Vol.2012-BIO-28 No.7
2012/3/28

遺伝子配列に対するペアワイズアライメントのGPUによる高速化

岡田啓佑[†1]　伊野文彦[†1]　萩原兼一[†1]

本稿では，GPU（Graphics Processing Unit）における高速な Smith-Waterman（SW）アルゴリズムの実装を示す．提案実装は 1 組の遺伝子配列に対するアライメントを高速に処理する．そのために，Striped SW アルゴリズムをタイリング技術とともに CUDA（Compute Unified Device Architecture）上に実装している．

GPU-Accelerated Pairwise Alignment for Genome Sequences

Keisuke Okada,[†1] Fumihiko Ino[†1]
and Kenichi Hagihara[†1]

This paper presents a fast implementation of the Smith-Waterman (SW) algorithm running on the graphics processing unit (GPU). Our implementation accelerates pairwise alignment of genome sequences. To achieve this, a striped version of the SW algorithm is implemented with a tiling technique on the compute unified device architecture (CUDA).

1. はじめに

近年，遺伝子データベースに登録されている遺伝子配列の数が増大している．そこで，1 対多の局所アライメントを高速化するために，GPU（Graphics Processing Unit）による Smith-Waterman（SW）アルゴリズム[1] の高速化が試みられている．GPU は CPU よりも 1 千倍ほど多くのスレッドを実行でき，それらは階層化されている．異なる組の局所アライメントは独立に処理できるため，多くの既存手法は複数の組を同時に処理する．

しかし，1 組の遺伝子配列を対象とするペアワイズアライメントに対しては，既存手法のような並列化は適用できない．特に，SW アルゴリズムはデータ依存の制約が強く，並列処理の効率を高めることは容易ではない．

そこで本稿では，ペアワイズアライメントの高速化を目的として，その GPU 実装を提案する．長い遺伝子配列に対して並列処理の実行効率を向上するために，提案実装は 2 つの工夫を持つ．まず，GPU 外部のオフチップメモリへの書き込み回数を削減するために，レジスタへの書き込みを増やせるよう計算対象の行列にタイリングを施す．次に，遊休スレッドの数を削減するために，タイル内に SSW（Striped SW）アルゴリズム[2] を適用する．提案実装は Compute Unified Device Architecture（CUDA）[3] 互換の GPU 上にて動作する．

2. 提案実装

SW アルゴリズムの性能ボトルネックは類似度を含む行列の生成にある．行列を構成する要素ごとの計算にデータ依存があり，並列に計算できる要素は逆対角線上に並ぶ（**図 1**(a)）．逆に，異なる逆対角線は並列処理できない．したがって，逆対角線を図の左上から右下にむけて逐次的に走査する必要がある．入力となる配列の長さをそれぞれ m および n とすれば（$m \geq n$），$m+n-1$ ステップで行列を生成できる．各ステップでは，高々m 個の要素を並列処理できる．ここで，初期および終期のステップにおいて並列性が減少することに注意されたい．

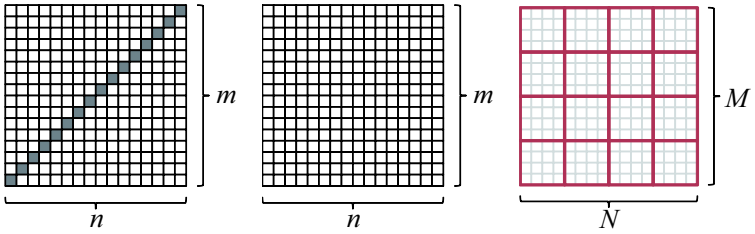
提案実装は，GPU 外部のオフチップメモリへの書き込みを削減するために，行列にタイリングを施す（図 1(c)）．タイル間のデータ依存は要素間のデータ依存と同一のパターンを持つため，要素に対する並列化をそのままタイルに適用できる．タイルの数を $M \times N$ 個とすれば（$M < m$ かつ $N < n$），$M+N-1$ ステップで行列を生成でき，各ステップでは高々M 個のタイルを並列処理できる．

各タイルの計算は GPU 上のスレッドブロックに割り当てる．これによりタイル内の要素はレジスタや共有メモリのみへの書き出しで計算できる．ただし，タイル間で要素値を受け渡しするために，オフチップメモリへの書き出しが必要である．したがって，タイリングによりオフチップメモリへの書き出しを $m+n-1$ 回から $M+N-1$ 回に削減できる．

タイル内の計算には SSW アルゴリズム[2] を適用する．SSW アルゴリズムは，一部のデータ依存を反映せずに同一列上の要素を並列処理する．これによりステップ初期および終期に

†1 大阪大学大学院情報科学研究科コンピュータサイエンス専攻
Department of Computer Science, Graduate School of Information Science and Technology, Osaka University

ⓒ 2012 Information Processing Society of Japan



(a) 並列性　(b) タイリング前　(c) タイリング後

図 1 Smith-Waterman アルゴリズムの並列性および行列のタイリング

おける遊休スレッドの発生を避け，実行効率を高める．ただし，各要素の値が閾値を超える場合についてのみ，データ依存を反映させるための再計算が必要である．再計算時には同一列上の要素を逐次処理する必要がある．

なお，提案実装はタイル間の計算には SSW アルゴリズムを適用しない．この理由は，再計算のオーバヘッドが大きいためである．再計算は同一列の要素を逐次処理する．したがって，すべてのタイルに再計算が必要な場合，m 個の要素を逐次処理することになる．一方，提案実装はタイル間の計算に SW アルゴリズムを適用しているため，同一列のタイルを同時に処理することはない．したがって，タイル内の再計算（$\lceil m/M \rceil$ 個の要素）を逐次処理することはあっても，M 個のタイルは並列処理できる．

3. 評価実験

GeForce GTX 480 および CUDA 3.2[3] を用い，提案実装の性能を計測した．提案実装は，128 個のスレッドを含むスレッドブロックを生成し，各スレッドは連続する 7 行の計算を担当している．これらの値は実験的に定めた．一致スコア，不一致スコア，ギャップ開始ペナルティおよびギャップ伸長ペナルティはそれぞれ $+1$，-3，-5 および -2 とした．また，タイルサイズは $h \times w = 896 \times 512$ 要素とした．

長さがそれぞれおよそ 23M 塩基対および 24M 塩基対の配列 NT_033779.4 および NT_037436.3 に対し，提案実装は 397 分で行列を生成できた．このときのスループットは 23.7 GCUPS（Giga Cell Updates Per Second）であり，配列間の類似度は 9063 である．1 対多の局所アライメントを処理する既存研究[4] と比較して，このスループットは 56%に相当する．

表 1 実測スループットと再計算回数（各配列の先頭 1M 塩基対のみを使用）

配列 1	配列 2	類似度	スループット（GCUPS）	再計算回数（$\times 10^8$）
BA000046.3	NC_000021.7	0	22.7	0.0
NC_005027.1	NC_003997.3	32	22.3	0.4
BA000035.2	BX927147.1	3,921	15.7	1.3
NT_033779.4	NT_037436.3	8,745	17.8	1.4
CP000051.1	AE002160.2	86,171	2.8	117.5
AE016879.1	AE017225.1	1,047,688	0.7	1,825.7

類似度がスループットに与える影響を調べるために，いくつかの配列の先頭 1M 塩基対のみを使用して行列を生成した（**表 1**）．提案実装では，類似度が小さいほど再計算の回数が少なく，スループットが高い．したがって，類似性の低い遺伝子配列を除外するための前処理として有用である．

最後に，タイリングの効果を調べるために，SW アルゴリズムの単純な実装との比較を示す．この実装は逆対角線ごとにカーネルを起動する．表 1 に示した配列に対し，いずれもスループットは 1.3 GCUPS であった．したがって，再計算が発生しない場合，提案実装は単純実装に対して 17 倍の速度向上を得ている．この速度向上は主にオフチップメモリへの書き出し削減による．一方，類似度の高い組に対しては再計算の回数が多く，単純実装の方が高速であることもある．

謝辞 本研究の一部は，科学研究費補助金若手研究（B）（23700057）および基盤研究（B）（23300007）の支援を受けた．

参考文献

1) Smith, T.F. and Waterman, M.S.: Identification of Common Molecular Subsequences, *J. Molecular Biology*, Vol.147, No.1, pp.195–197 (1981).
2) Farrar, M.: Striped Smith-Waterman speeds database searches six times over other SIMD implementations, *Bioinformatics*, Vol.23, No.2, pp.156–161 (2007).
3) NVIDIA Corporation: CUDA Programming Guide Version 3.2 (2010).
4) Ligowski, L., Rudnicki, W., Liu, Y. and Schmidt, B.: Accurate Scanning of Sequence Databases with the Smith-Waterman Algorithm, *GPU Computing Gems*, Morgan Kaufmann, San Mateo, CA, pp.155–172 (2011).

ⓒ 2012 Information Processing Society of Japan