別紙 1

論文の内容の要旨

論文題目	Structured Parallel Programming with Trees （和訳　木を用いた構造化並列プログラミング）
学位 申請者	佐藤 重幸

並列計算機の普及によって，並列プログラムの開発環境は身近なものとなり，今や誰でもが並列プログラムを開発すること（並列プログラミング）が可能になっている．しかし，並列プログラミング自体は未だに普及しているとは言えない．これは，逐次プログラミングに比べて並列プログラミングが難しいからである．計算機のコストモデルや負荷分散といった，通常の逐次プログラミングでは意識しない事柄を考慮しなければ，台数効果が得られる効率の良い並列プログラムを開発することはできない．プログラミングの観点では，このような一般のプログラマには難しい事柄を隠蔽し，プログラマが問題やアルゴリズムの記述に専念できるようにする，計算の抽象化が重要である．並列プログラミングにおいては，ハードウェアに近い立場の低水準抽象化は存在するものの，プログラマに近い立場の高水準抽象化は十分に確立していない．これが並列プログラミングの障害である．

抽象化の典型的な実現方法は，コンパイラによる自動化である．並列プログラミングの自動化，すなわち自動並列化は1980年代から研究されており，現状では，forループ内での規則的な配列アクセスについて並列化と最適化が可能になっている．しかし，それよりも複雑なデータ構造，例えばポインタ構造や疎表現を用いた計算については，自動化はおろか適切な抽象化すら十分に明らかとは言えない．そのような複雑な構造を用いた計算は，不規則アルゴリズムとして一括りに扱われている．

不規則アルゴリズムは，一般にグラフ構造を扱う．一般のグラフ構造上で，分割統治型計算を構成するのは難しい．分割統治は，並列アルゴリズムにおいて負荷分散を保証するために不可欠であり，良質な並列プログラムを構成する切り札である．それを利用できない一般のグラフ構造について，並列プログラミングの困難さは不可避である．しかしながら，少し制限された木構造であれば，自然に分割統治が実現できる．また，木構造は，プログラミングの道具として一般的で十分に強力である．そこで本研究では，木を用いた並列計算の抽象化を取り扱う．

まず初めに注目したのは，木スケルトンである．スケルトンとは並列計算のパターンであり，木スケルトンは，良質な負荷分散を保証する木上の計算パターンである．木スケルトンは，近年，特に松崎によって深く研究されており，その理論的基盤はほぼ確立している．しかしながら，その実用性や有用性については多くの改善の余地を残しており，現状において木スケルトンは使いやすいものではない．

本研究の第１の貢献は，木スケルトンの実装面を充実させ，木スケルトンを使いやすくしたことである．具体的には２つのことに取り組んだ．まず，C++とMPIで実装された，分散メモリ環境向けの木スケルトンライブラリを再設計し，スケルトンとデータ構造を疎結合にしたライブラリ実装を与えた．これによって，木構造をひとつのビューとして扱えるようになり，異なるデータ構造のスケルトンで1つのデータ構造の実体を扱えるようになった．もう１つは，松崎の理論基盤に基づいた木上の再帰関数の並列化器のコンパイラ実装である．これは逐次的なＣの再帰関数を受け取って，木スケルトンを暗黙裡に利用する並列プログラムを生成する．これによって，木スケルトンの複雑なAPIをプログラマから隠蔽し，簡潔で平易なアルゴリズム記述と良質な並列プログラムを両立させる．

これらの取り組みによって，木スケルトンの使いやすさは向上したが，十分な実用性を得るには至らなかった．そこで，この取り組みの教訓を生かして，２つの問題領域に焦点を当てて，並列計算の抽象化に取り組んだ．１つはプログラム解析，もう１つは近傍計算である．

プログラム解析では一般に，プログラムを制御フローグラフ(CFG)と呼ばれるグラフ構造で表現し，CFG上の計算によって解析結果を得る．すなわち，典型的な不規則アリゴリズムであり，分割統治型の並列計算が難しい．本研究の第２の貢献は，Rosenの高水準アプローチに基づいて，抽象構文木(AST)上の分割統治によるプログラム解析手法を開発したことである．提案手法は，任意のグラフ構造を持つCFGではなく，ASTを用いることで自然に分割統治を実現する．これまでASTが解析にあまり用いられてこなかった主な原因は，AST上で扱うのが難しい制御フローを生むgoto/label文の存在である．提案手法は，AST全体から見て少数であるgoto/label文の解釈を後回しにして，大部分を分割統治で計算する．その後，goto/label文に由来する小さい計算を行い全体の結果を得る．具体的にはTarjanの定式化に基づいたデータフロー解析と，関数的な定式化に基づいた値グラフの構築を扱った．

もう１つの問題領域である空間上の近傍計算では，負荷分散だけが問題にはならない．なぜなら，空間全体に対する１回の近傍計算には十分な量の独立した部分計算があるからである．問題なのは，多くの近傍計算は反復し，各反復を単純に並列化するとキャッシュミスが多発することで効率が悪くなることである．すなわち，負荷分散だけでなくキャッシュ効率化も関心事となる．しかし，キャッシュ忘却アルゴリズムの研究成果から，分割統治は局所性向上にも役立つことが知られている．本研究の第３の貢献は，木スケルトンの文脈で培った代数的定式化や木分割に基づく分割統治を，近傍計算に対するキャッシュ効率化手法に応用したことである．具体的には，まず，ステンシル計算と呼ばれる規則的な配列上の近傍計算を扱い，そして，高い応用性持つ空間分割木上の反復走査を扱った．

別紙２

論文審査の結果の要旨

学位申請者氏名　佐藤　重幸

審査委員主査　岩崎　英哉

委員　尾内　理紀夫

委員　沼尾　雅之

委員　成見　哲

委員　中山　泰一

本論文は，Structured Parallel Programming with Trees（木を用いた構造化並列プログラミング）という題で，全14章から構成されている．そのうち第2章から第13章までが，本論文における中心的な貢献内容であり，第2章～第5章を第I部，第6章～第9章を第II部，第10章～第13章を第III部とする３部構成になっている．

第1章は「Introduction」として，本論文の背景，動機について述べている．並列計算機が身近な存在になった現在，並列計算の非専門家のための並列プログラミングが望まれている．並列プログラミングを容易にする試みとして，並列計算のパターンを「並列スケルトン」として提供し，実装の詳細を気にしない高水準プログラミングを可能とする研究がある．しかし並列スケルトンの既存研究では，木というデータ構造を扱うものは十分には扱われていない．そこで，本研究では，木を用いる（特に）分割統治型並列計算を抽象化し，具体的な問題に応用できる木スケルトンの設計と実装を行うことを目標に掲げている．

第I部である第2章～第5章は，木スケルトンを用いたプログラミングについて述べている．

第2章は「Tree Skeletons」と題して，木スケルトンの定式化を与えている．

第3章は「Interface Between Data Structures and Skeletons」と題して，分散メモリ環境向けの木スケルトンライブラリを再設計し，木を表現するデータ構造と木スケルトンの間のインタフェースを疎にしたライブラリ実装を C++ と MPI を用いて与えた．これにより，木構造をひとつのビューとして扱えるようになり，異なるデータ構造のスケルトンで1つのデータ構造の実体を扱えるようになった．

第4章は「Tree Skeleton Hiding」と題して，木上の再帰関数の並列化器のコンパイラ実装に関して述べている．これは逐次的なCの再帰関数を受け取って，木スケルトンを暗黙に利用する並列プログラムを生成する．これにより，木スケルトンの複雑なAPIをプログラマから隠蔽し，簡潔で平易なアルゴリズム記述と良質な並列プログラムを両立させることに成功している．

第5章は「Limitations of Tree Skeletons」と題し，上で述べたように木スケ

ルトンの実装面を充実させ利便性を改善したが，実用性や有用性については依然として多くの改善の余地を残していることを指摘し，第I部を総括している．

第II部である第6章～第9章は，プログラム解析などの構文主導型のプログラミングに焦点を当てた並列計算の抽象化について述べている．ここでは，プログラム構文構造を表現する，抽象構文木(AST)と呼ばれる木構造を直接扱うことにより，並列計算を可能としている．

第6章は「Syntax-Directed Computation and Program Analysis」と題し，第II部の導入として，プログラム解析を扱う動機などを述べている．

第7章は「Syntax-Directed Divide-and-Conquer Data-Flow Analysis」と題して，AST上の分割統治によるプログラム解析手法を述べている．プログラム解析では一般に，プログラムを制御フローグラフ(CFG)と呼ばれるグラフ構造で表現しCFG上の計算によって結果を得るが，ここではASTを用いることで自然に分割統治型の並列計算を実現することを提案している．提案手法では，AST上で扱うのが難しいgoto文に関しても，行列演算を用いてボトムアップ的に対処できる．

第8章は「Syntax-Directed Construction of Value Graphs」と題して，構文主導型の値グラフの構築手法を述べている．値グラフとは，コンパイラの最適化で用いられるデータ構造であり，プログラムの冗長性の除去に用いられる．従来，値グラフはCFGなどを経由して構築されてきたが，本手法ではASTから値グラフを構築する手法を述べている．さらにgoto文への対処法についても議論している．

第9章は「Lessons from Syntax-Directed Program Analysis」と題して，第II部を総括している．

第III部である第10章～第13章は，近傍計算とそれに使われる空間分割木と呼ばれる木構造に焦点をあて，代数的定式化に基づくキャッシュ効率の良いスケルトンの設計について述べている．

第10章は「Neighborhood Computations」と題して，近傍計算の性質，キャッシュ効率が良い計算パターンの必要性などを述べ，第III部の導入としている．

第11章は「Time Contraction for Optimizing Stencil Computation」と題して，科学技術計算でしばしば現れるステンシル計算の最適化手法を述べている．提案手法では，ステンシル計算におけるループ構造に関して，Loop Contractionという方法により繰返しの回数を削減する．

第12章は「Locally Enhancement based on Segmentation of Trees」と題し，木を反復して辿る操作を並列パターンとして定式化している．その上で，「穴のあいた空間」を用いて区切られた木構造として空間分割木を捉え，うまく区切られた空間分割木上の処理は，キャッシュ計算量が抑えられることを示している．

第13章は「Towards Neighborhood Abstractions」と題して，第III部における近傍計算についてまとめている．

第14章は「Conclusion」と題して，木を用いた並列プログラミングでは木の解釈を考えるべきであり，木の解釈から生じたパターンこそが有用なスケルトンになるという知見を述べ，本論文を総括している．

本論文は，プログラミングの道具として一般的な木構造を用いた並列計算の高水準の抽象化を扱い，幅広い視点から多くの有用な知見を得ている．よって，本論文は博士（工学）の学位論文として十分な価値を有するものと認める．