トピックス

ICFP/PEPM '97 会議報告

浅井健一

1 はじめに

去る6月9日から13日まで，オランダの首都アムステルダムにて，第2回関数型プログラミング国際会議(International Conference on Functional Programming 略称 ICFP '97)，および部分評価とプログラム変換の国際会議(Symposium on Partial Evaluation and Semantics-Based Program Manipulation 略称 PEPM '97)が開かれた．また，これらの会議の直前には Haskell ワークショップと Types in Compilation ワークショップというふたつの併設ワークショップも開かれた．ここでは，そのときの模様を感想を交えて報告する．

2 Haskell ワークショップ

オランダの天気は，概してどんよりしていることが多いそうだが，筆者がオランダについたときはとても良い天気で，日中は少し汗ばむほどであった．そんな中，1週間にわたる国際会議は，まず Haskell ワークショップから始まった．場所は筆者の泊まっている Cok ホテルの地下会議室で，参加者は40人ほどである．ワークショップのまとめ役は OGI の John Launchbury で，その他の主な参加者としては Erik Meijer や Simon Peyton Jones, John Hughes といった遅延評価型の関数型言語を支えてきた人たちをあげることができる．このワークショップは，近年，より広く使われ始めた Haskell について，その表現力，技術的課題， Haskell の今後などを話し合うために開かれたもので，合計9件の発表が行なわれた．これらは，いずれも Haskell の拡張や実装上の問題点の克服が主で，具体的には Haskell の type class を拡張する話や，今までうまく扱えていなかったメモ関数の実装などが発表された．

論文発表の後は Haskell の今後についての議論がなされた．Haskell には，ますますいろいろな機能が付け加えられて便利になる一方，複雑になりすぎている，という指摘もある．この点が議論の中心となり，今後は各種の技術課題の克服に向けて取り組む一方で，Haskell の標準言語仕様(Standard Haskell)を定めて固定化し，より広いユーザの獲得をはかることとなった．この標準化の作業には Haskell の生みの親とも言うべき John Hughes があたることになった．詳しくは Standard Haskell のホームページ(http://www.cs.chalmers.se/~rjmh/Haskell/)を参照していただきたい．

3 Types in Compilation ワークショップ

翌日，場所を CWI に移して Types in Compilation ワークショップが開かれた．まとめ役は CMU の Robert Harper とボストン大学の Robert Muller である．参加者はおよそ80人と盛況で，当初 Cok ホテルの地下会議室で行なう予定だったのを CWI に場所を移しての開催となった．

招待講演は Mitchell Wand で，題は "Types in Compilation" とまさにワークショップの名称そのものである．ここでは，比較的，概念的な話がなされた．型の大雑把な導入があった後，型再構成とフロー解析がますます近くなってきていることが指摘され，その関係を明らかにする必要性が述べられた．

一般講演では合計10件の発表があった．中でも Franklyn Turbak らによる Church プロジェクトの

A report on ICFP/PEPM '97

Kenichi ASAI, 東京大学大学院理学系研究科情報科学専攻, Department of Information Science, Graduate School of Science, University of Tokyo.

コンピュータソフトウェア, Vol.15, No.3 (1998), pp.64–68.
[トピックス]1997年11月3日受付.

発表は，インタセクション型を使った型システムを示し，それを使ってフロー解析をするというもので，招待講演で指摘された型再構成をフロー解析に使った例として注目を集めた．

近年，コンパイルにおける型の重要性は広く認識されており，そのせいもあって，多くの活発な議論がなされた．このワークショップは，次回は日本 (京都) での開催で，日程は 1998 年 3 月 25 ~ 27 日，論文締切は 12 月 8 日である．

なお，このワークショップで発表された論文は，ボストン大学のテクニカルレポート [2] として入手することができる．

4 ICFP '97

ICFP は Lisp and Functional Programming (LFP) と Functional Programming Languages and Computer Architecture (FPCA) というふたつの関数型言語系の国際会議が統合されてできた国際会議で，今年で 2 回目になる．今年の投稿論文数は 78 で，その中から 25 件が採録された．そのうち，日本からの論文は 3 件である．組織委員長は Glasgow 大学の Simon Peyton Jones，プログラム委員長は Copenhagen 大学の Mads Tofte で，参加者は約 100 名ほどであった．

4.1 招待講演

招待講演は 2 件，行なわれた．1 件目は初日の最初，Cornell 大学の Greg Morrisett によるもので，題は予定では "Types in compilation" であったが，前日に行なわれたワークショップの招待講演の題名と重なったためか，"Compiling with types" に改められた．

講演の内容は，一言でまとめるならば，型をコンパイル中にも大いに利用し，コンパイラの信頼性を向上させよう，というものである．型の情報は，プログラムの信頼性を向上させるばかりでなく，最適化の助けとして近年，いろいろなところで利用されている．しかし，それらはコンパイルの最初の段階でのみ使われ，一度，型チェックを通ったら捨てられてしまうのが普通であった．そのため，コンパイラで行なわれる種々のプログラム変換は型のない世界で行なわれ，プログラム変換の信頼性を下げてしまう結果となっていた．

ここでの提案は，型情報を構文チェック後も捨てずに保持し，コンパイラの扱う中間言語も型付きにすること，すなわち「型付き中間言語 (Typed Intermediate Language 略して TIL)」の採用である．ユーザの入力したプログラムだけでなく，コンパイラの中の各プログラム変換後においても型が正しいことをチェックすることでプログラム変換の信頼性を高めよう，というわけである．

このままでは，コンパイル中に何度も型チェックが行なわれるためコンパイル効率の低下が予想されるが，一度，信頼性の高いコンパイラが出来上がってしまえば，後で途中の型チェックを省くことは簡単なので，効率上の問題はあまりないという主張である．型付き中間言語は，この招待講演ばかりでなく他の発表でも多く聞かれた言葉で，この世界のひとつのキーワードになってきているようである．

2 件目の招待講演は，3 日目の最初に Ericsson Telecom の Joe Armstrong によって行なわれた．題は "The development of Erlang" で，実地で使われている Erlang という言語の紹介である．実地で使うためには，大規模分散処理をサポートしていること，リアルタイムで反応できること，など多くの厳しい条件があり，それらがひとつでも欠けると使い物にならなくなってしまう．Erlang では，関数型言語や論理型言語の考え方を採り入れつつ，それらをうまく克服しているのだそうである．あまり技術的な話はなく，「システムを動かしたままシステムの一部を取り替えるのに成功した」などの苦労話が紹介された．

4.2 一般講演

ここでは，25 件の一般講演のうち，筆者の印象に残ったものをいくつか紹介していく．

Allyn Dimock, Robert Muller, Franklyn Turbak, J. B. Wells "Strongly typed flow-directed representation transformations" では，強力なλ閉包表現の変換方法を提示している．インタセクション型とユニオン型の機能を持つフロー型を導入し，型システムを使ってデータが作られる場所と使われる場所の関係を特定する．そして，作る側と使う側で矛盾が起きないように，データ表現を最適化するのである．従来の Closure Conversion

や inlining は特殊なケースとして含まれている．型付き中間言語を用い，型システムを使って解析を行なう，という最近はやりの手法を存分に使っている．

Martin Erwig "Functional programming with graphs" では，従来，扱いにくかったグラフをデータ型として表現し，関数型言語でリストを扱うのと同じような感覚で，グラフも扱えるようにしよう，という試みを紹介している．グラフ上の fusion などもできており，面白い試みではあるが，効率の方にはまだ問題があるようである．

Chris Okasaki "Catenable double-ended queues" では，ほぼ定数時間で append が可能な両方向出し入れ可能キューの実装法を純粋な関数型言語で示している．キューの構造を必要に応じて少しずつ変化させていくことで (amortized の) 定数時間を達成している．この論文の著者は毎年のように lazy な言語上でのデータ構造の実装法を提案している人で，遅延評価の動きがすべて把握できているようである．

J. Michael Ashley "The effectiveness of flow analysis for inlining" では，inlining においてフロー解析がどのくらい有効かを調べ報告している．inlining をするときには，まずフロー解析を行ない，ある変数には特定の関数しか来ない，という情報を得て，その呼び出しを inlining する．どのくらい inlining できるかは，解析時にどのくらいその情報を正確に得られるかにかかっている．その解析として polyvariant なものを使えば，確かにより多くの変数の値を正確に求めることができるが，コストもかかる．一方，sub0-CFA と呼ばれる精度の解析を用いると，それなりに低いコストでかなりの inlining ができることが報告された．ここで，sub0-CFA とは，monovariant なフロー解析で，解析の際に行なう反復の回数を 2 回など定数回で打ち切るものである．

Zhenjiang Hu, Hideya Iwasaki, Masato Takeichi, Akihiko Takano "Tupling calculation eliminates multiple data traversals" では，Tupling できる関数を hylomorphism と呼ばれる形の関数に制限することで，従来，難しかった Tupling するべき関数の発見，および新しい関数形の構築を機械的に行なえるようにした．Tupling の候補になる関数を hylomorphism に限定すると，通常の fusion 変換と同様の機械的操作によって Tupling が可能であることが示されている．

Julia L. Lawall, Harry G. Mairson "On global dynamics of optimal graph reduction" では，λ計算のグラフ簡約における最適な簡約法が提示された．ここで，グラフ簡約が最適である，とは，β基の簡約化される数が最小である，と定義されている．従来のグラフ簡約では，λ閉包の適用時にそのλ閉包の本体が複製されるため，必ずしも簡約数が最小にはならなかった．ここで提案された方法では，本体を複製する際，必要最小限の部分のみを複製し，残りを共有することで，簡約数を最小にしている．

Dirk Dussart, John Hughes, Peter Thiemann "Type specialization for imperative languages" では，Hughes の提案した Type specialization を副作用などを含む命令型の言語に拡張した．Type specialization とは，部分評価を行なう際，プログラムばかりでなく，型も特化してしまうもので，constructor specialization や data type specialization といったものの一般化になっている．ここでも，解析を行なうかわりに型システムを使っている．

Olin Shivers "Automatic management of operating-system resources" では，関数型言語から OS のリソースにアクセスするインターフェースを与えている．関数型言語ではストアに対するアクセスは GC を通じて，自動的に制御されている．これを OS の他のリソース (プロセス，I/O チャネル，シグナルなど) にも拡張しよう，という試みである．著者は実際に scsh という Scheme をベースにしたシェルを作成，公開している．

一般講演とは別に，別室ではポスターセッションも開かれ，16 件の発表が行なわれた．それらの内容は論文集の最後に各々 1 ページずつで要約されている．

なお，第 3 回の ICFP は 1998 年 9 月 27 日から 30 日まで，アメリカ，メリーランド州のバルティモアで開かれる．論文締切は 4 月 15 日である．今回に引き続いて，日本からの積極的な論文投稿を期待したい．

5 PEPM '97

PEPM はその名前が示す通り，部分評価を中心としてプログラム変換全般を視野に入れている会議である．ほぼ 2 年おきに開かれており，会議の開かれない年には

ワークショップなど小規模な集まりが開かれる．今年は会議の年にあたり，ICFP に引き続いて行なわれた．

投稿論文数は 32 で，例年に比べると少なかったが，これは部分評価の技術がほぼ定着しつつあり，特に PEPM ではなくても ICFP や POPL, PLDI や OOPSLA といった他の大きな学会に出す人が増えているためのようである．会議ではこのうち 17(うち日本からは 1)が採録されている．プログラム委員長は Charles Consel がつとめた．

5.1 招待講演

招待講演は Glasgow 大学の Satnam Singh で，題は "Using Partial Evaluation to describe Dynamic Hardware" である．近年，部分評価の技術は様々なところで応用されているが，この招待講演は，その影響がハードウェアまで及んでいることを示している．講演の内容は，FPGA などの回路を自由に変更させることができる素子に注目して，動的に回路自体を効率の良い物に変えてしまおう，というものである．例えば，一般的な乗算器の回路に対し，片方の入力が常に 2 であることがわかれば，それで特化することにより乗算器の回路は 1 ビット左シフトの回路になる．

ゲートレベルでは，AND ゲートの片方の入力が 1 であることがわかったらもう片方の入力を直接，出力とし，0 であることがわかったら定数 0 を出力する，といったことが行なわれている．FPGA では，各ゲートの再配置法が問題になるが，これに関しては今のところ何もしておらず，今後の課題である．しかし，ゲートがただの結線になるだけでも相当の速度向上が得られるため，現在のアプローチでも全体としてはかなり速くなるそうである．

5.2 一般講演

Luke Hornof, Jacques Noye "Accurate Binding-Time Analysis for Imperative Languages: Flow, Context and Return Sensitivity" は，フランスの IRISA で行なわれている TEMPO プロジェクトの一環で，C 言語に対して十分な部分評価を行なうための束縛時解析 (Binding Time Analysis) に必要な精度について報告している．具体的には，データフローの情報，コンテキストの情報，そして関数の返り値の情報が必要とされ，逆にこれらの情報が正しく推論できていれば Sun の RPC などのプログラムも十分に特化できるようである．

David Melski, Thomas Reps "Interconvertibility of Set Constraints and Context-Free Language Reachability" では，集合についての制約解消の問題と文脈自由文法の到達可能性の問題とが相互に変換可能であることを示している．この仕事の動機は，最近の解析でよく用いられる集合の制約解消に対する洞察を得ることにある．集合の制約解消は普通，プログラムの長さを n とすると $O(n^3)$ かかる．これは cubic bottleneck と呼ばれ，何とかこの壁を破れないかと多くの研究がなされてきた．この論文では，それを，すでに $O(n^3)$ かかることがわかっている文脈自由文法の到達可能性に変換できることを示すことで，その壁が破れないことを示している．

Olivier Danvy, Ulrik P. Schultz "Lambda-Dropping: Transforming Recursive Equations into Programs with Block Structure" では，部分評価後に得られるフラットな関数定義の列をブロック構造を持ったプログラムに変換する方法を示している．普通，部分評価を行なう際には，コードの再利用をするために，まず λ-lifting をしてネストのないフラットな関数定義にしてから部分評価を行なう．しかし，このようにすると結果もフラットな関数になってしまう．ここでは，そのフラットな関数に対し，まず，呼び出し関係をグラフで表現して強連結成分分解し，その構造に従ってブロック構造を作る．そして，不要なパラメタの受け渡しをなくすことで最終的な結果を得ている．

普通，部分評価後のコードを見ることはないので，こんなことをしても役に立たない，と思いがちだが，ブロック構造にした方が局所性が上がりその後コンパイラによる最適化が可能なこと，受け渡すパラメタの数が減るためレジスタがあふれにくくなること，などのため，コンパイル時間，実行時間，ともにかなりの速度向上が見られている．

Lennart Augustsson "Partial Evaluation in Aircraft Crew Planning" では，ドイツの航空会社 Lufthansa で実際に使われている乗務員スケジューリングシステムを部分評価を使って効率化した，という

話が紹介された．乗務員スケジューリングシステムというのは，各フライトに乗務員を割り当てるシステムで，そこには「乗務員は出発地に戻ってこなくてはいけない」「フライトとフライトの間には所定の休み時間がなくてはいけない」「週に一定回数以上，乗務してはいけない」等々の多くのルールがある．これらのルールは比較的頻繁に変更されるために，高レベルな専用言語で記述され，コンパイラにより C コードへ変換される．しかし，普通，これらのルールの数は膨大になるため，コンパイル時間および実行効率が大きな問題となっている．そこで，ルール記述言語のレベルで部分評価を適用し，効率化を試みた．ここで行なわれている部分評価は定数伝搬のような単純なもののようだが，効果は大きく，ルールの大きさは約 40%となり，そこから生成される C のコードサイズも半減した．その結果，最終的なオブジェクトコードを得るまでのコンパイル時間 (部分評価に要する時間を含む)，およびその実行時間は，ともに半減した．さらに，出力されたルールを乗務員スケジューリング職人がじっと眺めたところ，前よりも良いルールを発見できたこともあったそうである．

Saumya Debray “Resource-Bounded Partial Evaluation” では，システムのリソースを考慮に入れた部分評価法を提案した．部分評価を行なうときには普通，止まらなくなることがない限り，関数呼び出しは展開される．これは，展開した方が関数呼び出しのオーバーヘッドが減り，速度向上が見込めるからである．しかし，あまり展開しすぎるとその関数がキャッシュに入り切らなくなり，かえって遅くなってしまうこともある．ここでは，そういった情報も考慮に入れて行なう部分評価法を示している．具体的には，各基本命令のコストがわかっているとして，そこからその定義を展開したときのコストと速度向上を見積り，それが + になるときのみ展開を行なう．部分評価を必ず停止させるための道具と見ることもできる．

Walid Taha, Tim Sheard “Multi-Stage Programming with Explicit Annotations” では Lisp の quote, unquote, eval のようなものを型付きの言語に入れて，マルチレベルのシステムを実現している．これにより型付き言語の中でもプログラムを自由に扱えるようになるため，部分評価器の作成なども Lisp と同じような感覚でできるようになる．導入された機構は Scheme の hygenic macro に近いもののようだが，quote とは違って型がきちんと付くものしか書けないため，より安全なものになっているようである．コードを変更した上で実行する，などということもできるので自己反映計算とも関係が深い．

次回の PEPM は 1999 年 1 月にアメリカ，テキサス州のサンアントニオで POPL と共に開催される予定である．

6 終わりに

ワークショップふたつと国際会議をふたつ，すべてに参加すると，1 週間かけて全部で 85 の発表を聞くことになる．いささか大変なスケジュールではあったが，関数型言語の最先端をかなりの部分，網羅しており，充実した内容だったように思う．日本からも複数の論文が通っており，今後の発展が楽しみである．

なお，筆者は今回の会議に参加するにあたり，情報科学国際交流財団から援助を受けました．

参考文献

[1] *Proceedings of the Haskell Workshop*, Amsterdam, June 7, 1997.
[2] Workshop on Types in Compilation (TIC97), Boston College Computer Science Department Technical Report BCCS-97-03, Amsterdam, June 8, 1997.
[3] *Proceedings of the ACM SIGPLAN International Conference on Functional Programming (ICFP '97)*, Amsterdam, June 9-11, 1997, ACM SIGPLAN Notices, Vol. 32, No. 8, 1997.
[4] *Proceedings of the ACM SIGPLAN Symposium on Partial Evaluation and Semantics-Based Program Manipulation (PEPM '97)*, Amsterdam, June 12-13, 1997, ACM SIGPLAN Notices, Vol. 32, No. 12, 1997.