「情報処理学会論文誌：プログラミング」の編集について

プログラミング研究会論文誌編集委員会

情報処理学会では，研究会の活性化を目指して様々な改革を進めている．プログラミング研究会はこの流れを受けて，研究会のあるべき姿について徹底的な討論を行ってきた．その帰結として，研究会独自の論文誌の編集にいち早く踏み切ることを決定した．

研究会論文誌「情報処理学会論文誌：プログラミング」の特徴と意義は大きく 3 つある．第 1 は，従来の「論文」に対して想定されてきた対象分野や査読基準では必ずしもカバーしきれない，多様な成果の公表の場を提供することである．第 2 は，投稿論文の内容を研究会で発表することを義務づけることによって，迅速で的確な査読を実現するとともに，議論の結果の最終稿へのフィードバックを可能にすることである．第 3 は，研究内容の表現に必要であると認められれば，長大な論文も採録可能としている点である．

本論文誌を通じて，日本のプログラミング分野の研究活動を盛り上げていきたい．読者諸氏からの多くの論文投稿を期待する．

1. 対 象 分 野

プログラミングは，コンピュータの誕生と同時に生まれた伝統的な分野であるが，コンピュータがある限り不可欠な技術である．並列分散処理やマルチメディア応用など処理内容が高度になるにつれて，プログラミングの重要性は増すことがあっても減ることはないであろう．

「情報処理学会論文誌：プログラミング」は，プログラミングに関するテーマ全般を専門に扱う論文誌である．具体例として次のようなテーマがあげられる．

- プログラミング言語の設計，処理系の実装
- プログラミングの理論，基本概念
- プログラミング環境，支援システム
- プログラミング方法論，パラダイム

これらを応用したシステムの開発事例も対象に含まれる．また，上記以外でも，プログラミングに関する面白い話題であれば対象となる．

2. 編 集 方 針

本論文誌は，プログラミング研究会における発表と論文誌投稿が密接にリンクされている点に特徴がある．論文誌への投稿者が用意する研究会発表用の資料が，そのまま本論文誌への投稿論文となる．

研究会発表をせずに本論文誌に投稿することはできないが，逆に，本論文誌への投稿をともなわない研究会発表は可能である．そのような発表や，論文が不採録となった発表については，アブストラクトが本論文誌に掲載される．従来のプログラミング研究会の研究報告は廃止し，その代わりとして，研究会登録者には本論文誌が配布される．

本論文誌に掲載する論文は，通常のオリジナル論文と，サーベイ論文の 2 種類とする．どちらの種類であるかは，著者自身の指定によって決まる．論文の記述言語は日本語，英語のいずれかとする．論文の長さに制限は設けない．

3. 査 読 基 準

基本的に，減点法に陥ることを避け，論文の良い点を積極的に評価するという方針を貫く．具体的には，新規性，有効性などの評価項目のうち，どれか 1 つの点で特に優れていると認められれば採録する．体裁のみが整った論文より，若干の不備はあっても技術的な貢献の大きい論文を積極的に受け入れる．

このような観点から，たとえば次にあげるような，従来は論文としてまとめることが難しかった内容について論じた論文もできるだけ受け入れる．

- プログラミング言語の設計論
- システムの開発経験に関する報告
- 斬新なアイディアの提案
- 概念の整理，分類法，尺度の提案
- 複数のシステムその他の比較

4. 投稿から掲載までの流れ

本論文誌への投稿希望者，および研究会での発表希望者は，発表会開催日の 2〜3 カ月前までに発表申込みをする．具体的な方法は研究会ホームページ `http://www.ipsj.or.jp/sig/pro/` を参照していただきたい．申し込みの際には，本論文誌への投稿の有無，オリジナル論文とサーベイ論文の種別指定を明記する．また，アブストラクト（和英両方，和文は 600 字程度）を添付する．

論文投稿を希望した場合は，研究発表会の 3 週間前までに，別に定めるスタイル基準に従ったカメラレディ形式で論文を提出する．

毎回の研究発表会の直後，編集委員会が開催され，各論文について 1 名の査読者が決定される．査読報告をもとに，編集委員会は採録，条件付き採録，不採録のいずれかの判定を行い，発表会開催後 3 週間程度で発表者に採否通知を行う．照会の手続きはないが，論文改善のための付帯意見が添付される場合がある．この場合は，3 週間以内に改良版を作成する．

5. 研究発表会

プログラミング研究会では，発表会ごとに特集テーマを設けている．ただし各発表会では，特集以外の一般の発表もつねに受け付けている．

2001 年度の発表会日程は次のとおりであり，各発表会の特集テーマは，今後数年間はそのまま維持する予定である．

6 月 21 〜 22 日［プログラミング言語の設計と実装］
7 月 25 〜 27 日［SWoPP—並列/分散/協調プログラミング言語と処理系］
10 月 22 〜 23 日［理論］
1 月 29 〜 30 日 [並列・分散処理]
3 月 15 〜 16 日 [プログラミング言語一般]

6. 編 集 母 体

本論文誌は，下記のプログラミング研究会論文誌編集委員会の責任で編集を行う．各研究発表会ごとに担当編集委員が割り当てられ，投稿論文の査読プロセスを主導する．2000 年度より論文誌編集委員会メンバを増やし，各研究発表会を 2 名の編集委員で担当している．

本号の編集にあたって

2001 年度第 2 回研究発表会（特集：並列/分散/協調プログラミング言語と処理系）
担当編集委員　高木浩光，八杉昌宏
2001 年度第 3 回研究発表会（特集：理論）
担当編集委員　小川瑞史，高橋和子

本号は，2001 年度第 2 回プログラミング研究会（通算第 35 回）と 2001 年度第 3 回プログラミング研究会（通算第 36 回）からの採録論文 6 件からなる．

第 2 回研究会は，2001 年 7 月 26，27 日に沖縄コンベンションセンターにおいて，「2001 年並列/分散/協調処理に関する『沖縄』サマー・ワークショップ（SWoPP 沖縄 2001）」（7 月 25 日〜27 日開催）の一環として開催された．会場や発表スケジュールの調整では SWoPP 幹事のお世話になっている．第 3 回研究会は，2001 年 10 月 22 日，23 日に高知工科大学で開催された．第 3 回研究会の会場や懇親会の手配については高知工科大学の菊池豊氏のお世話になっている．ともにあつく御礼申し上げます．

研究会のテーマは，第 2 回が「並列/分散/協調プログラミング言語と処理系」，第 3 回が「理論」とし，幅広く論文を募集した．研究会論文誌への投稿をともなう発表のほかに，論文投稿をともなわない発表を歓迎したことも，これまでと同様である．その結果，第 2

回では 13 件，第 3 回は 12 件の発表が行われた．

第 2 回，第 3 回研究会とも，研究会当日の昼休みや発表終了後に編集委員ならびに編集委員会が出席を依頼したメンバが集まり，複数回にわたって編集委員会を開催した．編集委員会では，その委員会直前またはその前のセッションで発表された各論文について，発表から時間を置くことなく議論を行った．ただし，投稿論文の共著者となっているメンバは，その論文についての議論の間は退席している．委員会では先の節に記した対象分野，編集方針および査読基準に従って，各論文の評価できる点について意見が交され，その場で可能なかぎり査読者の選定を行うようにした．各査読者は，編集委員会での議論をふまえ査読を行った．

結果として，第 2 回研究会からは 4 件，第 3 回研究会からは 2 件の通常論文が採録された．これ以外の発表については，各々について 1 ページの概要を掲載した．

以下，掲載論文について，簡単に紹介する．

「属性文法の系統的デバッグ法（Systematic Debugging Method for Attribute Grammars）」では，属性文法による記述をデバッグするための枠組みを定式化し，従来のアルゴリズミックデバッギングとスライスによる方法の両者を包含した，より一般的な枠組みについて述べている．さらに，このデバッグ法を，属性文法デバッガ Aki を拡張することで実装して，評価実験を行っている．

「Addistant：アスペクト指向の分散プログラミング支援ツール（Addistant: An Aspect-oriented Distributed-Programming Helper）」では，バイトコードをロード時に編集することで，Java プログラムを半自動的に分散化するシステム Addistant について述べている．分散化のポリシを明示的に分離したファイルに記述することで，アスペクトを分けることができる．また，バイトコード編集の方針を 4 つ提示し，その使い分け方について議論している．

「入れ子関数を利用したマルチスレッドの実現（Implementation of Multiple Threads by Using Nested Functions）」では，言語レベルのマルチスレッドの実現の方法として，入れ子関数を用いる方法を提案している．入れ子関数を，一種のクロージャとして扱い，関数の中に外側の関数の実行再開後の処理を記述し，それを保存して再開後の処理を先行して実行したい場合に呼び出せるようにすることによってマルチスレッドを実現している．また，これを GCC（GNU C Compiler）の入れ子関数を用いて実現するときの性能を改善する方法についても述べている．

「Java バイトコード変換による細粒度 CPU 資源管理（Java Bytecode Transformation for Fine Grain CPU Resource Management）」では，Java VM 上でアプレットなどの信用できないプログラムを安全に実行するために，プログラムに割り当てる CPU 資源を制限する機構を実現する手法として，クラスロード時のバイトコード変換によって対象プログラムに資源管理用コードを挿入する手法について述べ，実行頻度で重み付けされたコントロールフローグラフを用いたコード挿入アルゴリズムを提案している．そして，実アプリケーションでのコード量の増減，実行時間の増減について，定量的評価を行っている．

「名前呼び環境 PCF 計算の意味論（Semantics of Call-by-name Environment Calculus）」では，λ 計算にファーストクラス環境を組み込んだ環境 λ 計算の枠組みとして，再帰演算子をもつ単純型付き λ 計算である PCF を拡張した環境 PCF PCF_{env} を提案している．さらに PCF の名前呼び評価戦略を PCF_{env} に拡張したものを定義し，PCF_{env} を PCF 上で解釈することで，評価戦略と意味論の基本的性質である健全性や計算的適切さ（Computational Adequacy）を証明している．

「最大マーク付け問題の効率的プログラムの自動生成（Automatic Generation of Programs Based on High Level Strategy Description）」では，関数型プログラムの変換手法の 1 つであるプログラム運算を記述する言語 CCP（Calculation Carrying Program）の処理系を提案している．プログラム運算は，大きな効率化をはかることが可能であるが，しばしばその導出には人間のガイドが必要となる．CCP は変換対象のプログラムと変換規則・変換戦略の記述からなり，この処理系では CCP の実行から変換の各ステップを記述する変換スクリプトで対話的に実行可能なものを自動的に生成する．実例として最大マーク付け問題を示し，この処理系が変換規則・変換戦略の開発やデバッグに有効であるとしている．

最後に，研究会開催，論文査読，論文誌編集にご協力を賜りました皆様に深く感謝いたします．