Vol.2021-HPC-178 No.12
2021/3/15

Urgent Computing に向けたアプリケーション

中島研吾[†1 †2]　住元真司[†3]　塙 敏博[†1]

地震，津波，水害等の自然災害発生時にはスーパーコンピュータ（スパコン）によるシミュレーションが緊急時対応に大きな役割を果たすものと期待される。このような緊急時の計算，すなわち Urgent Computing では，スパコンのリソースの適切な配分が重要となる。本研究は，スパコンを Urgent Computing に利用するために，稼働するアプリケーションが備えるべき機能について検討するものである。

1. はじめに

1.1 Urgent Computing とは？

Urgent Computing とは，地震，津波，台風，森林火災等大規模な自然災害等の緊急事態が生じた場合に，スーパーコンピュータのリソースを集中的に投入して，迅速な被害想定と減災のための意志決定を実施することを目的としたものである。近年 Urgent Computing は世界的に注目されており，UrgentHPC Initiative〔1〕は SC18 で BoF ミーティング，SC19，SC20 ではワークショップを開催している。2019 年に発足した VESTEC（Visual Exploration and Sampling Toolkit for Extreme Computing）〔2〕は Urgent Computing のためのインフラ構築に関連する EU のプロジェクトである。

自然災害の多い我が国では，Urgent Computing に向けた取り組みは，各分野における研究と並行して継続して実施されており，サイバー空間（仮想空間）とフィジカル空間（現実空間）を高度に融合させたシステムにより、経済発展と社会的課題の解決を両立する、安全・安心な人間中心の社会，すなわち Society 5.0〔3〕の実現に資するものと期待されている。

東北大学サイバーサイエンスセンター〔4〕ではシステムレベルを常時実施することによるチェックポインティングにより，大規模な地震，津波等に対応した Urgent Computing に対応している。また，ゲリラ豪雨予測（理化学研究所等）〔5〕，リアルタイムデータ同化融合地震シミュレーション（東京大学地震研究所，東京大学情報基盤センター）〔6,7〕（図 1）など，シミュレーションとリアルタイム観測データ処理・データ同化を融合させた取り組みも行われている。

Urgent Computing の実現には，通常運転モードから，迅速で安定した緊急時運転モードへの切り替え，また，観測データなど，外部からの大規模データを大量に取得し，迅速に処理することが不可欠である〔8〕。昨今はワークロードの多様化により，スーパーコンピュータに要求される能力も多様化し，計算科学シミュレーションのみならず，データ処理・解析，機械学習，及びそれらの融合に対応したシステムが求められるようになっており，そのようなシステムは Urgent Computing にも適している。東京大学情報基盤センターで 2021 年 5 月から稼働開始する「『計算・データ・学習』融合スーパーコンピュータシステム（Wisteria/BDEC-01）」〔9,10〕においても，Urgent Computing は重要な利用分野であり，リアルタイムデータ取得・解析と計算科学シミュレーションの融合への利用が期待されている。

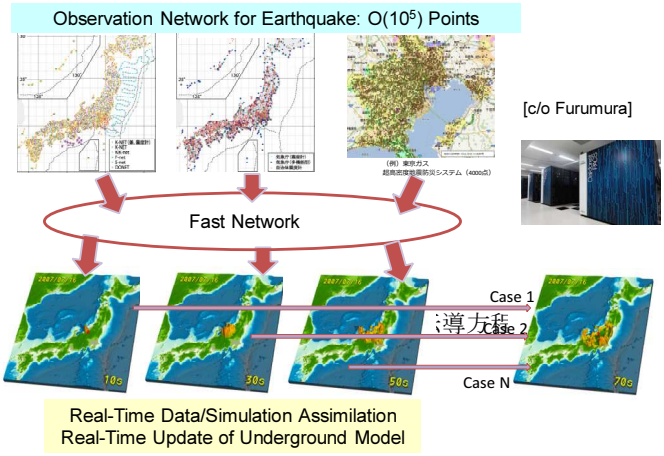

図 1　リアルタイムデータ同化融合地震シミュレーションの概要〔6〕

1.2 COVID-19 時代の Urgent Computing

さて，現在，人類と地球は新型コロナウイルス感染症（COVID-19）という未曾有の危機に直面している。問題解決に向けては「防疫」，「治療」，「創薬」など広範囲にわたり様々な手法による研究開発が急務であり，スーパーコンピュータの有する高速な計算能力，データ処理能力の貢献が期待されている。このような状況の下，我が国では，HPCI（革新的ハイパフォーマンス・コンピューティング・インフラ）において，関係機関の協力のもと，関連する研究が必要とする計算資源を提供する臨時の課題募集「新型コロナウイルス感染症対応 HPCI 臨時公募課題」がおこなわれている〔11〕。アメリカにおいても NSF（National Science Foundation）傘下の XSEDE（Extreme Science and Engineering Discovery Environment）によって，COVID-19 HPC Consortium〔12〕が形成されており，このコンソーシアムには理研，KISTI（韓国）等アメリカ以外の機関も参加している。表 1 は「新型コロナウイルス感染症対応 HPCI 臨時公募課題」において JCAHPC を含む東大情報基盤センター

†1 東京大学情報基盤センター
Information Technology Center, The University of Tokyo
†2 理化学研究所計算科学研究センター
RIKEN, Center for Computational Science (R-CCS)
†3 富士通
Fujitsu

のスパコンを使用した課題をまとめたものである〔13〕。

表 1　新型コロナウイルス感染症対応 HPCI 臨時公募課題（東大情報基盤センター関連分）〔13〕

課題名	代表者（所属）	利用システム
新型コロナウイルスの主要プロテアーゼに関するフラグメント分子軌道計算	望月 祐志（立教大学）	Oakforest-PACS（JCAHPC）〔14〕
COVID-19 治療の候補薬: chloroquine, hydroxychloroquine, azithromycin の催不整脈リスクの評価ならびにその低減策に関する研究	久田 俊明（株式会社 UT-Heart 研究所／東大）	
新型コロナウイルス表面のタンパク質動的構造予測	杉田 有治（理化学研究所）	
計算機解析による SARS-CoV-2 増殖阻害化合物の探索	星野 忠次（千葉大学）	Oakbridge-CX〔15〕
室内環境におけるウイルス飛沫感染の予測とその対策：富岳大規模解析に向けたケーススタディ	坪倉 誠（神戸大学）	
Spreading of polydisperse droplets in a turbulent puff of saturated exhaled air	Marco Edoardo Rosti（OIST）	

COVID-19 によって，我々の生活様式も大きく変化している。「防疫」，「治療」，「創薬」は重要かつ緊急な課題である一方，短期的に解決できない問題も多く，日常的な経済活動を継続しつつ問題の解決に当たることの難しさを身を以て知ることになった。

東大情報基盤センターでは，表 1 に示すような課題のジョブに対しては，ジョブの優先度を高めることで対応してきた。スーパーコンピュータシステムによっては，実行優先度と計算機資源消費率の異なる複数のジョブクラスを提供している場合もあるが，このような制度を適用すると，計算機リソースを予め優先度の高低によって分割しておく必要があり，計算機資源の効率的な利用が困難となる。

突発的な自然災害に迅速に対応することが中心であった従来の Urgent Computing では，通常のジョブは全て一旦停止して，災害対応のワークロードに資源を集中して配分することが主要な課題であった。COVID-19 時代の Urgent Computing では，対応は長期間にわたる一方，創薬にむけた取り組み，変異種への対応など迅速な対応が求められるフェーズもあり，より柔軟な Urgent Computing の枠組みが求められている。

1.3 本研究の概要

本研究は，このような背景のもと，より柔軟な Urgent Computing を実現することを目指したものである。自然災害等突発事象等を想定した従来型の Urgent Computing では，基本的にシステム側で全ての対応が可能であった。本研究で目指す柔軟な Urgent Computing においては，アプリケーション側でも一定の準備が必要となる。本稿では，COVID-19 時代の柔軟な Urgent Computing におけるアプリケーションが備えるべき機能について検討した。本研究では，MPI を使用した並列アプリケーションを対象として，ULFM-MPI（User Level Failure Mitigation）〔16〕によって，ジョブの実行途中で利用計算機資源量を変化させる場合を想定し，アプリケーションを試作し，Oakbridge-CX〔15〕上での評価を実施した。

2. ターゲットアプリケーション：pHEAT-3D

本研究の対象は，図 2 に示すような単位立方体から構成される直方体（NX，NY，NZ は各軸方向の総節点数，要素数は各々NX-1，NY-1，NZ-1 となる）において式 (1) の非定常熱伝導方程式を有限要素法によって解くアプリケーション（pHEAT-3D）である：

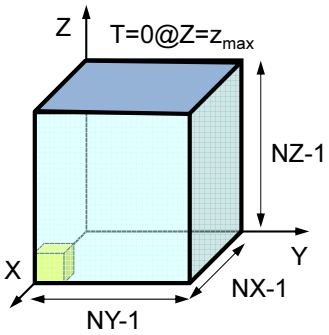

図 2　本研究の計算対象

$$\rho c \frac{\partial T}{\partial t} = \nabla \cdot (\lambda \nabla T) + q \qquad (1)$$

ここで，T：温度，ρ：密度，c：比熱，λ：熱伝導率，q：単位体積時間当たり発熱量，である。Trilinear 三次元六面体要素を使用，時間方向には後退 Euler スキームを適用し，連立一次方程式解法は点ヤコビ前処理による並列共役勾配法（Conjugate Gradient，CG），並列データ構造は〔17〕に示す GeoFEM 型データ構造を使用した。pHEAT-3D は Fortran で記述され，MPI・OpenMP ハイブリッド並列化に対応している。本研究では，式 (1) において，密度，比熱，熱伝導率，発熱量は全て 1 に固定している。初期状態では全節点の温度は 0, Z=Z_{max} 面での境界条件は T=0 である（図 2 参照）。図 3 に示す 32^3 節点（＝32,768），29,791 要素の計算モデルを使用して，T=1,000.まで計算した結果を図 4 に示す。

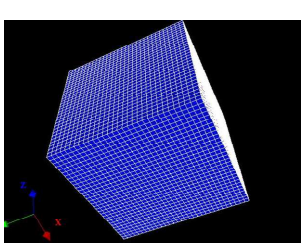
図 3　32,768 節点, 29,791 要素の計算モデル

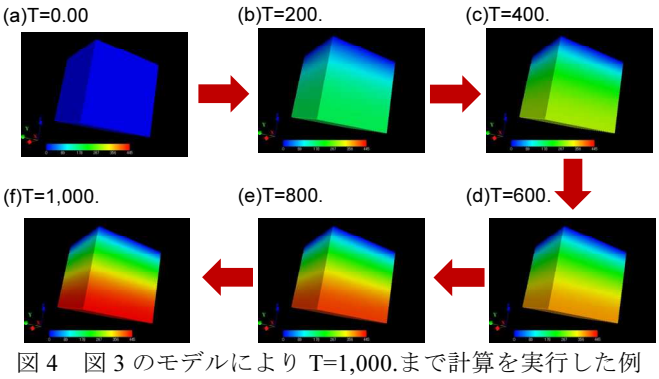

図 4　図 3 のモデルにより T=1,000.まで計算を実行した例

pHEAT-3D の構成と処理の流れは図 5，表 2 に示すようになっている。

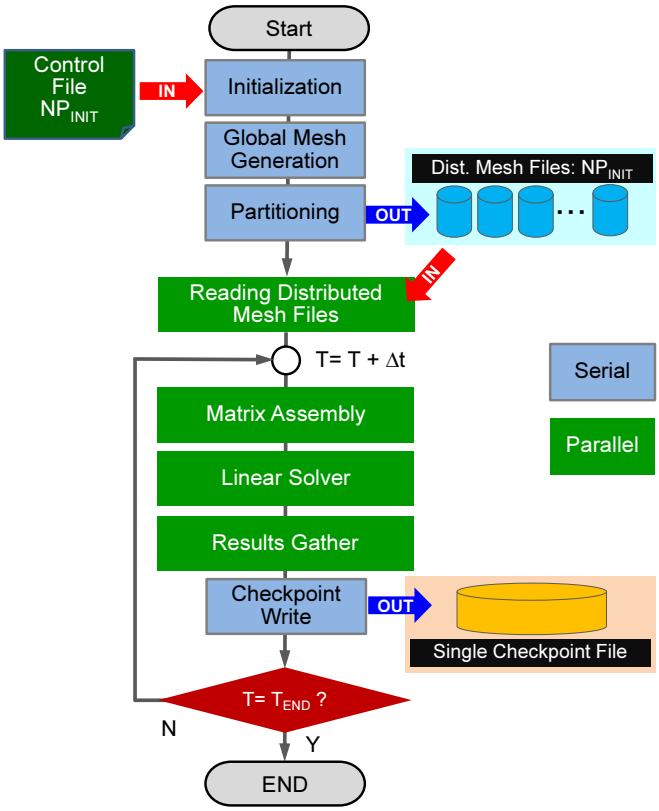

図 5 pHEAT-3D の処理の流れ

表 2　pHEAT-3D の実施内容，ファイル名・変数名

名称	実施内容
Initialization	制御情報ファイル読み込み
Global Mesh Generation	初期全体メッシュ生成
Partitioning	領域分割，分散メッシュファイル生成
Reading Distributed Meshes	分散メッシュファイル読み込み
Matrix Assembly	行列生成
Linear Solver	前処理付き CG 法による求解
Results: Gather	結果情報 0 番プロセス集約
Checkpoint: Write	チェックポイントファイル書き出し

ファイル名・変数名	説明
Control File	制御情報ファイル
Distributed Mesh Files	分散メッシュファイル
Single Checkpoint File	チェックポイントファイル（単独ファイル）
NP_{INIT}	初期領域分割数（MPI プロセス数）
T	計算実施時間
ΔT	各ステップ時間刻み
T_{END}	計算終了時間

pHEAT-3D は「Initialization」で，制御情報ファイル（Control File）から，全体メッシュ規模（図 2 の NX，NY，NZ），領域分割数（NP_{INIT}），計算終了時間（T_{END}），各タイムステップあたりの時間刻み（ΔT，固定），チェックポイントファイル書き出し間隔等を読み込み，「Global Mesh Generator」で初期全体メッシュを生成する。「Partitioning」では，制御ファイルで指定された領域分割数（NP_{INIT}，実行時の初期 MPI プロセス数と等しい）に従って METIS〔18〕による，初期全体メッシュの領域分割を実施し，並列計算用の分散メッシュファイル（Distributed Mesh Files）を生成する。METIS は階層型グラフ処理による領域分割ツールで，本研究では，各領域間の負荷バランスを最適にする pmetis を使用している。ここまでは 1 プロセスによる serial 処理として実施される。「Reading Distributed Mesh Files」で生成された分散メッシュファイルを読み込み，計算を開始する。本研究における計算では，各タイムステップ当たりの時間刻みは固定，物性値も固定のため，連立一次方程式の係数行列は不変であるが，各タイムステップにおいて「Matrix Assembly」において係数行列を計算し，「Linear Solver」において前処理付き共役勾配法によって連立一次方程式を解いている。「Matrix Assembly」，「Linear Solver」の部分が計算時間の大半を占めるが，この部分は MPI・OpenMP によって並列化されている。チェックポイントファイルは制御情報ファイル（Control File）で指定した書き出し間隔に従って，各節点の温度が書き出される。この際，各 MPI プロセスで計算された結果は，MPI_Gatherv を使用して「Results Gather」によって 0 番プロセスに集約され，single ファイルとして出力される（図 5，表 2）。

メッシュ数が非常に大きくなると，そもそも初期全体メッシュを生成することが困難となるが，チェックポイントファイルの生成，出力がボトルネックとなる可能性がある。

3. Urgent Computing の実現へ向けた検討

3.1 ULFM-MPI による耐故障研究における先行研究事例

1.3 で述べたように，本研究は，COVID-19 時代の，より柔軟な Urgent Computing を実現することを目指すものである。2.で紹介した pHEAT-3D のような MPI を使用した並列アプリケーションを対象として，ULFM-MPI〔16〕によって，ジョブの実行途中で利用計算機資源量を変化させる場合を想定している。

User Level Failure Mitigation（ULFM）〔16〕は，MPI の拡張規格，または実装であり，利用者が MPI におけるプロセス故障を管理できるようになっている。現在 ULFM によって提供される関数群や定数群は MPI 標準規格には含まれていないが，既に実アプリケーションも含めた応用事例が多数ある（例えば〔19〕）。Shahzad 等の開発した CRAFT（Checkpoint-Restart and Automatic Fault Tolerance）は，ULFM-MPI を利用したチェックポイントリスタートに基づく，並列科学技術計算向けの耐故障性を有するライブラリであり，様々な利用事例がある〔20〕。Fukasawa 等による pFEM-CRAFT〔21〕は，MPI を使用した並列有限要素法による非定常熱伝導アプリケーションに対して，CRAFT

を使用して，Spare Node を使用しない耐故障フレームワークを提案，実装，動作を検証したものである。

3.2 本研究における手法

本研究で目指す COVID-19 時代の柔軟な Urgent Computing では，ジョブ実行途中で利用計算機資源量を変化させる場合を想定する。より具体的には，ある一定のノード数で開始したジョブが，よりプライオリティの高いジョブに，利用している計算ノードの一部を供出するような場合である。図 6 はその一例で，Job-A，Job-B，Job-C が実行中に，より優先度，緊急性の高いジョブ Job-X が投入されたとする。本研究では，資源のマイグレーション・再配分を実施し，Job-A，Job-B，Job-C の一部のノードを Job-X へ移動することによって，先行 3 ジョブを停止することなく，Job-X を実行，終了後は元のノード数を回復している。

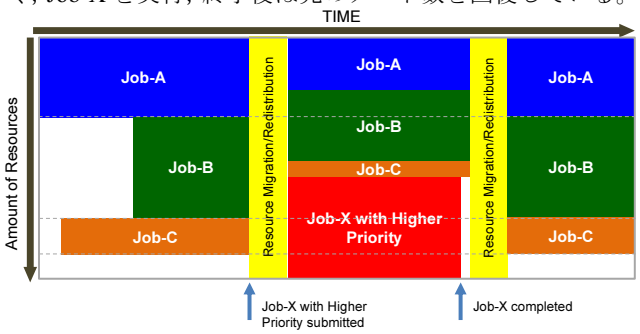

図 6　より優先度・緊急性の高いジョブに計算機資源を動的に割り当てる事例

自然災害等突発事象等を想定した従来型の Urgent Computing では，基本的にシステム側で全ての対応が可能であったが，図 6 に示すような事例では，システムとともに，個々のアプリケーションにおいても対応，準備が必要である。耐故障アプリケーションに関する研究，特に Spare Node を持たず，故障発生時にはノード数を減らして実行を継続するケースは，図 6 の状況と類似しており，本研究への応用が可能である。

3.3 Dynamic pHEAT-3D

本研究では，Fukasawa 等の pFEM-CRAFT〔21〕のアイディアに基づき pHEAT-3D を改良し，動的にプロセス数を変化できる Dynamic pHEAT-3D を整備した。pFEM-CRAFT の原型コードは pHEAT-3D と同じものであり，本稿の著者の一人は，pFEM-CRAFT の研究にも参加している。

Dynamic pHEAT-3D の構成と処理の流れを図 7, 表 3 に示す。図 5, 表 2 に示す pHEAT-3D とオーバーラップがあり，図 7 の（2）に示す部分は基本的に pHEAT-3D と同じである。制御情報ファイル中，T_{CHANGE} で指定される時間になると，図 7 の（1）の「Y」側になり，MPI プロセス数を NP_{INIT} から NP_{NEW} へ減じて，データを再構成して，並列実行を継続する。その際，「Checkpoint : Read」では，チェックポイントファイルを最初から読み込み，最後に書き出さ

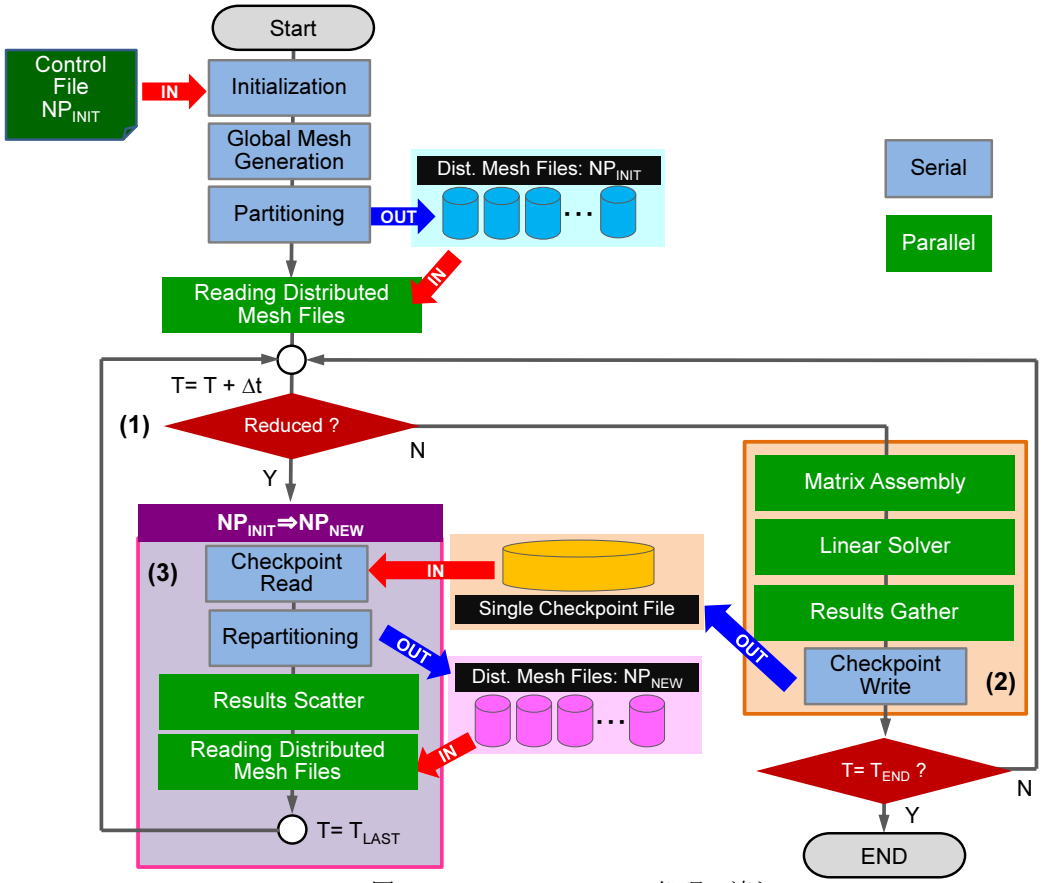

図 7 Dynamic pHEAT-3D の処理の流れ

れている時間（T_{LAST}）に戻って計算を再開する。

計算の再開時には領域分割数を NP_{NEW} に変更して，新たな分散メッシュファイルを生成し，「Results : Scatter」によって，新しい領域分割に応じて，チェックポイントファイルの内容を各 MPI プロセスの初期状態として送信する，という手順になっている。

表 3　Dynamic pHEAT-3D の実施内容，ファイル名・変数名

名称	実施内容
Initialization	制御情報ファイル読み込み
Global Mesh Generation	初期全体メッシュ生成
Partitioning	領域分割，分散メッシュファイル生成
Repartitioning	領域再分割，分散メッシュファイル再生成
Reading Distributed Meshes	分散メッシュファイル読み込み
Matrix Assembly	行列生成
Linear Solver	前処理付き CG 法による求解
Results: Gather	結果情報 0 番プロセス集約
Results: Scatter	結果情報各プロセスへ通信
Checkpoint: Write	チェックポイントファイル書き出し

ファイル名・変数名	説明
Control File	制御情報ファイル
Distributed Mesh Files	分散メッシュファイル
Single Checkpoint File	チェックポイントファイル（単独ファイル）
NP_{INIT}	初期領域分割数（MPI プロセス数）
T_{CHANGE}	MPI プロセス数を変化させるタイミング
NP_{NEW}	再分割後の領域分割数（MPI プロセス数）
T	計算実施時間
ΔT	各ステップ時間刻み
T_{END}	計算終了時間
T_{LAST}	チェックポイントファイルに書き出した最後の計算実施時間

4. 計算例

4.1 Oakbridge-CX

3.で紹介した Dynamic pHEAT-3D を東京大学情報基盤センターの Oakbridge-CX（OBCX，大規模超並列スーパーコンピュータ）〔15〕上で評価した。OBCX は全 1,368 ノードから構成される富士通製のシステムで，各ノードは 28 コアを有する Intel Xeon Platinum 8280（Cascade Lake，第 2 世代 Xeon スケーラブルプロセッサ（SP））を 2 ソケット搭載している。総ピーク性能は 6.61 PFLOPS である。OBCX は全 1,368 ノードのうち 128 ノードに高速の SSD が搭載されており，計算科学の他，データ科学分野でも広く使用されている。更に SSD 搭載 128 ノードのうち 16 ノードは外部ネットワークに直接接続されており（外部接続ノード），観測データ等のリアルタイム取得が可能である。

日本列島には約 2,000 の高感度地震観測点と約 120 の広帯域地震観測点が設置され，全国の 9 国立大学，気象庁，防災科学技術研究所，海洋研究開発機構，産業技術総合研究所などにより運用されている。JDXnet (Japan Data eXchange network)〔22〕はこれらの地震観測点のデータの全国規模のリアルタイム流通ネットワークである。SINET と JGN-X の 2 つの広域 L2 網をバックボーンとして大学など関係十数機関を結び，各機関がそれぞれの観測点から収集したリアルタイムデータを，広域 L2 網上で相互にブロードキャストすることにより，データ交換データ流通が行われている。現在，OBCX の外部接続ノードを使用して，JDXnet から地震観測データをリアルタイムで取得可能となっている〔7〕。

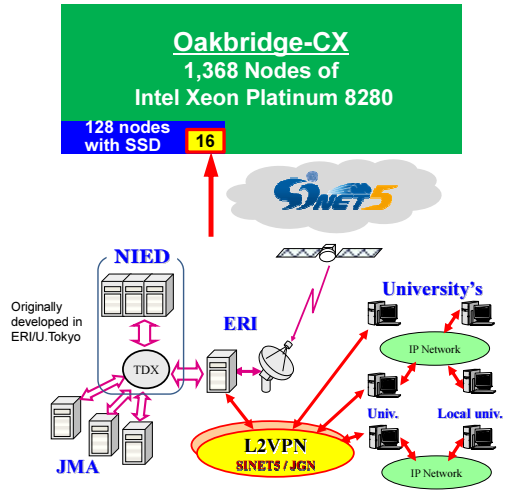

図 8　Oakbridge-CX 外部接続ノードによる JDXnet 観測データの取得〔7〕

4.2 計算結果

本研究では，図 2 において NX=NY=NZ=128，すなわち 2,097,152 節点，2,048,383 要素の問題に対して，OBCX の 8 ノード（SSD 非搭載）を使用してプログラムの動作確認を実施した。各ノードは 56 コアを有しているが，1 プロセス当たり 6 スレッド（6 コア使用）のプロセスを各ノード当たり 8 個立ち上げ（56 コアのうち 48 コア使用），全 8 ノード，64 プロセスを使って計算を実施する。Δt=1.00 として，全 10,000 タイムステップ（T_{END}=10,000）とし，1,000 タイムステップ毎にチェックポイントファイルを書き出した。

本研究においては T=2,500 または T=7,500 となった時点で MPI プロセス数を 64 から，60，32，24 または 16 に変更するケースを想定し，プロセス数変更をしない場合と合わせて，合計 9 ケースの計算を実施した（表 4，図 9）。図 10 は各ケースにおける I/O も含む総計算時間（秒）である。初期 Partitioning（■）には各ケース 36 秒程度を要しており，64_0000 以外の各ケースでは，Repartitioning（■）に 30 秒程度を要している。Partitioning，Repartitioning は並列化されていないこともあり，分割数変更後の MPI プロセス数が少ないほど計算時間が短くなっている。

表 4　計算ケース

ケース名	プロセス数の変更タイミング	変更後（減少後）のプロセス数
16_2500	T=2,500	16
16_7500	T=7,500	
24_2500	T=2,500	24
24_7500	T=7,500	
32_2500	T=2,500	32
32_7500	T=7,500	
60_2500	T=2,500	60
60_7500	T=7,500	
64_0000	変更無し	64

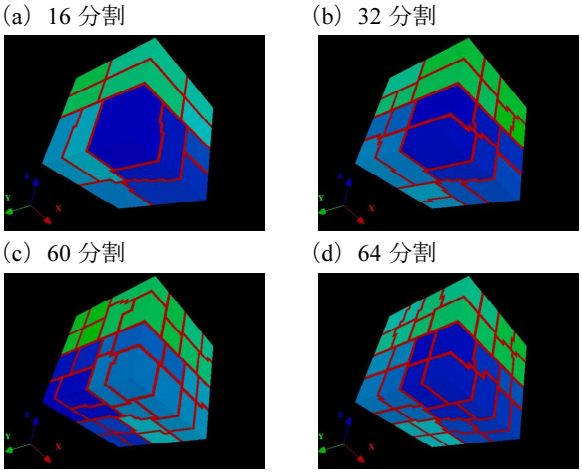

図 9　pmetis による領域分割の事例（32^3=32,768 節点, 29,791 要素の例, 赤い部分は領域間のオーバーラップ要素を示す）

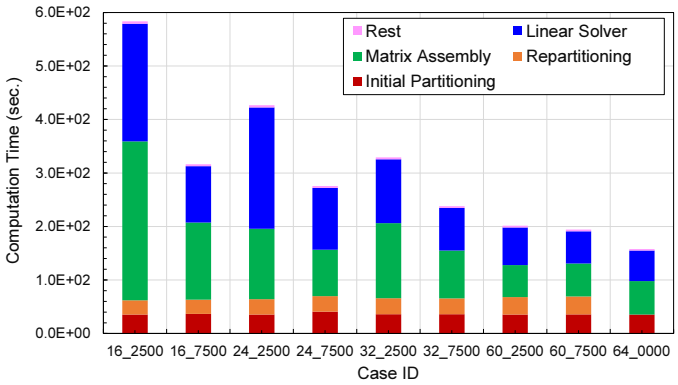

図 9　各ケースの計算時間・内訳

図 10 の Rest（■）はチェックポイントファイルの I/O 時間を含んでいるが，ほぼ無視できる程度に小さいことがわかる。全般的に NP_{NEW} の値が小さく，プロセス数変更（減少）のタイミングが早い（T=2,500）の場合ほど，計算時間が長くなっている。計算結果，各タイムステップにおける共役勾配法の反復回数は各ケースで完全に一致していることを確認した。

図 11 は，図 10 のうち各タイムステップにおける Linear Solver（■），Matrix Assembly（■）の合計時間について，64-MPI プロセスの場合を基準として，使用プロセス数によって並列化効率を示したものである。64 プロセスの場合を基準とすると，理想的には 32 プロセスでは計算時間は 2 倍になるはずであるが，図 10 で 80%となっているのは，実際は計算時間が 64 プロセスの場合の 2÷0.8＝2.50 倍になっていることを示している。

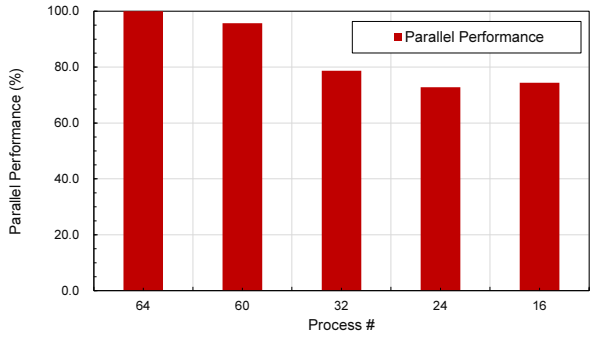

図 11　使用プロセスによる並列化効率

5. まとめ

本研究は，ジョブの実行を止めることなく，より柔軟に Urgent Computing を実施することを目的として，そのような場合に求められるアプリケーションの備えるべき機能を検討するために実施されたものである。ULFM-MPI に基づく耐故障アプリケーションに関連した手法に基づき，有限用要素法による非定常熱電動シミュレーションについて，計算途中で MPI による並列ジョブのプロセス数を動的に変更するためのプロトタイプ Dynamic pHEAT-3D を作成し，Oakbridge-CX システム上で動作確認を実施した。図 7，表 3 に示すような機能を付加することによって，有限要素法，差分法，有限体積法等については Urgent Computing 向けの並列アプリケーションを少ない追加作業で開発できる見通しがついた。

現在は，ULFM-MPI を実際に使用して，プロセス数の増減に対応したアプリケーションの作成を進めている〔23〕。

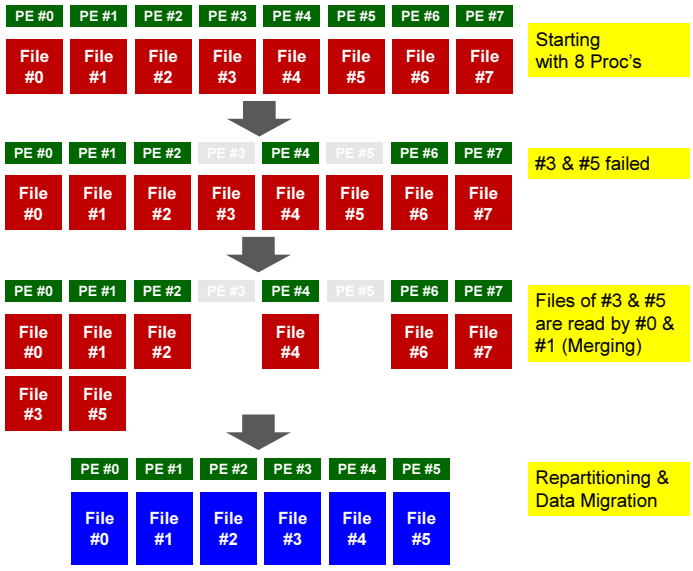

図 12　pFEM-CRAFT におけるデータの Merging・Repartitioning・Migration〔21〕

本稿で紹介したアプリケーションは，簡単のため，チェックポイントファイルの読み書きや初期メッシュ生成，partitioning，repartitioning はシリアル処理で実施している。現状では問題規模が小さいため，問題はないが，$O(10^7)$以上の規模の節点，要素を扱う場合には，図 7 に示した手法では対応が難しくなる可能性がある。実際，pFEM-CRAFT〔21〕では，著者による局所細分化・動的負荷分散に関する先行研究〔24〕で開発したツールを使用し，METIS の分散並列版である ParMETIS を適用し，図 12 に示すような，Mering⇒Repartitioning⇒Migration を実現している。ただ，現状では，特に図 12 の Merging・Migration 部分の効率が非常に悪く，計算に時間を要していることから，今回は図 7 に示すような一部シリアル処理を含む実装を採用している。今後は図 13 に示すような完全分散並列処理を目指す。

本稿では，COVID-19 時代のより柔軟な Urgent Computing のためにアプリケーションが備えるべき機能について検討した。今後は関係各位との協力のもと，スーパーコンピュータの実運用への適用を継続して検討していく予定である。

謝辞

本研究の一部は科学研究費補助金（19H05662，代表：中島研吾）の助成を受けたものである。

参考文献

1) HPC for Urgent Decision Making (UrgentHPC): https://www.urgenthpc.com/
2) VESTEC Project (Visual Exploration and Sampling Toolkit for Extreme Computing): https://vestec-project.eu/
3) Society 5.0（科学技術政策，内閣府）：https://www8.cao.go.jp/cstp/society5_0/
4) 東北大学サイバーサイエンスセンター：https://www.cc.tohoku.ac.jp/
5) 30 秒ごとに更新するゲリラ豪雨予報－首都圏でのリアルタイム実証実験を開始－（理化学研究所）https://www.riken.jp/pr/news/2020/20200821_1/index.html
6) Oba, A., Furumura, T., Maeda, T., Data‐assimilation‐based early forecasting of long‐period ground motions for large earthquakes along the Nankai Trough, J. Geophys. Res., https://doi.org/10.1029/2019JB019047, 2020
7) Nakajima, K., Iwashita, T., Yashiro, H., Shimokawabe, T., Matsuba, H., Nagao, H., Ogita, T., Katagiri, T., h3-Open-BDEC: Innovative Software Platform for Scientific Computing in the Exascale Era, ISC High Performance 2020, (Project Poster), June 2020
8) Gibb, G., Nash, R., Brown, N., Prodan, B., The Technologies Required for Fusing HPC and Real-Time Data to Support Urgent Computing, IEEE/ACM Proceedings of 2019 UrgenHPC Workshop in conjunction with SC19, 24-34, 2019
9) 東京大学情報基盤センターが Society 5.0 実現へ向けた「計算・データ・学習」融合スーパーコンピュータシステムの導入を決定：https://www.cc.u-tokyo.ac.jp/public/pr/pr-wisteria.ph

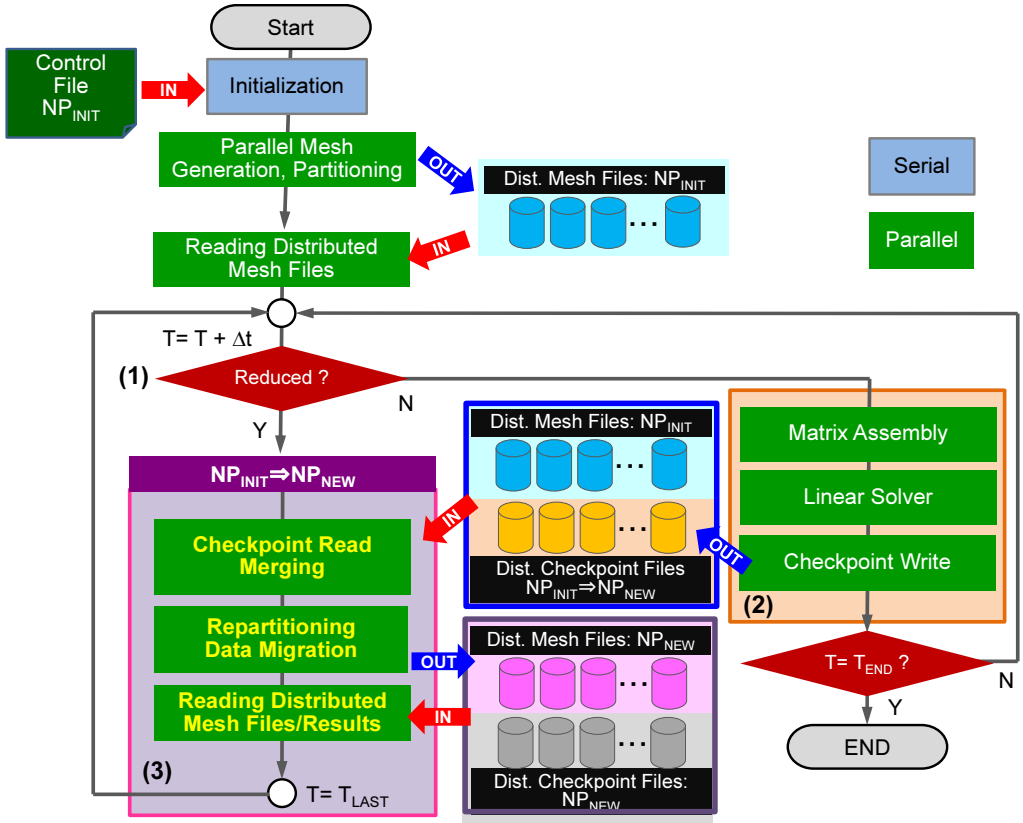

図 13 pFEM-CRAFT と同様に完全に分散並列化を適用した Dynamic pHEAT-3D の処理の流れ

10) 中島研吾，塙敏博，下川辺隆史，坂本龍一，有間英志，星野哲也，伊田明弘，三木洋平，河合直聡，芝隼人，Society 5.0 を実現する BDEC システム，大学 ICT 推進協議会 2020 年度年大会（AXIES 2020）（オンライン，2020 年 12 月）
11) 新型コロナウイルス感染症対応 HPCI 臨時公募課題：https://www.hpci-office.jp/pages/adoptionlist2020_25
12) COVID-19 HPC Consortium (XSEDE): https://www.xsede.org/covid19-hpc-consortium
13) 第 9 回 JCAHPC セミナー（第 4 回 OFP 利活用報告会）「人類と地球を護るスーパーコンピューティング」（2020 年 10 月 15 日）https://www.cc.u-tokyo.ac.jp/events/jcahpc/09.php
14) Oakforest-PACS スーパーコンピュータシステム：https://www.cc.u-tokyo.ac.jp/supercomputer/ofp/service/
15) Oakbridge-CX スーパーコンピュータシステム：https://www.cc.u-tokyo.ac.jp/supercomputer/obcx/service/
16) ULFM: https://fault-tolerance.org/
17) Nakajima, K., Parallel iterative solvers of geofem with selective blocking preconditioning for nonlinear contact problems on the earth simulator, IEEE Proceedings of SC'03, 2003
18) METIS: http://glaros.dtc.umn.edu/gkhome/views/metis
19) Rizzi, F., Morris, K., Sargsyan, K, Mycek, P., Safta, C., Debusschere, B., LeMaritre, O., Knio, O.M., ULFM-MPI Implementation of a Resilient Task-Based Partial Differential Equations Preconditioner, FTXS '16: Proceedings of the ACM Workshop on Fault-Tolerance for HPC at Extreme Scale, p.19–26, 2016
20) Shahzad, S., Thies J., Kreutzer, M., Zeiser, T., Hager, G., Wellein, G., Craft: A library for easier application level checkpoint/restart and automatic fault tolerance, IEEE Transactions on Parallel and Distributed Systems, Vol. 30, No. 3, pp. 501–514, 2019
21) Fukasawa, T., Shazad, F., Nakajima, K., Wellein, G., pFEM-CRAFT: A library for application-level fault-resilience based on the craft framework. Poster at the 2018 SIAM Conference on Parallel Processing for Scientific Computing (SIAM PP18), 2018
22) 卜部卓，鷹野澄，鶴岡弘，中川茂樹，地震観測データ流通システム JDXnet の現状とクラウド化，電子情報通信学会研究報告 113 巻-256 号，21-23，2013
23) 住元真司，塙敏博，中島研吾，ULFM を用いた動的プロセス再構成ランタイムの試作，情報処理学会研究報告（2020-HPC-178-11），2021（in press）
24) 中島研吾，局所細分化を考慮した階層型ボクセル格子の並列データ構造，情報処理学会研究報告（2008-HPC-116-28），2008