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

令和　元 年　 ６ 月　 ７ 日現在

機関番号：１７１０４
研究種目：基盤研究(B)（一般）
研究期間：2014～2018
課題番号：２６２８００２３

研究課題名（和文）計算状態の精密操作に基づく高性能・高信頼システム技術

研究課題名（英文）Technology for Efficient and Reliable Systems Based on Accurate Execution Stack Access

研究代表者
八杉　昌宏（Yasugi, Masahiro）

九州工業大学・大学院情報工学研究院・教授

研究者番号：３０２７３７５９

交付決定額（研究期間全体）：（直接経費）　　13,300,000円

研究成果の概要（和文）：コンピュータで計画的に計算を行っている最中に、将来の予定に含まれるデータをいざとなれば参照したり変更したりできる機構を備えとして用いて、複数CPUによる並列計算をうまく行ったり、安全性を高めるためのメンテナンスを行ったりする方式を実現しています。本研究では、これまでより性能や移植性のよい機構の実現を試み、「評価器」に応用したときに良好な結果を得ました。また、並列化が進む多様で複雑な計算環境を効率よく簡単・安全に利用できるように、本機構を活用した並列プログラミング言語の提案や改良を行いました。

研究成果の学術的意義や社会的意義
運用コストを改善したり実行性能を高めたりするために、計算環境の並列化・多様化・複雑化が進んでいます。本研究の成果を生かすことで、並列化が進む多様で複雑な計算環境を効率よく簡単・安全に利用できるようにします。また、そのための高水準プログラミング言語の実装を容易にして学術研究を促進します。

研究成果の概要（英文）：By using mechanisms for occasionally inspecting/modifying data for future plans during the planned execution on computers, we can develop technologies for parallel computing and reliable/sustainable computing. We investigated more efficient and portable mechanisms and obtained a good result when applying a mechanism to evaluators. In addition, we proposed and improved parallel programming languages implemented with our mechanisms for exploiting highly parallel/heterogeneous/complex computing environments efficiently, easily, and safely.

研究分野：計算機科学

キーワード：プログラミング言語　計算機システム　ソフトウエア開発効率化・安定化　ディペンダブル・コンピューティング　ハイパフォーマンス・コンピューティング　入れ子関数　実行スタック　並列処理

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９、ＣＫ－１９（共通）

１．研究開始当初の背景

計算システムは十分な信頼性・性能・容量が求められるのに加えて、最近は消費電力量の削減も求められる。並列化率向上によりクロック周波数向上と同じ性能が得られれば消費電力量の大幅な削減が可能となるため、マルチコア、計算機クラスタ、クラウド計算など、さまざまなレベルで並列システムの提供が拡大している。加えて、仮想化・階層化・ヘテロ化などで多様化・複雑化する計算システムを、そのままの形で効率よく簡単・安全に利用することは困難であり、適切に抽象化されたプログラミングインタフェースが求められている。そのようなプログラミングインタフェースを提供する高信頼・高性能プログラミング言語の処理系を実装するには、しばしば、「ごみ集め」など計算中のソフトウェアの動的再構成・保全の機能が必要となる。

我々は、L-closure と呼ぶ安全な計算状態操作機構の研究で、拡張 C 言語の仕様として入れ子関数(クロージャ)を利用して呼出し元に眠る変数の値への安全で正式・精密なアクセスを可能とし、負荷分散、ごみ集め、マイグレーション、チェックポインティングなど、計算中のソフトウェアの動的再構成や保全を記述可能とした。実装技術としては、初期化や保存を呼出しまで遅延させることで、クロージャ生成コスト削減や、変数へのレジスタ割当てを可能とし、提案機構の追加を意識させない高い実行性能を得た。これに基づき、アセンブリ言語レベルの技術を用いた実装と、標準 C 言語への翻訳方式での実装を開発済である。

実際に利用できるメモリの不足、人に起因する誤りの他、近年のセキュリティ侵害、停電といった不確実性があるなかで、高性能を犠牲にしない計算状態の保全技術の重要性は高まっている。また、今後、低消費電力への要求によるハードウェアの微細化・低電圧化が進み不確実性が高まるにつれ、本研究の重要性は増大していく。

２．研究の目的

本研究では、これまでの研究成果を発展させ、高性能・高信頼システムのための計算状態操作機構の応用技術に磨きをかけるとともに、設計・実装面から同機構を確立・改良することを目的とする。

より具体的には、ソフトウェアの動的再構成・保全の面からは、ごみ集めや一級継続がサポートされた高水準言語の実装、冗長実行や負荷分散やマイグレーションにより仮想化･階層化･ヘテロ化なども想定して並列システムにおける資源を有効活用する手法の開発、などがあり、設計・実装面からは、新しい処理系や言語仕様の調査・比較やベースとしての活用、計算状態操作機構を持つ拡張 C 言語の言語仕様や実装・性能モデルの改良、などが挙げられる。

これにより、仮想化・並列化・階層化・ヘテロ化などでますます多様化・複雑化する計算システムを、抽象化されたプログラミングインタフェースにより効率よく簡単・安全に利用することを可能とする。

３．研究の方法

本研究においては、高水準言語の実行ステップ（セマンティクス）に関する一貫性・頑健性・時間効率・空間効率を冗長性・後戻り・再実行・予測近似といった手段により実現したい。その際に、「隠された計算状態」が存在すると抜本的再構成・保全が不可能となるが、そのような「隠された計算状態」の存在を防ぐために計算状態操作機構を利用する。その結果、ソフトウェアの動的再構成・保全を行うという考え方に基づく活用・応用としての多様な実装技術に取り組むことができる。

研究代表者は全体設計、個別の設計ならびに処理系実装・改善、応用研究も行った。学外研究分担者（平石）は主に拡張 C 言語の翻訳ベース処理系ならびに高水準言語の処理系の実装と評価を分担した。学内研究分担者１名（光来）は仮想化に関連する課題等を分担した。研究協力者（小出）は処理系実装・改善の課題等について協力した。学内研究協力者（江本）は並列プログラミングや高水準言語の設計・実装について協力した。

(1) 計算状態操作機構を持つ拡張 C 言語の言語仕様や実装・性能モデルの改良

GCC 4 以降（や LLVM）では、RTL レベル以降のコンパイルフェーズの改造により L-closure を実装することは困難になった。これは、RTL レベルより前のコンパイルフェーズにおいてクロージャがアクセスする変数を構造体のフィールドに変換してしまうためである。このため、変換において 2 つの場所（通常のレジスタ割り当て可能な変数と明示的スタック上の場所）を導入する変換ベースの L-closure の実装も開発してきた。

この方向で研究成果を発展させ、変換先に例外処理機能を持つ言語を利用することでさらに高性能を達成しようと試みた。将来的には C 言語の新たな拡張方式の研究も視野に入れつつ、まずは Java 言語や C++言語などの既存の言語・処理系を用いての試験実装を検討した。本研究では、まずは Java 言語を対象として変換ベースの試験実装を進めた。

また、拡張 C 言語の変換ベースの実装において L-closure よりも中庸的な計算状態操作機構 closure の変換ベースの実装も行うこととした。

(2) 並列システムにおける資源を有効活用する手法

① 基礎的研究としては、並列計算のための理論と言語、仮想環境における並列性能の研究を行

った。
② バックトラックに基づく負荷分散は、本研究課題で基軸としている計算状態操作機構を用いて高性能実装されている。この研究を発展させ、仮想化・階層化・ヘテロ化なども想定して並列システムにおける資源を有効活用する手法の開発を進めた。
・仮想環境への対応として、例えば、利用可能な本物のコア数（物理 CPU 数）が減った場合の振る舞いや対策についての研究を行った。
・ヘテロ化への対応として、Intel Xeon Phi に代表される MIC（Many Integrated Core）アーキテクチャを対象に実装を行った。
・多数のコアを持つプロセッサにおいてボトルネックを生じさせずにその資源を有効活用するための研究の他、例外処理などを総合的に追求した。
③ 階層的計算省略に基づく並列実行モデルは、急所(single point of failure; 単一障害点)を持たない並列実行方式であり、上記のバックトラックに基づく負荷分散と同様に計算状態操作機構による実行時の再構成を利用した処理系として開発を進めた。
・複数のプロセスがメッセージ交換をしながらある程度独立して計算を進めるモデルを想定しており、その特性を活用した新たなメッセージ媒介システムの構成が必要となっている。その実装を進め、新しい実行方式のコンセプトの検証を進めた。
・高水準プログラミング言語の設計と処理系実装を行うとともに、メッセージ媒介システムのスケーラブルな実装も行い、高可用かつ高性能である点について確認するとともに、MIC（Many Integrated Core）アーキテクチャも対象に実装を行った。
④ 評価用アプリケーション・実アプリケーションの開発
バックトラックに基づく負荷分散も、階層的計算省略に基づく並列実行方式も、不規則な並列アプリケーションを扱うことができ、並列化が従来困難な探索問題を含めた、評価用アプリケーション・実アプリケーションの開発を進めた。

(3) ごみ集めや一級継続がサポートされた高水準言語の実装
Lisp インタープリタ（正確に言えば Scheme 言語のインタープリタ）を対象に、変換ベースの計算状態操作機構によるごみ集めや一級継続のサポートを行い、(1)で追加した計算状態操作機構 closure の変換ベースの実装を含めた性能評価を進めた。

４．研究成果
(1) 計算状態操作機構を持つ拡張 C 言語の言語仕様や実装・性能モデルの改良
① 本研究では、Java 言語を対象として例外処理を活かした変換ベースの試験実装を進めた。計算状態操作機構を備えた拡張 Java 言語を設計するとともに、試験実装として、変換の具体例を作成した。ただし、変換を支えるための研究は当初の予想よりも奥が深く、S 式ベースの拡張可能 Java 言語処理系を実装するとともに、「S 式ベース言語処理系」に部分列に関する条件が扱えるようなパターンを考案・追加した上で、新しいパターンを用いた S 式ベースの Java 言語処理系の改良も行った。また、型情報追加変換についても整備を進めた。以上により、Java ベースという形で、例外処理を活かした計算状態操作機構の実現準備がほぼ整えられた。
② 本研究では新規に、L-closure よりも中庸的な計算状態操作機構 closure の変換ベースの実装を行った（雑誌論文①）。拡張 C 言語（あるいは拡張した S 式ベース C 言語）において、計算状態操作機構 closure は入れ子関数定義から生成される標準的なクロージャである。closure では通常考えられる程度の生成コストと維持コストを要するが、呼出しコストは通常の関数と同程度である。さらにいえば、本研究以前に開発した変換ベースの L-closure 実装において初期化や明示的スタックへの値の保存を遅延すべきかを判定するコスト（遅延判定コスト）が比較的高いことが本研究では新たに分かった（学会発表 6）ため、遅延判定コストを持たない変換方式として変換に基づく closure に注目することになった。
　拡張した S 式ベース C 言語から標準 C 言語への変換による移植性に優れた実装が本研究の成果として得られた。ここでは、本研究以前に行った（GCC 4 をベースとした）拡張 C コンパイラによる実装の基本的考え方を踏まえた。具体的には、入れ子関数からアクセスする変数を closure の環境を成す構造体のフィールドに変換する。実行再開のために基本的にはすべての変数のデータを明示的スタックに（から）退避（復元）するようにする L-closure 用の変換とは異なり、closure 用の変換では、入れ子関数からアクセスされない変数は変換対象としない。また、フィールドへの変換後は、仮引数を除き変数自体は残らない。標準 C 言語への変換は、拡張 C コンパイラとしての既存実装よりも移植性に優れた実装であるとともに、標準 C 言語からのコンパイル時の最適化がよい効果を持つ可能性もある。

(2) 並列システムにおける資源を有効活用する手法
① 基礎的研究としては、ドメイン特化型並列計算向け高水準言語（雑誌論文③、学会発表 18,25）、並列計算量の理論的扱い（学会発表 12）、信頼性（学会発表 19,27）、仮想環境における並列性能の最適化／性能確保方式（学会発表 3,4,10,11,20,28,33）に関する成果を得た。
② バックトラックに基づく負荷分散に関しては、以下の研究成果を得た。ワークスティールがない限り各ワーカは論理的なスレッドさえも生成しない（必要なときのみ真のタスク生成）という手法（計算状態操作機構を用いて実現）は国内外で唯一といえるもので、国内学会で研究

賞を得た学会発表を含め注目されている。
・多数のコアを持つプロセッサを有効活用するために確率的に局所性を高める方式を開発し、性能解析、改良、評価した。例えば、ワークスティールの成功確率を設定可能とした確率的ガードと仮想確率的ガードを提案し（学会発表 21)、比較的大きな作業空間を要するアプリケーションも用いて分散環境でも評価した（雑誌論文②)。また、仕事量の見積りというヒントを優先度または重みとして用いる方式を考案し有効性評価を行った（学会発表 16)。
・MPI を利用した処理系の開発ならびに大規模環境での評価（Intel Xeon Phi や京スーパーコンピュータで実行）を行った（学会発表 22)。また、MPI を利用した処理系の改良を続けた（学会発表 8,23)。
・タスク並列言語用に中断処理を伴う例外処理方式を提案した（雑誌論文⑤、学会発表 23)。これに基づき並列探索の動的な枝刈りを例外処理により行う処理系／手法を開発した（学会発表 24,31、このうち学会発表 31 は研究賞受賞）他、分散環境への対応を行った（雑誌論文④、学会発表 14)。
・仮想化における計算環境の変化に関する調査し、仮想環境を考慮した負荷分散方式の評価（学会発表 13）と Cilk との性能比較を行った。実環境向け高速化手法であるビジーループを実環境ならびに仮想環境において評価した。
・階層行列構築アプリケーションも開発した（学会発表 1、研究賞受賞)。
③ 階層的計算省略に基づく並列実行モデルについては以下の成果を得た。本モデルは耐障害性を最初から備えた独特な並列計算モデルであり、今後さらなる成果が期待される
・階層的計算省略に基づく並列実行モデル HOPE を提案した（学会発表 30)。
・このモデルでは、複数のプロセスがメッセージ交換しながら計算を進めるが、これを支えるスケーラブルなメッセージ媒介システムの実装し、改良を行った。例えば、不要メッセージ削除機能を実現した（学会発表 29)。
・高水準プログラミング言語の設計と処理系実装を行い（学会発表 15)、通信障害模擬機能の実装を行って（学会発表 7)、これを用いて耐障害性評価を行った。

(3) ごみ集めや一級継続がサポートされた高水準言語の実装
計算状態操作機構の変換ベースの実装を利用可能な Lisp インタープリタにより、(1)で追加した変換ベースの closure の評価を行い、高い性能を確認した。同時に計算状態操作機構 L-closure の変換ベースの実装も含めて評価を行い、この場合、変換ベースの closure の性能が良いとの結果を得た（雑誌論文①)。

５．主な発表論文等

〔雑誌論文〕（計 5 件）

① Masahiro Yasugi, Reichi Ikeuchi, Tasuku Hiraishi, Tsuyeyasu Komiya. Evaluating Portable Mechanisms for Legitimate Execution Stack Access with a Scheme Interpreter in an Extended SC Language. Journal of Information Processing, 査読有, Vol. 27, pp. 177-189, 2019
DOI: 10.2197/ipsjjip.27.177

② Hiroshi Yoritaka, Ken Matsui, Masahiro Yasugi, Tasuku Hiraishi, Seiji Umatani. Probabilistic guards: A mechanism for increasing the granularity of work-stealing programs. Parallel Computing, 査読有, Vol. 82, pp. 19-36, 2019
DOI: 10.1016/j.parco.2018.06.003

③ Kento Emoto, Fumihisa Sadahira. A DSL for graph parallel programming with vertex subsets. The Journal of Supercomputing, 査読有, Vol. 75 (Online First), 2019
DOI: 10.1007/s11227-019-02821-w

④ Shingo Okuno, Tasuku Hiraishi, Hiroshi Nakashima, Masahiro Yasugi, Jun Sese. Parallelization of Extracting Connected Subgraphs with Common Itemsets in Distributed Memory Environments. Journal of Information Processing, 査読有, Vol. 25, pp. 256-267, 2017
DOI: 10.2197/ipsjjip.25.256

⑤ Tasuku Hiraishi, Shingo Okuno, Masahiro Yasugi. An Implementation of Exception Handling with Collateral Task Abortion. Journal of Information Processing, 査読有, Vol. 24, No. 2, pp. 439-449, 2016
DOI: 10.2197/ipsjjip.24.439

〔学会発表〕（計 52 件）

1. Zhengyang Bai, Tasuku Hiraishi, Hiroshi Nakashima, Masahiro Yasugi, Akihiro Ida. Parallelization of Matrix Partitioning in Construction of Hierarchical Matrices using Task Parallel Languages, The 3rd cross-disciplinary Workshop on Computing Systems, Infrastructures, and Programming (xSIG 2019), 2019
2. 八杉 昌宏, 平石 拓, 光来 健一. 移植性に優れた計算状態操作機構の評価. 第 10 回 自

動チューニング技術の現状と応用に関するシンポジウム，2018
3. 村岡 裕二, 光来 健一. VM 専用仮想メモリとの連携による VM マイグレーションの高速化. 情報処理学会第 144 回 OS 研究会 (SWoPP 2018)，2018
4. Masato Suetake, Takahiro Kashiwagi, Hazuki Kizu, Kenichi Kourai. S-memV: Split Migration of Large-memory Virtual Machines in IaaS Clouds. the 2018 IEEE International Conference on Cloud Computing (CLOUD 2018)，2018
5. Tasuku Hiraishi. Dynamic Load Balancing for Construction and Arithmetic of Hierarchical Matrices, SIAM Conference on Parallel Processing for Scientific Computing (SIAM PP18)，2018
6. 八杉 昌宏，池内 嶺知，平石 拓，小宮 常康，重本 孝太. 拡張 SC 言語で記述した Scheme インタプリタによる計算状態操作機構の評価. 日本ソフトウェア科学会プログラミング論研究会第 20 回プログラミングおよびプログラミング言語ワークショップ (PPL 2018)，2018
7. 西牟禮 亮，八杉 昌宏，平石 拓，馬谷 誠二. 並列分散フレームワークの耐障害性評価のための通信障害模擬機能. 日本ソフトウェア科学会プログラミング論研究会第 20 回プログラミングおよびプログラミング言語ワークショップ (PPL 2018) (カテゴリ 3，ポスター発表)，2018
8. 平石 拓，村岡 大輔，八杉 昌宏. タスク並列言語におけるノード間通信の実装方式の検討. 情報処理学会第 59 回プログラミング・シンポジウム，2018
9. 八杉 昌宏，平石 拓，光来 健一. 計算状態操作機構による並列言語実装と性能改善. 第 9 回 自動チューニング技術の現状と応用に関するシンポジウム，2017
10. Kenichi Kourai, Sungho Arai, Kousuke Nakamura, Seigo Okazaki, Shigeru Chiba. Resource Cages: A New Abstraction of the Hypervisor for Performance Isolation Considering IDS Offloading. the 9th IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2017)，2017
11. 髙山 都旬子，光来 健一. CPU オーバコミット時における並列アプリケーション実行の最適化. 第 29 回コンピュータシステムシンポジウム (ComSys 2017)，2017
12. 白水 駿，江本 健斗. 並列プログラム計算量の系統的機械証明手法の開発. 日本ソフトウェア科学会第 34 回大会，2017
13. 良本 海，八杉 昌宏，平石 拓，馬谷 誠二. 仮想環境を考慮した要求駆動型負荷分散. 日本ソフトウェア科学会第 34 回大会，2017
14. 奥野 伸吾，平石 拓，中島 浩，八杉 昌宏，瀬々 潤. 分散環境での並列グラフマイニングにおけるタスク中断処理による冗長探索削減. 情報処理学会第 115 回プログラミング研究会 (SWoPP2017)，2017
15. 重本 孝太，八杉 昌宏，平石 拓，馬谷 誠二. HOPE コンパイラのプロトタイプ実装. 情報処理学会第 115 回プログラミング研究会 (SWoPP2017)，2017
16. 寄高 啓司，八杉 昌宏，平石 拓，馬谷 誠二. 優先度ならびに重みを用いたワークスティールフレームワークの性能改善. The 1st cross-disciplinary Workshop on Computing Systems, Infrastructures, and Programming (xSIG 2017)，2017
17. 八杉 昌宏，平石 拓，光来 健一. 計算状態操作機構による並列言語実装と評価. 第 8 回 自動チューニング技術の現状と応用に関するシンポジウム. 2016
18. Kento Emoto, Kiminori Matsuzaki, Zhenjiang Hu, Akimasa Morihata, Hideya Iwasaki. Think Like a Vertex, Behave Like a Function! A Functional DSL for Vertex-Centric Big Graph Processing. the 21st ACM SIGPLAN International Conference on Functional Programming (ICFP 2016)，2016
19. 八杉 昌宏. 複数言語による正しい並列プログラミングのための計算基盤の検討. 日本ソフトウェア科学会第 33 回大会 (ポスター発表)，2016
20. 髙山 都旬子，光来 健一. VM が利用可能な CPU 数の変化に対応した並列アプリケーション実行の最適化. 日本ソフトウェア科学会第 33 回大会，2016
21. Hiroshi Yoritaka, Ken Matsui, Masahiro Yasugi, Tasuku Hiraishi, Seiji Umatani. Extending a Work-Stealing Framework with Probabilistic Guards. Ninth International Workshop on Parallel Programming Models and Systems Software for High-End Computing P2S2 2016 (held in conjunction with ICPP 2016)，2016
22. Daisuke Muraoka, Masahiro Yasugi, Tasuku Hiraishi, Seiji Umatani. Evaluation of an MPI-Based Implementation of the Tascell Task-Parallel Language on Massively Parallel Systems. Ninth International Workshop on Parallel Programming Models and Systems Software for High-End Computing P2S2 2016 (held in conjunction with ICPP 2016)，2016
23. Tasuku Hiraishi, Shingo Okuno, Daisuke Muraoka, Masahiro Yasugi. Exception Handling with Collateral Task Abortion in Distributed Memory Environments. HPC in Asia Poster session, ISC High Performance 2016，2016
24. Shingo Okuno, Tasuku Hiraishi, Hiroshi Nakashima, Masahiro Yasugi, Jun Sese. Reducing Redundant Search in Parallel Graph Mining using Exceptions. 21st International Workshop on High-Level Parallel Programming Models and Supportive

Environments HIPS 2016 (held in conjunction with IPDPS 2016), 2016
25. 江本 健斗, 松崎 公紀, 胡 振江, 森畑 明昌, 岩崎 英哉. 大規模グラフ並列処理のための関数型領域特化言語. 第 18 回プログラミングおよびプログラミング言語ワークショップ (PPL2016), 2016
26. 八杉 昌宏, 平石 拓, 光来 健一. 多様な計算環境/計算モデルにおける計算状態操作機構. 第 7 回 自動チューニング技術の現状と応用に関するシンポジウム, 2015
27. 八杉 昌宏. 安全な並列計算向け型検査方式の例題を用いた検討. 日本ソフトウェア科学会第 32 回大会 (ポスター発表). 2015
28. Kenichi Kourai, Riku Nakata. Analysis of the Impact of CPU Virtualization on Parallel Applications in Xen. The 13th IEEE International Symposium on Parallel and Distributed Processing with Applications, 2015
29. 諏訪 将大, 八杉 昌宏, 平石 拓, 馬谷 誠二. 分散進捗管理のためのメッセージ媒介システムにおける不要メッセージ削除機能. 情報処理学会第 105 回プログラミング研究会 (SWoPP2015). 2015
30. Masahiro Yasugi, Tasuku Hiraishi, Seiji Umatani. Towards a New Parallel Execution Model Based on Hierarchical Omission. Annual Meeting on Advanced Computing System and Infrastructure (ACSI) 2015
31. Shingo Okuno, Tasuku Hiraishi, Hiroshi Nakashima, Masahiro Yasugi, Jun Sese. Reducing Redundant Search using Exception Handling in a Task-Parallel Language. Annual Meeting on Advanced Computing System and Infrastructure (ACSI) 2015, 2015
32. 八杉 昌宏, 平石 拓, 光来 健一. 計算状態操作機構の新展開に向けて. 第 6 回 自動チューニング技術の現状と応用に関するシンポジウム, 2014
33. Kenichi Kourai, Kousuke Nakamura. Efficient VM Introspection in KVM and Performance Comparison with Xen. The 20th IEEE Pacific Rim International Symposium on Dependable Computing, 2014

〔その他〕
ホームページ等
http://super.para.media.kyoto-u.ac.jp/tascell/index.html

６．研究組織

(1)研究分担者
研究分担者氏名：平石　拓
ローマ字氏名：(HIRAISHI, Tasuku)
所属研究機関名：京都大学
部局名：学術情報メディアセンター
職名：助教
研究者番号（8 桁）：60528222

研究分担者氏名：光来　健一
ローマ字氏名：(KOURAI, Kenichi)
所属研究機関名：九州工業大学
部局名：大学院情報工学研究院
職名：教授
研究者番号（8 桁）：60372463

(2)研究協力者
研究協力者氏名：小出　洋
ローマ字氏名：(KOIDE, Hiroshi)

研究協力者氏名：江本　健斗
ローマ字氏名：(EMOTO, Kento)

※科研費による研究は、研究者の自覚と責任において実施するものです。そのため、研究の実施や研究成果の公表等については、国の要請等に基づくものではなく、その研究成果に関する見解や責任は、研究者個人に帰属されます。