学会,2015, 花菱ホテル（別府市・大分県）

6. 辻美和子, "マルチ SPMD 環境に向けた XMP/YML の活用", 第 2 回 XcalableMP ワークショップ, 2014.10.24,2014, 秋葉原 UDX（千代田区・東京都）

7. 辻美和子,佐藤三久, "マルチ SPMD 環境における耐故障性実現に向けた OmniRPC-MPI の拡張", 情報処理学会研究報告 ,2014-HPC-146,On Line,2014.10.02-03, 情報処理学会, 2014, 沖縄産業支援センター（那覇市・沖縄県）

６．研究組織
(1)研究代表者　辻 美和子（TSUJI, Miwako）
国立研究開発法人理化学研究所・計算科学研究機構・研究員
研究者番号：80466466

(2)研究分担者
（　　）

研究者番号：

(3)連携研究者
（　　）

研究者番号：

(4)研究協力者
（　　）