6N-1

木構造機械語型計算機のアーキテクチャ

有田　隆也　森下　巖
(東京大学工学部)

1．はじめに

従来の汎用計算機アーキテクチャの研究において、オブジェクトプログラムが高水準であるか、あるいはハードウェアとソフトウェアの分担はどうするかというテーマに関しては、比較的よく議論されてきたが、オブジェクトプログラムを一次元のテキスト列でなく、なんらかの構造をもつものとして扱うアーキテクチャについては、あまり検討されていない．

本稿では、プログラム構造に直接結び付く解析木から冗長性を除去した抽象構文木を機械語として実行する新しい計算機について検討する．木構造の表現単位であるノード自体を従来の計算機における機械語命令とみなして実行し、ノードを従来のようにリニアなメモリ上にあるとしてアクセスせず、木構造としてアクセスする．

このような計算機では、プログラム入力中に抽象構文木をインクリメンタルに作成する構造エディタ[1]を使用することにより、プログラムの入力作成からコンパイル処理なしで実行に移ることができる．これを即時実行と呼ぶ．

2．実行系の構成

実行ユニット PU (Processing Unit)、データメモリ DM (Data Memory)、抽象構文木表現を格納するメモリ TM (Tree Memory)、データアドレッシングユニット DAU (Data Addressing Unit)、プログラムアドレッシングユニット TAU (Tree Addressing Unit)によって実行系は構成されている(図１)．PU と TAU の構成を図２と図３に示す．

TM には、プログラムである抽象構文木が格納されている．抽象構文木のノードは３つの固定長フィールド、ノードの種別(Type)、そのノードの右ノードへのポインタ、なければ親ノードへのポインタ(RUPtr)、最後がそのノードの最左子ノードへのポインタかデータ自体(LDPtr)、から構成されている．

ノードの実行過程において次に実行すべきノードが決定されアドレッシングユニットがフェッチする．隣接ノードへの遷移を指定する主要なマイクロ命令を表１に示す．これらの命令により、PU は遷移を指定する制御信号を TAUに送り、TAU はそのノードの TM でのアドレスをレジスタあるいはトゥリースタックから選択し TM に送る．

TM から読み出されるノード表現のうち、種別 (Type) フィールドはオペコードとして PU に送られ Ireg に格納される．隣接のノ

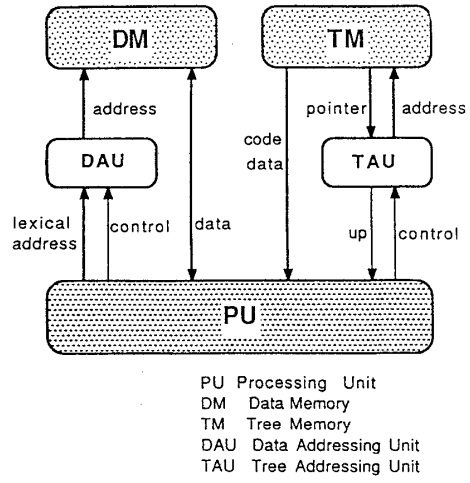

図１　全体の構成

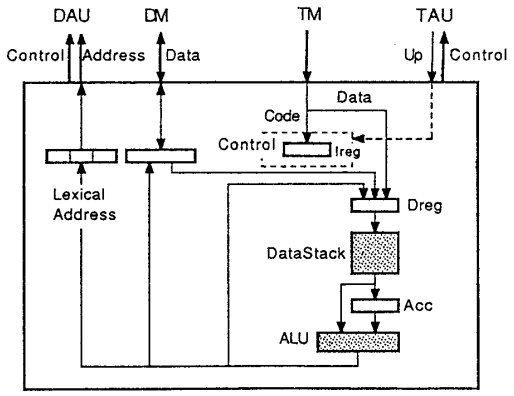

図２　PU の構成

Architecture Design of a Tree-Structured Machine-Language Computer
Takaya ARITA, Iwao MORISHITA
University of Tokyo

ードへのポインタ RUPtr と LDPtr は、TAU に戻される．名前、データ、ブロックレベルが読み出される場合には、Dreg に格納される．

3．実行ユニット（PU）の命令

PU の制御回路内で、オペコードに対応したマイクロ命令列へのポインタが生成される．各ノードに至ったときに実行すべき処理がマイクロ命令列で記述されている．スタックマシン型アーキテクチャを採用し、静的タイプ情報を実行以前に展開しているので、マイクロ命令の総数は少ない[2]．PUのマイクロ命令は次の４種類に分けることができる．

1) 演算
スタックからオペランドをポップして各種演算を行ない、結果をプッシュする．

2) データの読み書き
データをフェッチすると、LoadID 命令により TAU がアドレスを出し、データバスを通して字句レベルアドレスを PU に読み込む．それを LoadValue 命令で DAU に送り、物理アドレスに変換後、データメモリからが読み出す．書き込みは、Assign 命令による．

3) 制御の移動
Call 命令による信号が TAU に渡されると、TAU はノードポインタをトゥリースタックにプッシュ後、子ノードをフェッチする．PU 内でもマイクロポインタをマイクロスタックにプッシュし、子のノードの 種別フィールドを取り込む．Return 命令でそれぞれポップする．CallP 命令は離れたノードへ制御を移す．

4) 特殊命令
初期化命令、終了命令、ブロックへの出入りの命令などが用意されている．

4．即時実行型システムの試作

実行系のシミュレータと構造エディタからなる即時実行型システムを C を用いて UNIX 上で作成した．PL/0[3]に C 風の記述などを付加した言語を対象としている．素数を求めるプログラムなど４個のサンプルプログラムをテストした．

作成された抽象構文木表現は、抽象化しない解析木に比較して、構成ノード数が 1/3 程度に減少する．表２に上記のサンプルプログラムの実行時に実行されたマイクロ命令の総数を示す．１ノードを３マイクロ命令以下で実行できることが知れる．

5．おわりに

抽象構文木を中心にとらえることにより、エディタ系と実行系が密接に結合したシンプルなアーキテクチャの即時実行型システムが実現できることを示した．DAU におけるアドレスの変換法と PU のマイクロ命令表現の変更で複数言語に柔軟に対応することが可能である．

参考文献

1) 有田隆也、永松礼夫、森下巖：コンパイラを支援する構造エディタについて、情報処理学会第３３回全国大会、pp.763-764(1986).
2) Schulthess,P.U.:A Reduced High-Level-Language Instruction Set,IEEE MICRO,vol.4,no.3,pp.55-67(1984).
3) Wirth,N.:Algorithms + Data Structures = Programs, pp.307-319, Prentice - Hall, Englewood Cliffs,N.J.(1976).

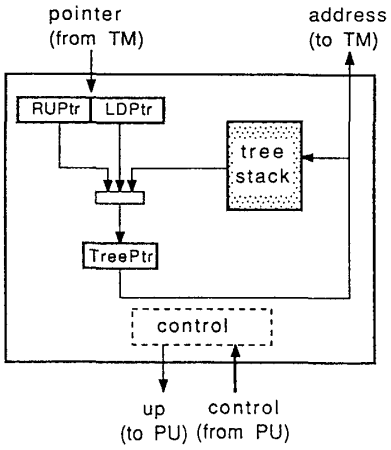

図３　TAU の構成

表１　ノードフェッチのための主要なマイクロ命令

マイクロ命令	処理内容
Call0	最左子ノードに移る．
Call1	第２子ノードに移る．
Call2	第３子ノードに移る．
Return	親ノードに戻る．
MoveR	右ノードに移る、右ノードがない場合は親ノードに戻る．

表２　実行されたマイクロ命令総数

	プログラムノード数	実行された総ノード数	実行総ステップ数
プログラム１	65	94	209
プログラム２	53	380	852
プログラム３	55	2679	7720
プログラム４	125	4838	12253