1J-3 連想プロセッサ DREAM-II の文字列処理への応用と評価

後藤 厚宏，田中 英彦，元岡 達
（東京大学 工学部）

1. はじめに

連想プロセッサDREAM-II[1]の特徴を以下に示す。

(i) 二次元アクセス記憶等のビット処理専用ハードウェアの集合と汎用マイクロセッサによって小規模処理モジュール（連想モジュール）を構成する。

(ii) 処理アルゴリズムはマイクロプログラムで記述し、各モジュールはマイクロプログラムレベルでSIMD制御される。

(iii) 多数の処理モジュールを一次元的に配置し、モジュール間の結合は簡素なものにとどめる。

(iv) ホスト計算機に結合され高レベルなコマンドによって処理が進む。

DREAM-IIは現在稼動中であり、二値図形処理への応用について評価を行なってきた。

本報告では、図形処理と並んで重要な非数値処理のひとつである文字列処理とDREAM-IIアーキテクチャとの適合性を検討した。

2. 連想モジュールによる文字列照合

文字列処理には、連結，部分列抽出、転換，置換等の操作があるがその基本操作は文字列照合である。

DREAM-IIでは、文字列照合を各々のモジュールで並列に実行する。

(1) パターンの正規表現を許す文字列照合

並列比較セルによって正規表現を許した文字列照合をバックトラックなしに実行する方法が、Mukhopadyhyay[2]によって提案されている。これをDREAM-IIによって実行する為には16個の文字比較器（Match Cell Array：MCA）とビットマトリクス演算器を専用ハードウェアとして各モジュールに追加すればよい。

MCAは入力文字Cと各セルが持つn文字（$X_1 \sim X_n$ ： $n \leq 16$）とを比較し、nビットの比較結果（$m_1 \sim m_n$）を出力するものである。

if $C = X_i$ then $m_i = 1$ else $m_i = 0$
（$i = 1 \sim n$）

MCAを用いることにより、入力文字列 $C_1 C_2 C_3 \cdots$ と文字列パターンとの照合はMCAの出力 $M(m_1 \sim m_n)$ に対するビット処理に置き換えることができる。

各比較セルの時刻 t における状態 $S_i(t)$ を考え、

$$f_i(t) = m_i(t) \cdot S_i(t) \qquad i = 1 \sim n$$

とすると、Mukhopadyhyayの方法におけるセル結合は正規表現に対応したビットマトリクス N（ex1～ex3）を用いて次式で表現できる。

$$S(t+1) \leftarrow N \cdot F(t)$$

但し

$F(t) = (f_1(t), f_2(t), \cdots, f_n(t))$
$f_i(t)$ は比較セル出力

$S(t+1) = (S_1(t+1), S_2(t+1), \cdots, S_n(t+1))$
$S_i(t+1)$ は次ステップにおけるセル状態

これは図1に示すビットマトリクス演算器で実行する。

(ex 1)
'$X_1 X_2 X_3 X_4$'

$$N = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ 1 & 0 & 0 & \cdot \\ 0 & 1 & 0 & \cdot \\ 0 & 0 & 1 & \cdot \end{pmatrix}$$

(ex 2)
'$X_1 \begin{pmatrix} X_2 \\ X_3 \end{pmatrix} X_4$'

$$N = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ 1 & 0 & 0 & \cdot \\ 1 & 0 & 0 & \cdot \\ 0 & 1 & 1 & \cdot \end{pmatrix}$$

(ex 3)
'$X_1 (X_2 X_3)^* X_4$'

$$N = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ 1 & 0 & 1 & \cdot \\ 0 & 1 & 0 & \cdot \\ 1 & 0 & 1 & \cdot \end{pmatrix}$$

(注)
・ は Don't Care Bit

SNOBOL言語における錨揚げモードの照合では、

初期状態　$S_i(0)=0$　$i=1\sim n$

錨入力　$S_1(t)=1$　for all t

であり、$S_n(t_{post})=1$ となる t_{post} が一致部分列のポストカーソル位置を示す。

(2) Don't Care 文字を許す文字列照合

Don't Care文字は、MCAの出力Mにおいて Don't Care文字に対応するビットを常に1にすることに相当する。

可変長 Don't Care (VLDC)は、パターン表現における"繰り返し"と1文字の Don't Care によって実現できる。

(3) 文字列照合の性能

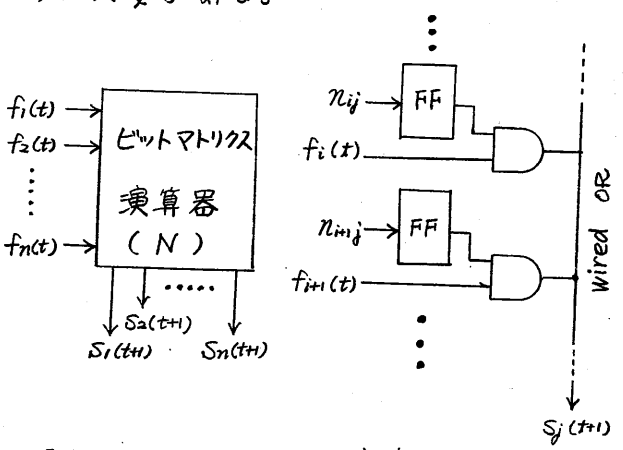
DREAM-IIの処理モジュールは処理幅が16ビットである為、パターン表現に必要な文字数が最大16文字までの文字列照合が各モジュールで実行できる。

照合操作をマイクロプログラムで記述することにより、二次記憶からのデータ転送速度に整合した照合が可能である。

照合操作はパターンに依存しない為、モジュール毎に別々のパターンについて同時並列に実行できる。

3. 一致部分列の抽出

上記の文字列照合ではパターンの正規表現を許している為、一致部分列は可変長になる。照合操作に続いて一致部分列の抽出が必要である場合、ポストカーソルに対応するプリカーソル位置を求める必要がある。



[図1] ビットマトリクス演算器

プリカーソル位置は、ポストカーソル位置から逆方向の錨入れモードパターン照合によって求められる。

DREAM-IIでは、ポストカーソル検出時刻以前のセル出力 $F(t_0-t)$ を各モジュールでバッファPに保存しておき

$f_i(t_0-t)=f_i(t_0-t)\cdot S_i(t_0-t)$　$i=1\sim n$

$S(t_0-t-1)\leftarrow N'\cdot F'(t_0-t)$

N'：逆パターンのビットマトリクス

を実行する。但し

$S_n(t_0)=1$、　$S_i(t_0)=0$　$(i\neq n)$

$t_0=t_{post}$　：ポストカーソル位置

プリカーソル位置は

$S_1(t_{pre})=1$

となる t_{pre} によって示される。

逆方向パターン照合は

$S(t_0-t)=(0,0,\cdots,0)$

となるまで続けることにより、ポストカーソルが重複する全てのプリカーソル位置を求めることができる。

4. DREAM-IIによる文字列処理の問題点

- 各モジュールで照合可能なパターン表現がモジュールの処理幅で制限される。
- モジュール間の結合が疎である為、ひとつのモジュールで照合できないパターンはサブパターンに分解する必要がある。そのパターンがサブパターンのセレクション，コンカティネイション等の簡単な構造でない場合、オーバヘッドが増加する。
- プリカーソルの検出はモジュール毎に実行する。この為、照合操作に続く一致部分列の抽出，置換、転換等の処理ではモジュールレベルの並列性が失なわれる。

[文　献]

[1] 後藤，大和，上森，元岡："多重モジュール構成連想プロセッサ DREAM-II のハードウェア"　信学技報 EC 79-64

[2] A. Mukhopadyhyay "Hardware Argorithms for Nonnumeric Computation"　IEEE Trans. on C C-28, 6, 1979, pp 384~394