令和４年度科学技術試験研究委託事業

「次世代計算基盤に係る調査研究」

（システム研究調査チーム）

成果報告書

令和５年５月３０日

国立大学法人神戸大学

牧野淳一郎

目次

本報告書は、文部科学省の令和4年度科学技術試験研究委託事業による委託業務として、国立大学法人神戸大学が実施した令和4年度「次世代計算基盤に係る調査研究」（システム研究調査チーム）の成果を取りまとめたものです。

事業の名称
「次世代計算基盤に係る調査研究」（システム研究調査チーム）

１．事業の目的
ポスト「富岳」時代の次世代計算基盤について、我が国として独自に開発・維持するべき技術を特定しつつ、要素技術の研究開発等を実施し、具体的な性能・機能等についての検討を行うことを目的とする。このため、国立大学法人 神戸大学を代表機関として、参画機関である株式会社 Preferred Networks、国立大学法人 東京大学、大学共同利用機関法人 情報・システム研究機構 国立情報学研究所、国立研究開発法人 海洋研究開発機構、国立大学法人 東海国立大学機構 名古屋大学、順天堂大学、国立大学法人 広島大学、東洋大学、国立研究開発法人 国立環境研究所、独立行政法人国立高等専門学校機構 松江工業高等専門学校、公立大学法人 会津大学と連携し、研究開発を推進する。

２．2022 年度（報告年度）の実施内容
２－１．当該年度（2022 年度）の業務計画
以下を目標とし研究開発を推進する。

① アーキテクチャ調査研究
2022年度は商用プロセッサについて、2028-2030年までのコストパフォーマンス、電力性能を予測すると共に、独自アクセラレータアーキテクチャについて、レファレンス設計を定め、N7ないしN5のプロセスルールでの動作速度・電力性能を評価する。

② システムソフトウェア・ライブラリ調査研究
独自アクセラレータのレファレンスアーキテクチャをアーキテクチャ調査研究グループと共同で定義し、アプリケーションを記述可能なデータ並列言語のプロトタイプを定め、処理系の開発を始める。また、粒子系、構造格子系向けのフレームワーク/DSLの仕様定義を行い、非構造格子向けフレームワークの検討を始める。

③ アプリケーション調査研究
想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/Oの必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにする。

また、分担機関と連携し、再委託によって、以下の研究開発に取り組む。

④ アーキテクチャ調査研究の統括およびアクセラレータ評価
（再委託先 株式会社 Preferred Networks）
アーキテクチャ調査研究全体の統括を行うと共に、独自アクセラレータアーキテクチャについて、レファレンス設計を定め、N7ないしN5のプロセスルールでの動作速度・電力性能を評価する。

⑤ CPU評価
（再委託先 国立大学法人 東京大学）
RISC-V等による独自実装の汎用CPUおよび商用CPUの評価のため、独自アーキテクチャについてレファレンス設計を定め、動作速度・電力性能を評価する。

⑥ ネットワークアーキテクチャ評価
（再委託先 大学共同利用機関法人 情報・システム研究機構 国立情報学研究所）
ネットワークアーキテクチャについて、レファレンス設計を定め、バンド幅・レイテンシ・電力性能を評価する。

⑦ システムソフトウェア・ライブラリ調査研究の統括およびコンパイラ検討
(再委託先 公立大学法人 会津大学)
システムソフトウェア・ライブラリ調査研究全体の統括を行うと共に、アプリケーションを記述可能なデータ並列言語のプロトタイプを定める。

⑧ DSL検討
(再委託先 独立行政法人国立高等専門学校機構 松江工業高等専門学校)
粒子系、構造格子系向けのフレームワーク/DSLの仕様定義を行い、非構造格子向けフレームワークの検討を始める。

⑨ アプリケーション調査研究の統括
(再委託先 順天堂大学)
アプリケーション調査研究全体を統括し、問題規模の設定、アルゴリズムの検討に対して必要なサポートを行う。また他グループへのアプリケーションからの提案点要求を整理し、コデザインを有効に進める。

⑩ 創薬と深層学習応用アプリケーションの検討
(再委託先 株式会社 Preferred Networks)
創薬と深層学習応用アプリケーションについて想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/0の必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにする。

⑪ ゲノム科学アプリケーションの検討
(再委託先 国立大学法人 東京大学)
ゲノム科学アプリケーションについて想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/0の必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにする。

⑫ 気象・気候シミュレーションアプリケーションの検討
(再委託先 国立研究開発法人 国立環境研究所)
気象・気候シミュレーションについて想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/0の必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにする。

⑬ 地震と構造物シミュレーションアプリケーションの検討
(再委託先 国立研究開発法人 海洋研究開発機構)
地震と構造物シミュレーションについて想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/0の必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにする。

⑭ ものづくりアプリケーションの検討
(再委託先 東洋大学)
ものづくりについて想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/0の必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにする。

⑮ マテリアルサイエンス応用アプリケーションの検討
(再委託先 国立大学法人 東海国立大学機構 名古屋大学)
マテリアルサイエンス応用について想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/0の必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにする。

⑯ 素粒子・原子核物理応用アプリケーションの検討

（再委託先 国立大学法人 広島大学）
素粒子・原子核物理について想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/0の必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにする。

⑰ 宇宙・惑星科学応用アプリケーションの検討
（再委託先 国立大学法人 東京大学）
宇宙・惑星科学について想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/0の必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにする。

⑱プロジェクトの総合的推進
プロジェクト全体の連携を密としつつ円滑に運営していくため、運営委員会や技術検討会の開催等、参画各機関の連携・調整にあたる。特に、プロジェクト全体の進捗状況を確認しつつ計画の合理化を検討し、必要に応じて調査或いは外部有識者を招聘して意見を聞くなど、プロジェクトの推進に資する。プロジェクトで得られた成果については、積極的に公表し、今後の展開に資する。

２－２．実施内容（成果）

本項目では、実施内容の概要を述べると共に、成果の詳細について記載する。

⓪ 本調査研究の内容と各調査研究グループの目標

0-1. 本調査研究の前提としての調査研究の概要と神戸大チームへの要望

「次世代計算基盤に係る調査研究」公募要領では、以下のように内容がまとめられている。(公募要領 P6)

２．事業の概要

（１）調査研究の内容

本調査研究では、ポスト「富岳」時代の次世代計算基盤の具体的な性能・機能等について、サイエンス・産業・社会のニーズを明確化し、それを実現可能なシステム等の選択肢を提案する。その際、我が国として独自に開発・維持するべき技術を特定しつつ、要素技術の研究開発等を実施する。

具体的には、システム（アーキテクチャ、システムソフトウェア・ライブラリ、アプリケーション）、新計算原理、運用技術を対象に、技術動向等を調査し、技術的課題や制約要因を抽出しつつ、次世代計算基盤に求められる具体的な性能・機能を明らかにする。

また、留意すべき事項として以下があげられている(公募要領 P7,8)。

○ 「富岳」の成果及び課題
○ 評価指標として考慮すべき項目
○ 技術的課題や制約要因の抽出
○ 日本が独自に保有すべき技術と国際協調する技術の特定

評価指標については以下のように述べられている。

評価指標には以下を含めることが望ましい。

演算性能、電力性能比、I/O 性能、コスト、運用可能性、生産性（アプリ開発のしやすさ）、商用展開・技術展開、カーボンニュートラルへの対応等

技術的課題や制約要因について、要点をまとめると、

● 半導体等技術等の原理的限界によるものと、条件により技術的に突破できるものとを精査する。
● 技術的課題以外でも、アカデミア・産業界双方における人材不足等次世代計算基盤の構築にあたって想定される課題・制約要因を考慮する必要がある。

となる。また、日本が独自に保有すべき技術と国際協調する技術の特定については、要点は以下の通りである。

システム（アーキテクチャ、システムソフトウェア・ライブラリ、アプリケーション）、新計算原理、運用技術について、国内外の情報を調査し、国内の開発対象とすべき範囲を提示する。更に、これまで培った技術の継承とともに、我が国として新たに強みを持つことができる、あるいは我が国の強みを見出せる可能性のある領域を特定する。

また、神戸大学の提案が採択されるにあたっては、神戸大学チームとして全ての可能性を検討する、というよりは、理研チームと協力・分担して補完的な役割を果たすことが望まれている、と理解している。すなわち、上記の目標を、検討対象となる全てのハードウェアアーキテクチャに対して行う、というよりも、理研チームでは検討対象になっていない、国内のベンチャー企業等の開発した、HPC ないし AI・深層学習をターゲットにしたプロセッサ技術をベースにした場合に、ポスト「富岳」時代の次世代計算基盤の具体的な性能・機能がどのようなものになりえるか、また、それは同じ時期に利用可能になると想定される国内外の商用システムに比べてどのような利点・欠点をもつのか、を検討対象とする。

もちろん、

○ 「富岳」の成果及び課題
○ 評価指標として考慮すべき項目
○ 技術的課題や制約要因の抽出
○ 日本が独自に保有すべき技術と国際協調する技術の特定

については必要な検討を行う。

0-2. 本調査研究の背景とアプローチの概要

0-1.にまとめた調査研究全体と神戸大チームへの要請を、本調査研究では以下のアプローチによって実現する。
本調査研究では、PFN・神戸大学の持つ世界最高性能のアクセラレータ実現技術とAI応用フレームワーク/アプリケーションソフトウェア技術を最大限に活用しつつ、従来から重要なHPCアプリケーションおよびAI利用等新しい応用の双方について高い実行効率を実現できるシステムの構成・評価をアプリケーショングループとの密接な協力によって実現する。

まず、なぜこのようなアプローチが有効と考えられるかをこれまでのHPCシステムのトレンドとそれを形作ってきた制約要因からまとめ、さらに今後の課題は何か、それに対してどのようなアプローチが必要か、という観点からまとめる。

0-2-1. 過去のHPCシステムのトレンドとその技術的背景

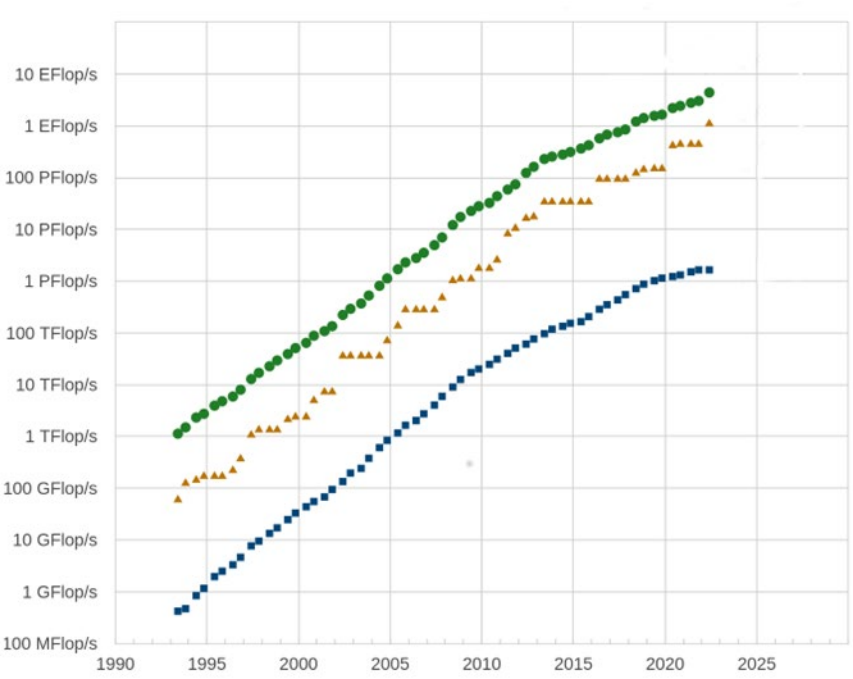

図1 Top500のトレンド

図1に、Top500リストにおけるTop1システム(オレンジ三角)、500位システム(青四角)、全体合計(緑丸)の1993年から2022年までのトレンドを示す。まず明らかにわかることは、Top1と全体合計の両方が、2012-13頃を境にしてそれ以降の性能向上率がそれ以前より明らかに低下していることである。
Top1は1993年のおよそ100GFから2013年前後の34PFまで、20年間で34万倍、すなわち1年で1.9倍の増加率で指数関数的な性能向上を示してきた。一方、2013年から2022年までの9年間では34PFから1.1PFまでのおよそ30倍、1年あたりでは1.46倍とほぼ半分の増加率となっている。

電力性能を示すGreen500を見ると2012-13年の2-3GF/W程度から2022/6の62.7GF/Wまで、およそ25倍の向上が実現されている。すなわち、最近10年間のTop1性能向上は、ほぼ電力性能の向上だけによっている。

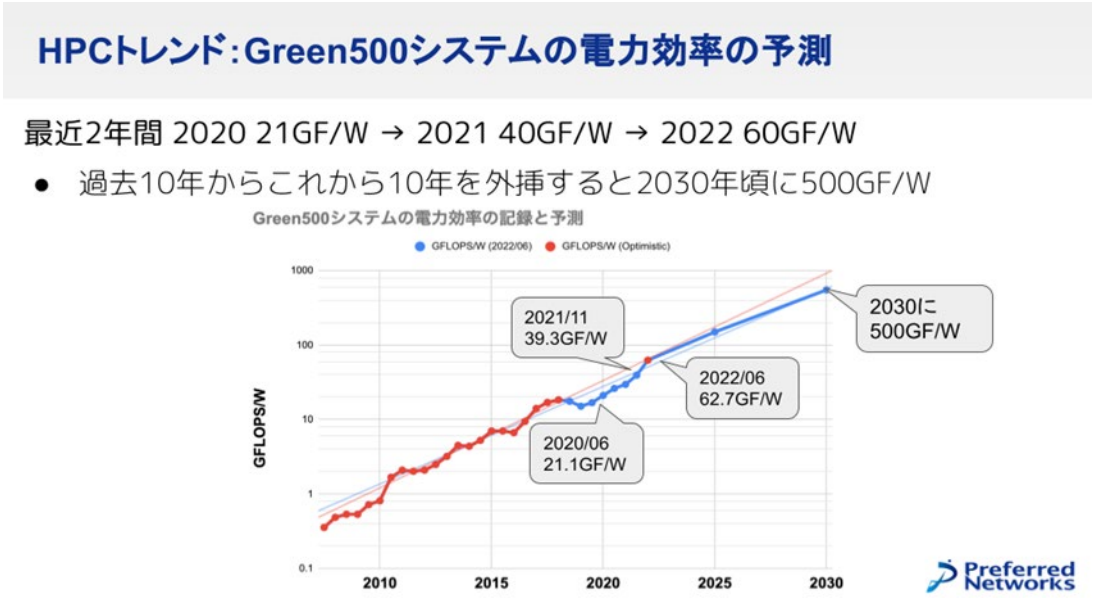

図 2 電力性能のトレンドと 2030 年予測

一方、1993/11 に Top1 となった数値風洞の消費電力は 1MW であり、電力性能は 0.24Mflops/W であるが、これは当時の典型的な HPC システムとは言いがたい。事実上最後の ECL ロジックのシステムだからである。ほぼ同時期の Cray T3D はおよそ 0.5Mflops/W であるので、こちらを典型的とすると、1993 年から 2013 年までの 20 年間の電力性能の向上はおよそ 5000 倍である。つまり、この期間は、10 年間あたりの電力性能向上はおよそ 70 倍であり、さらに 9 倍程度の消費電力の増加によってほぼ 10 年間に 600 倍の性能向上を実現してきたということになる。

一方、2013 年以降は電力性能の向上率が 10 年で 30 倍程度に低下し、さらに Top10 クラスのシステムの消費電力が 20-40MW と巨大になったため、それ以上に消費電力を増加させることが現実的ではなくなり、電力性能の向上率の低下もあわさって Top1 システムの性能向上率が大きく低下した。

1993 年から 2013 年までの HPC システムの性能向上にもっとも大きく貢献したのは半導体技術の進歩である。デザインルールは 0.8 μmから 45 nmまで 1/18 に低下し、コア電圧は 3.3V から 1V 前後に低下している。理論上はこれらによって 180 倍の電力性能の向上が実現できる。実際の向上は上でみたように 5000 倍であるので、あと 28 倍程度を主にアーキテクチャの改良で実現したことになる。但し、45nm の典型例は IBM BG/Q であるが、これは SOI プロセスを使っており半導体技術による改良分がさらに 2 倍程度あるかもしれない。それでも、10 倍以上をシステムアーキテクチャの改良によって実現している。

「システムアーキテクチャの改良」の内訳を見ていく。

T3D では、システム全体の消費電力のうちプロセッサの消費電力はごくわずかであった。1 ノードプロセッサの消費電力はおよそ 300W だが、プロセッサチップの消費電力は 20W 前後(150MHz 動作の場合)である。メモリおよびネットワークインターフェースが電力の大半を消費していることがわかる。

一方、BG/Q では、ノードあたり 80W 程度の消費電力のうち 55W をプロセッサチップが占める。メモリおよびネットワークの消費電力を相対的に大きく減少させていることがわかる。すなわち、全消費電力に対するプロセッサチップの消費電力の割合がほぼ 8 倍に増加しており、これが「アーキテクチャの改良」のほとんどを占める。これらは、基本的にはメモリバンド幅およびネットワークバンド幅の相対的な低下によって実現したものである。メモリ B/F では 2.5 から 0.2、ネットワーク B/F では 12 から 0.2 と大きく低下している。

このため、プロセッサアーキテクチャ自体の改良による消費電力の削減はそれほど大きなものではない。もっとも効果が大きいと思われるのはプロセッサコア当りの演算数の増加である。Cray T3D システムに採用されていた Alpha21064 プロセッサはサイクル毎に加算または乗算の 1 命令であったが、BG/Q は 4 FMA 8 演算となっている。この効果は 2-3 倍程度であることがわかる。

2013 年から 2022 年までの電力性能の 25 倍の向上の内訳をみていく。2022 年の電力性能のトップを実現したのは Frontiers であり、AMD MI250X GPU を使っている。従って、BG/Q とほぼ同時期に同様の消費電力を GPU で実現した Titan と比べるのが適切であろう。Titan は NVIDIA K20x であり、BG/Q とは異なり TSMC 28nm プロセスである。一方、MI250 は TSMC の「N6」プロセスである。

TSMC の場合 16FF 以降の FinFET プロセスでは CMOS スケーリング則の根拠となる「同一形状でのトランジスタの小型化」が成り立たっていないので、TSMC の発表による消費電力の低下をレファレンスとする。28HPM、16FF、16FF+、N7 の間の電力削減はそれぞれ 48%、30%、60%とされているので、電力効率向上は 7 倍となる。従って、3 倍強をアーキテクチャ改良で実現していると考えられる。このうち最大のものは、ベクトル演算から行列演算への移行であると考えられるが、AMD MI250X アーキテクチャは倍精度演算に特化する等特殊な特徴をもつためその寄与も大きいと思われる。

なお、この、9 年間で 7 倍の半導体の性能向上は、それまでの 20 年間の 180 倍に比べてそれほど小さくない。前者は年間 24%であるのに対して後者は 30%である。すなわち、半導体技術の進歩のスローダウン、いわゆるムーアの法則の限界は、2022 年までにおいては HPC システムの性能向上のスローダウンの主要な要因ではなかったことがわかる。プロセッサチップ以外の消費電力を限界近くまで削減したこと、消費電力が限界近くまで増加したことの 2 つの効果が大きい。

表 1 に K20x と MI250X の比較を示す。MI250X の名目倍精度ピーク性能は 95.7TF と AMD の公式資料にはあるが、Top500 サイトの名目ピーク性能と公式の Frontiers の構成からはその半分以下の 44.8TF がピーク性能となっており、なんらかの矛盾がある。ここでは消費電力は名目の値とし、ピーク性能は AMD 公式の半分とした値を参考値とした。

表 1 NVIDIA K20x と AMD MI250X

	FP64 peak(TF)	メモリバンド幅(GB/s)	消費電力 (W)	電力性能 (GF/W)	B/F
K20x	1.31	250	235	5.57	0.19
MI250X	95.7(47.9)	3276.8	560	171(85.5)	0.03(0.07)

名目の電力性能は 15 倍であるが、B/F 値はほぼ 1/3 になっていることがわかる。AMD 公称値の半分のピーク性能としても 0.07 は富岳の 1/5 であり、多様なアプリケーションで高い性能を実現するのは容易ではない。仮に 95.7TF が実現されているなら、B/F 値は実に 0.03 となり、極めて演算密度の高いアプリケーション以外では高い実行効率を実現することは困難ではないかと予想される。

すなわち、過去 10 年間の傾向をそれ以前と比較すると

- プロセッサチップの消費電力がシステム全体の消費電力の大きな割合を占めるようになった
- システムの消費電力が現実的な限界である 20-40MW に到達した
- 半導体プロセスの微細化による消費電力の削減の速度が若干低下した

ことによって HPC システムの性能向上率はそれまでのほぼ半分になったといえる。この性能向上率は

- B/F 値の大幅な低下
- 倍精度まで行列演算命令を採用することによる電力性能の向上

によってようやく実現できたものであり、どちらも多様なアプリケーションで高い実行効率を実現することを困難にするものである。

一方、この 10 年で飛躍的に発展し、今後もさらなる発展が期待できる深層学習分野では、16 ビット浮動小数点等の短い語長での行列・行列積が計算の主要部分となっており、これを高速に実現できることが極めて重要になった。NVIDIA は 2017 年に汎用 GPU としては世界で初めて 16 ビット行列乗算でユニットをもつ Tesla V100 を発表し、16 ビット行列乗算で前世代の P100 の 6 倍以上の性能を実現した。深層学習では、他の HPC アプリケーションに比べて低い B/F 値が大きな制約にならないことが多く、おそらく市場規模として人工知能・深層学習のほうがその他の従来型の HPC よりも既に大きくなっている現在、市場原理に従って開発されていく HPC システムは深層学習を主要なアプリケーションとして開発されていくことになろう。

0-2-2. 今後の HPC システムのトレンドの概要

前節で見たように、過去 30 年間については、HPC システムの性能向上は以下の 4 要因によっていた。

(1) プロセッサチップ以外の周辺ロジック、メモ等の電力の削減
(2) 消費電力の増加
(3) 半導体技術の進歩による 1 トランジスタの消費電力の削減
(4) プロセッサアーキテクチャの進歩による演算あたりの電力の削減

今後は(1)、(2)は期待できないというのは既にみた通りである。(3)、(4)についてはどうだろうか？詳細な分析は次章のアーキテクチャグループ報告で述べるが、以下に 2029 年に稼働するシステムを前提にして概要を述べる。

まず(3)の半導体技術についてである。現在のところ TSMC は 2025 年後半に N2 プロセスでの量産を始める計画であり、これまでと同様なペースでプロセス開発をつづけるなら 2028 年には N1.4 が利用可能になると期待できる。N7、N5、N3、N2 の世代毎の電力削減は 30%、30%、15%であるので、N2 から N1.4 を楽観的に 30%とし、さらに現在の計画通りに開発が進むとして、N7 からの電力性能向上は 3.4 倍となる。GAA プロセスの開発が多少遅延すると 2.5-3 倍となるであろう。

(4)のプロセッサアーキテクチャについては予測は難しいが、外挿としてありえるのは行列乗算の単位を大きくすることでの省電力化であろう。倍精度で2×2の行列を扱っているものを4×4にするものである。これによる電力性能向上は行列命令の導入ほどは大きくなく、1.5 倍程度と考えられる。

従って、(3)、(4)から総合的には、3.75-5.1 倍程度の電力性能向上、すなわち Green500 で 240-320GF/W、ボードレベルのピークではその 1.5 倍、360-480GF/W となる。

これは N1.4 の場合の推定値であり、N2 の場合は Green500 で 170-225GF/W、ボードレベルで 250-335GF/W となる。

Green500 で 320GF/W、平均電力が最大電力の 90%とすると、30MW の電力リミットでの HPL 性能は 8.64EF となる。

0-2-3. 実行効率と「使いやすさ」のトレンド

特にここ 15 年程度の HPC システムの、使う側からみた大きな問題は、高い実行効率を実現することが非常に困難、あるいは不可能になってきた、ということである。「京」でも、非常によく最適化・チューニングされた大規模アプリケーションの典型的な実行効率は 10-15%であった。もちろん、「京」は 2011、2012 年の 2 年連続でゴードン・ベル賞を獲得しており、それらでは 50%前後の実行効率を実現したが、これらが典型的とは言いがたい。規則格子の数値粒子コードで 15%、不規則格子の有限要素法では 2-3%というのが典型的な効率であった。

富岳では、この効率が大きく低下した。「京」のコードそのままでは、規則格子等では 15%の効率が 5%以下、また 50%に近い効率を実現していた粒子法コードでも 10%以下等である。以下に富岳で実行効率が低くなる主要な要因をまとめる。

(a) 非常に大きな演算、L1 および L2 レイテンシ
(b) SIMD 命令セットの HPC-ACE から SVE への変更によるアーキテクチャレジスタ数の減少
(c) x86 に比べて半分程度の OoO(アウトオブオーダー実行)資源
(d) x86 に比べて半分程度の L1、L2 バンド幅
(e) 比較的貧弱な分岐予測機構

この中で、(a)、(b)、(c)はそれぞれが単独で問題というよりはこの 3 つが組み合わさって大きな問題になっているものである。例えば x86 は(b)のアーキテクチャレジスタ数問題は抱えているが、(a)、(c)では富岳より余裕があり、結果的に実行効率低下の程度が小さい。

一方、上にみたような設計パラメータの選択は、(b)、(e)を別にすると高い電力性能とメモリバンド幅を汎用 CPU 設計で実現するためには必要なものであったように見える。大きな演算レイテンシはおそらく低電圧での比較的高いクロック実現に貢献しているであろうし、OoO 資源やキャッシュバンド幅の削減は大きな消費電力削減に直結する。

なお、「京」においても、多くのコードの実行効率は決して高くはない。地球シミュレータのような古典的ベクトルプロセッサであれば、規則格子、不規則格子のどちらでも 50%ないしそれ以上の実行効率を実現できたであろう。

「京」において多くのアプリケーションの実行効率が地球シミュレータよりも大きく下がった基本的な要因は B/F 値の低下である。地球シミュレータは B/F=4 を維持していたのに対して、「京」の B/F は 0.5 である。それでも単純に 1/8 まで効率が落ちるわけではなく、典型的には 1/4 程度であるのは、キャッシュ等がある程度は働いているからであるが、逆にいうとこれらのコードと「京」のアーキテクチャの組合せではキャッシュの有効性がその程度にとどまるということである。

ここで注意するべきことは、「京」における効率低下も、富岳における上でみた理由による性能低下も、原理的な限界ではない、ということである。

この問題は、以下のように理解することができる。

地球シミュレータのような、古典的なベクトルアーキテクチャに最適化されたアプリケーション向けに、キャッシュアーキテクチャを最適化することを考える。ベクトルアーキテクチャに最適化されたアプリケーションコードは、

- なるべく長いベクトルを
- なるべく連続に

アクセスするようなループ構造をもつ。これは、複数のプロセッサがそれぞれ高い並列度でメモリアクセスしても実行効率が高くなるように、ループ繰り返し数を大きくする必要があるからである。このため、必然的にキャッシュに入らないような広い領域を連続にアクセスすることになり、キャッシュデータの再利用率は低くなる。

このようなメモリアクセスに対するなるべく高い実行効率を実現するためには、主記憶のバンド幅を、特に連続アクセスに対して高くとることが最重要であり、L2、L1 キャッシュのバンド幅は相対的に重要でなくなる。さらに、特に L2 キャッシュのレイテンシはあまり重要ではなくなる。再利用されるキャッシュというより、ストリームアクセスのためのバッファメモリ的な使われ方になるからである。

一方、規則格子差分法のアルゴリズム自体を、メモリ階層がある、すなわちオンチップメモリは比較的低いレイテンシと高いバンド幅でアクセスでき、オフチップメモリは高いレイテンシと低いバンド幅にならざるを得ないマシンで実行することを考える時、まずバンド幅のことだけを考えるなら、最適なアルゴリズムはキャッシュブロッキング、すなわち、例えば3次元格子なら、それをキャッシュメモリに入る程度の小領域に分割し、順番に処理する、という方法であるはずである。

特に、近年はテンポラル・ブロッキングと呼ばれる、そのようなブロックに対して1ステップではなく複数時間ステップの積分を行うことで必要メモリバンド幅をさらに減少させる方法の研究が世界的にも活発に行われている。

しかし、「京」・富岳のアーキテクチャではこのような方法で実効効率をあげることが困難になっている。これは、特にLLCであるL2キャッシュのバンド幅が小さく、レイテンシが大きく、容量が小さいためである。

すなわち、日本の国家プロジェクトである地球シミュレータ、「京」、富岳の系列では、前世代のアーキテクチャに最適化されたアプリケーションコードに対して、次世代の半導体技術の制限の中でアーキテクチャを最適化する、というアプローチに結果的になってしまったため、原理的なアプリケーション・アルゴリズムに対する最適解とは違うものになってしまっている。特に、上でみたように、アーキテクチャが本来有効な最適化の効果を阻害するため、結果的にアプリケーションの進化も阻害し、そのことがさらにアーキテクチャの進化を阻害する、という悪循環になる。

なお、これは「京」・富岳だけの問題というわけではなく、x86やArm等の汎用プロセッサでも、またGPUでも同様にある問題であり、特にGPUは「京」・富岳とほぼ同様な問題を抱えている。非常に高い演算性能に対してキャッシュメモリは十分な大きさと速度をもっていないため、高い性能をだすにはレジスタデータの再利用が必須となっている。

一方、x86等は、実際にキャッシュデータ再利用ができるアプリケーションコードに対して高い実行性能を実現できるが、そのために短いループ長でも性能が落ちない低レイテンシのキャッシュ、演算ユニット、さらに少ないアーキテクチャレジスタでもパイプライン実行ができるための莫大なOoO資源が必要になり、これらの電力消費が高い電力効率の実現を妨げている。

ここまでをまとめると、地球シミュレータ・「京」・富岳の系列においては、「京」・富岳アーキテクチャデザインの時にレファレンスとなるアプリケーションコードが基本的に地球シミュレータ向けに最適化されたコードであったために、例えば規則格子差分法というアルゴリズムとしては基本的なものに対しても、アーキテクチャ設計が原理的な制約の下での最適点ではなく、前世代のアーキテクチャ向けに最適化されたアプリケーションコードに対する最適点となるため、アプリケーションとアーキテクチャの健全な共進化を阻害している。これは日本だけの問題ではなく、GPUアーキテクチャもまた汎用CPUアーキテクチャも同様な困難を抱えている。

本調査研究では、上記の理由から、検討対象とするアプリケーションエリアに対して、既存のアプリケーションコードそのものを評価対象にするのではなく、検討するアクセラレータアーキテクチャ向けにデータ構造・ループ構造等、場合によっては数値スキーム自体の書き変えを行った場合の性能を評価し、その評価に基づいてアプリケーションとアーキテクチャのコデザインを進める。その時に、現在および将来の半導体技術の発展方向も考慮することで、長期にわたって有効なアプリケーションを開発するための方法を明らかにする。以下、調査項目別に成果を記載する。

① アーキテクチャ調査研究

2022年度は商用プロセッサについて、2028-2030年までのコストパフォーマンス、電力性能を予測すると共に、独自アクセラレータアーキテクチャについて、レファレンス設計を定め、N7ないしN5のプロセスルールでの動作速度・電力性能を評価した。具体的には、半導体プロセスのトレンド及びNVIDIA/AMD社のアクセラレータのトレンドを調査し、2028年のプロセッサ性能は倍精度演算で173TFLOPS、消費電力は1060W

と予測した。レファレンスアーキテクチャについて、N7、N5での電力推定の結果と、半導体技術の進歩の予測から、これよりもかなり高い電力性能を実現できる見込みを得た。また、レファレンスアーキテクチャの汎用性を高めるため、高いバンド幅でDRAMを接続する方法の検討を進めた。詳細は④～⑥で述べる。

② システムソフトウェア・ライブラリ調査研究

システムソフトウェア検討チームでは、独自アクセラレータのレファレンスアーキテクチャをアーキテクチャ調査研究グループと共同で定義し、アプリケーションを記述可能なデータ並列言語のプロトタイプを定め、処理系の開発を始めた。具体的には、OpenCLおよびOpenACCをMN-Core後継アーキテクチャに移植する方法について検討を始めた。

また、粒子系、構造格子系向けのフレームワーク/DSLの仕様定義を行い、非構造格子向けフレームワークの検討を始めた。具体的には、HPCおよび機械学習のためのプログラミングフレームワークとDSLについて調査・検討した。アクセラレータ向けのプログラミングフレームワークについては、外部より講師を招きセミナーを実施し情報収集をした。また、データ並列言語のプロトタイプ仕様を検討するため、MN-Core用の演算カーネルを実装し動作確認をした。粒子系フレームワークの仕様を検討するために、アクセラレータを搭載した計算機においての重力N体シミュレーションのパフォーマンスモデルを作成し性能評価、アルゴリズムの検討などを行った。詳細は⑦、⑧で述べる。

③ アプリケーション調査研究

想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/Oの必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにした。具体的には、分子動力学、ゲノム解析、気象・気候シミュレーション、地震・構造物シミュレーション、構造解析(ものづくり)、マテリアルサイエンス、素粒子・原子核・宇宙の各アプリケーション領域で、想定するアプリケーションを定め、演算カーネルの抽出、ノード間通信、I/O量の推定を行った。一部のアプリケーションではカーネルのMN-Coreアクセラレータへの実装、性能評価を行った。詳細は⑨～⑱で述べる。

④ アーキテクチャ調査研究の統括およびアクセラレータ評価

(再委託先 株式会社Preferred Networks)

アーキテクチャ調査研究全体の統括を行うと共に、独自アクセラレータアーキテクチャについて、レファレンス設計を定め、N7ないしN5のプロセスルールでの動作速度・電力性能を評価した。具体的には、NVIDIA、AMDのそれぞれのプロセッサの性能を比較評価し、TSMCのN16からN3までの性能評価を行った。結果は以下にまとめる。

4-1. 商用アクセラレータの電力性能トレンドの推定

商用アクセラレータとしてNVIDIA社およびAMD社のGPGPUのこれまでのトレンドを調査し、将来の予測を実施した。

まず、GPGPUの世代間の性能向上が、TSMC社のプロセスノードの性能改善との相関を調査した。TSMC社のプロセスノード間の性能改善が表2である。

表 2 TSMC 社のプロセス情報

プロセス情報	N16/12	N10	N7	N5	N3	N3E	N2
Speed Improvement@same Power	50%	15% vs N16	20% vs N10	20% vs N7	15% vs N5	18% vs N5	15% vs N3E
Power Reduction@Same Speed	60%	35% vs N16	40% vs N10	40% vs N7	30% vs N5	34% vs N5	30% vs N3E
logic density		x2	x1.6	x1.8	x1.7	x1.6	不明
出典	TSMC HP	TSMC HP	TSMC HP	TSMC HP WikiChip	TSMC HP	TOM's Hardware	TOM's Hardware

上記TSMC社のプロセスノード間性能改善と、NVIDIA社GPGPUの電力性能トレンドとの相関の調査及び、将来予測結果を示したものが、表3である。

NVIDIA社のGPGPUの世代間の演算性能及び消費電力の改善は、基本的にはTSMC社の公称改善率以内に収まることが確認できた。

V100及びH100 SXMでは演算性能が前の世代から大きく改善しているが、ここはTensorCoreの採用やTensorCoreの大型化、半導体プロセスの改善以上の周波数向上等により演算器あたりのトランジスタ数を改善したことによると思われる。

その結果、前の世代に比べてダイサイズの大型化、トランジスタあたりの消費電力量が改善していない等の状況が見て取れ、基本的にはTSMC社の公称改善率以内での改善に収まるペースでの改善トレンドであると言える。

それをもとに将来の予測をしたものが表内のHopper Next及びHopper Next Nextの数値である。

この予測においては、TSMC社の半導体プロセスN3E,N2の情報を元に、実際のNVIDIA社GPGPUの改善幅を仮定して算出している。

富岳NEXT世代では、半導体プロセスの量産化が順調に進めばN2世代のHopper Next Nextの世代、若干の遅延が生じるとN3EのHopper Next世代となると思われる。それぞれの倍精度演算性能、電力、電力効率は、Hopper Nextが108TF,843W,127.62GF/W@FP64、Hopper Next Nextは173TF,1060W,162.78TF/Wと予想される。

表 3 NVIDIA 社 GPGPU のトレンドと将来予測

		既製品				予測	
製品名		P100	V100	A100 SXM	H100 SXM	Hopper Next	Hopper Next Next
製品発表年		2016	2017	2020	2022	2025	2028
プロセス(量産開始年)		N16(2015)	N12(2017)	N7(2018)	N4(2021)	N3E(2023)	N2(2026)
Spec	演算性能@FP16(TF)	21.2	125	312	989	1721	2761
	演算性能＠FP64(TF)	4.7	7.8	19.5	67	108	173
	FP16/FP64	4.5	16.0	16.0	14.8	16	16
	周波数(MHz)	1328	1455	1410	1780	1905	2038
	TDP(W)	300	300	400	700	843	1060
	電力効率 GF/W @FP64	15.67	26.00	48.75	95.71	127.62	162.78
	B Tr 数	15.3	21	54	80	120	180
	Die Size(mm^2)	610	815	826	814	814	814

表4はAMD社GPGPUのトレンドと将来予測であり、こちらもNVIDIA社GPGPUと同様の分析を実施した。

こちらもNVIDIA社と同様に、基本的にはTSMC社の公称改善率以内に収まることが確認できた。それをもとにMI250Xより将来予測をすると、MI250Nextは倍精度で115TF,672W,171.13GF/W、MI250Next Nextでは185TF,846Wで218.28GF/Wと予想される。

表 4 AMD 社 GPGPU のトレンドと将来予測

		既製品					予測	
製品名		MI8	MI25	MI50	MI100	MI250X-OAM	MI250 Next	MI250 Next Next
製品発表年		2017	2017	2018	2020	2021	2025	2028
プロセス(量産開始年)		N28	GF14	N7	N7	N6(2020)	N3E(2023)	N2(2026)
Spec	演算性能@FP16(TF)	8.2	26.4	26.5	186.4	383	920	1477
	演算性能@FP64(TF)	0.512	0.768	6.6	11.54	47.85	115	185
	FP16/FP64	16.0	34.4	4.0	16.2	8.0	8.0	8.0
	周波数(MHz)	1000	1500	1725	1502	1700	2040	2183
	TDP(W)	175	300	300	300	560	672	846
	電力効率 GF/W @FP64	2.93	2.56	22.00	38.47	85.45	171.13	218.28
	B Tr 数	8.9	12.5	13.2	21	58.2	116.4	174.6
	Die Size(mm^2)	596	510	331	763.2	1448	1450	1450

NVIDIA社、AMD社ともに予測値は当然上振れ、下振れが考えられるが、それを加味しても300GF/Wの達成には電力効率の大幅な向上が必要なため困難に見える。

次に、アクセラレータの金額の分析を実施した。
16/12nmから5nmまで、ファウンドリから販売されるWaferの価格を調査し、それをもとに3nm,2nmの価格を予測したものが表5である。

表 5 Wafer 価格と価格上昇率

プロセスノード	Wafer価格	価格上昇率
16nm/12 nm	$3,984	-
10 nm	$5,992	1.50倍
7 nm	$9,346	1.56倍
5 nm	$16,988	1.82倍
3 nm	$30,901（予測）	1.82倍（予測）
2 nm	$56,209（予測）	1.82倍（予測）

Wafer価格の出典:
https://www.techpowerup.com/272267/alleged-prices-of-tsmc-silicon-wafers-appear

ここから、ファウンドリから販売されるWaferの価格は世代が進むごとに指数関数的に上昇しており、さらに、世代が進むごとに金額の上昇率自体が上がっている傾向が見てとれた。

3nm,2nmでは5nm時点の1.82倍以上の倍率となることが容易に想像できるが、最低ラインである同じ倍率で試算している。

富岳では7nmのプロセスを使用していたが、仮に本プロジェクトにて3nmを採用すれば、wafer価格は最低でも3.3倍、2nmを採用すれば最低でも6倍のwafer価格となることになる。

本プロジェクトでは、この半導体価格の上昇が大きな制約となることが予想される。

4-2. 商用 CPU の電力性能トレンドの推定

商用CPUのトレンド調査のため、2015年以降に発売されたIntel社CPUのうち、サーバー向けであるPlatinumランクからモデルを抜粋したものが表6である。

2015年発売のSkylake世代から、2022年発売のSapphire Rapids世代までの8年間で、最大演算性能は3.66倍に上昇している。しかしながら、2016年発売のNVIDIA社GPGPU P100と2022年発表のH100 SXMは、7年間で倍精度演算の演算性能が14倍まで上昇していることを考慮すると、CPUの演算性能の向上は極めて緩やかであることが伺える。

この演算性能の向上は最大周波数および演算コア数によってもたらされているが、Sapphire Rapidsはチップレット技術を採用することで、より多くのコア数を実現している。これを除けばコア数はほぼ横ばいであり、その演算性能の向上は周波数の向上によってもたらされていることがわかる。

また、電力効率も8年間で2倍程度の改善と、NVIDIA社GPGPUが5.22倍であることを踏まえると、こちらも極めて緩やかであると言える。

表 6 Intel 社 CPU のトレンド

コードネーム	発売年	ランク	モデル	コア数	最大周波数 [GHz] (AVX512)	TDP [W]	最大演算性能	GF/W
Sapphire Rapids	2022	Platinum	8490H	60	2.9	350	5568	15.9
IceLake	2019	Platinum	8368Q	38	3.3	270	4013	14.9
Cascade Lake	2019	Platinum	8280L	28	2.4	205	2150	10.5
skylake	2015	Platinum	8180M	28	1.7	205	1523	7.4

4-3. MN-Core 後継の可能な電力性能の推定

PFN・神戸大学で、これまで開発を行ってきたアクセラレータについて実機での計測結果および設計時の評価結果もとに電力性能の推定を行った。これは、MN-Core(12FFCプロセス)、MN-Core2(N7プロセス)、MAU-Shuttle(N5プロセス)について、MABと呼んでいるプロセッサブロックの電力性能と、その中の演算器の部分のみの電力性能をまとめたものである。MAU-Shuttleについてはシミュレーション、他は実測である。

4-4. 独自アクセラレータアーキテクチャの検討

現時点で、アクセラレータのアーキテクチャとしては様々なものが提案・実装されている。本節では、まず 4-4-1節でそれらを概観したあと、4-4-2節以降で独自アクセラレータアーキテクチャについて検討する。

4-4-1. 既存アクセラレータアーキテクチャの比較

以下のものを検討対象とする。

- NVIDIA GPU
- PEZY SCx
- Sunway SW26010(Pro)
- Intel Max GPU
- MN-Core

これらについて、命令処理方式とメモリ階層、コア間通信の実現方法、結果としての面積・消費電力の観点から考察する。

4-4-1-1. 命令処理方式

ここで比較するのは、SIMDかMIMDかである。Pezy SCxとSunwayは完全なMIMDである。Sunwayはコア内でSIMDを採用しており、SIMD幅は256ビット(Proでは512ビット)である。Pezy SC2まではスカラー演算、SC2は128ビットSIMDである。これらにおけるSIMD方式はx86のSSE/AVX等と同様であり、1命令で処理されるデータ幅が広い。レジスタ内では要素毎の独立な並列演算となる。メモリアクセスはおそらくunaligned loadはサポートしているものと思われる。ストライドアクセスや間接アクセスについては不明である。

Pezy SCxでは最大8スレッドが時分割でサイクル毎に切り替わって動作する。この機能により、演算やL1アクセスのレイテンシを隠蔽でき、コンパイラによる命令スケジューリングやハードウェアによるOoO実行を不要にしている。SunwayではこのようなSMTのサポートはなく、OoO実行もリネームレジスタもないため、高い性能をだすには物理レジスタを注意深くアサインする必要がある。

NVIDIA GPUでは、[SM]という単位内でのSIMDである。A100では、1チップに128個のSMがあり、これがFP32 Cudaコア64個、Tensorコア4個をもつ。

1つのTensorコアは、16ビットの場合4×4の行列に対して$A \times B + C$の演算を行う。従って、データ幅は256ビット(積算項はその2倍)である。これが4つあることでデータ幅は1024ビット、積算項については2048ビットとなる。演算数は256FMAである。

Cudaコアにおける行列演算でないFP32/FP64演算の性能はTensorコアによる行列演算性能の1/16であるので、おそらくFP32ユニットは実装されておらず、FP16ユニットがSMあたり16個あるものと推測される。レジスタファイルはSMあたり256kBである。NVIDIAの資料では、Tensor core毎にL0 Iキャッシュ、Warp SchedulerとDispatch unitを持つことになっているので、この場合SMの中でTensor core毎の4ユニットがMIMD動作し、データ幅は256ビットということになる。つまり、AVX(2)命令をもつx86プロセッサのような、256ビット幅のSIMD命令をもつプロセッサの集合体とみなすこともできる。

NVIDIA GPUの特徴は、Tensor core毎に64kBと巨大なレジスタファイルをもつこと、この巨大なレジスタファイルを利用して、非常に多数のスレッドを起動できることである。ここで注意するべきことは、この「スレッド」はSIMD ユニットの中で動いているので、通常のスレッドのようなMIMD動作するものではなく、ソフトウェアから見えるSIMDユニットの幅が増えたようなものであることである。これは、ハードウェアのレイテンシを効率的に隠蔽する効果をもち、またアプリケーションの並列性の記述を簡潔にできる、極めて有効な機構である。

主記憶アクセスに対しては間接アクセス、すなわち、その時のスレッド毎に独立なアドレスをもって主記憶アクセスができる。これはもちろんレイテンシが高く、アドレスの分布によってはスループットも低いが、スレッド数が大きい時にはレイテンシは隠蔽される。

Intel MAX GPUは、単一のXeコアにVector Engine(VE)、MatrixEngine(XMX)がそれぞれ8個搭載され、VEはFP64/32/16演算を32/32/64個、XMXはFP16で512演算となっている。一方、データ幅は512ビットと記載されている。FP32に対してはサイクル毎に16データで演算幅とあっているがFP64では不足しているように見える。

Intel発表資料ではチップ全体でFP64/32 52Tflops、レジスタバンド幅419TB/s、L1バンド幅105 TB/sとなっており、VE数は1024であるので動作クロックは1.6 GHz、ベクトル演算のデータ幅はFP64で1024ビットとなる。FMA演算では2演算に対して3入力1出力であり、サイクル毎にVE1つで4096ビットのレジスタアクセスが発生するはずであるが、Intel資料からは1サイクルでVEがアクセスできるレジスタは2048ビットで不足している(FP32では問題ない)。

レジスタ構成、スレッドをサポートしているかどうか、主記憶アクセスにスレッド毎の独立アドレスをサポートするかどうか等は不明である。但し、レジスタファイルはVeあたり64kBと巨大であり、NVIDIA GPUと同様のSMTサポートがある可能性が高い。

MN-Coreは完全SIMDであり、チップ上の全てのユニットが同期して同一命令を処理する。間接アクセスは各ユニットが持つローカルメモリに対して行われる。

アプリケーションの実行効率については、MIMDが高く、SIMDでは低いのではないかと考えがちであるが、これに明確な根拠があるとはいいがたい。その理由の1つは、MIMDといいながら、Pezy SCx以外の全てのプロセッサはコア内やSM内はSIMDであることである。このため、条件分岐はマスク演算に置き換えられる。MIMDが有効に働くためには、プロセッサコアやSM単位で違う内容を、条件分岐の形ではなく実行するようなコードである必要がある。これは、通常のループ並列化やOpenCL、Cuda等による並列化で発生することは稀である。

特にコア数が大きい時には、並列化の効率に大きな影響をもつのは同期や放送・縮約等の大域通信のオーバーヘッドである。これについては4-4-1-4で述べる。

4-4-1-2. メモリ階層

全てのプロセッサはメモリ階層をもつが、その実装はプロセッサ毎に大きく異なる。以下それぞれについて述べる。

Pezy SCxはノンコヒーレントな3層キャッシュをもつ。SCの特徴は、キャッシュラインがL2、L3と大きくなることであり、メモリコントローラやキャッシュコントローラを複雑にせずに高い実効転送バンド幅を実現している。ノンコヒーレントなので、他のコアが変更したデータを触るためには明示的にキャッシュをフラッシュする必要がある。フラッシュ命令は階層毎に用意されている。

もうひとつの特徴は、各PEがキャッシュ階層とは独立にローカルメモリをもつことである。

Sunwayでは、プロセッサは1個のManagement Processor Element(MPE)と64個のComputer Processor Element(CPE)に分かれており、MPEはIキャッシュとL1、L2 DキャッシュをもつがCPEはIキャッシュは持つが Dキャッシュはもたない。その代わりに64kBのローカルメモリをもち、主記憶とローカルメモリの間はDMAでデータ転送する。主記憶のストライドアクセスはあると推測される。間接アクセスはなく、必要なら複数のDMAを起動する。

64個のCPEは8×8の2次元格子構造をもち、x方向、y方向のそれぞれでPoint-to-point、放送、総和をレジスタ間通信で行うことができる。この機能は並列化効率の向上に非常に有効である。

NVIDIA GPUはPezyと同様なノンコヒーレントな階層キャッシュをもつ。ラインサイズは128byte固定である。L1を分割してローカルメモリとして使用できる。

Intel MAX GPUは階層キャッシュである。コヒーレンシサポート、ラインサイズは不明である。L1を分割してローカルメモリとして使用できる。

4-4-1-3. オフチップメモリ

全てのプロセッサで、なんらかの形のオフチップメモリをつけている。
表８に外部メモリの主要なパラメータについてまとめる。

表 8 オフチップメモリ構成

	ピーク演算性能(FP64)	ピークメモリ転送速度	B/F	備考
PEZY SC2 4.1	4.1	95GB/s	0.023	DDR3
Sunway SW26010	3.06	137GB/s	0.045	DDR4
NVIDIA A100	19.5	1.56TB/s	0.080	HBM2
Intel Max GPU 52	52	3.2TB/s	0.062	HBM2e

(MN-Core については省略する)

Intel MAX GPUは、FP64のベクトル演算に対する値としている。FP32等では行列演算があり、FP64でも行列演算のサポートがあるとピーク性能が2倍、B/Fは半分の0.03になる。

「京」でB/F=0.5、富岳でも0.36を維持したことを考えると、これらのアクセラレータではB/Fの値が非常に低くなっていることがわかる。特に、Intel Max GPUではL2キャッシュのバンド幅も0.25と極めて低いものになっており、アプリケーション性能へのインパクトがあるのではないかと想像される。

4-4-1-4. コア間通信について

前節で、メモリ階層についてハードウェアの観点からまとめたが、本節ではどのような操作にどのようなオーバーヘッドがあるか、という観点からチップ内コア間通信がどのように実現されるかをまとめる。

* Point-to-point転送

SunwayとMN-Core以外の全てのプロセッサで、キャッシュを経由する。このため、キャッシュラインのフラッシュ、同期、リードのステップが必要となり、オーバーヘッドは大きくなる傾向がある(キャッシュフラッシュには容易にマイクロ秒オーダーの時間がかかる)。

Sunwayでは、64個のCPEグループの中では、任意のコア間でx方向、y方向の2ステップのレジスタ間通信によってデータを送ることができる。このレイテンシはナノ秒オーダーであり、非常に高速である。

MN-Coreでは、各階層の共有メモリを経由することでデータ転送が可能である。SIMDである関係上コア間でランダムな通信では多数回のアクセスが必要になるが、規則的な移動ならば効率的な転送が可能である。

* 同期・放送・縮約

SunwayとMN-Core以外の全てのプロセッサで、キャッシュを経由するPtP通信によって実装される。このため、関係するプロセッサ数の対数オーダーのオペレーションが必要になり、レイテンシは極めて大きくなる。NVIDIA GPUでは典型的には100マイクロ秒程度が総和にはかかる。この時間は多くアプリケーションでは致命的といえるほどに長く、アクセラレータの広い範囲のアプリケーションへの適用を妨げる大きな要因になっている。

Sunwayでは、64個のCPEグループの中では、縦、横の2回のオペレーションで同期・放送・縮約ができる。このレイテンシはナノ秒オーダーであり、非常に高速である。

コア間縮約のオーバーヘッドは、多くのアプリケーションで並列化効率に直結する。単純な例としては行列ベクトル積がある。コア間縮約が遅いため使えない場合には、結果の1要素の演算は1コアで行う必要があり、行列の1次元方向の要素数よりも多くのコアを使うことは難しい。縮約が十分速いなら、1つの結果要素のための内積演算を複数コアで並列に行えるため、並列化効率が飛躍的に向上する。また、多くのアプリケーションで全体での最大値や総和といった演算が実際に発生する。典型的な例は有限要素法の反復計算で現れる内積演算である。MPIによるノード間通信での総和は、近年はIBスイッチが縮約をサポートすることもあり高速化しており、チップ内での総和のサポートは必要度が高いものになっている。

4-4-2. 電力コストについて

ここでは、命令実行方式、メモリ階層構造及びオフチップメモリについて、消費電力の観点から検討する。

まず、MIMD/SIMDの選択と電力コストについては、命令フェッチ・デコード・スケジュールユニットのコストを考えるとその個数が少ないことが望ましいのは自明である。但し、汎用CPUのようなコア内SIMD命令では、無制限にSIMD幅を増やすことはどこかからの時点で消費電力増加につながる。これは、レジスタ内のシャッフル命令等がビット幅に比例以上のコストがかかること、演算ユニット、メモリユニットともに物理的に大きく、配線長が長くなるため、配線遅延をカバーするためには大電力のドライバが必要になること等による。これを避けるためには、原理的にはシャッフル命令を排除、またはそのスループットを下げ、さらにメモリ・レジスタ間転送で非アラインド転送を排除して、演算器の近くにL1Dをもってくればよい。これはしかし、キャッシュ階層をもつアーキテクチャでは困難である。

これらの観点から、「電力コストだけを考えた時には」命令実行方式としてはなるべく大きな単位でのSIMD方式が望ましいが、幅広いSIMD方式ではメモリ階層の構造も合わせて考える必要があることがわかる。

次に、メモリ階層構造について検討する。4-1節の数値からは、2029年頃のアクセラレータについて、半導体技術の進展、アーキテクチャの進歩から想定されるボードレベルの電力あたり性能は500GF/Wとなる。この値から、「1演算につきデータをどの程度の距離チップ内で移動可能か」が決まる。チップ内の電気信号によるデータ移動に必要な消費電力は、配線長さ 1、の単位長あたりのキャパシタンス c と信号の電圧振幅 V によって、以下のように与えられる。

$$w = \frac{cV^2}{2}$$

配線からグラウンドプレーンまでの距離と配線の幅ないし高さが同程度で絶縁体の誘電率があまり大きくない時、cは次式の程度である

$$c = 0.2(\mathrm{fF}/\mu\mathrm{m})$$

電圧振幅Vを0.6V、配線長を10mmとした時、Wは0.36pJである。従って、64ビットのデータを5mm移動すると、12pJのエネルギーが消費される。

仮に電力性能の目標を1Tflops/Wとすると、これは1演算あたり1pJである。すなわち、64ビットデータを0.43mm移動するだけで1pJ消費する。B/F=1のLLCがあるとして、演算器からメモリセルまでの平均配線長が$0.43 \times 8 = 3.5$mmを超えると、LLCまでのデータ移動だけで1Tflops/Wを実現するための電力をすべて消費してしまうことがわかる。従って、ある程度の規模でチップ内コアから物理的に共有されるLLCをもつ限り、2029年頃に半導体技術的には実現可能と考えられる1TF/Wを実現するのは実際には不可能である。

これに対する対応としては以下のようなものが考えられる。

(1) 共有キャッシュを排除し、SunwayのようなメインメモリへのDMAのみ、あるいはMN-Coreのようなチップ内ネットワークのみとする。
(2) キャッシュメモリのバンド幅を下げる。
(3) 3次元実装、チップ上光通信等の技術でデータ移動のエネルギー消費を下げる。
(4) 上の(1)-(3)の組合せ等。例えば、PEZY-SCのようにキャッシュとローカルメモリを組み合わせて、さらにDMAやチップ内ネットワークももつ。

消費電力とコストだけを考えると(1)が望ましいが、既にGPGPUにポートされているアプリケーションや、特にOpenMP等で記述されているアプリケーションのポータビリティを考えると、キャッシュがあるメモ

リ構成が望ましい。一方、OpenACCの場合には配列のレイアウト等を指定できることもあり、むしろDMAやチップ内ネットワークのほうが効率的な実装ができる。

最後に、オフチップメモリについて検討する。

HBM3を使った場合に、データ転送のエネルギー消費は2-3 pJ/bitと言われている。これは、DRAMチップ上、サブストレート上、プロセッサチップ上でそれぞれ5-10 mm程度のデータ移動があり、特にDRAMチップ上では駆動電圧をあまり下げられないことから原理的な限界といえる。このことから、DRAMへのデータ移動の消費エネルギーが全体の20%、目標電力性能が1 TF/Wとすると、1演算あたりDRAMアクセスの消費エネルギーが0.2 pJ、移動できるデータは演算あたり0.07-0.1ビット、B/Fの値としては0.008-0.0125となる。

B/Fが0.001ということは、オフチップメモリ1語アクセスにつき1万演算程度しないとメモリバンド幅リミットになる、ということである。大規模な密行列の操作が主体となるアプリケーションではそのような状況もありえるが、大多数のアプリケーションではこれは現実的ではない。

このことは、HBMxの現在の実装の延長では、多くのアプリケーションが有効に利用できるオフチップメモリをアクセラレータにつなげることは困難である、ということを意味する。従って、より低い消費電力でアクセスできるオフチップメモリ実装方式の検討は必須である。そのオプションとしては以下のようなものが考えられる。

(a) HBMxメモリの設計はそのままで、ロジックダイ上に3次元実装する
(b) HBMxメモリの実装を変更し、特にDRAMダイ上でのデータ移動を減らす
(c) カスタムあるいは今後期待される標準品DRAMをロジックダイ上に3次元実装する
(d) SRAMダイをロジックダイ上に3次元実装する

本調査研究では、これらのオプションについて、消費電力とアプリケーション性能への影響の観点から比較検討する。

4-4-3. メモリ階層のオプション

本調査研究では、人的な資源の制約もあり、チップ内メモリ階層については(1)のチップ内ネットワークをベースにしてアプリケーション実装はどのようになるか、キャッシュに比べてどのような優劣があるかを検討する。オフチップメモリについては、コスト・バンド幅の観点からは最善といえる(c)のカスタムDRAMメモリの3次元実装をレファレンスとし、他のオプションではどのようになるかを比較検討する。

4-4-4. MN-Core 後継のベースラインアーキテクチャの決定

ここまでのアーキテクチャ、メモリ階層の検討結果を踏まえ、本調査研究でのアプリケーション評価、アーキテクチャ評価のレファレンスとなるアーキテクチャを策定した。

⑤ CPU 評価

(再委託先 国立大学法人 東京大学)

商用CPUの長期トレンドをまずみておく。動作クロックは、1990年代初頭の20MHz前後から、2000年代前半の最大3.4GHzまで上がったあと、ゆっくりと低下して2GHz程度になっていたが、上昇に移っている。表９にAMD EPYCプロセッサの最近3世代のベースクロックを、最大のベースクロックを実現するコア数のもの、最大コア数の中で最大クロックのものの組合せで示す。下にいくほど新しいモデルである。

表９ EPYC プロセッサの最大ベースクロックのコア数と最大コア数のクロック

	最大ベースクロック	最大コア数	備考
7002	3.2GHz/8c	2.65GHz/64c	
7003	3.7GHz/8c	2.45GHz/64c	
9004	4.1GHz/16c	2.4GHz/96c	3.1GHz/64cモデルが存在

最大コア数ではベースクロックはまだ若干減少しているが、8-16コア程度のモデルではベースクロックが急激に上昇していることがわかる。

このベースクロックの上昇は、世代毎の性能向上を実現するためには必要なことと考えられるが、同一コア数でもソケットあたりの電力増加をもたらしている。例えば最大の64コアの場合、TDPがこの3世代で280Wから360Wに増加した。2ソケットのシステムではCPUだけで720Wを消費することになる。

AMD EPYCに比べてIntel Xeonの消費電力はさらに高くなっており、どちらも消費電力を増加させてソケットあたり及びシリコン面積あたりの処理能力をあげる傾向が明白になっている。

一方、Armは、AppleがMacに全面的に採用し、従来のIntel x86ベースに比べて劇的な高性能・低電力化を実現した。また、NeoverseベースのGraviton2/3/3EがAWSで提供されて、利用価格あたりではIntel/AMDに比べて高い処理性能を提供している。

さらに、まだ採用実績があるわけではないが、RISC-Vでもx86やArmの高性能コアに匹敵するような高性能とさらに高い電力あたり性能を目標とするコア(例えばTenstorrentのAscalon)が発表されており、x86/Armに比べて命令デコード数等のパラメータを変えることでの電力最適化ができるという利点もあることがわかった。

RISC-V等による独自実装の汎用CPUおよび商用CPUの評価のため、独自アーキテクチャについてレファレンス設計を定め、動作速度・電力性能を評価した。実際の作業としては、RISC-Vについては商用IPの代表的なものを検討し、特に性能と、複数の実装オプションがあることから第一の可能性としてTenstorrentのAscalonをベースに検討を進めることとした。

CPUとアクセラレータの結合方式、必要な機能については、アクセラレータの側でレファレンスアーキテクチャとして採用したMN-Core後継をベースに検討を行った。想定するインターフェースはPCIe(UCIe)として、転送バンド幅からチップレット実装が必要と考えられる。

CPU側では共有キャッシュベースのコア間通信で十分なのか？というのは大きな問題である。A64FXにいたる富士通のHPCむけプロセッサでは、L2Dを共有するプロセッサコア間での高速な同期をハードウェアでサポートすることで、キャッシュベースのコア間通信のレイテンシの問題を相当程度軽減してきた。特に、OpenMPや自動並列化によるループの並列化で、比較的繰り返し回数が少なく実行時間が短いループでもある程度の性能向上を実現するには高速な同期は必須である。また、CPUとアクセラレータの間の通信レイテンシやアクセラレータのカーネル起動のオーバーヘッドを小さくすることは並列性能、特に強スケーリング性能の向上に極めて重要だが、ホストCPU側の同期や、DRAMからのデータ転送の起動にそれ以上に時間がかかっていては意味がない。

このため、CPU側でも、アクセラレータ側でみたようなコア間の高速な同期・通信のためのメカニズムがアプリケーション性能向上に大きく貢献すると考えられる。このようなメカニズムについて検討を始めた。

⑥ ネットワークアーキテクチャ評価

(再委託先 大学共同利用機関法人 情報・システム研究機構 国立情報学研究所)

まず、計算ノードのレファレンス設計に対して、ネットワーク側に必要な性能・機能を評価した。最終的にはこれはアプリケーション性能評価に基づいて決定する必要があるが、まだアプリケーションのネットワーク要求の詳細は定まっていないので、まずHPLとHPCGベンチマークを対象とした。HPLは演算に対してメモリバンド幅、ネットワークバンド幅への要求が極度に低いと思われがちだが、メモリバンド幅については確かににその傾向があるものの、ネットワークバンド幅については必ずしもそうではない。これは、どちらも演算速度に比例、メモリ量の平方根に反比例、という形になるため、必要メモリバンド幅と必要ネットワークバンド幅の比は定数になるからである。

詳細は省略するが、HPLである程度の実行効率をだすために必要なネットワークバンド幅(縦横合計)は次式で与えられる。

$$b > b\_min=32c/(x^{1/2})$$

bはバイト/秒単位のネットワークバンド幅、cは1秒あたりの演算回数、xはバイト単位の主記憶サイズである。c=256TF、x=128GBとすると、

$$b\_min=22.9GB/s$$

となる。つまり、ネットワークを2次元格子にマップした時に、4つのリンクのそれぞれの転送バンド幅が11.5GB/s以上あればよい。合計でほぼ50GB/sである。これは直接網の場合で、間接網で縦・横の次元のそれぞれで放送をサポートすればこの半分の25GB/sでよい。

HPCGでは主要な通信は袖交換である(マルチグリッド法なのでレイテンシもある程度重要だがここでは省略する)。レファレンスアーキテクチャでアクセラレータ主記憶が32GBとするとほぼ500^3程度のグリッドが入る。B/F=0.1を仮定すると実行効率は1.6%となり、この時の1イテレーション(最大格子サイズ)の実行時間は1.2ミリ秒となる。袖のサイズは 12 MB程度なので、通信時間を計算時間の1/3の0.4ミリ秒として必要なネットワーク速度は6方向合計で30GB/s程度でよい。

最近、必要となるネットワークバンド幅を抑えるために、アーキテクチャレベルで通信データの圧縮を行い、通信量を削減する実装が登場している。例えば、2倍の圧縮率が得られた場合、必要となるネットワークバンド幅を半分に抑えることができる。そこで、圧縮技術について検討を行った。Anton3、あるFPGA専用計算機では、対象とするアプリケーション(例:Molecular Dynamics(MD)シミュレーション)の通信データの内容に局所性があることを利用して、実際に転送するデータ転送量を削減している。Anton3の相互結合網では、先頭ビットから連続して"0"が生じるようにINZ(interleaved non-zero encoding)を採用し、そのビットを除去して転送することでデータ量を削減している。佐野らのFPGAネットワークの研究では、2次元Lattice Boltzmann Methods(2D-LBMs)において、同一送受信間の通信において前後の転送データに依存関係があることを利用して圧縮することで高い圧縮率を得ることができ、その結果1.7倍の性能向上を達成したことが報告されている。これらの通信データの圧縮の例は、(1)データが32ビット表現である点、(2)アプリケーションで発生する通信データの局所性を利用したシンプルな可逆圧縮方式を採用している点が共通している。一般的に、64ビットの浮動小数点数を対象とする可逆圧縮アルゴリズムは処理が複雑である。そのため1μ秒以下の通信時間が前提の相互結合網に導入できる技術は限られている。特にAnton3の相互結合網は1hopあたり55nsと極限まで小さな通信遅延である。そこで、特定の下位ビットを単純に削除し、誤差を許容する非可逆圧縮方式を検討した。

ネットワークトポロジ・機能についても検討を始めた。チップ内の通信と同じく、大域通信としてもっと

も重要なのは放送と縮約である。典型的な縮約操作であるMPI_Allreduceについて検討した。

「京」、富岳で採用されているTofuネットワークでは、「京」9216ノードでメッセージ16KiBの時に50MB/s、32KiBの時に100MB/sの速度が得られている。すなわち、これらのメッセージサイズでレイテンシは320マイクロ秒である。富岳では若干高速化しているはずだが大きな差はない。

一方、Mellanoxのスイッチを使った1500ノードのネットワークでは、MellanoxのISC2019での発表によれば、スイッチ側で縮約を行うSHARPを使わない場合に60マイクロ秒程度、SHARPを使うと10マイクロ秒程度となり、ノード数が若干少ないとはいえ「京」とは非常に大きな差がある。

Tofuネットワークではネットワークの直径が間接網に比べて大きいことが、レイテンシを非常に大きくしていることがわかる。今後、他のオペレーションやアプリケーションの実際の通信パターンに対する評価を進めていく。

なお、最近、ネットワークトポロジを光サーキットスイッチ(OCS)により、動的に更新する技術が研究開発され、実用化されつつある。その点が新たなトレンドであり直接網、間接網のネットワーク設計に影響を与える可能性がある。Googleでは、学習用アクセラレータであるTPUv4、Jupiterデータセンターネットワークでは、光サーキットスイッチを利用したネットワーク構成を採用した。TPUv4クラスタでは、トーラスネットワークトポロジに対して光サーキットスイッチを導入した。Jupiterデータセンターネットワークは、これまで採用してきたClos網ではなく、パケットスイッチ間に光サーキットスイッチで構成された中間層を導入する直接網を採用した点に特徴がある。そして、Clos網と同スループットかつ、送受信スイッチ間にサーキットを設定することで平均レイテンシを削減できることが報告されている。光サーキットスイッチのHPCシステムへの導入に関する研究は従来から行われており、Jupiterデータセンターネットワークのように単純に相互結合網の性能向上を意図したものが多い。一方でTPUv4クラスタは、隣接したTPUノード群にジョブを割り当てるジョブスケジューリング(いわゆるトポロジエンベッディング)、セキュリティ向上のためにユーザ毎に異なる資源群の利用、故障箇所の迂回など従来と異なる目的である点が注目される。そこで、様々なネットワークトポロジをベースにして、光サーキットスイッチが部分的に導入される前提にて、理論的なトポロジエンベッディングや故障箇所を避けたジョブパーティショニングについて検討を行った。

⑦ システムソフトウェア・ライブラリ調査研究の統括およびコンパイラ検討

(再委託先 公立大学法人 会津大学)

システムソフトウェア・ライブラリ調査研究全体の統括を行うと共に、アプリケーションを記述可能なデータ並列言語のプロトタイプを定めた。具体的には、HPCおよび機械学習のためのプログラミングフレームワークとDSLについて調査・検討した。特にアクセラレータ向けのプログラミングフレームワークについては、外部より講師を招きセミナーを実施し情報収集をした。データ並列言語のプロトタイプ仕様の詳細を検討するため、MN-Core用の倍々精度演算カーネルを実装し動作確認をした。

さらに、機械学習向けフレームワークのMN-Core向け実装について、その考え方で各種のHPCアプリケーションを記述可能かどうかの検討をアプリケーショングループと共同で行った。MN-Core向け深層学習フレームワークは基本的にはPyTorchで記述した深層学習ネットワークからアプリケーションコードを生成することができ、深層学習ネットワークの記述から自動微分を使った誤差伝搬学習コードの生成、そのMN-Coreむけの最適化等を行うことができる。もっとも、並列、言い換えると配列や演算を分割して演算コアに割り当てる方法等は、自動割当も可能だが最適化にはOpenACCのプラグマと同様な考え方で指定することが必要である。

基本的に、規則格子や密行列を扱うところが計算の主要な部分であるようなアプリケーションでは、その部分を深層学習向けフレームワークで記述することは、PyTorchやそのベースであるnumPyの経験があればそれほど困難ではないことがわかった。但し、深層学習向けフレームワークは、ニューラルネットに最適化されているため、最内側計算カーネルが行列積である時にもっとも高い性能がでる。言い換えると、通常の差分法等は、記述はでき、実行もできるが生成されたコードの最適化は十分ではない。従って、コ

ード生成・最適化部分に改良が必要であることが確認できた。

OpenCL、OpenACC等の従来のHPCアプリケーションのアクセラレータでの記述に使われている環境の実現についても検討を始めた。

OpenCLについては、プライベートメモリ・ローカルメモリ・グローバルメモリ等の階層が仕様に存在しており、プライベートメモリはそれを所有するプロセッサコアだけがアクセス可能なローカルメモリに対応するので、実行モデルとしてはMN-Coreに非常に近い。但し、MN-Coreではそれぞれのプロセッサコアがキャッシュ経由でDRAMをアクセスする、というデータパスが存在しないため、各スレッドが自分でアドレスを生成してDRAMにアクセスすることはできない。DRAMとのデータ転送は、DRAMからブロックを読み出して、それをチップ内ネットワークを経由してプロセッサコアに分配することになる。

このことは、しかし、アプリケーションを記述する上で大きな問題となるものではないと考えられる。アプリケーションを記述する側の立場で考えると、実際にしたいことはまさにDRAMからブロックを読み出してそれをプロセッサコアに分配することであって、そのために各プロセッサコアに自分でアドレスを計算させてみたDRAMをアクセスさせるのはむしろ意味的には遠い操作になっているからである。

もちろん、速度を問題にしなければ、DRAMを各コアが読み出す、というOpenCLコードから、実際に対応するデータをDRAMから順番に読み出してプロセッサコアに格納する、という実行コードを生成することも可能であり、OpenCLコードそのものを実行させることもできる。

OpenACCについては、実際の記述の内容はPyTorch/numPyと同様であり、OpenCLよりも意味論的な問題は少ないことがわかった。

⑧ DSL 検討

(再委託先 独立行政法人国立高等専門学校機構 松江工業高等専門学校)

粒子系、構造格子系向けのフレームワーク/DSLの仕様定義を行い、非構造格子向けフレームワークの検討を始めた。具体的には、アクセラレータを搭載した計算機においての重力N体シミュレーションのパフォーマンスモデルを作成し性能評価、アルゴリズムの検討などを行った。

粒子系向けフレームワークについては、理研計算科学研究機構(当時)の粒子系シミュレータ研究チームで開発を始めたFDPSを使うことを想定し、実際のMn-Coreへの移植を行った。現状では、FDPSは粒子間相互作用の計算のみをアクセラレータにオフロードする。相互作用計算の記述は、Cuda等で直接行うこともできるが、相互作用記述の専用DSL PIKGを使うこともできる。PIKGからMN-Coreの最適化されたアセンブラとホストとのデータ交換記述を生成するシステムを開発し、性能評価を行った。

カーネル部分についてはほぼ理論限界を実現するコードを生成できた。一方で、現在のMN-Coreのチップ内データ転送命令では、適切な分配を行うためにホスト側で複雑なデータの並べかえを行う必要があり、そのオーバーヘッドがデータ転送速度を低下させることがわかった。この知見はデータ転送命令の仕様に反映させる。

⑨ アプリケーション調査研究の統括

(再委託先 順天堂大学)

アプリケーション調査研究全体を統括し、問題規模の設定、アルゴリズムの検討に対して必要なサポートを行った。また他グループへのアプリケーションからの提案点要求を整理し、コデザインを有効に進めた。具体的には、８つのアプリケーション分野それぞれに対し、まずは、現在の計算速度では計算時間に問題があり、次世代のスーパーコンピュータを必要とし、今後発展するであろうコードを選択してもらった。さらに、そのコードの現在の計算規模、および今後の計算規模を見積もってもらうとともに、カーネルの抽出に取り組んでもらった。この時、計算規模の見積もりも同時に行った。アルゴリズムに関しては計算と通信のオーバーラップが重要で、このために計算速度と通信速度の比が重要なパラメータとなる。現在、この比の見積もりの精度を確認しているところであり、アプリケーション全体で整理し、コデザイ

ンをさらに有効に進めていきたい。

アプリケーション側からPreferred Networksの開発したMN-Coreを使ってみたいという要望があり、使い方のレクチャーとリモートでの接続を提供してもらった。

また、分担機関と連携し、再委託によって、以下の研究開発に取り組んだ。

⑩ 創薬と深層学習応用アプリケーションの検討
(再委託先 株式会社 Preferred Networks)
創薬と深層学習応用アプリケーションについて想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/Oの必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにした。具体的には、創薬については、分子動力学シミュレーションソフトウェアLAMMPSのベンチマークから一つ選び、ボトルネックである相互作用計算カーネルのMN-Coreでの最適化実装を与え、相互作用リストによる粒子データの索引がボトルネックとなることを明らかにした。深層学習応用については、独自アクセラレータアーキテクチャの先端プロセスでの性能諸元を仮定して、MN-Coreで動作確認済みの既存アプリケーションのいくつかについて性能評価を行った。

⑪ ゲノム科学アプリケーションの検討
(再委託先 国立大学法人 東京大学)
ゲノム科学アプリケーションについて想定するアプリケーションを定め、研究グループと共有することで新しいアルゴリズムの検討の方向を明らかにした。具体的には東大医科研スパコンShirokaneで実施しているNVIDIA社パイプラインを紹介し、実行時間やボトルネックについて議論した。

NVIDIA社のものを含めて、従来の標準的な実装や、高速化されたものはいずれも、マッピングについてはオープンソースのbwa-memに実装されているアルゴリズムを元にしている。また、マッピング後の変異の解析(いわゆる variant call)については、これも公開されているgatk toolkitが使われている。これらの標準ツールでヒトゲノムの30倍程度のカバレージのデータの解析を行った場合、通常のCPUサーバー(2CPUのIntelXeonサーバー等)で1日前後、NVIDIA DGXで25分前後かかっていた。

ところが、最近になって、CPUベースでこの解析を20分程度で行えると主張する解析ツールが複数アナウンスされた(QIAGEN CLC Light Speed Module https://digitalinsights.qiagen.com/plugins/clc-lightspeed-module/ 等)。

bwa-memの場合、レファレンスゲノムに対してBWTを行ってできるテーブルと補助データ構造を、リード配列に応じてほぼランダムにアクセスするため、主記憶のランダムアクセス性能がボトルネックとなる。アクセスする範囲は数GB程度となる。bwa-memの実行プロファイルを調査した結果、bwa-memはCPU向けには高度にチューニングされており、メモリアクセス回数から期待できる性能がでているとわかった。

このため、より高速なソフトウェアでは、bwa-memとは異なるアルゴリズムが採用されていると考えられるが、その内容は明らかになっていない。高速アルゴリズムを調査する他、本FS内でアルゴリズム高速化を検討していく必要があるとわかった。

⑫ 気象・気候シミュレーションアプリケーションの検討
(再委託先 国立研究開発法人 国立環境研究所)
気象・気候シミュレーションについて想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/Oの必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにした。具体的には、検討アプリケーションとして全球非静力大気モデルNICAM（準構造格子ステンシル）を選定し、切り出し演算カーネルの最新化を行った。加えて、演算加速機（特にGPU）への最適化、またニューラルネットワークでの気象モデルの置き換えを実施した国内外の事例について調査を実施した。

〈想定するアプリケーションの選定〉
気候・気象シミュレーションに用いられる代表的なアプリケーションとして、大気モデル、陸域モデル、海洋モデルが挙げられる。大気モデルは大気の風速、気圧、気温、水物質等の時間発展を球面上の流体現象として解く部分（流体力学過程）と、大気放射過程、水物質の微物理過程、サブグリッドスケールの乱流拡散過程、下端境界（地表面）での水熱交換過程、化学物質の微物理過程や化学反応過程をそれぞれ解く部分（諸物理過程）を持つ。大気モデルには地球全体の大気を同時に解く全球モデルと、限定した地域のみを対象とし側面境界条件を与える領域モデルがある。それぞれについて、スペクトル展開して波数空間上で流体力学を解くスペクトル法モデルと、格子に区切って計算する格子法モデルが存在する。計算並列化では水平方向に領域分割を行い分散メモリでの計算を行うことが主流である。大気の鉛直方向には大気放射過程や乱流混合過程のように、陰解法で解くことが必要な計算があるため、鉛直方向の計算プロセス分割は一般的ではない。陸域モデルは、地球の陸面上を規則的または不規則的な領域で区切り、各領域での植物、土壌を介したエネルギー収支、水・炭素量等の物質収支を解く部分と、河川等による領域間の水の移動を解く部分を持つ。河川以外の過程のほとんどは領域間の依存関係がないため、水平方向に分割して並列化を行うことが容易である。河川の移流計算のためのネットワーク通信は複雑であるが、シミュレーション中に通信相手が変わることはなく、また全体からみて所要計算時間は短い。海洋モデルは大気モデルと同様に、流体力学過程と諸物理過程を持つが、海洋の場合陸地が存在するため、水平方向の境界条件がより複雑である。そのため全球海洋モデルにおいてもスペクトル法はほとんど用いられず、格子法モデルが一般的である。上記で挙げたシミュレーションモデル以外にも、土石流モデルや生態系モデル、電場を解く雷モデル等が存在する。また、気象・気候分野は古くから地球観測データ等を用いたデータ科学と密接に連携して研究を進めてきており、変分法やカルマンフィルタに代表されるデータ同化システムを利用することは大前提となる。それぞれのシミュレーションで必要となる時空間離散化やアルゴリズムは多種多様であり、検討するアーキテクチャにおいてこれらすべてを網羅可能であるかを検討することは人的・時間的制約のため難しい。そこで本研究では、最も代表的なアルゴリズムを持つアプリケーションを評価対象として選定する。

本研究では、格子法を用いた大気モデルの流体力学過程を対象とする。アプリケーションとしては、隣接格子の値を用いた演算（ステンシル計算）を行う全球非静力大気モデルNICAMを選定した。NICAMは水平方向の格子点をプロセス分割する際に、プロセス間のトポロジーを非構造的に、プロセス内の格子点を構造的に扱う「準構造格子系」を採用している。ただし、ステンシル計算を行う場合のアクセスパターンはシンプルなカーテシアン座標系とは異なっており、CFD分野で用いられる構造格子系、非構造格子系のアプリケーションとは異なる演算バターンを評価することが可能である。NICAMのステンシル演算カーネルは、富岳開発プロジェクトであるフラッグシップ2020プロジェクトや、日独仏エクサスケールコンピューティングのためのソフトウェア開発研究事業SPPEXAの一つであるAIMESプロジェクトにおいてすでに抽出・整備されており、これらを用いたソフトウェアパッケージ「IcoAtmosBenchmark v1」を用いる。本パッケージには、６つの演算カーネル（水平拡散、水平発散、鉛直三重対角行列演算、水平移流フラックス計算、水平移流制限項計算、鉛直移流制限項計算）とステンシル計算に用いるメトリクス計算カーネル、さらにプロセス間通信カーネルが含まれている。

〈計算規模設定〉
気候・気象分野での科学的なマイルストーンとして、現在各国の研究機関は2030年に全球1km程度の水平解像度での気候シミュレーションを、24時間で1年分計算可能（1 Simulation Year Per wall-clock Day [SYPD]）にすることを目指している。本研究においても、同様の目標設定が妥当であると考える。NICAMはスーパーコンピュータ「京」の1/4系を用いて、世界で初めて全球870mメッシュでの気象計算を実現した。この時の計算速度は0.0006SYPDに相当した。現在、富岳の1/4系を用いて同じ問題サイズは0.024SYPDで計算可能であり、およそ40倍の高速化を実現している。これを踏まえ、日本の次世代フラッグシップマシンで再び40倍の高速化を実現可能であれば、1SYPDを達成可能である。ただし、「京」から「富岳」への高速化はハードウェアの性能向上だけで実現したものではなく、一部はアプリケーションの改良（特に、混合精度演算の積極的な利用）に起因している。次世代計算基盤においても、ハードウェアの性能向上のみではなく、ソフトウェアの改良を含めて、上記の実現目標を達成するものと想定する。

プログラミングモデルやソフトウェア再実装のコストを度外視して、今回検討するアーキテクチャへの

最適化が理想的（少なくとも現行のアプリケーション並み）に完了できると仮定して、その演算性能を引き出すために重要な要素としては３点が挙げられる。一つ目は大きな並列度の確保である。演算加速機構はL2、L1、演算ユニットと階層構造になっており、それらを使い切るためにはSIMDで動作する十分に大きな並列度を確保することが必要であり、NICAMではデータ構造の最内に配置される水平格子数を並列度の指標としてみることができる。二つ目の要素は演算加速機構が主に用いる高速メモリの容量であり、これは一つ目の要素とも密接に関連している。１時刻ステップをできる限りに高速に計算するという命題において、ホスト側の大容量メモリへのアクセスは無視できないオーバーヘッドとなる。最近のGPUを用いた計算においても、課題は同一であり、GPU上のメモリで全ての演算を完結させてはじめて、アーキテクチャの性能に見合う計算速度を得ることが出来る。高速メモリの容量によって、確保可能な並列度は制限を受ける。３つ目の要素は高速メモリの転送速度である。一般的に速いメモリはコストが高く容量が限られており、演算加速器用に大容量の高速メモリを搭載しようとすると、メモリ転送性能が犠牲になる。演算密度が低い気象・気候アプリケーションの演算性能は、ほぼメモリ転送性能によって決定される。そのため、並列度＝メモリサイズとメモリ転送性能のトレードオフの中で、性能要求を考える必要がある。

表9に、NICAMを870mメッシュで計算する場合の、分割数と要求されるマシン性能についてまとめる。NICAMは空間領域を水平方向に二次元タイル分割することでプロセス並列を実現しており、分割数によって水平方向のタイルあたり格子数が変動する。表より、想定するメモリ容量に収まるのは10000タイル以上に分割したケースであるが、40000タイル以上に分割すると必要とする並列度の1/4程度しか確保できないということがわかった。十分な並列度とメモリ容量制限をクリアする設定では、要求されるメモリ転送性能が極端に高く（富岳のCMGあたりメモリ転送性能の500倍以上）、現実的ではない。このことからも、ハードウェア性能の向上のみで目標とする計算性能を達成するのは難しいことが明らかになった。

ノード間通信性能及びファイルI/Oの見積もりについても検討を行った。現行のシミュレーションでは、通信、ファイルI/O共に総計算時間の5%程度に抑えるような実験設定で計算を行っており、ストロングスケーリングによって利用するプロセス数を増加させると、それらの過程に要する経過時間の比率は大きくなる。本検討では経過時間を10%まで許容したが、並列度として最適なプロセス分割数（10240並列）を選んだ場合には、必要となる通信性能は60GB/s、ファイルI/O速度は430MB/sと、それぞれ富岳の2.2倍、14倍となる。

表9：ターゲット問題のプロセス並列数の違いによる要求性能のまとめ

分割タイル数	タイルあたり水平格子数	袖領域格子数	鉛直層数	時間刻み（力学コア）	タイルあたりメモリ消費量	時間刻みあたり必要経過時間[1]	ファイルI/O頻度	必要となるメモリ転送速度[2]	必要となる通信速度[3]	必要なファイルI/O速度[4]
2560	264196	2052 (0.8%)	78	2sec	63.0GiB	0.0055sec	every 10sec	573TB/s	119GB/s	1708MB/s/die
10240	66564	1028 (1.5%)	78	2sec	15.9GiB	0.0055sec	every 10sec	143TB/s	60GB/s	430MB/s/die
40960	16900	516 (3.1%)	78	2sec	4.0GiB	0.0055sec	every 10sec	36TB/s	30GB/s	109MB/s/die
163840	4356	260 (6.0%)	78	2sec	1.0GiB	0.0055sec	every 10sec	9TB/s	15GB/s	28MB/s/die

1) 870mメッシュ全球シミュレーションを24時間で1年分計算する(1 SYPD)際に必要となる計算速度。
2) 仮定：シミュレーション全体の平均としてピークメモリ転送性能の10%で実行可能であること。
3) 要件：１回の時間ステップの中で、4ノードへのバイセクション通信を50回行う経過時間が、総経過時間の10%に収まること。
4) 要件：シミュレーション内の１時間ごとに、三次元変数20個を出力する経過時間が、総経過時間の10%に収まること。

〈国内外のアプリケーション事例調査〉
本研究では、気象・気候アプリケーションの演算加速器に最適化した研究成果についてのサーベイを行った。また、機械学習を用いて既存のアルゴリズムを代替するような研究成果についてのサーベイも加えて行った。機械学習によって置き換えられるモデルコンポーネントは、元の物理法則に基づくアルゴリズムとは独立した演算パターンを有し、演算加速機との親和性が高いことが期待される。

*気象・気候シミュレーションモデルのアクセラレータ移植状況に関する調査
NVIDIA GPUについては、以下の５件の研究に関するサーベイを行った。1)欧州の領域気象モデルCOSMOの

GPU最適化に関しては、力学コアをC++テンプレートフレームワークであるstella、物理過程をOpenACC利用によってオフロードし、スイススーパーコンピューティングセンターの所有するPiz Daint(Cray XC30)上で性能ベンチマーク結果が得られている。2)米国次世代全球大気モデルMPASの雲微物理スキームは領域気象モデルWRFと共有されており、このスキームをOpenACC利用によってオフロードした成果について報告されている。3)米国次世代気候モデルE3SMに関して、高解像度の二次元領域雲解像モデルを解像度の粗い全球モデルの各格子に埋め込んだE3SM-MMFシステムのGPU最適化が報告されている。雲解像モデル部分をOpenACC利用によってオフロードし、オークリッジ国立研究所の所有するsummitでの性能ベンチマーク結果が得られている。4)欧州の全球大気化学モデルEMACの大気化学反応モジュールのGPUオフローディングの成果について報告されている。この研究ではPythonで記述されたSource-to-source parserを用いて、Fortranの元コードからCUDAのコードを生成するという手法をとっており、想定した性能向上率の80%を達成している。5)米国のまた別の非静力大気モデルであるNIMについて、OpenACC利用によってオフロードし性能評価を行った成果が報告されている。

AMD GPUについては、あまりこのGPUに特化して最適化した成果について公開された文書に乏しい。ただし、米国ではkokkosフレームワークを用いた全球気象・気候モデルの書き換えが進行しており、これによってAMD GPUへの対応を果たしていると考えられる。また、前述のMPASの力学コアは、Cray社の提供するCray FortranでのOpenMP offloading機能を用いて、オフローディングを行うことが可能であった。

*気象・気候シミュレーションモデルでのAI技術利用状況に関する調査
以下の３種類のAI技術利用を対象に調査を行った。

① 予測シミュレーション全体をデータ駆動モデルで置き換えたもの
② シミュレーション結果から気象現象を特定して抽出するもの
③ シミュレーションモデルの構成要素をデータ駆動モデルで置き換えたもの

上記①の予測シミュレーション全体をデータ駆動モデルで置き換えるものについては、解像度の低い気候モデルの結果や高解像度モデルの結果を訓練データに使用しているも多く見られた。再解析データ(ERA5)を学習に使用しているが、観測データ(例えば衛星データ)を使用したものは見当たらなかった。上記②の気象現象を特定して抽出するものについては、高解像度気候モデルの出力から、機械学習を使用して、熱帯低気圧（TC）と大気の川（AR）をセグメンテーションするものが多く見られた。また、ラベルデータを必要とする機械学習手法が使われており、データへのラベル付け自体も課題があると感じられた。上記③のシミュレーションモデルの構成要素をデータ駆動モデルで置き換えたケースが、本FS研究課題において最も検討すべきものである。このケースに該当する既往研究には、雲プロセス(気候モデルに埋め込まれたCRMにおけるSuper Parametrizationなど)を置き換えるものが多く見られた。

⑬ 地震と構造物シミュレーションアプリケーションの検討
(再委託先 国立研究開発法人 海洋研究開発機構)
地震と構造物シミュレーションについて想定するアプリケーションを地震動計算および地盤破壊計算と定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/Oの必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにした。具体的には、地震アプリのベースとなる低次非構造有限要素と地盤アプリの個別要素法のカーネル部分を対象として、実機で想定される加速処理やデバイスあたりの問題サイズなどを検討し、GPU実装での経験をもとに有限要素法におけるelement by element法の効率的な実装や、個別要素法における条件分岐のマスク処理などで演算性能を強化しつつtime_to_solutionを改善するというアルゴリズム開発の方針を定めた。また、これらを実施するための疎行列ループの切り出しを行い、ターゲットとなるカーネルコードの整備を進めた。これらの実機での実行は、次年度以降に、現在開発中のOpenCLやOpenACCを用いて行う予定である。

⑭ ものづくりアプリケーションの検討
(再委託先 東洋大学)
ものづくりについて想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/Oの必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにした。具体的には、

ものづくり分野のソフトウェアであるADVENTUREによる高速アクセラレータMN-Coreの評価を行った。既存のコード資産を有効活用、かつ低コストな新規コードの開発を方針とし、ADVENTUREをライブラリ化、PyTorchでアプリを開発し、ソルバーをMN-Coreで実行した。C言語約5万行で開発されているADVENTUREの一部機能をライブラリ化し、数十行のPyTorchコードによりctypes経由で利用した。ADVENTUREデータの読み込みと有限要素法による線形方程式の組み立てまではADVENTUREで実施、COO形式行列を係数とする方程式の求解をPyTorchコードで実施し、PyTorchコード部分をMN-Coreで実行可能とした。非構造格子・小規模問題の実行時間について、CPU(Xeon Platinum 8260M,8threads)において21.502 [sec]、MN-Core(offloading実行)において10.124[sec]と、CPUと同程度の速度で計算できることを確認した。

⑮ マテリアルサイエンス応用アプリケーションの検討

(再委託先 国立大学法人 東海国立大学機構 名古屋大学)

マテリアルサイエンス応用での重要なアプローチは、マテリアル構成要素である原子核と電子がしたがう方程式を定量的に解いて現象を解明・予測すること、すなわち量子論に基づく多体の基本方程式を高速・高精度で解くことである。そのための理論的ベースは、現時点では密度汎関数理論 (Density Functional Theory: DFT) であり、次世代スーパーコンピューター上でのマテリアルサイエンスの推進においても、DFTの重要性は低下することはないだろう。

DFT計算が可能なアプリケーション・コードには大別して平面波基底を用いるものと実空間格子を用いるものがあるが、本委託研究においては、後者の範疇に属し、課題メンバーが開発・高度化・高速化を行ってきたRSDFTコード (Real-Space DFTコード) をターゲット・アプリケーションと定めた。その理由は、基本的に超並列アーキテクチャである今日のコンピュータ上では、実空間と逆格子空間を行き来するアルゴリズム (高速フーリエ変換) が不可欠である平面波基底法に比して、実空間のみで計算が実行できるRSDFTコードには原理的な優位性があるためである。それは2011年度Gordon Bell賞 (peak performance) 受賞でも実証されている。

「富岳」コンピュータ上でのRSDFTコードの実行により、その重要なカーネルは、部分空間反復法 (Rayleigh-Ritz法) による外部固有値問題の求解部分であり、特に問題サイズが大きい場合 (10万原子規模) は、Gram-Schmidt直交化および部分空間対角化がボトルネックとなり、いずれもターゲット物質のサイズNの3乗に比例する計算コストを要することが明らかになっている。ただしボトルネック部は行列積演算が主となるため、計算全体でも40～50%の高い実行効率を達成することが可能である (10000ノード使用時)。一方、小～中規模サイズ (1万原子以下) では差分行列および非局所擬ポテンシャルと呼ばれるランク1の行列の演算部分が、N3のボトルネック部に比べ無視できない計算時間を要するようになり、それに伴って実行効率は下がる。RSDFTの発展形であり、マテリアルサイエンスにおいて重要なMolecular Dynamics (MD) 計算をDFTの枠内で実行することも可能となっており (コード名:RS-CPMD,Real-Space Car-Parrinello Molecular Dynamics)、このRS-CPMDコードでの重要カーネルは、直交化に対応するラグランジュ乗数の計算であり、システムサイズ3乗に比例する計算コストとなる。また一般的にMDは時間方向になるべく長くシミュレーションを行う必要があり、１ステップに要する計算時間を抑えるために系のサイズを大きく取れない。そのために、高効率で実行可能なN3のボトルネック部の計算と、低効率の差分および非局所ポテンシャルの行列演算の計算時間が同程度となり、全体的な実行効率も下がる傾向にあることが明らかとなってきた。

ノード間通信は、N3ボトルネックに伴うAllreduceと、差分および非局所擬ポテンシャル行列演算に伴う隣接通信が主となる。並列化は空間および固有ベクトルの軸について行うことが可能で、大規模系では両者の軸についての並列化を組み合わせた場合、計算時間全体の30%ほどが通信時間となることがわかっている。小～中規模系では隣接通信のコストが主となるが、この部分のコスト削減については今後の検討課題となっている。I/Oについては、計算の最後に、大きくても使用メモリサイズ相当のデータを書き出すのみで、現在のターゲットサイズ、10000-100000原子規模計算では、それほどの重大な問題は引き起こさないことがわかっている。

RSDFTコードを加速装置 (GPU) に対応させる試みも行った。Si8原子の系で仮想的にグリッドサイズを1603にしたモデルケースにおいて、A100のGPU４台を搭載した計算機１ノード (4MPIプロセス) でテストを

行ったところ、GPUを用いない場合、反復計算1ステップで14秒を要する計算が、GPUを使用すると、80秒になるという結果を得た。しかしながら、RSDFTには元々、差分および非局所擬ポテンシャルの通信を束ね、通信回数を削減する機能が実装されており、この機能を有効にしてGPUを使用すると１ステップ11秒という結果となった。本来MPI通信回数の削減が目的で実装された機能であるが、それがそのままGPUオフロードの効率化に利用できることがわかった。問題サイズや並列化との組み合わせにより加速装置がどの程度有効かを見るのは今後の課題となる。

⑯ 素粒子・原子核物理応用アプリケーションの検討

(再委託先 国立大学法人 広島大学)

素粒子・原子核物理について想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/Oの必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにした。具体的には、素粒子分野の格子量子色力学（格子QCD）アプリケーションの中で特に計算時間のかかるクォーク伝搬関数計算プログラムの性能推定を行った。クォーク伝搬関数計算プログラムでは、クォーク運動方程式を差分化した大規模疎行列係数を持つ連立方程式を反復法(BiCGStab法)により解いている。また、このプログラムの計算カーネルの大規模疎行列ベクトル積に対応する４次元格子上のステンシル計算部分を抽出した。

格子QCDのターゲット問題の格子サイズ512^4格子をMN-Coreのような加速装置で実行することを見据えて、1加速装置あたりの格子サイズを$16^3 \times 32$とおいて、クォーク伝搬関数計算プログラムの性能を机上見積もりした。倍精度で反復法を2000反復行ったときの計算時間として4.1秒と見積もった。同様の計算をGPU(NVIDA RTX A2000 12GB)を用いて単精度で行った場合の実測時間は5.12秒であった。机上見積もりではメモリバンド幅が律速となっており、GPUのメモリバンド幅とMN-Coreのメモリバンド幅の比較から妥当な机上評価であるとみている。

机上見積もりの精度を上げるため実機での測定を目指した。格子QCDステンシル計算カーネルのMN-Core上での動作確認のためにPython/PyTorch移植を行った。また、類似ステンシルアプリケーションである姫野ベンチの移植も行った。クォーク伝搬関数計算プログラムステンシル計算カーネルと姫野ベンチは、ともにMN-Core上での実行ができた。今後実機上での計測と机上見積もりの比較を行うとともに並列計算時のボトルネックの精査を行う予定である。

⑰ 宇宙・惑星科学応用アプリケーションの検討

(再委託先 国立大学法人 東京大学)

宇宙・惑星科学について想定するアプリケーションを定め、既存のアルゴリズムでの演算カーネルの抽出、ノード間通信、I/Oの必要量の推定を行い、新しいアルゴリズムの検討の方向を明らかにした。具体的には、銀河形成シミュレーションコード「ASURA-FDPS」のベンチマークを行い、必要な演算量を見積もり、それによって可能となるサイエンスの検討を行った。また、ハミルトニアン分割を行う新しいアルゴリズムの検討、機械学習の活用による新しいアルゴリズムの検討、一部のアルゴリズムについてはそのプロトタイプの開発を行った。さらに、FDPSについて各部分の詳細な実行プロファイルの計測を「富岳」上で行い、アクセラレータシステムでのホストとアクセラレータの演算量の比、ホスト側の必要な演算やメモリアクセス能力の推定を始めた。

以下に実行プロファイルの一例を示す(表10)。これは、200万粒子、富岳1024ノードでの計測例である。1ノードあたりの粒子が極端に少ないため、典型的なものではない、むしろ、強スケーリングを試みた時にどのようなボトルネックが現れるかを検討した例である。

表 10 FDPS の実行プロファイル。富岳 1024 ノード、200 万粒子の例

全体	7.2ms
領域分割	0.41
粒子交換	0.78
相互作用計算	5.8
ツリー構築	1
LET構築	1.6
LET交換	0.4
ツリー探索	0.44
相互作用カーネル	1.57

粒子数が多いと、ノード間並列化のために必要な領域分割、粒子交換、LET構築、交換の計算時間は相対的に小さくなるので、ツリー構築、ツリー探索と実際の相互作用カーネルが残ることがわかる。

富岳では、ツリー構築に非常に時間がかかっていることがわかる。これは、この実装の段階では富岳上に高速な並列化ソートライブラリがなかったためである。ソート自体は、SIMD命令を使ったバイトニックソート、SIMD命令を使ったクィックソート、並列化したサンプルソートを組み合わせた高速なライブラリを開発し、性能評価中である。

FDPSはある程度のネットワーク性能があれば粒子数が大きい場合には並列化効率は大きな問題にならないので、今後1ノード性能の精密な評価を進める。

⑱プロジェクトの総合的推進

プロジェクト全体の連携を密としつつ円滑に運営していくため、運営委員会や技術検討会の開催等、参画各機関間の調整を行った。

２－３．活動（研究会の活動等）

①ミーティングの実施

研究開発課題責任者、コーディネーター、各分担先代表者を構成メンバーとするミーティングを設置し、開催した。

- アプリケーショングループ+全体ミーティング
  8 月 19 日、9 月 2 日、10 月 6 日、10 月 31 日、12 月 1 日、12 月 23 日、1 月 6 日、1 月 20 日、2 月 3 日、2 月 17 日、3 月 3 日、3 月 17 日、3 月 31 日
- アーキテクチャグループミーティング(隔週木曜)
- システムソフトウェアグループミーティング(2022 年度は不定期)
- Himeno-bench Meeting（毎週火曜）

②CPS セミナー「MN-Core 向け相互作用計算カーネルコンパイラ PIKG/MN-Core」10 月 28 日 オンライン

③High Performance Computing Physics(HPC-Phys)勉強会

2 月 8 日　神戸大学 惑星科学研究センター セミナー室、サロン

④次世代計算基盤に係る調査研究に関する合同ワークショップ～次世代高性能計算基盤の開発に向けて～

2 月 22 日　コングレスクエア日本橋 2 階 コンベンションホール A・B

２－４．実施体制

業　務　項　目	実　施　場　所	担　当　責　任　者
①アーキテクチャ調査研究	兵庫県神戸市灘区六甲台町1番1号	理学研究科　教授　牧野 淳一郎
②システムソフトウェア・ライブラリ調査研究	兵庫県神戸市灘区六甲台町1番1号 国立大学法人神戸大学	理学研究科　教授　牧野 淳一郎
③アプリケーション調査研究	兵庫県神戸市灘区六甲台町1番1号 国立大学法人神戸大学	理学研究科　教授　牧野 淳一郎
④アーキテクチャ調査研究の統括およびアクセラレータ評価	兵庫県神戸市灘区六甲台町1番1号 国立大学法人神戸大学	理学研究科　教授　牧野 淳一郎
⑤CPU評価	東京都文京区本郷七丁目３番１号 国立大学法人東京大学	大学院情報理工学系研究科 准教授　塩谷 亮太
⑥ネットワークアーキテクチャ評価	東京都立川市緑町１０番３号 大学共同利用機関法人情報・システム研究機構 国立情報学研究所	アーキテクチャ科学研究系 教授　鯉渕 道紘
⑦システムソフトウェア・ライブラリ調査研究の統括およびコンパイラ検討	福島県会津若松市一箕町大字鶴賀字上居合９０番地 公立大学法人会津大学	コンピュータ理工学部 教授　中里 直人
⑧DSL検討	島根県松江市西生馬町１４－４ 独立行政法人国立高等専門学校機構 松江工業高等専門学校	情報工学科　講師　岩澤 全規
⑨アプリケーション調査研究の統括	東京都文京区本郷二丁目１番１号 順天堂大学	健康データサイエンス学部開設 準備室　特任教授　姫野 龍太郎
⑩創薬と深層学習応用アプリケーションの検討	東京都千代田区大手町一丁目６番１号　大手町ビル２階 株式会社Preferred Networks	代表取締役　最高研究責任者 岡野原 大輔
⑪ゲノム科学アプリケーションの検討	東京都文京区本郷七丁目３番１号 国立大学法人東京大学	大学院新領域創成科学研究科 教授　鎌谷 洋一郎
⑫気象・気候シミュレーションアプリケーションの検討	茨城県つくば市小野川１６番地２ 国立研究開発法人国立環境研究所	地球システム領域 主任研究員　八代 尚
⑬地震と構造物シミュレーションアプリケーションの検討	神奈川県横須賀市夏島町２番地１５ 国立研究開発法人海洋研究開発機構	付加価値情報創生部門 部門長　堀 宗朗
⑭ものづくりアプリケーションの検討	東京都文京区白山五丁目２８番２０号　東洋大学	総合情報学部　教授　塩谷 隆二
⑮マテリアルサイエンス応用アプリケーションの検討	愛知県名古屋市千種区不老町１番 国立大学法人東海国立大学機構 名古屋大学	未来材料・システム研究所 特任教授　押山 淳
⑯素粒子・原子核物理応用アプリケーションの検討	広島県東広島市鏡山一丁目３番２号 国立大学法人広島大学	大学院先進理工系科学研究科 准教授　石川 健一
⑰宇宙・惑星科学応用アプリケーションの検討	東京都文京区本郷七丁目３番１号 国立大学法人東京大学	大学院理学系研究科 准教授　藤井 通子
⑱プロジェクトの総合的推進	兵庫県神戸市灘区六甲台町1番1号 国立大学法人神戸大学	理学研究科　教授　牧野 淳一郎

様式第２１

学　会　等　発　表　実　績

委託業務題目「次世代計算基盤に係る調査研究」（システム研究調査チーム）

機関名　国立大学法人神戸大学

１．学会等における口頭・ポスター発表

発表した成果（発表題目、口頭・ポスター発表の別）	発表者氏名	発表した場所（学会等名）	発表した時期	国内・外の別
「研究報告Ⅱシステム研究調査の概要と検討状況」（口頭発表）	牧野 淳一郎	コングレスクエア日本橋2階コンベンションホールA・B、オンライン（次世代計算基盤に係る調査研究に関する合同ワークショップ～次世代高性能計算基盤の開発に向けて～）	2023年2月22日	国内

２．学会誌・雑誌等における論文掲載

掲載した論文（発表題目）	発表者氏名	発表した場所（学会誌・雑誌等名）	発表した時期	国内・外の別
なし				