

Flat Concurrent Prolog のインプリメンテーション

これは、Taylor 氏の行なつた Flat Concurrent Prolog (FCP) のインプリメンテーションの概要である。

1. FCP

FCP は Concurrent Prolog に対し次のような制約を入れた言語である。

- ガード部のリテラルはテストだけを行なう (変数のインスタンス化を行なわない) システム述語に限る。

ガード部のリテラルはテスト述語だけであるから、Concurrent Prolog のようにガード部の計算が何重にもネストされることはない。この制約により、Concurrent Prolog のプログラミングの語彙をあまり失うことなしに、Concurrent Prolog をインプリメントする際の様々な問題点を回避することができると設計者である Shapiro が主張している。

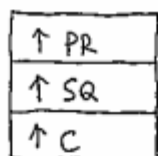
## 2. イニクレーション

### 2.1 リゾリューション

ゴール  $G$  が与えられたときに、 $G$  をリゾルブする節の探索は逐次的に行なわれる。まず、空の suspension record というものを用意し、各節について順々にコミット可能な節がどうかを実際にヘッド・エニフィケーション、ガード部のラスト述語の実行を行なうことにより確かめる。この際、もし節がコミットできない節の場合、ヘッド・エニフィケーションで行な、 $\tau$  バインディングを  $\tau$  から  $\tau'$  に戻す。ただし、コミットできない理由が suspension によるものであった場合はこの節と suspension の原因とを、 $\tau$  変数を suspension table に登録しておく。逐次的に節を探索する過程で、コミットできる節が見つかる、その時点で以後の探索を打ち切り、その節のボディ部でゴール  $G$  を置き換えてしまう。コミットする節がなく、 $\tau$  への節を調べ終ったときは、suspension table を調べ、それが空の場合はゴールを失敗させ、そうでない場合は次節に述べたように suspension record を作る。

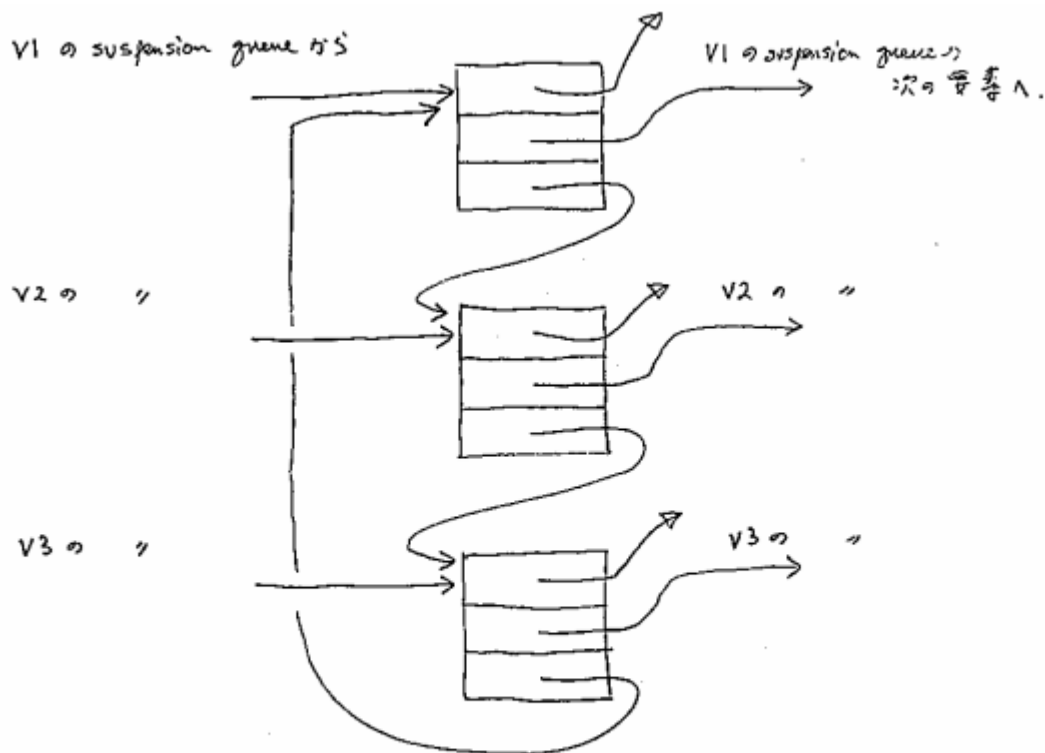
## 2.2 suspension record

ゴールをリゾルブする辞 (コミットできる辞) がなく、かつ、suspend している辞が1つ以上ある場合には、suspension Table の内容に基づき次のような suspension record が作られる。今、suspension Table に3つのエントリ、 $(c_1, v_1)$ ,  $(c_2, v_2)$ ,  $(c_3, v_3)$  があつたとき、ただし、 $c_i$  は辞の識別子、 $v_i$  は suspend を起した変数へのポインティである。このとき、次のような words のアーク構造が各エントリごとに作られ、



↑ PR : ジョビエスワードへのポインティ  
↑ SQ : suspension queue の次要素へのポインティ  
↑ C : エントリ向のポインティ

それぞれが、各  $v_i$  の suspension queue に入れられる。



将来、変数  $V_i$  が non-variable term にインスタンス化された場合は、 $V_i$  の suspension queue の各要素に対して、その PR フォールドにあるものを ready queue に登録し、つなびた <sup>FCP</sup> ている  $V_j$  の suspension record をそれが属している suspension queue から取り除く。

### 3. Concurrent Prolog インタプリタとラージョンとの比較.

Concurrent Prolog のインタプリタとラージョンと FCP のインタプリタとラージョンを比較した場合、FCP には次のような利点がある。

(i) プロセス管理関連で用いるデータ構造が単純になる。

(ii) Concurrent Prolog の場合のように AND-OR 木の深い大域的なデータ構造を必要としない。

### Curriculum Vitae

Stephen Taylor

Stephen Taylor received a H.N.C. Degree in Electrical and Electronic Engineering in 1977. He has been involved in Industrial Research at S.T.C., Marconi Avionics, British Aircraft Corporation, N.C.R. and C.D.C. He received a B.S. degree in Computer Science from Essex University, England in 1982.

He is presently studying for a P.H.D. Degree in Computer Science at Columbia University, New York, U.S.A. His research interests include Artificial Intelligence Techniques, Logic Programming Languages and Parallel Architectures. Since joining Columbia he has been associated with the DADO project which is a special purpose parallel Architecture for Artificial Intelligence applications.