6Q-7

マルチ PSI/V2 におけるコード形式について

宮崎芳枝[1], 中越靖行[1], 堂前慶之[1], 稲村雄[2], 木村康則[2], 中島克人[2]
1: (株) 富士通ソーシアルサイエンスラボラトリ
2: (財) 新世代コンピュータ技術開発機構

1 概要

第五世代コンピュータ・プロジェクトの一環としてマルチ PSI/V2 システムを開発している。マルチ PSI/V2 上には並列論理型言語 KL1 の分散処理系が実装されている。本稿ではこの KL1 分散処理系における、プログラムコードの形式ついて述べる。

現在のところコード生成はマルチ PSI/V2 のフロントエンド・プロセッサである PSI-II 上のクロスシステムでおこない、フロントエンド・プロセッサからマルチ PSI 本体内の要素プロセッサの 1 つにロードされる。その後コードは実行時に必要に応じて各要素プロセッサに分散される。

2 モジュール形式

2.1 モジュール

KL1 プログラムは述語の集まりであるモジュールという単位にまとめて記述されている。KL1 プログラムはモジュール単位にコンパイルされ、それらがリンクされて、ロードされる。デマンドロードやロードしたコードが不要になった時のガーベジコレクション (以下 GC) を容易にするため、モジュールは KL1 の通常のデータ型の一つとして扱われる。

2.2 GC 領域とコード領域

述語を呼び出すために張られたポインタは GC 時には張り直さなければならない。ポインタにはモジュール内の述語同士に張られたものとモジュール外の述語に張られたものがある。

モジュール内の述語呼出しに関してはポインタを相対アドレスで表現することにしておけば、他のアトミックなデータと同様に扱うことができて GC 時のメンテナンスは不要になる。GC 時にはモジュール単位にコードが移動するためモジュール外の述語呼出しのポインタと構造体定数 (3 章) はメンテナンスの対象となる。このような絶対アドレスポインタがモジュール中に占める比率は小さい。もし、絶対アドレスポインタをモジュール内の一個所に集めることが出来れば、GC 時にはモジュール全体をメンテナンスの対象とする必要は無くなり、効率が良い。そこで、モジュールを絶対アドレスポインタを含む領域 (GC 領域) とそれ以外の領域 (コード領域) の 2 つに分けた。具体的には、GC 領域にはモジュール外述語呼出し用の述語記述子 (2.3 章) と構造体定数への絶対アドレスポインタが格納される。コード領域には命令語、モジュール内相対アドレス、整数・アトムなどのアトミック・データのみが入っている。

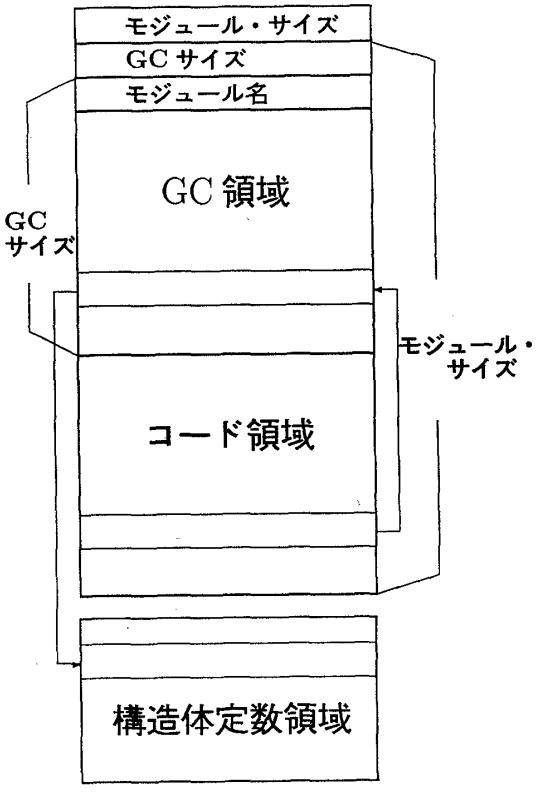

図 1: モジュールの形式と構造体定数領域

Code representation of Multi-PSI/V2
Yoshie MIYAZAKI [1], Yasuyuki NAKAGOSHI [1], Yoshiyuki DOUMAE [1],
Yu INAMURA [2], Yasunori KIMURA [2], Katsuto NAKAJIMA [2]
1: Fujitsu Social Science Laboratory Ltd. 2: Institute for New Generation Computer Technology

2.3 モジュール外呼び出し

モジュール外述語の呼出しは、モジュールへのポインタ、述語名アトム、引数個数の3つの情報を用いて行なわれる。呼び出される側のモジュールのコード領域の先頭には述語エントリ表と呼ばれるハッシュ表が置かれる。述語名アトムと引数個数をエンコードした値をキーとしてこの表を検索することにより、呼び出すべき述語のコードアドレスを求めることができる。

呼出し側モジュールには各呼出し述語に対して4語の述語記述子がGC領域中に設けられ、以下のような情報が格納される。

- モジュールフィールド
  呼出し先モジュールへのポインタを格納する。
- 述語名フィールド
  呼出し先モジュールの述語エントリ表を検索するためのキー、即ち、呼出し述語の述語名アトムと引数個数をエンコードした値を格納する。
- モジュール名フィールド
  呼出し先モジュールのモジュール名アトムを格納する。これはデバッグ用である。

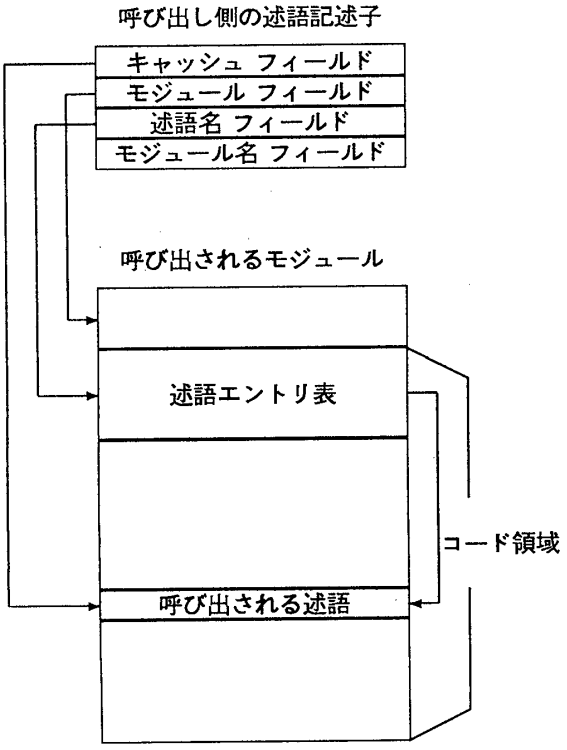

図 2: モジュール外呼び出し

- キャッシュフィールド
  モジュール外呼出し述語のコードアドレスを直接格納する。一度呼出した述語のアドレスをこのフィールドにキャッシュしておくことにより、2回目以降の呼出しを高速化することができる。GC時にはこのフィールドはクリアされる。

3 構造体定数の処理

3.1 構造体定数

構造体定数とは、プログラム中に陽に記述される構造体の中で、その要素として変数を含まないものの総称である。例えば、ベクタ{a, b}や8ビットストリングascii#"program"、あるいはネストした構造体[a,{b, c}, d]などである。WAMをベースにした処理系では構造体は実行時に生成されるが、変数を含まない構造体定数の場合はコンパイル時に予め生成しておくことができるため、実行時間および生成のための命令コードの両方の節約が可能である。定数表を検索するようなプログラムでは特に効果が大きい。

3.2 構造体定数の生成

コンパイラはソースプログラムに構造体定数を見つけると、構造体をモジュールの外部に生成し、その構造体への絶対アドレスポインタをモジュールのGC領域に置く(図1参照)。そしてコード中のその構造体を使用する場所には、

```
put_structured_constant   Ai, Lab
```

なる命令を配置する。この命令は構造体定数へのポインタを引数レジスタAiにロードするためのもので、Labは GC領域中の構造体へのポインタの格納場所を示す相対ポインタである。

謝辞

日頃御指導頂いている内田室長をはじめとするICOT第4研究室の方々に感謝します。

参考文献

[1] Y.Kimura and T.Chikayama: An Abstract KL1 Machine and Its Instruction Set, Proc. of SLP'87

[2] K.Nakajima et al.: Distributed Implementation of KL1 on the Multi-PSI/V2, ICOT TR-439

[3] D.H.D.Warren: An Abstract Prolog Instruction Set, Technical Note 309, Artifical Intelligence Center, SRI, 1983