㊲ Guarded Horn Clause Languages: Are They Deductive and Logical ?

C.Hewitt and G.Agha (MIT,米国)

発表要旨

guarded Horn clause languages（GHCL）のよいモデルは，proof theoryでは与えられない。これらの言語は，プログラムからは導けないarrival order assumptionを動的に付加する。したがって，プログラムはlogical dedectiveシステムを記述しない。また，GHCLの実行モデルは，通信の"送達の保証"を仮定しているが，GHCLプログラムの論理的解釈の表現力は，そのような保証からの帰結を推論するには不充分である。アクターのsemanticsは，GHCLの適切なモデルを与えることができる。

質疑応答

質問：グローバルクロックかが必要になるのは分かりましたが，あなたは，一般にstraight-forwardな…likeな宣言的semanticsに関して，より一般的な論理については絶望しているようですが，serialiser，あるいはグーロバルタイムの概念（notion）を用意して，それを言語に加えると，どういうことになるのですか。

回答：グローバルクロックについてちょっと説明させてください。グローバルクロックの概念（notion）について，間違いがあるようです。このシステムには，グローバルクロックはありません。グローバルクロックは，retroactiveな概念（concept），…な概念（concept）です。異なるグローバルクロックは，…。これが意味するところは，システム内で，誰かがイベントを観測しており，その参照によって，イベントの直線的な順序が決まる，ということです。因果律，あるいは並列処理の法則が述べているのは，すべてのイベントは，あるオブザーバによって順序づけることができる，ということです。

質問：しかし，何か足りないように思います。何故なら，何らかのfirst-orderのformulationによって，このaxiomatisationを与えられるとは思えないからです。それができるとすると，first-order logicと任意のHorn clause logicがcomputabilityとどういう関係にあるかについて，非常に強力な結論が得られることになり，この，オープンであることに対する要求と，何かが本質的にnon-deductiveであるという主張は矛盾するように見えます。

回答：実際のところ，universeの小さな部分，たとえば，FGHCプログラムのbodyの内部，あるいはアクタープログラムのbodyの内部などですが，これに関する，理想化され，axiomatisedされた，マイクロセオリーを持つことには問題はありません。

コミットメントが衝突する，複数の参加者が関与するオープンなシステムでは…したがって，"What's going on?"といいたくなるような，矛盾が存在することになります。logical deductionは，矛盾するようなステートメントがあるような領域においては，非常に適したツールではありません。

質問：では，高いレベルにおけるaxiomatisationは行うべきではないと？

回答：いいえ。各参加者（participant）はそれぞれ独自のマイクロセオリーを持っていなければなりません。これは，各個人（indivisual）にとって，非常に重要なツールになります。矛盾が生じるのは，２人の個人の，複数のマイクロセオリーの間でです。それらを大きな整合するセオリーにまとめあげることはできません。

質問：それは，deductionを使うべきではないということではないのですか。

回答：いいえ。グーロバルな状況がない時には，複数の参加者が無い時には，依然として非常に強力なテクニックです。

もう１つの論点としては，deductionは，これらの言語のsemanticsに起こっていることを表してはいない，ということがあります。例えば，transition systemか何か，別のものが必要で，この言語について，アクター言語についてと同様に，何が起こっているかを説明することとなります。

質問：もっと議論のための時間があるといいのですが。ここでは，少しコメントを述べるだけにします。並列論理型言語を，全ての宣言的側面を無視して，完全に操作的に理解することはできません。イデオロギーはともかく，完全に操作的な基盤に立って，このモデルを理解しようと試みることはできます…あるモデルを別のモデルに埋め込むのが容易かどうかを調べることができます。私が思うには，並列論理型言語にアクターを埋め込むことについては，いくつかの肯定的な例があるようです。アクターを並列論理型言語に埋め込む場合，例えば，VulcanをFCPに，あるいは，A'UMをflat GHCに，という場合，などです。今の発表の中では，逆の埋め込みについて言及していました。並列論理型言語をアクターに埋め込んだ具体的な例はあるのでしょうか。

回答：既にあります。第一にその埋め込みは行われていますし，第二に，その意見には不賛成です。アクター言語をGHCLに埋め込む方法については，あまりよく分かりません。というのは，GHCLは，静的な構造…reflective capabilitiesについて，たとえて言えば，Common LispとSchemeのようなもので，Schemeはある程度のreflective powerを持っていますが，それは，Common Lispのような言語では明らかでないものです。例えば，call/ccのようなものです。

質問：実際にそういったことは行われています。A'UMはflat GHCにコンパイルされることによって，実際に走っています。

回答:A'UMが充分発達した言語かどうかは，完全に理解しているとはいえないので，私には分かりません。

質問：問題は，埋め込みがどの程度自然に行えるか，です。

回答：私が知る限り，すべての例について，straight-forwardな形で行われています。

質問：第２の点は，GHCLをdeclarative semanticeを並列論理型言語として見ようとすると，同期や，非決定性などの側面を無視して，純粋に論理的なプログラムとして，捉えると，いくらかの情報が失われてしまうのです。抽象化を行って，宣言的な側面だけを眺めるわけにはいきません。しかし，１時間ほど前に紹介された試みのように，declarative semanticeを並列論理型言語に与えようとした例はあります。この場合，domain…domainは単なる置換ではなく，置換の系列であり，これはデッドロックしたり，失敗したりします。したがって，おそらく…domainでも，これらのdomainの上に関係を定義するaxiomとして，並列論理型言語を宣言的に読むことはできます。

回答：インプリメンテーションの観点からみると，アクターモデルはハードウェアにより近いと思い

ます。…アクターは，実際のところ，マシンの内部で起きる，マルチコンピュータ上でのメッセージパッシングなどを表しています。したがって，GHCLは，インプリメントの際には，アクターモデルである，メッセージパッシングによって作られることになると思います。

㊳ Lazy Evaluation of FP Programs: A Data-Flow Approach

Y.H.Wei（IBM,米国）

発表要旨

リストを扱う関数型言語に遅延評価を導入したシステムの紹介をする。ベースとなる言語は，バッカスのFPである。これに，どのデータが要求されているか，という情報を返す関数や，リストのこの位置にあるものは必要とされている，ということを示すオブジェクトを加えた，FPのスーパーセット，DFPを導入する。

こうした上で，FPのeagerなプログラムを，それと同じ結果を返す，DFPのlazyなプログラムに変換する。

このlazyなプログラムは，必要なデータのみを，きちんと全て評価するという，効率的な性質をもっている。また，この様な制御を行うと，効率があがるばかりでなく，無限の長さを持ったストリーム（ここでは自然数の自乗が無限にならんだ数列の例が示された）など，eagerな制御をしたのでは，扱うことのできないデータを取り扱うことができる。

質疑応答

質問：この様な，色々な関数記号，オブジェクト記号をあらたに取り入れたプログラムに変換すると，オーバーヘッドが伴うのではないか，と予想される。講演のなかでは，効率の良くなる例が示されたが，そうでない例もあるのではないかと思われる。如何？

回答：オーバーヘッドは確かにあるが，必要なデータのみを評価するという，利点の方が，大きい。