1K-5

BLASTアルゴリズムのハードウェア化の検討

石川淑† 田中飛鳥† 宮崎敏明†
会津大学大学院コンピュータ理工学研究科†

1. はじめに

Basic Local Alignment Search Tool(BLAST)は最も有名なシーケンスアライメントツールの一つである．シーケンスアライメントとはタンパク質（またはDNA)配列データベース(DB)内のシーケンスと検索対象となるタンパク質（またはDNA）配列(クエリシーケンス)を比較し,配列同士の類似部分検索を行うものである．シーケンスアライメントは，生物学上の進化や遺伝子系図を調べる上で重要であることから，バイオインフォマティクス分野では欠かせない情報となっている．今日，DBの急激な肥大化に伴い，シーケンスアライメントの高速化が求められている．そのため，BLAST のハードウェアによる高速化手法が多く提案されてきた[1,2]．しかし，それらの手法は，BLAST アルゴリズムの一部のハードウェア化に留まっており，処理全体のハードウェア化は試みられていない．本稿では，BLAST 処理全体のハードウェア化を提案する．

2. BLASTアルゴリズム

BLAST アルゴリズムは，前処理と，Seeding(ステップ 1)，Ungapped extension(ステップ 2)，Gapped extension (ステップ3)の3つのステップからなる．前処理では，検索に使用される隣接ワード（Neighborhood word）と呼ばれる文字列を生成する．隣接ワードはクエリシーケンスとスコア行列を用いて生成される．スコア行列とは，タンパク質を構成する 20 種のアミノ酸同士の類似度を数値化した表であり，一般に20×20の行列形式で表現される．スコア行列は複数存在するが，本稿では，その内，BLOSUM50と呼ぶスコア行列を使用する．クエリシーケンスをクエリワードと呼ぶk文字（通常，タンパク質配列ではk=3，DNA配列ではk=11）に分割し，そのクエリワードの3文字を他のアミノ酸の20文字と1文字ずつ比較しスコアを計算する．3 文字のスコアの合計が閾値 T（通常 T=12）以上となったワードが隣接ワードとなる.Seeding(ステップ1)では，前処理で生成した隣接ワードを使用して DB シーケンスとの seed を探す．seed とは，DB シーケンス上で隣接ワードが一致した位置のことである．ステップ 2，すなわち Ungapped extension ではステップ 1 で見つかった seed を拡張開始点とし,一致点の拡張を行う．拡張は合計スコアが最大値から閾値 S（ユーザが指定）下がるまで行う．拡張されたシーケンスペアのうち HSP（high score pair）と呼ぶ組み合わせだけが次のステップ 3 の入力となる．Gapped extension(ステップ3)ではHSPのシーケンスペア中で最も類似度が高い部分を探索し，求まった解を最適なシーケンスアライメントとして出力する．Gapped extension では Smith-Waterman アルゴリズムという動的計画法に基づく手法が使用される．

3. 提案手法

従来，BLAST のハードウェア化は，前述のステップ 3 を中心に行われてきた．前処理である隣接ワードの生成は，ホスト計算機上のソフトウェアで行う必要があり，クエリシーケンスを変更する度にホスト計算機との通信が必要になるため，通信ボトルネックが生じる可能性がある．本稿では，隣接ワード作成処理（前処理）を含むBLASTアルゴリズム全体をハードウェア化することを提案する.ここでは,BLASTP（タンパク質配列のシーケンスアライメントツール）を対象とし，ワード長3（k=3），隣接ワード閾値12（T=12）とする．図1に前処理部の提案回路を示す．前処理部では，まず，クエリシーケンスを 3 文字のワードに分割する.タンパク質は20種類のアミノ酸で構成されているため，3 文字のワードは 20×20×20=8000 種類存在する．そのため，クエリワードから隣接ワードを生成するためには，クエリワードごとに，8000 種類の 3 文字のアミノ酸列との間で個々に合計スコアを計算し，隣接ワード閾値 T を超えたものを列挙する必要がある．ここでは，その計算量を軽減するために，図2に示したように，オリジナルのスコア行列の情報から,アミノ酸ごとに，ペアとなる 20 種類のアミノ酸をスコア値が高い順に並べた Protein LUT（lookup table）と呼ぶ参照テーブルを用意し，合計スコアの高い順に隣接ワード候補を生成するようにする．これにより，合計スコアが閾値 T 以下になった時点で，そのワードに対する隣接ワードの生成を即座に停止することができるため，8000 種類全ての隣接ワード候補の合計スコアを算出し，それを吟味する必要がなくなる．

ステップ 1 以降で必要となるデータは，隣接ワードがクエリシーケンス上のどの位置で生成されたかを示す位置情報である．その位置情報を保持するために，隣接ワードメモリを用意する．クエリワードに対し生成された隣接ワードを使用し，隣接ワードメモリのアドレスを生成する．クエリシーケンスの長さに依存して，メモリに保持する隣接ワードの位置情報の数が増加するため，一つの隣接ワードに対し

An approach to hardware implementation of BLAST
†Shizuka Ishikawa, †Asuka Tanaka, †Toshiaki Miyazaki
†Graduate School of Computer Science and Engineering, The University of Aizu of Aizu

て位置情報の数が予め用意したメモリ領域を超える可能性がある．それに対処するために，一つの隣接ワードに対して格納出来る位置情報の最大数を超えた場合，未使用のメモリ領域を検出し，そこに位置情報を保持する機構も導入した．

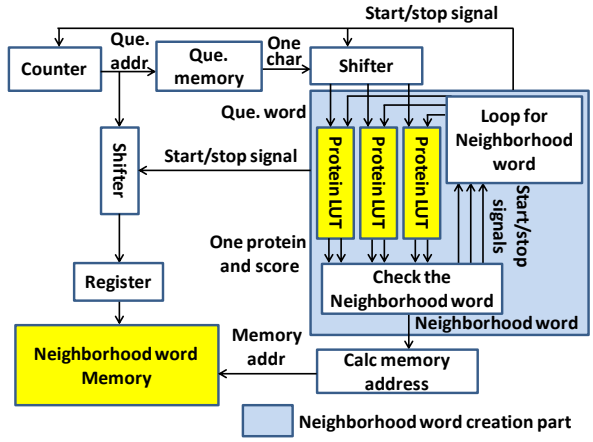

図 1 前処理を実行する提案回路

図 2 Protein LUT の内部構造

4. 全体構成

図 2 に提案回路の全体構成を示す．提案回路は，前処理部，Two-hit(Seeding)部，Ungapped extension 部，Gapped extension 部の 4 つの部分に分かれている．各部は，前述した隣接ワードを求める前処理，ステップ 1,2,3 に対応する．前処理部では，入力されたクエリシーケンスを用いて隣接ワードメモリの内容を生成する．この隣接ワードメモリは，ステップ 1 の処理で，何度も参照される．ステップ 1 では，3 文字に分割された DB シーケンスを隣接ワード用メモリに送り一致点を探す．Two-hit(Seeding)部では，ステップ 2 で拡張を行う seed 数を減らすために Two-hit[2,3]法と呼ぶ手法をハードウェア化している．Two-hit 法を適用することにより，最初に求まった seed 数を約 1/8 まで減らすことができる．Two-hit とは 2 つの seed のクエリシーケンスと DB シーケンスの位置情報の差が同じであり，seed の間隔が検索幅以内である場合に拡張を行うというものである．Two-hit で拡張する位置が見つかった場合，次の Ungapped extension 部分に対して拡張の開始信号と，クエリシーケンスならびに DB シーケンスの位置情報を送る．Ungapped extension 回路では，文字列の左右方向に拡張を行うのではなく，一方向への み拡張を行うようにしている．Ungapped extension 部の処理で使用したクエリシーケンスと DB シーケンスの文字列情報は時間調整用の FIFO を介して，最終処理部である Gapped extension 部へ送られる．Gapped extension 部では，Smith-Waterman アルゴリズムに基づいたシーケンスアラインメント処理が行われる．

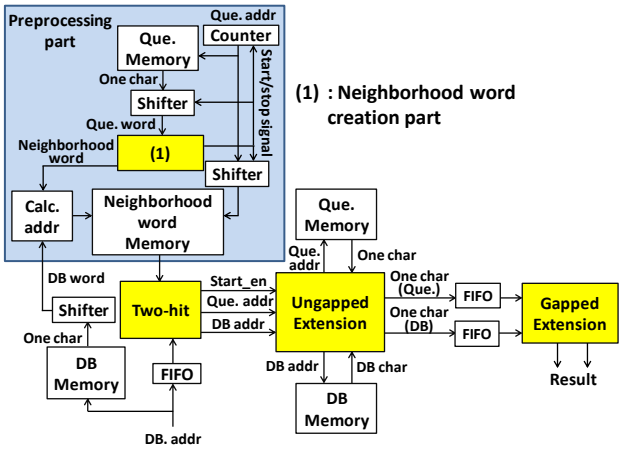

図 3 BLAST 処理を実行する提案回路の全体構成

5. 評価

動作確認のため，FPGA を用いて提案回路の実装を進めている．表 1 は，図 1 に示した前処理部の実装結果である．評価に用いたクエリシーケンス長は 100 である．ここでは、最大 8 個の位置情報を保持できるサイズの隣接ワード用メモリを用意した。使用した FPGA は Cyclone-IV E EP4CE115F29C7 であり，回路実装には同社の設計ツール QuartusII 10.1sp1 を使用した．提案回路の最大動作周波数は 81.48 MHz，実行時間は 81 μsec であった．

表 1 前処理部の FPGA 回路規模

	使用量	使用率
ロジックエレメント数	1,621	1 %
レジスタ数	101	<1 %
メモリビット数	512,500	13 %

6. おわりに

BLAST 処理全体を実行するハードウェアアーキテクチャを提案した．今後は，全体回路の動作確認を行い，多くのデータを用いて，より詳細な評価を実施する．

参考文献

[1] K. Benkrid, Y. Liu and A. Benkrid, “A Highly Parameterized and Efficient FPGA-Based Skelteon for Pairwaise Biological Sequence Alignment,” IEEE, Vol.17, No. 4, April 2009.
[2] S. Kasap, K. Benkrid and Y. Liu, “Design and Implementation of an FPGA-based Core for Gapped BLAST Sequence Alignment with the Two-hit Method,” LAENG, Vol. 16, Issue. 3, No. 25, August 2008.
[3] A. Jacob, J. Lancaster, J. Buhler, R. D. Chamberlain “FPGA-accelerated seed generation in Mercury BLASTP,” in Proc. of 15th IEEE Symposium in Field-Programmable Custom Computing Machines (FCCM), April 2007.