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　 ５ 月 ３０ 日現在

機関番号：１２６１２
研究種目：基盤研究(B)（一般）
研究期間：2014～2018
課題番号：２６２８００２０

研究課題名（和文）大規模グラフ並列処理のための代数的構造に基づく理論基盤とプログラム開発基盤の構築

研究課題名（英文）Developing an algebraic model and program development platform for large-scale graph processing

研究代表者

岩崎　英哉（Iwasaki, Hideya）

電気通信大学・大学院情報理工学研究科・教授

研究者番号：９０２０３３７２

交付決定額（研究期間全体）：（直接経費）　　13,100,000円

研究成果の概要（和文）：本研究は，大規模なグラフデータを処理対象とする効率的な並列アプリケーションの開発を支援するシステムを構築し．広く一般の利用に供することを目的として推進した．その結果，次のような研究成果を得ることができた．(1) グラフとその上の同期的頂点主体計算の関数的モデル化．(2) 前項のモデル化に基づく関数型領域特化言語Fregelの提案．(3) メッセージ数削減の最適化機構を持ち複数の既存の並列グラフ処理フレームワークのプログラムへ翻訳可能なFregelコンパイラの実装．(4) 実用的な例題に基づく有効性の実証．開発したFregelシステムは，オープンソースソフトウェアとして公開している．

研究成果の学術的意義や社会的意義
本研究は，グラフ中に潜む代数的データ型に基づく構造に注目したモデル化を行っている点に，従来の研究にはほとんどみられない学術的な特徴がある．本研究の成果により，従来はプログラマが手で記述していたグラフデータの並列処理に関して，処理の自然な記述を可能とする領域特化言語と，気軽な利用を可能とする並列実行系が一体となり，プログラマの負担が大きく軽減される．さらにプログラムの代数的な性質に基づく最適化処理をコンパイラが内部で自動的に行うため，プログラマは効率のための細部に囚われる必要はなくなる．Webに代表される巨大なグラフデータに対する処理への需要が大きくなっている今日，その意義は大きい．

研究成果の概要（英文）：In this research, we developed a system named Fregel as an open source software that encourages the programmer to develop efficient parallel applications for large-scale graph processing. Technical contributions of this research can be summarized as follows. First, we abstracted and formalized the synchronous vertex-centric computation as a higher-order function. Second, on the basis of the functional model, we proposed Fregel, a functional domain specific language for declarative-style programming on large graphs. Third, we developed a Fregel compiler that transforms a Fregel program into programs for existing vertex-centric frameworks such as Giraph and Pregel+. This compiler applies optimization methods for a target Fregel program by removing unnecessary communications between vertices on the basis of algebraic properties of the programs. Fourth, we confirmed that Fregel programs compiled by the Fregel compiler can be executed with reasonable performance through some experiments.

研究分野：プログラミング言語

キーワード：プログラミング方法論　グラフ並列処理　領域特化言語　代数データ型　関数型言語

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９、ＣＫ－１９（共通）

１．研究開始当初の背景

(1) 近年，道路網ネットワーク，ソーシャルネットワーク，生命情報科学ネットワークなど，グラフやネットワーク構造を持つ大規模データ（以後「グラフデータ」と呼ぶ）が，身近な存在となってきている．それに伴い，これらのグラフデータを処理対象とし，新しい知識を獲得することを目的とするアプリケーションが増えている．

(2) グラフデータを処理対象とするアプリケーションの開発支援に共通する一般的な課題は，次のような点である．(a) 多くのアプリケーションに共通して現れる計算構造の簡潔な記述を可能とすること．(b) 獲得した知識を利用してグラフデータを更新し，複数のグラフデータ処理の直列的な適用を可能とすること．(c) グラフデータの大規模化に対応し，効率のよい並列分散処理を可能とすること．グラフデータは実世界の多くの場面で自然に現れるため，これらの課題を解決することは，極めて重要である．

２．研究の目的

本研究は，大規模なグラフデータを処理対象とする効率的な並列アプリケーションの開発を支援するシステムを構築し．広く一般の利用に供することを目的とする．この目的を実現するために，具体的には次のような目標を掲げる．

(1) グラフデータ中の代数的構造を適切に捉えた代数データ型に基づき，グラフデータ自身とそれに対する処理をモデル化する．
(2) グラフ処理を自然に記述できるように，グラフ処理に特化した領域特化言語を制定する．
(3) この記領域特化言語で記述されたプログラムの効率的な並列実行系を，既存の大規模データ処理フレームワークを利用して実現する．これらのフレームワークの利用は，適切な抽象化の下に隠蔽し，プログラマには意識させないようにする．
(4) 実用的な例題アプリケーションを領域特化言語を用いて記述し，実並列環境で作動させることにより，提案システムを定性的かつ定量的に評価する．

３．研究の方法

本研究では，大規模グラフデータに対するスケーラブルな並列分散処理のためのモデルとして，Google による Pregel に注目した．Pregel の大きな特徴は，バルク同期並列に基づく頂点主体の計算モデルを採用している点である．このモデルでは，グラフ内の頂点はクラスタ中の計算ノードに分散され，各頂点は必要に応じて互いにメッセージを交換しながら，スーパーステップ（superstep，以下 SS と略記）と呼ばれる一単位の処理を同期的に繰り返し実行する．

頂点主体のバルク同期並列に基づく Pregel は，プログラマはデータフローや同期のことを考える必要がないという意味で，自然なプログラミングモデルを提供する．しかし，Pregel モデルに基づいて実際にプログラミングを行おうとすると，(a) 明示的な副作用に依存したプログラミング，(b) 明示的な状態制御，(c) 明示的な計算停止制御，等の煩雑なプログラミングを強いられるという問題点があることがわかった．本研究は，これらの問題点の根本的な原因を明らかにした上で，それを解決するためのモデルを構築し，構築したモデルを基礎とする領域特化言語を設計し処理系を実装する，という方法で進めていった．

４．研究成果

(1) 同期的頂点主体計算の関数的モデル化

グラフを相互再帰的な構造を持つ代数データ型として定義した上で，Pregel による頂点主体のバルク同期並列を，グラフの構造に基づく構造的再帰関数としてモデル化した．頂点主体の並列計算では，各頂点は以下のような処理を繰り返し実行する．

(a) 隣接頂点から，1 回前の繰返し処理において計算されたデータを受信する．
(b) 受信データと自身の 1 回前の繰返し処理での計算結果を用いて，必要な計算を行う．
(c) 出力辺で接続されている全隣接頂点に，(b)における計算結果のデータを送信する．各隣接頂点は，このデータを次の繰返し処理で受け取る．

Pregel で頂点間通信を行うためには，送信を行う SS と受信を行う SS を分断して記述しなければならない．そのため，(a)–(c)の論理的なひとまとまりの処理をまとめて記述することができず，明示的な状態制御を行う見通しの悪いプログラムにならざるを得ない．

以上の考察に基づき，提案モデルでは(a)-(c)のひとまとまりの繰返し処理 1 回分を「論理スーパーステップ（logical superstep，以下 LSS と略記)」と呼び，LSS を単一の関数 (LSS 関数と呼ぶ) として記述する．LSS 関数は，「何回目の繰返し処理か」(これをクロックと呼ぶ) を表す数値を引数として取る．LSS 関数ではデータの送受信を明示的には記述せず，かわりに一つ前のクロック値に対する自身の再帰呼出しを用いる．

さらにこのモデルでは，各頂点における LSS 関数の計算結果から構成されるグラフの無

限リストが，頂点主体計算の各クロックにおける結果を表すものと考える．現実には無限に計算を続けるのではなく，計算結果が変化しなくなり定常状態に陥った時点など，適切な時点で計算を打ち切り，その時点でのグラフを最終結果とする必要がある． そこで，この無限リストの外側から，適切な時点におけるグラフを最終結果として取得する関数を適用し，これを頂点主体計算を表現する新しいモデルとした．

LSS 関数は，入力グラフの構造を基礎とする一種の構造的再帰関数として捉えることができる．グラフは全体としては循環的な構造をしているが，LSS 関数の再帰呼出しが無限再帰に陥らないのは，クロックを減らして再帰呼出しを行うためである．さらに，同一の引数に対する LSS 関数の呼出しを重複して行わないようなメモ化機構を前提として考えると，我々のモデルはグラフ上の動的計画法として捉えることができる．

(2) 頂点主体並列計算のための関数型領域特化言語 Fregel の提案

本研究では，(1)のモデル化に基づく関数型領域特化言語 Fregel（Functional Pregel）を提案した．Fregel における頂点主体グラフ並列処理は，LSS を繰返し実行する．その際，手続き的なデータ送受信のかわりに，明示的に隣接頂点の値にアクセスする記述を行う．

Fregel の構文は，基本的には Haskell の構文に沿って定義されており，Haskell 上で実行やデバッグを行うことができる．Fregel は，頂点主体計算の抽象的な記述を支援する「グラフ高階関数」と呼ばれる 4 個の高階関数群を提供する．最も基本的的なグラフ高階関数は関数 `fregel` であり，次の形をしている．

`fregel` *init step term graph*

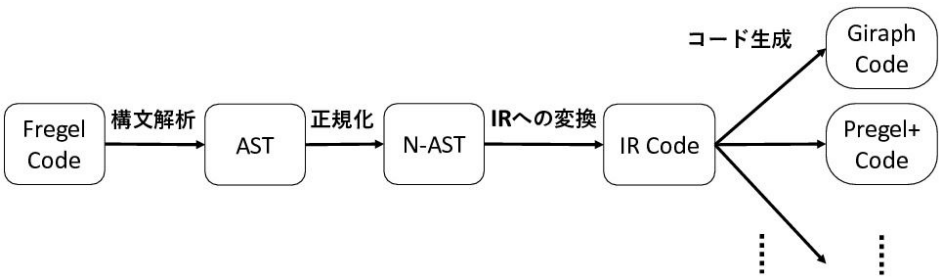
ここで *init* は各頂点における初期化関数，*step* は各頂点における LSS 処理 1 回分を記述する LSS 関数，*term* は計算の終了条件，*graph* は処理対象のグラフである．LSS 関数 *step* には，隣接頂点間のメッセージ送受信を明示的には記述しない．*step* の引数は，自頂点 `v`，全頂点における前回の LSS における値を保持する表 `prev`，今回の LSS における値を保持する表 `curr` である．1 回前の LSS の結果として隣接頂点 `u` が持つ値は，`prev u` として得る．また，`is v` によって，自分への入力辺の重み `e` とその辺で結ばれた隣接頂点 `u` の対`(e,u)`のリストを得る．`is v` をジェネレータとする集約表現の中で `prev u`（`u` は隣接頂点とする）を用いることにより，隣接頂点の前 LSS の計算結果値にアクセスすることができ，データ受信を隠蔽した記述が可能となっている．

(1)と(2)の研究成果は，日本ソフトウェア科学会第 33 回大会にて登壇発表を行い高橋奨励賞を受賞した．さらに，プログラミング言語の研究分野で最高峰の国際会議 ICFP 2016 にも論文が採択され登壇発表を行った．

(3) Fregel コンパイラの実装

本研究では，Fregel プログラムのコンパイラを，既存の頂点主体分散並列グラフ処理フレームワークのプログラムへの翻訳系として実装した．このコンパイラの特徴は，複数のフレームワークへの翻訳を可能としている点である．現状で対象とするフレームワークは，Pregel のオープンソース実装である Giraph と Pregel+ であるが，今後はさらに別のフレームワークにも対応する予定である．

下図に，Fregel コンパイラの処理の一連の流れを示す．途中で施す最適化については，(4)で詳しく述べる．ユーザが記述した Fregel プログラムをはじめに構文解析し，抽象構文木（AST）へ変換する．次に抽象構文木レベルで正規化処理を行い，`fregel` 等のグラフ高階関数の複数の出現をひとつにまとめて，Pregel の起動コストをなくす．次に，正規化された抽象構文木（N-AST）を，手続き型言語風のブロック構造を持つフレームワーク独立な中間表現（Fregel Intermediate Representation，以下 FregelIR と略記する）に変換する．最後にその中間表現から，各フレームワークのプログラムに変換する．このように共通の中間表現を経由することで，各フレームワーク向けのプログラム生成をモジュラかつ容易に行うことができる．



N-AST を FregelIR に変換する際には，論理的なひとまとまりの繰返し処理である LSS を，Pregel における複数の SS に分断する．Fregel では，隣接頂点の値を `prev u` のようにして直接参照する形で記述するが，FregelIR ではこの処理をメッセージの明示的な送信と受信に変換する．このようにメッセージ送受信が発生する際に，LSS を分断する必要が生じる．さらに LSS を複数の SS に分断した結果，SS をまたいで保持する必要が生じた局所変数は，頂点が保持する変数として定義するなどの処理も行う．

Giraph，Pregel+など，本コンパイラが翻訳対象とするフレームワークは，Java，C++といった言語の違いはあるが，C 風のブロック構造，制御構造を持つという意味で共通点が多い．一方で，SS を記述する関数の定義方法，頂点間のメッセージ送受信の方法など，フレームワークによって大きく異なる部分もある．FregelIR は，フレームワークごとの違いを吸収できる程度の抽象度を持ちつつ，ブロック構造や制御構造などの共通部分に関しては，フレームワークのコードを出力するという目的を達成するのに必要十分な機能に限定した上で，出力言語の違いを吸収可能な最大公約数的な表現を持つように設計した．

(4) Fregel プログラムの最適化手法の提案と実装

Fregel プログラムを単純にコンパイルすると，翻訳結果の Giraph や Pregel+ のプログラムにはメッセージ送受信に関するオーバーヘッドがあり，十分な実行性能が得られない．たとえば，LSS における計算結果値がひとつ前の LSS と同じである場合，ある条件下ではその値を隣接頂点に送信しても無駄であるが，現状は必ず送信処理を行ってしまう．

そこで，メッセージ送受信の削減に注目した最適化を，(a) 重複する値の送信の削減，(b) 頂点における計算の単位元値の送信の削減，(c) 頂点の非活性化に基づく終了判定のための値の集約の削減，(d) 同一頂点に対するメッセージの結合に基づく受信の削減，の 4 つの視点から検討し，Fregel コンパイラにおいて実現した．

最適化が可能かどうかを判断するための手法としては，次の 2 つを考察し実際に実装を行った．第一の手法は，受信したメッセージを二項演算子によって結合するという前提のもと，メッセージ最適化が可能な二項演算子やその単位元に関する情報をあらかじめ「知識」としてコンパイラに保持させ，その知識に該当する場合にのみ最適化を行うというものである．第二の手法は，メッセージ送受信の削減が可能な条件，および，頂点の非活性化が可能な条件を定式化し，対象プログラムがこれらの条件を満足するか否かは SMT ソルバを用いて判定するという手法である．この手法による最適化は，利用する SMT ソルバの能力にも依存するが，第一の手法のように最適化可能な Fregel プログラムの形が限定されず，より幅広いプログラムに対して最適化可能と判断できる可能性がある

実装は，Fregel コンパイラ内部の AST 変換および N-AST から FregelIR への変換部分を拡張した．第一の手法と第二の手法は最適化が可能かどうかの判定方法が異なるが，いったん最適化可能と判定されれば，どちらの手法でも同じ翻訳結果が出力される．

第一の方法は，国際会議 FLOPS 2018 の論文として発表した．第二の方法は，国内ワークショップ PPL 2018 で発表を行い(論文賞受賞)，その推薦論文がコンピュータソフトウェア誌に採択，掲載された．

(5) Fregel の性能評価

いくつかの Fregel プログラムの実行時間を，Giraph，Pregel+ それぞれのフレームワークにおいて直接手書きして最適化して記述したプログラムの実行時間と比較した．この評価実験で用いたプログラムは，単一始点最短路問題，全到達可能性問題，最大値問題を解くものである．実験環境は，CPU は AMD Opteron Processor 6380×4 (4×16 コア)，メモリは 128GB，Hadoop は version 1.2.1，Giraph はバージョン 1.2.0 である．実験用のグラフデータは，頂点数は 100 万，有向辺数は 1 千万のグラフと，頂点数は 100 万，有向辺数は約 1 億のグラフであり，いずれも乱数を用いて作成したものである．

Fregel コンパイラで，(4)におけるメッセージ結合以外のすべての最適化が適用されたプログラムの実行時間は，手書きのプログラムの実行時間の 1.3 倍未満に抑えることができた．また Pregel+ のプログラムを出力する際にメッセージ結合の最適化を行うと，メッセージ結合を行わない場合と比較して，実行時間が最大で約半分と大きな効果が得られることがあることも判明した．

最適化の効果などの詳しい性能評価は，(4)で述べたコンピュータソフトウェア誌に採択，掲載された論文で述べられている．

(6) その他の主な研究成果

① 辺で直接接続されていない遠隔頂点へのリモートアクセスが記述可能な頂点中心グラフ処理のための高水準言語 Palgol を設計した．また，Palgol プログラムから効率の良い Giraph プログラムを生成するシステムを実現した．この成果は，国際会議 APLAS 2017 に論文が採択され，発表を行った．

② Fregel を用いても記述しにくい頂点の部分集合を陽に扱うプログラム記述を支援するために，Fregel よりも一段高いレベルにある（Fregel とは別の）領域特化言語を設計し，その領域特化言語のプログラムから Fregel プログラムへの変換系を実装した．この成果は，論文誌 The Journal of Supercomputing に採録された．

③ 既存研究では，頂点主体計算の他に，部分グラフ単位で並列に計算を行う部分グラフ主体計算も提案されており，どちらの性能が優っているかは問題，グラフの形状などに依存する．そこで，頂点主体/部分グラフ主体のいずれかを選択してコードを出力することが可能な（Fregel とは別の）領域特化言語を設計し，実装した．

５．主な発表論文等

〔雑誌論文〕（計６件）

① Kento Emoto，Fumihisa Sadahira，A DSL for graph parallel programming with vertex subsets，The Journal of Supercomputing，査読有，採択済.
DOI: 10.1007/s11227-019-02821-w

② 加藤 直斗，岩崎 英哉，Fregel コンパイラにおける不要な値送受信の削減，コンピュータソフトウェア，査読有，Vol. 36，No. 2，2019，pp. 28−46．DOI: 10.11309/jssst.36.2_28

③ Le-Duc Tung，Zhenjiang Hu，Towards Systematic Parallelization of Graph Transformation Over Pregel，International Journal of Parallel Programming，査読有，Vol. 45，No. 2，2017，pp. 320−339．DOI: 10.1007/s10766-016-0418-5

④ Chong Li，Le-Duc Tung，Xiaodong Meng，Zhenjiang Hu，Let High-level Graph Queries Be Parallel Efficient: An Approach Over Structural Recursion on Pregel，Journal of Information Processing，査読有，Vol. 24，No. 6，2016，pp. 928−936.
DOI: 10.2197/ipsjjip.24.928

⑤ Kiminori Matsuzaki，Reina Miyazaki，Parallel Tree Accumulation on MapReduce，International Journal of Parallel Programming，査読有，Vol. 44，No. 3，2016，pp. 466−485．DOI: 10.1007/s10766-015-0355-8

〔学会発表〕（計１９件）

① Yongzhe Zhang，Zhenjiang Hu，Composing Optimization Techniques for Vertex-Centric Graph Processing via Communication Channels，33rd IEEE International Parallel and Distributed Processing Symposium (IPDPS 2019)，査読有，2019，採択済.

② Timothy A. K. Zakian，Ludovic A. R. Capelli，Zhenjiang Hu，Incrimentalization of Vertex-Centric Programs，33rd IEEE International Parallel and Distributed Processing Symposium (IPDPS 2019)，査読有，2019，採択済.

③ 加藤 直斗，岩崎 英哉，頂点主体並列分散処理のメッセージ送信遅延による通信数削減，日本ソフトウェア科学会 第 21 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 2019，ポスター)，査読無，2019.

④ 小西 篤志，江本 健斗，Fregel から GraphX へのコンパイルにおける SMT ソルバを用いた不要な通信の削減，日本ソフトウェア科学会 第 21 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 2019，ポスター)，査読無，2019.

⑤ 秋葉 柊哉，岩崎 英哉，グラフ並列分散処理のための計算モデルを選択可能な領域特化言語， 情報処理学会 第 60 回プログラミング・シンポジウム報告集，査読無，2019，pp. 7−16.

⑥ Akimasa Morihata，Kento Emoto，Kiminori Matsuzaki，Zhenjiang Hu，Hideya Iwasaki，Optimizing Declarative Parallel Distributed Graph Processing by using Constraint Solvers，14th International Symposium on Functional and Logic Programming (FLOPS 2018)，Lecture Notes in Computer Science 10818，査読有，2018，pp. 166−181.
DOI: 10.1007/978-3-319-90686-7_11

⑦ 加藤 直斗，岩崎 英哉，Fregel コンパイラにおける不要な値送受信の削減，日本ソフトウェア科学会 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)，査読有，2018.（論文賞受賞）

⑧ 近松 万由子，岩崎 英哉，中野 圭介，関数型言語の初学者のための Haskell ビジュアルプログラミング環境，日本ソフトウェア科学会 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)，査読有，2018.

⑨ 駒村 春野，岩崎 英哉，頂点主体グラフ並列処理のための Giraph 用クラスライブラリ，情報処理学会 第 59 回プログラミング・シンポジウム報告集，査読無，2018，pp. 119−131.

⑩ Yongzhe Zhang，Hsiang-Shang Ko，Zhenjiang Hu，Palgol: A High-Level DSL for Vertex-Centric Graph Processing with Remote Data Access，15th Asian Symposium on Programming Languages and Systems (APLAS 2017)，Lecture Notes in Computer Science 10695，査読有，2017，pp. 301−320．DOI: 10.1007/978-3-319-71327-6_15

⑪ 松崎 公紀，岩崎 英哉，江本 健斗，胡 振江，森畑 明昌，複数の頂点主体グラフ計算フレームワーク向けの中間表現とコード生成器，日本ソフトウェア科学会第 34 回大会講演論文集，査読無，2017.

⑫ Onofre Coll Ruiz，Kiminori Matsuzaki，Keeping Control Away from Computation: A Computation Control Layer over the Vertex-centric Graph Processing Model，10th International Symposium on High-Level Parallel Programming and Application (HLPP 2017)，査読有，2017.

⑬ Amogh Rathore，Hideya Iwasaki，Implementing Distributed Backpropagation Algorithm using Apache Giraph，日本ソフトウェア科学会 第 19 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 2017)，査読有，2017.

⑭ 定久 典久，江本 健斗，頂点部分集合変数を備えた大規模グラフ計算用領域特化言語，日本ソフトウェア科学会 第 19 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 2017)，査読有，2017.
⑮ 中島 拓，江本 健斗，Spark GraphX への Fregel コンパイラ，日本ソフトウェア科学会 第 19 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 2017，ポスター)，査読無，2017.
⑯ Onofre Coll Ruiz，Kiminori Matsuzaki，Shigeyuki Sato，s6raph: Vertex-Centric Graph Processing Framework with Functional Interface，5th ACM SIGPLAN Workshop on Functional High-Performance Computing (FHPC 2016)，査読有，2016，pp. 58−64. DOI: 10.1145/2975991.2976000
⑰ Kento Emoto, Kiminori Matsuzaki, Zhenjiang Hu, Hideya Iwasaki, Think Like a Vertex, Behave Like a Function! A Functional DSL for Vertex-Centric Big Graph Processing, 21st ACM SIGPLAN International Conference on Functional Programming (ICFP 2016)，査読有，2016，pp. 200−213．DOI: 10.1145/2951913.2951938
⑱ 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 Fregel とその評価，日本ソフトウェア科学会第 33 回大会講演論文集，査読無，2016．(高橋奨励賞受賞)
⑲ 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，日本ソフトウェア科学会 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 (PPL 2016)，査読有，2016.

〔その他〕
ホームページ等
① Fregel Project
https://ipl.cs.uec.ac.jp/~iwasaki/Fregel
② Fregel Project（日本語）
https://ipl.cs.uec.ac.jp/~iwasaki/Fregel/index-j.html
上記ホームページでは，本研究で開発した Fregel コンパイラを，オープンソースソフトウェアとして公開している．

６．研究組織

(1)研究分担者
研究分担者氏名： 胡 振江
ローマ字氏名： Zhenjiang Hu
所属研究機関名： 国立情報学研究所
部局名： アーキテクチャ科学研究系
職名： 教授
研究者番号（8 桁）： 50292769

研究分担者氏名： 松崎 公紀
ローマ字氏名： Kiminori Matsuzaki
所属研究機関名： 高知工科大学
部局名： 情報学群
職名： 教授
研究者番号（8 桁）： 30401243

研究分担者氏名： 江本 健斗
ローマ字氏名： Kento Emoto
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： 准教授
研究者番号（8 桁）： 00587470

(2)研究協力者
なし