様　式　Ｃ－１９、Ｆ－１９、Ｚ－１９　（共通）

# 科学研究費助成事業　　研究成果報告書

平成 ２７ 年　 ６ 月 １１ 日現在

機関番号：８２６２６
研究種目：若手研究(B)
研究期間：2013～2014
課題番号：２５８７１１９９

研究課題名（和文）ポストペタスケール計算機環境に向けた高可用分散協調スケジューリングの研究

研究課題名（英文）A Study of a Highly Available Distributed Scheduling Scheme for Post-petascale Computing Environments

研究代表者

竹房　あつ子（Takefusa, Atsuko）

独立行政法人産業技術総合研究所・情報技術研究部門・主任研究員

研究者番号：７０３４５４１１

交付決定額（研究期間全体）：（直接経費）　　2,800,000円

研究成果の概要（和文）：ポストペタスケール計算環境でアプリケーションの実効性能を維持しつつプログラムの継続実行を支援する高可用分散協調スケジューラの実証を目的とし、高可用分散協調スケジューラのプロトタイプシステムを設計，開発，評価した。
H25年度は、高可用分散協調スケジューラを設計してApache ZooKeeperを用いてJavaで実装し，その高可用性とスケーラビリティを調査した。H26年度は，User Level Fault Mitigation (ULFM) MPIをもちいてC言語で実装しているfalanxミドルウェアに提案する高可用分散協調スケジューラを実装し、提案手法の実現可能性を示した。

研究成果の概要（英文）：Fault resiliency is an important issue for post-petascale computing environments. In order to achieve fault resiliency of application programs running on such computers, we propose, design and develop a prototype system of a highly available distributed scheduler and investigate its performance characteristics.
We first designed a highly available distributed self-scheduler and developed a Java-based prototype system using Apache ZooKeeper, and showed its availability and scalability. Then, we implement the proposed scheduler into the C-based falanx middleware developed using User Level Fault Mitigation (ULFM) MPI. We showed the feasibility of the proposed scheduler from the experiments.

研究分野：並列・分散処理

キーワード：並列分散処理　耐障害性　ポストペタスケール計算　スケジューラ　資源管理

## １．研究開始当初の背景

2018～2020 年にポストペタスケール(エクサスケール)計算機が実現すると米国，日本の計算機科学者らで作成されたロードマップで報告されている。ポストペタスケール計算機は，十万プロセッサ，数千万 CPU コア規模で構成されると予想されるため，従来のデータ並列型プログラムでは問題サイズを維持したまま並列化効率を上げる(強スケーリング)ことが非常に困難になる．よって，そのような環境ではアプリケーションプログラムのタスク並列の各タスクに並列性を内包させた図 1 に示すような階層型タスク並列が計算効率を高めるための有望なプログラミングモデルの 1 つである．
しかしながら，そのような環境では障害発生間隔が 5 分程度になると予測されており，アプリケーションプログラムには障害が発生しても継続して実行できる耐障害性が求められる．タスク並列型アプリケーションでは，障害により失敗した部分について局所的に再計算して実行フローを維持する方法や，部分的に失敗した箇所があっても処理結果に影響を与えないようにアルゴリズムそのものに冗長性を内包させる方法により，耐障害性を持たせるように設計できる．しかしながら，そのような実装を実現するためには，アプリケーションプログラムの計算ロジックとは無関係な部分で煩雑なコーディングが必要となる．

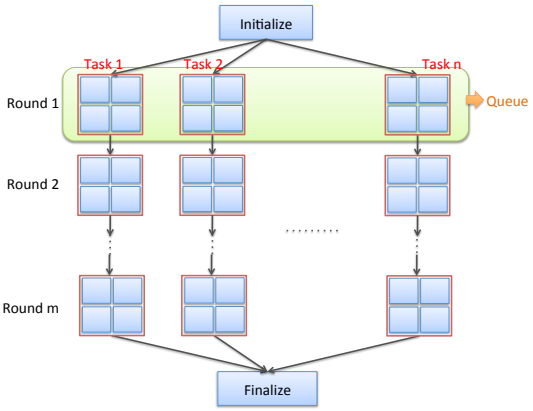


図 1．階層型タスク並列アプリケーションプログラム例

## ２．研究の目的

ポストペタスケール計算機環境でアプリケーションの実効性能を維持しつつプログラムの継続実行を支援する高可用分散協調スケジューラを実証する．
本研究では，階層型タスク並列アプリケーションプログラムの耐障害性を支援する高可用分散協調スケジューラのプロトタイプシステムを開発し，ポストペタスケール計算機環境を模擬した環境で性能特性を調査して設計指針を示す．

## ３．研究の方法

(1)高可用分散協調スケジューラは，スケーラブルであること，スケジューラ自身に耐障害性があることが求められる．よって，スケーラブルかつ高可用性を実現するため，複数プロセスが協調して資源管理を行う分散協調セルフスケジューリング機構を提案，設計した．また，プロトタイプシステムを実装し，そのスケーラビリティと耐障害性を評価した．

(2)産業技術総合研究所では，ポストペタスケール計算機環境において耐障害性を有した階層型タスク並列アプリケーションプログラムの開発を支援する Falanx ミドルウェアを開発している．MPI および C 言語で実装された Falanx ミドルウェアに，本研究で提案するスケジューリング機構を実装し，その有効性を調査した．

## ４．研究成果

(1) 提案する高可用分散協調スケジューラは，複数資源管理プロセスを分散協調させてタスクキューを管理し，タスクキュー内のタスクを各計算ノード上の実行デーモンプロセスが自律的に取得して実行する．また，各計算ノードの死活監視を行い，実行中に故障が発生した場合は選択的に再実行または削除する仕組みを提供するようにした．

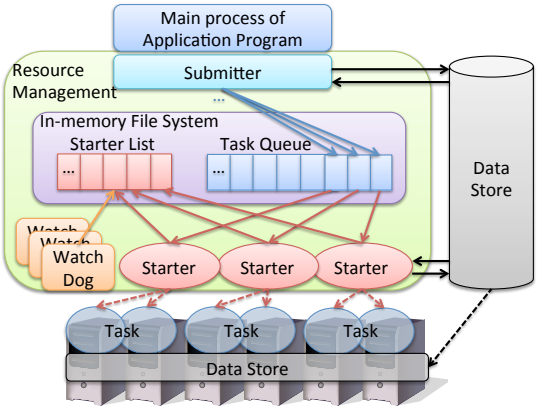


図２．高可用分散協調スケジューラの概要

図２に提案したスケジューラの概要図を示す．Submitter では実行可能なタスクをタスクキューへ投入するとともに、アプリケーションプログラマの指示に従い、失敗したタスクを再実行または破棄する。各 Starter は，ノード群を管理しており，タスクキューに投入されたタスクをその小ノード群上で実行する．In-memory File System は複数ノードのメモリ上で資源管理情報を高速に冗長管

理し，タスクキューやタスクの処理状況，Starter 情報等を保持する．Submitter や Starter は，In-memory File System を介して処理対象のタスクの情報を授受する．また，WatchDog は計算ノードの死活監視を行う．Starter または Starter が管理するノード群に属する計算ノードに障害が発生した場合，そのノード群が担当していたタスクの状況を失敗状態とし，Submitter が再実行または破棄できるようにする．各タスクの引数や実行結果等の情報は，別途分散データストアで管理する．

資源管理プロセスの耐障害性と資源管理情報の永続化を実現するため，Apache ZooKeeper を用いてこれらの機能を Java で実装した．データストアには，Apache Cassandra を用いた．本実装で 100 Starter，10000 タスクのアプリケーションを用いた評価から、スケジューラの耐障害性とスケーラビリティを示した．

(2) Falanx は産業技術総合研究所で開発しているミドルウェアである．データストア機構と資源管理機構からなり，アプリケーションプログラムを単一 MPI ジョブとして実行させるとともに，実行障害発生時には選択的にタスクを再実行／破棄させることができる．しかしながら，Falanx の既存実装では資源管理機構を MPI プロセスの 1 ランクで処理しているため，本研究で提案する高可用分散協調セルフスケジューリング機構を Apache ZooKeeper を用いて Falanx ミドルウェアに実装した．また，分散管理によるオーバヘッドを低減するため，3 通りのタスクキューの実装を行った．

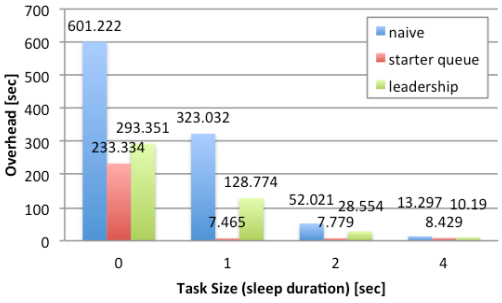


図３．分散協調資源管理によるオーバヘッド

図３に提案手法のオーバヘッドを調査した結果を示す．naïve は単純な実装，starter queue は高速化を図っているが，タスクの優先度制御はできないもの，leadership は高速化を図りつつ，優先度制御も可能な実装を表す．図３から，タスクサイズがある程度大きければ分散管理によるオーバヘッドは無視できること，またタスクキューの実装方法により，オーバヘッドを低減できることが確認できた．また，実アプリケーションである並列フラグメント分子軌道計算プログラム OpenFMO (Fragment Molecular Orbital)を用いた評価実験も行った．実験結果から，提案手法では高可用性を保ちつつ，1 ランクでの資源管理と同等の性能を保つことが示された．

これらの研究成果は，国内研究会で発表するとともに，ACM HPDC'14 および IEEE/ACM SC14 のポスターセッション、および査読付き国際会議 ACM IMCOM 2015 において発表した。

## ５．主な発表論文等

〔雑誌論文〕（計３件）


① Atsuko Takefusa, Hidemoto Nakada, Tsutomu Ikegami, Yoshio Tanaka. A Highly Available Distributed Self-Scheduler for Exascale Computing. Proc. ACM IMCOM (ICUIMC) 2015, 7-3, pp. 1-8. 2015 年. 査読有. http://dl.acm.org/citation.cfm?id=2701214

② 竹房あつ子，中田秀基，池上努，戸澤貴之，田中良夫．耐障害性ミドルウェア falanx への高可用分散協調スケジューラの実装．情報処理学会研究報告 2014-HPC-145(8), pp. 1-8. 2014 年. 査読無. http://ci.nii.ac.jp/naid/110009808103

③ 竹房あつ子，中田秀基，池上努，田中良夫．パーシステントストレージを利用した高可用分散協調スケジューラの実装．情報処理学会研究報告 2013-HPC-140(20), pp. 1-6. 2013 年. 査読無. http://ci.nii.ac.jp/naid/110009588140


〔学会発表〕（計５件）


① Atsuko Takefusa. A Highly Available Distributed Self-Scheduler for Exascale Computing. ACM IMCOM (ICUIMC) 2015, 2015 年 1 月 9 日，バリ（インドネシア）.

② Atsuko Takefusa. Scalable and Highly Available Fault Resilient Middleware for Exascale Computing. IEEE/ACM SC14 (Poster Presentation), 2014 年 11 月 18 日．ニューオーリンズ（米国）.

③ 竹房あつ子．耐障害性ミドルウェア falanx への高可用分散協調スケジューラの実装．情報処理学会 HPC 研究会，2014 年 7 月 28 日，朱鷺メッセ 新潟コンベンションセンター（新潟県・新潟市）.

④ Atsuko Takefusa. A Study of a Highly Available Distributed Self-Scheduler for Exascale Computing. ACM HPDC'14 (Poster Presentation), 2014 年 6 月 25

日．バンクーバー（カナダ）．

⑤ 竹房あつ子．パーシステントストレージを利用した高可用分散協調スケジューラの実装．情報処理学会 HPC 研究会，2013 年 8 月 1 日，北九州国際会議場（福岡県・北九州市）

６．研究組織

(1)研究代表者

竹房　あつ子（TAKEFUSA ATSUKO）

独立行政法人産業技術総合研究所・情報技術研究部門・主任研究員

研究者番号：70345411