## ㉛ Transformation Rules for GHC Programs

K.Ueda ( ICOT , 日本 )

発表要旨

（Flat）GHCプログラムの変換規則を紹介する。これらの規則はunfold/foldに基づいている。新しい点として同一化を束縛の方向づけによって行うという点がある。また，これまでのシステムと比べるとモードのないシステムが想定されている点と，foldのルールが含まれているという点でより一般性の高いものになっている。

意味的なモデルは複数の集合の（代入を通じての）処理の可能な有限の列にゴールの複数の集合を連想するものである。変換によってユニフィケーションが失敗する可能性なしに作られる処理列の集合は保存される。

このモデルはユニフィケーションの失敗やデッドロックといったような変則的な事象も扱う。また，変換技術を応用することによって，並行プロセスの通信も融合して含むことができる。

質疑応答

**質問**：あなたは意味論を処理列の集合であると述べたが，プログラムからその集合をどうやって導くかは述べなかった。説明して欲しい。

**回答**：意味論には内部的なものと外部的なものと2層ある。内部的な意味論はreductionの列である。これはただの操作的意味論である。外部的な意味というのは抽象化によって得られる。いくつかのreductionをまとめたものを処理する。処理は，有限個のreductionからなる。

**質問**：導出可能な処理の技術的な定義をしてほしい。

**回答**：これは，基本的には操作的意味論である。たとえば，正常な処理はreductionの列をいくつか見つけることによって得られる。これは，観察可能な出力を代わりに作ることができる。それを人間が観察し抽象化することによって処理の列が得られる。これは，複雑だが，方法としては安定している。

**質問**：FORTRANにおいて0で割るようなことがGHCにおける失敗であるということなのか。

**回答**：GHCにおける失敗というのは例外的なことであって普通のプログラムでは起きないことだと想定している。

**質問**：普通のunfoldingな物と比べてcase splitされたものは節の数が増えると思われるが実際にはどうか？

**回答**：case splitはgoalと節の全ての対を考えるが，ほとんど全ての対がunify不可能である。そのため，実際の例では増える節は非常に少ない。

**質問**：あなたは，失敗がGHCにおいては例外的なものだと言ったが，メタインタプリタの中でotherwiseを実現するにはどうしたらよいか？

**回答**：現在のところGHCではotherwiseを定義していない。otherwiseを実現するとしたら，他の節

の否定を並べることになるだろう。そして，この定義に基づくGHCでは強力なメタインタプリタを書くことができない。

## ㉜ A Tutorial Introduction to Metaclass Architecture as Provided by Class Oriented Language

P.Cointe ( Rank Xerox , フランス )

発表要旨

クラス／インスタンス関係を持つオブジェクト指向言語におけるリフレクションについて述べている。

システム内部に自分自身の記述をfirst class dataとして持つことは，リフレクションの実現に不可欠である。オブジェクト指向言語では，クラスがオブジェクトの構造 / 動作を記述しており，クラスそのものをオブジェクトとして扱うためにメタクラスが必然的に導入される。

本論文では，Smalltalk，ObjVlisp，CLOSについて，それぞれのメタクラス構造を解説している。さらにメタクラスを用いたリフレクティブなプログラム例をいくつか示している。

発表ではSmalltalkを例にとり，そのメタクラス・カーネルを解説した。そしてエラーハンドリングを行うためのメタクラスや，Thinglabシステムを構成するためのクラス / メタクラスを例として挙げていた。

質疑応答

**質問**：論文ではシンプルな例で説明してあるが，これをフレームのような知識表現に応用してみたことはあるか？

**回答**：特にそういうことはまだやってみていない。現在のところ我々はメタクラスベースのプログラミングスキーマを研究しているので。

**質問**：メタクラスによって，ハイレベルの知識表現がより簡単に実現できて興味深いと思う。

**回答**：そのとおりで，例えばCLOSでは各スロットもオブジェクトとして扱えるので，デフォルト値や要求駆動などは簡単にインプリメントできる。