計算機アーキテクチャ 145−13
（２００１．１１．２９）

FPGAによるゲノムシーケンス解析専用プロセッサの設計

水原 隆道 [1)]　　中西 恒夫 [1)]　　福田 晃 [2)]

[1)] 奈良先端科学技術大学院大学情報科学研究科
{*takami-m,tun*}*@is.aist-nara.ac.jp*
[2)] 九州大学大学院システム情報科学研究院
fukuda@f.csce.kyushu-u.ac.jp

概要

ゲノム情報学分野における相同性検索や立体構造予測などのアプリケーションは，巨大なゲノム配列データに対して解析を行うため，特に解析精度を求める場合には、高速な計算機が要求される．本稿では，このようなゲノム情報学アプリケーションに特化した専用プロセッサを設計し，ハードウェアによるその高速処理を図る．同専用プロセッサは，ゲノム情報学アプリケーションによく用いられる動的計画法をデータフロー並列処理により高速化する．ソフトウェアシミュレーションによる予備評価の結果，PentiumIII 1GHz と比べて，約 13.5 倍の処理速度が得られることを確認した．

Design of an FPGA-based Processor for Genome Sequence Analyses

Takamichi Mizuhara[1)]　Tsuneo Nakanishi[1)]　Akira Fukuda[2)]

[1)] Graduate School of Information Science
Nara Institute of Science and Technology
[2)] Graduate School of Information Science and Electrical Engineering
Kyushu University

Abstract

Genome informatics applications such as homology search or protein structure prediction, which deal with a huge amount of DNA or amino acid sequences, requires extremely high performance computers especially for accurate analysis. This paper describes design of an application-specific for genome informatics to achieve high performance computing by hardware. The processor accelerates dynamic programming frequently employed by genome informatics applications by data-flow parallel processing. A preliminary experiment by software simulation shows that the processor can perform dynamic programming 13.5 times as fast as purely software processing by PentiumIII of 1GHz CPU clock.

1　はじめに

ヒトゲノムの解読が完了した今日においても，その意味的解析は課題として残されており，ゲノム情報の配列解析は依然重要な基礎的技術である．日々増加するさまざまな生物の膨大なゲノム情報やタンパク質構成の情報空間から，遺伝生物学的かつ化学的意味を考慮した，高速で厳密な解析を施すゲノム情報解析システムが望まれている．FASTA[5], Smith-Waterman 法 [3], Needleman-Wunsch 法 [3]，および 3D-1D 法 [2] 等の多くの解析アルゴリズムにおいて、動的計画法 (DP:Dynamic Programming) が用いられている．本稿では，DP をハードウェアによって高速処理する専用プロセッサを提案し，ゲノム情報学アプリの特性に合わせたその最適設計について述べる．また，同プロセッサを用いて、代表的な相同性検索アルゴリズム Smith-Waterman 法を高速に処理する．

2　相同性検索アルゴリズム

ゲノム情報学で扱う遺伝子情報配列のシーケンスは，4 種類の塩基記号 A,C,G,T を並べた長さ 10^3から 10^8 程度の塩基文字列や，20 種類のアミノ酸記号を並べた長さ 10^3から 10^5 程度のアミノ酸文字列集合である．本稿で扱うゲノムシーケンスにおける相同性検索アルゴリズム Smith-Waterman 法は，複数のシーケンス間の類似度を DP を用いてスコア値として計数し，その値を比較することによって相同性を評価する．

Smith-Waterman 法は，DNA 全体に対する各塩基もしくはアミノ酸文字の出現頻度，アミノ酸の化学的性質をもとに，統計的に算出された各文字間のスコア表 (アミノ酸では BLOSUM 行列) を参照してより尤度が最大となる類似部位を探る．また，Smith-Waterman 法は，その類似部分において最適アラインメント，すなわち一方の文字列にスコア的に最小となる変形を施して，いかに他方の文字列と一致させるかの方法を探る．Smith-Waterman 法では，図 1 に示すようなテーブルを作成する．テーブルの x 軸，y 軸には比較する文字列が並べられている．手順 1 に示すアルゴリズムによりテーブルの各節点のスコア値を求める．このスコア値は，当該節点の位置における部分文字列の相同性の強さに相当する．

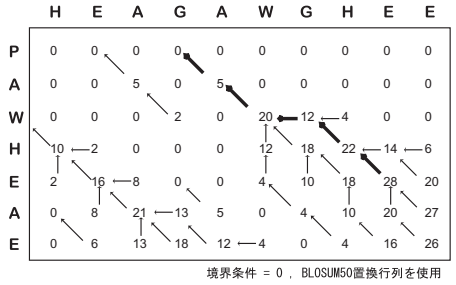

図 1: Smith-Waterman 法による相同性検索

手順 1 【スコア算出アルゴリズム】
2 つの文字列長 m, n であるゲノムシーケンス s_1, s_2 を入力として受け取り，x 軸，y 軸に対応させる．節点 (i, j) の持つ伝搬スコアを $F(i, j)$ とし，挿入および欠損によるギャップペナルティを d とする．

1. シーケンス間の各文字同士のスコア $s(i, j)$ を，BLOSUM 行列等のスコアテーブルを参照し決定する．
2. $i = 0, j = 0,$境界条件 $= 0$ として，作成したテーブルの節点 $(0, 0)$ から節点 $(m-1, n-1)$ に向かって DP を行う．
   (a) $s(i, j)$ を保有する当該節点は，すでに計算の完了している 3 方向の節点 $(i-1, j)$,$(i, j-1)$,$(i-1, j-1)$ からのスコアの伝搬を受ける．
   (b) 式 (1) を適用し，3 方向の伝搬スコアおよびギャップ等を考慮した中で，最高スコアを示すもののみを選択して当該節点の伝搬スコアとする．また，同時に選択した方向を伝搬元として保存する．

$$F(i,j) = \max \begin{cases} 0 \\ F(i-1,j-1) + s(x_i, y_i) \\ F(i-1,j) - d \\ F(i,j-1) - d \end{cases} \quad (1)$$

   (c) 次節点 3 方向に対して，当該節点の伝搬スコア $F(i, j)$ を伝搬する．
   (d) (a) に戻り $i = m-1, j = n-1$ まですべての節点について繰り返す．
3. $i = m, j = n$ の伝搬が完了した時点で，各節点の伝搬スコアと伝搬元を出力する．
4. アラインメントを得るために，伝搬スコア値が最大となる節点を見つけ，節点で保存した伝搬元を辿りその (i, j) を出力する．　□

手順 1 の結果，各節点における伝搬スコア値と，伝搬元方向が得られる．図 1 では太い矢印部分となるが，特に，伝搬スコア値が最大となる節点から伝搬元方向を辿ったアラインメントが最も類似している部分となる．

今回は，利用頻度の高い局所アラインメントを得ることができる Smith-Waterman アルゴリズムを実装するが，DP のスコア伝搬方法を変更することで大域アラインメントを求める Needleman-Wunsch アルゴリズムもスコア値伝搬の方法を変更することによって実現可能である．

3　プロセッサアーキテクチャ

本節では，2 つのシーケンスのスコアとアラインメントを得ることができる専用プロセッサアーキテクチャの概要を述べる．

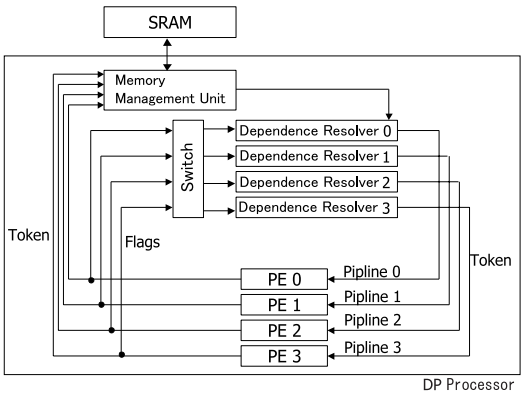

図 2: プロセッサアーキテクチャ

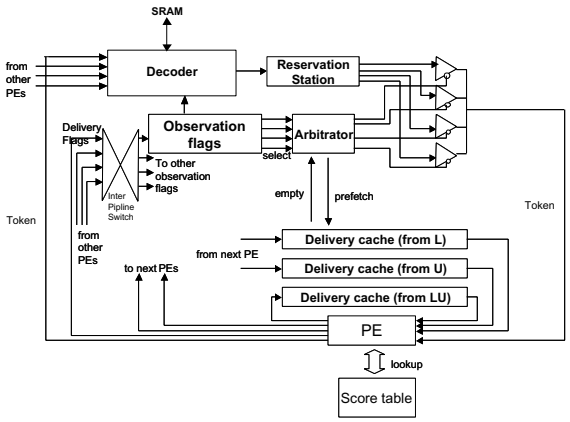

図 3: ストリーム構成

提案するアーキテクチャは，Smith-Waterman 法で用いられる DP を高速処理する準データフローアーキテクチャであり，図 2 に示す構成をとる．

DP テーブルの各節点の情報は，固定長トークンに格納され環状パイプラインによる，データフロー並列処理によって DP を高速処理する．

各 PE は，依存解決機構および発火機構によってパイプラインへ発火されたトークンを処理する．PE で処理された伝搬スコア値決定済みのトークンは，SRAM に書き戻される．各 PE で処理されたトークンは，伝搬スコア値と伝搬フラグに分割され，伝搬スコア値は，隣接節点を処理する隣接 PE のキャッシュに転送される．また，伝搬フラグは，伝搬スコア値の転送完了のシグナルとして Switch へ転送され，割り当て予定の Dependence Resolver によって処理される..

各パイプラインは，次の部品によって図 3 に示す構成を行う．

1. デコーダ
2. トークンプリフェッチ機構 (Reservation Station)
3. データ依存監視および発火機構 (Observation Flags and Arbitrator)
4. スコア値を計算するプロセッサ (PE:Processor Element)
5. 伝搬されたトークンが格納されるキャッシュメモリ (Delivery Cache)
6. スコアテーブル
7. 並列化された際に PE 間で情報を交換するスイッチ (IPS:Inter Pipline Switch)

3.1 トークン

2 つのシーケンスは，図 4(a) に示す書式のトークンとして SRAM に格納され，多重化されたパイプラインによって並列に処理される．このトークンは，PE によって処理されスコア値が決定すると，図 4(b) の書式となり，再び SRAM に格納される．

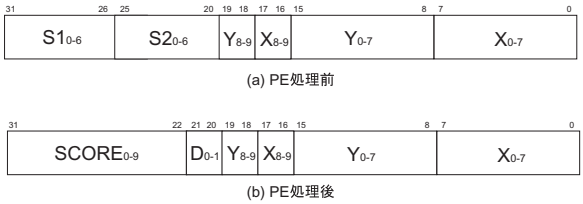

図 4: トークン

トークンに格納される情報は次の通りである．

- **文字** $[S1_{0-6}, S2_{0-6}]$**:** 塩基やアミノ酸文字を格納する．
- **座標** $[X_{0-9}, Y_{0-9}]$**:** トークンの位置を示すもので，トークンの識別や各 PE へのディスパッチに使用される．
- **伝搬スコア値** $[SCORE_{0-9}]$**:** PE での処理前は不定値であり，PE で処理された結果，伝搬スコア値が格納される．また，この値はアラインメントを辿る際や局所的な類似性の発見に使用される．

- **伝搬元方向フラグ** [D_{0-1}]**:** スコア値が伝搬されてきた方向を示すフラグ．3 方向を識別する．

各部品間は，トークンのキューとなるパイプラインによって接続される．また，IPS によって，各パイプライン間が互いに結合されトークン情報の一部が交換される．SRAM に格納されるトークンは，高速アクセスと後述するキャッシュ容量削減の点から最小容量で構成する．

3.2 PE へのディスパッチアルゴリズム

DP 実行時に展開されたテーブルの節点 (i, j) の処理を，式 (2) によって与えられる番号 PE_{num} の PE に割り当てる．ただし P は，プロセッサ内の PE 数である．

$$PE_{num}(i,j) = (|j-i|+1) \bmod P \qquad (2)$$

図 5 は、式 (2) による PE の割り当てを図示したものである．節点 (i, j) と $(i+1, j)$，$(i-1, j)$ は，隣接する PE に割り当てられる．つまり，当該節点において $k = i + j$ の時，その他の各節点における $i + j$ が k となり，かつ隣接しているか近隣の節点は異なった PE へ割り当てる．DP のスコア伝搬は，$i = 0, j = 0$ から右下方向に Wave Front で行われる．その結果，k が同値の節点は近時刻に依存解決され，PE の利用率が向上する．ただし，波面上の節点が同期的に処理されるわけではないことに注意されたい．

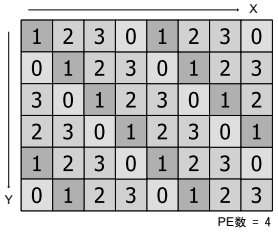

図 5: 節点のディスパッチ

3.3 パイプライン間スイッチ

本アーキテクチャで使用する DP の計算パターンでは，隣接節点へのスコア伝搬が必要となる．節点でのスコア計算を並列処理によって，異なるタイミングで処理するため，計算された節点の伝搬スコアを隣接する次処理節点に伝搬する必要がある．

Smith-Waterman 法では，スコア値の伝搬先が隣接節点に限られていることに着目し，複雑化しがちなデータフローアーキテクチャの交換網の設計を簡素化する．Smith-Waterman 法の任意の節点 (i, j) におけるスコア値の伝搬先は，節点 $(i+1, j), (i+1, j+1), (i, j+1)$ であり，どの節点においても同じ方向に伝搬される．そこで，パイプライン間のデータ交換網である IPS を図 6 のように構成し，スイッチを削減し，回路規模の大幅な縮小と処理時間の短縮を図る．

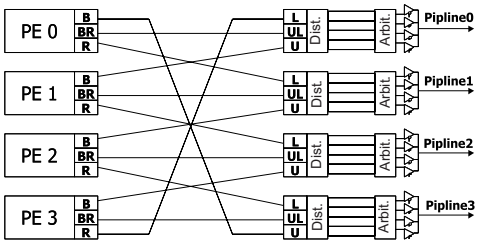

図 6: パイプライン間スイッチ

3.4 伝搬キャッシュ

ある時刻に各 PE で処理されている節点は，DP の特性上近隣の節点である．また，該当節点の処理結果であるスコアの伝搬先は，式 (2) より隣接 PE となる．そこで各 PE に，隣接 PE からの伝搬スコアを ISS を介さずに直接キャッシュする伝搬キャッシュ機構 (図 3 を参照) を実装する．

前節から，スコア値の伝搬方向が固定されるため PE からの伝搬スコアのキャッシュも特定され，交換網が不要となる．

3.5 データ発火機構とプリフェッチ機構

SRAM に格納されているトークンは，データ依存の解決を待っている．節点 (i, j) のデータ依存が解決し発火可能となる条件は式 (3) となる．

$$\begin{cases} \text{節点 } (i-1,j),(i,j-1),(i-1,j-1) \\ \qquad\qquad \text{のスコア伝搬が完了} \quad \text{かつ} \\ PE_{num}(i,j) \text{ の PE が empty} \end{cases} \qquad (3)$$

そこで，データ依存監視および発火機構を図 7 の

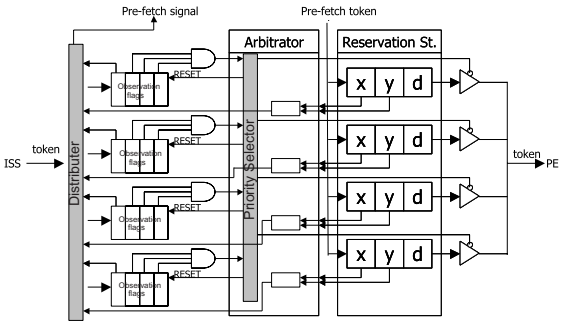

図 7: データ依存監視および発火機構

ように構成する．このデータ発火機構は，パイプラインの数だけ用意され次のモジュールで構成される．

1. 分配器 (Distributor)
2. 監視フラグ (Observation Flags)
3. 調停器 (Arbitrator)
4. 先読み予約レジスタ (Reservation Station)
5. 発火ゲート (Ignition Gate)

Observation Flags は，発火待ちのトークンデータ依存待ちを示すフラグであり，発火待ちのトークンを識別するための節点情報 (i, j) を参照する．このフラグは，3 方向からそれぞれスコアが伝搬されるとフラグがセットされ，すべてのフラグがセットされた時点で，データ依存が解決される．その後パイプラインに空きが生じた時点で Arbitrator によって，Ignition Gate に発火信号が伝達され，Reservation Station に格納されているトークンがパイプラインに送出される．

Distributor は，複数存在する Observation Flags に伝搬済みフラグをセットする．あるトークンに対して，既に伝搬情報が到達している場合は，既存の Observation Flags の中の該当するフラグにセットされる．初めて，伝搬情報が届いたトークンは，SRAM にデコーダを経由してプリフェッチ命令が発行され，Reservation Station にトークンがロードされる．また，同時に Observation Flags の該当するフラグがセットされる．

Arbitrator は，プライオリティセレクタで構成され，複数ある Observation Flags を監視し発火可能なトークンを選出する．

3.6 PC との連携

これまでに述べたデータフローアーキテクチャを，FPGA にプログラムし，DP 専用プロセッサとする．実装予定のボードとして，3 万ゲートの FPGA を 2 個，4MByte の SRAM と PCI バスのインターフェイスを装備する PCI ボードを用いる．これを，PC に接続し，PC で動作するアプリケーションの DP 処理コードを FPGA を駆動するコードに置き換えて使用する．

本稿で取り扱うようなゲノム情報学アプリケーションのシーケンスは長いため，その全てを SRAM 上に展開できない．DP のテーブルを PC において SRAM に展開可能な大きさに分割し，そのシーケンスをボード上の SRAM に展開する．

図 8(a) に示すように，2 次元の DP の計算量は、シーケンス長の 2 乗に比例する．また，この計算量を示す面積は，SRAM 上に展開されるトークンの数と一致する．そのため，一括計算可能な面積は制限される．そこで，図 8(b) のように一括計算可能な面積に分割し，DP を実行する．この分割の単位を示す矩形をウィンドウフレームと呼ぶ．

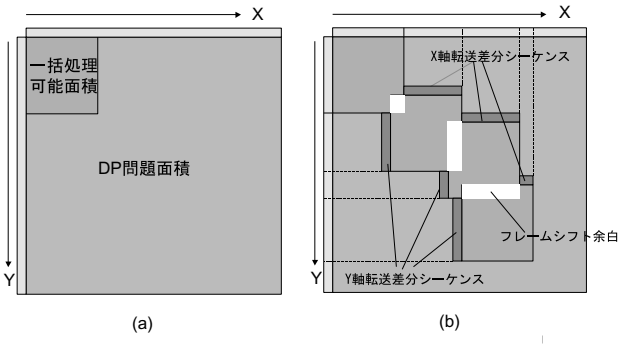

図 8: ウィンドウフレームシフト

DP 専用プロセッサは，DP 実行中には PC との通信が発生しないことを利用して，計算が完了する前にウィンドウフレームの左上部分を計算完了地点にシフトさせ，新しいウィンドウフレームのシーケンスを PC から転送する．このように，計算完了前に PC に割り込みを発生させ次に必要なシーケンスが提供されることにより PE は有効に利用され，処理時間の短縮が望まれる．

4 シミュレーションによる性能評価

DP を全てソフトウェアによって実行した場合と、本稿で提案した DP 専用プロセッサを用いた場合の実行時間について予備評価を行う．評価対象のシーケンス長は，評価ボードに搭載されている 4MByte の SRAM に展開可能な，長さ 1024 の 2 つの塩基文字列とし，スコア値決定の方法は文字比較による計算によって行う．プロセス起動，メモリ初期化，シーケンス転送などのオーバヘッドは含まない．つまり，双方ともにデータがメモリ上に存在している状態で DP を実行し，結果が得られる状態になるまでの時間を求める．

C++コンパイラを用いて生成した，DP を実行するソフトウェアを，PentiumIII 1GHz 搭載 PC で実行時間を測定する．また，DP 専用プロセッサに搭載する PE 数を 1 から 1024 まで変化させ，伝搬キャッシュはそれぞれの PE 数に応じて適当な容量を確保し，ソフトウェアシミュレーションによって実行に要するクロックの算出を行う．DP 専用プロセッサのクロック周波数は 33MHz とする．

ソフトウェアでの実行速度は，実時間で約 0.85sec であり，これを 1 としたときの DP プロセッサでの実行速度比を図 9 に示す．1PE で 6.5 倍程度を示し，16PEs で 13.5 倍の速度が得られることがわかった．

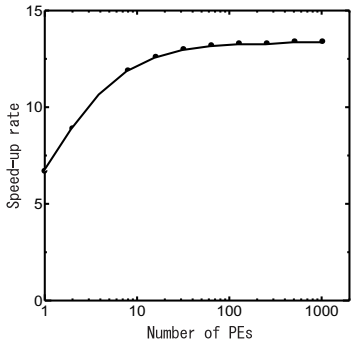

図 9: DP プロセッサによる速度比 (シミュレーション)

5 まとめ

本稿では，ゲノム情報学分野のアプリケーションプログラムを高速化するための DP 専用プロセッサの提案を行った．本 DP 専用プロセッサを用いることで，ゲノムシーケンス解析における相同性検索が，純ソフトウェア的手法と比べて高速に実現が可能となる．ソフトウェアによるシミュレーションでは，PentiumIII 1GHz のおよそ 13.5 倍程度の高速化が達成できることを確認した．また，本稿では，Smith-Waterman 法の専用プロセッサによる高速化を示したが，Needleman-Wunsch 法をはじめとして他の DP を用いる多くのゲノム情報学アプリケーションも高速化が可能となる．しかし，このシミュレーションは，PC と SRAM メモリ間のシーケンス情報や実行結果の転送などを考慮していない．また，DP を実行するシーケンス長によってもスループットは変化する．DP プロセッサを FPGA に実装し，評価ボードを用いた開発を今後の課題としたい．

謝辞

本研究は，平成 13 年度科学研究費補助金・特定領域研究 (C)(2) (題名:ゲノム情報学アプリケーションの専用プロセッサ協調型並列処理) の助成を受けている．

参考文献

[1] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, “Basic local alignment search tool,” *Journal of Molecular Biolog,* vol.215, pp.403–410, 1990.

[2] J. U. Bowie, N. D. Clarke, C. O. Pabo, R. T. Sauer, “Identification of protein folds: matching hydrophobicity patterns of sequence sets with solvent accessibility patterns of known structures,” *Proteins,* vol.7, pp.257–264, 1990.

[3] R. Durbin,S. Eddy,A. Krogh, and G. Mitchison, *Biological sequence analysis,* Cambridge University Press, 1998.

[4] D. T. Hoang, “Searching Genetic Databases on Splashzz 2,” *Proc. of IEEE Workshop on FPGAs for Custom Computing Machines,* pp.185–191, April 1993.

[5] W. R. Pearson, “Rapid and sensitive sequence comparision with FASTP and FASTA,” *Methods in Enzymology,* Academic Press, vol.183, pp.63–98, 1990.