「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なってきた．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三つある．第一は，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．第二は，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．第三は，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ゆきたい．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挙げられる．

- 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伴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，論文のよ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，どれか一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，たとえば次に挙げるような，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か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ない，発表会開催後3週間程度で発表者に採否通知を行なう．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常に受け付けている．

1999年度の発表会予定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17～18日 [プログラミング言語の設計と実装]
8月 4～ 5日 [SWoPP －並列／分散／協調プログラミング言語と処理系]
10月29～30日 [理論]
1月18～19日 [並列・分散処理]
3月23～24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なう．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．必要に応じて，副担当編集委員をおいて，編集作業を分担することもできる．副担当編集委員は編集委員会メンバ以外から選任することもある．

本号の編集にあたって

八杉 昌宏， 舘村 純一

1999年度第2回プログラミング研究会は，1999年8月4，5日に山口県下関市の海峡メッセ下関において，「1999年並列/分散/協調処理に関する『下関』サマー・ワークショップ（SWoPP下関'99）」（8月2日-5日開催）の一環として開催された．

昨年度に引き続き，発表予定者には，従来のSWoPPへの発表申し込みに加えて，プログラミング研究会への発表申し込みも行っていただいた．論文投稿の有無などの指定はプログラミング研究会への申し込みの方にのみ明記することになる．発表申し込みの時期はSWoPPへの発表申し込みの時期に合わせているので，やや早いスケジュールとなっている．

発表枠としては最大14件を想定していた．SWoPPでの他の研究会と比べて件数が少ないのは，発表と質疑の時間を合計45分（発表25分，質疑20分）としているためである．実際の発表スケジュールの調整では，SWoPP幹事の方に大変お世話になった．発表募集を行ったところ，予定していた枠とほぼ同数の13件の申し込みがあった．方式が変更された昨年度，今年度とも，申し込みに対して発表枠が不足することはなく，当初懸念されていた大幅な発表枠不足はいまのところ生じていない．

研究会は8月4日と5日に開催され，SWoPPの他の参加者も交えた活発な質疑討論が行われた．台風の影響も懸念されたが，さほど問題はなく，発表はスケジュール通り行われた．また，論文誌の創刊に伴う変更として従来の研究報告が廃止されているため，発表者が発表資料を必要部数，持参する形となっている．このため，SWoPP参加者は自由に発表資料を入手できるようになっている．今回，最終的には人気のある発表資料が不足する事態に陥ったため今後は注意が必要である．

投稿原稿の査読を議論する編集委員会会合は，編集委員ならびに編集委員会が出席を依頼したメンバーで現地にて開催した．各論文を議論するのに十分な時間を確保するために，会合は4回に分け，昼休みの時間などのまとまった空き時間のほとんどを利用した．十分な議論の後，各投稿論文について担当の査読者を決定し，査読を行ってもらった．

最終的に，投稿を希望したうちの8件の論文（すべて通常論文）が採録となった．これらの論文の掲載に

続き，それ以外の発表については1ページの概要を掲載してある．掲載順序は，論文，概要それぞれについて当日の発表順に従うことにした．

掲載した論文について，以下，簡単に紹介する．

「OpenJIT フロントエンドシステムの設計」では，自己反映計算をベースとして Java 言語の Just-In-Time コンパイラに言語拡張や最適化のためのモジュールを組み込み，動的な言語拡張や最適化を可能とする技術を開発している．この論文では高レベルな中間表現での最適化・特化を支援するフロントエンドシステムの実現について述べている．

「並行論理型言語における同期ポイントの移動の安全性について」では，並行論理型言語において，プロセスが何かを出力するために必要な入力を事前に待ってから実行を進めるようするプログラム変換（同期ポイントの移動）について述べている．この論文では，必要な入力を求めるための抽象実行の方法を示すとともに，同期ポイントの移動の安全性を表示的意味論の点から示している．

「ソースコード変換技術を用いた Java 言語におけるスレッドのモビリティの実現法」では，Java 言語において，実行中のスレッドの（実行状態の）移動をソースコード変換を用いて実現する手法を示している．この論文では，ソースコード変換において，実行状態の保存・復元のためのコードを埋め込むことによる通常実行時オーバヘッドを極力低減化するための工夫について述べている．また，他の言語拡張との組み合わせが容易となるよう，ソースコード変換には拡張可能プリプロセッサを用いている．

「通常の実行効率をほとんど損わないスレッドマイグレーションが可能な C++」は，C++言語において，実行中のスレッドの（実行状態の）移動をソースコード変換を用いて実現する手法を示している．この論文では，ソースコード変換において，実行状態の保存・復元のためのコードを埋め込むことによる通常実行時オーバヘッドを極力低減化するための工夫について述べている．特に C++ コンパイラの最適化を妨げないための工夫を示している．

「OpenMP におけるネストした並列性の実装と評価」では，並列プログラミングを容易に行うための API である OpenMP を用いて並列化されたプログラムの実行を対象として，そのタスクスケジューリングに遅延タスク生成に基づく動的負荷分散機構を導入している．これにより，入れ子状に並列化指示子が書かれている場合でも効率的な実行が可能となっている．また StackThreads/MP ライブラリを用いた実装と，その評価を行っている．

「サブタスク間の依存関係に基づくスケジューリング機構を備えた並列プログラミング環境の開発」では，大規模並列・分散システムにおいて，基盤となるハードウェア環境およびアプリケーションに柔軟に適応できる資源管理機能を備えた並列プログラミング環境 Parsley の設計と性能評価について述べている．Parsley は，並列処理可能なサブタスクを単位として，サブタスク間の依存関係に関する情報をもとに，実行時間を短縮するスケジューリングを行うものである．

「細粒度スレッド対応デバッガのポータブルな実装方式」では，細粒度スレッドの実行効率の高い実装方式におけるスレッド管理の複雑さに対応したデバッガの実装について述べている．そのような細粒度スレッド機構を備えた Cilk などの並列言語の処理系が C 言語を中間言語として用いていることを利用し，C デバッガを改変することなく用いることで移植性を高めている．並列言語レベルにおいてトレースなどのデバックを可能とするために，少数のランタイムフックと C デバッガを制御するインタフェース部を用いたデバッガシステムを提案している．

「Web 対応事務処理スクリプト言語「COBOL スクリプト」」では，金額計算に必要な正確な精度の 10 進形式のデータやその演算機能といった COBOL 言語の長所を，Web を利用した事務処理でも利用できるようにしたなどの特徴を持つ「COBOL スクリプト」の設計と評価について述べている．

最後になりましたが，研究会開催，論文誌編集に御協力を賜りました皆様に感謝いたします．