Vol.2010-BIO-21 No.30
2010/6/19

GPUによるDNA断片配列の高速マッピング

鈴木脩司[†1]　石田貴士[†1]　秋山泰[†1]

次世代 DNA シーケンサーは従来型と比較して数百倍から数千倍も高いスループットを持つが，得られるデータは短い断片配列であるため，既知ゲノムへの張り付けで位置を特定する配列マッピング処理が必要となる．特にメタゲノム解析では，環境中に含まれる微生物全てのゲノム配列が既知であることは稀であり，配列マッピングの際の感度を増すためアミノ酸配列への翻訳を行い，多数のミスマッチやギャップを許容する必要があるため，通常の配列マッピングに比べて長い計算時間が必要となる．本研究ではメタゲノム解析にも対応可能となるよう Smith-Waterman アルゴリズム (動的計画法) によりアライメントを行うことで曖昧さを許容し，また GPU を用いることで高速に実行可能となる配列マッピングのアルゴリズムを提案した．その結果，従来用いられてきた BLAST と比較してタプル長が K=4 のとき，1 GPU を使用した場合で約 50 倍，4 GPU を使用した場合で約 200 倍の高速化を達成した．

Fast short DNA sequence mapping on GPUs

Shuji Suzuki,[†1] Takashi Ishida[†1]
and Yutaka Akiyama[†1]

We developed a new efficient algorithm for mapping fragment sequences produced by a next-generation sequencer and have implemented the mapping system on GPUs. We designed the algorithm to map sequences even with ambiguous matches to be used for metagenomic studies by the Smith-Waterman algorithm and to be able to execute at high speed on GPUs. As results, the system achieved a speedup of about 50 times with respect to the BLAST using 1 GPU, and about 200 times with 4 GPUs, in the case of taple length of K=4.

†1 東京工業大学 大学院情報理工学研究科 計算工学専攻
Graduate School of Information Science and Engineering, Tokyo Institute of Technology

1. はじめに

近年，次世代シーケンサーと呼ばれるハイスループットなゲノム解読装置が登場した．この次世代シーケンサーの特徴として，数十塩基程度の短い断片配列を大量に読めること，1 ランで数十億塩基以上が解読可能であること，従来のシーケンサーと比べると数百倍から数千倍のスループットであることがある．この次世代シーケンサーによって得られるデータは短い DNA 断片情報であるため，これらのデータを利用するためには配列マッピングと呼ぶ処理が必要となる．これはシーケンサーが読み取った短い DNA 断片が元の配列のどの部分のものであったかを同定する処理である．配列マッピングを行うツールとして RMAP[1] や Bowtie[2] 等がある．単一の生物に対するゲノム解読ではシーケンサーの読み取った DNA 断片と元の配列とでは一致率が高いと考えられるため，これらのツールでは文字列の完全一致，または数塩基の違いしか許容しないという特徴がある．一方，次世代シーケンサーを用いた他の解析にメタゲノム解析がある．これは土壌や腸内等に生息する複数の微生物の DNA を分離・培養を経ずに直接シーケンサーで読み取ることで，単一の生物ではなく，ある環境における遺伝子の分布を解析しようというものである．環境中に含まれる微生物全てのゲノム配列が既知であることは稀であるため，このメタゲノム解析では遠縁のゲノム配列しか参照できない場合が多く，通常の単一の生物に対する配列マッピングに比べ，配列マッピングの際に多くのミスマッチとギャップを許容する必要があり，複雑なものとなっている．このためさらに，メタゲノム解析では感度を増すために DNA 配列をタンパク質配列に変換してから解析する場合がある．DNA 配列は A，T，G，C の 4 文字で表現されるのに対し，タンパク質配列は 20 種の標準アミノ酸からなっており，20 文字で表現される．さらに，DNA 配列の比較は一致か不一致との 2 状態しか区別しないが，タンパク質配列ではアミノ酸種間でその性質の類似性に差があるため，アミノ酸種毎に置換スコアが付されているスコア行列が用いられる．このため，ミスマッチの場合でもペナルティはアミノ酸種に依存し，これは配列マッピングをさらに困難なものとしている．このようなミスマッチやギャップを許容した複雑な配列間の比較には，もっとも正確なアラインメントが可能である動的計画法を用いた Smith-Waterman アルゴリズム[3] の実装である SSEARCH[4] 等を利用することが望ましいが，これは低速であるという問題がある．このため現在では，高速に配列比較が可能な近似的手法の一つである BLAST[5] が利用されている．しかし，次世代シーケンサーは一度に大量の DNA 断片を出力するため，すべてをタンパク質配列に変換してから配列マッピングを行うには BLAST を用いてすらまだ長い計算時間が必要となる．このため，本

Vol.2010-BIO-21 No.30
2010/6/19

研究ではメタゲノム解析にも対応可能となるように曖昧さを許容し，また，GPU を用いることで高速に実行可能となる配列マッピングのアルゴリズムを提案し，その実装を行った．

2. GPU について

GPU とはパソコンの拡張スロットに取り付けるビデオカードに搭載されている描画処理に特化したチップや，CPU に付随するチップセットの中に組み込まれる描画処理機能の部分のハードウェアである．現在では NVIDIA 社と AMD 社が高性能な GPU を開発している．この開発の過程で GPU を画像処理だけでなく汎用計算にも利用しようという GPGPU（General-Purpose computing on Graphics Processing Units）が生まれた．GPGPU 向けの総合開発環境に NVIDIA 社が提供する CUDA がある．CUDA はコンパイラやライブラリなどから構成されている．CUDA で GPU 内で実行する関数をカーネル関数というが，カーネルには以下のような制約がある．

- CPU 側のメモリへのアクセスはできない
- 再帰処理ができない
- static 変数なし

また，CUDA のメモリにはいくつかの種類がある．それぞれの特徴を表 1 に示す．

GPU の性能を引き出すためにはこれらのメモリを効果的に使用する必要がある．

関連技術として OpenCL(Open Computing Language) がある．現在，NVIDIA 社 GPU 用の CUDA， AMD 社 GPU 用の Brook+，Cell 用の Cell SDK とさまざまなプログラミング環境が乱立している状態にあり，各計算デバイスに対応する開発言語を用いなければならない．これを統合する目的で OpenCL が登場した．OpenCL は多種の CPU，GPU，その他のアクセラレータ等の差異を吸収して，異種混在の計算資源を並列コンピューティングに利用するためのフレームワークである．このため OpenCL で作られたプログラムは CUDA とは異なり，NVIDIA 社の GPU が搭載されていないパソコンでも他のアクセラレータを用いて高速に実行することができる．また，OpenCL は使用用途にパソコンの他に携帯機器などへの利用も想定されている．

表 1 CUDA の各メモリの特徴
Table 1 The Characteristics of the various CUDA memories

メモリの種類	置かれている場所	アクセス	キャッシュ
Register	チップ上	読み書き可	—
Local Memory	チップ外	読み書き可	なし
Shared Memory	チップ上	読み書き可	—
Global Memory	チップ外	読み書き可	なし
Constant Memory	チップ外	読み込みのみ	あり
Texture Memory	チップ外	読み込みのみ	あり

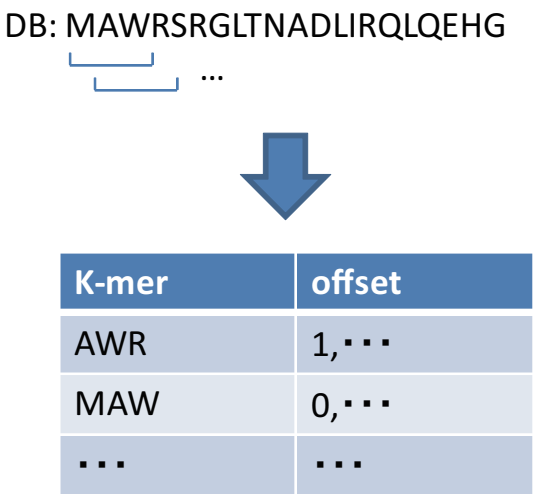

図 1 インデックスの構築
Fig. 1 Constructing the index

3. 手　　法

3.1 提案する配列マッピングアルゴリズム

本研究のアルゴリズムでは，まずインデックスを用いて DB に対してマッピング位置の候補を探索し，その後，その周辺を動的計画法により詳細に探索して最適アラインメントとそのスコアを計算する．以下に詳細を示す．

3.1.1 前　処　理

マッピングの対象となる既知のタンパク質配列の DB の中には多数のタンパク質配列が含まれている．しかし，複数の配列よりも単一の配列の方が扱い易いため，これらの配列を区切り文字を入れて連結していく．次に連結した配列を 1 文字ずらしで K-mer(長さ K の文字列) に区切って，出来た配列を元に key を生成していく．この様子を**図 1** に示す．

そして，key の配列が連結配列上のどこの位置で発見されたかを各 key 毎にまとめる．こ

Vol.2010-BIO-21 No.30
2010/6/19

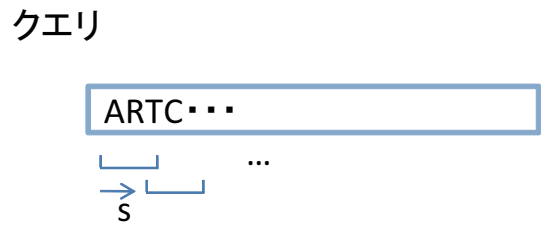

図 2 クエリの K-mer の生成
Fig. 2 Generating the K-mer on queries

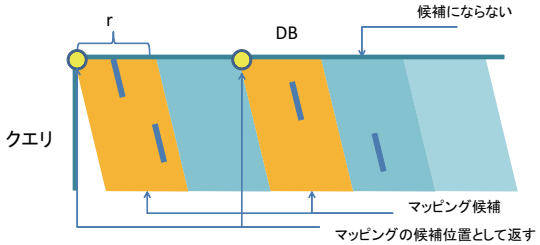

図 3 マッピングの候補探索
Fig. 3 Searching mapping candidates

れをインデックスとしてマッピング候補探索の際に使用する．また，シーケンサーが読み取った DNA 断片を順方向の 3 フレームと逆方向の 3 フレームの計 6 通りのすべての読み枠をタンパク質配列に翻訳しておく．

3.1.2 マッピング候補探索

まず，クエリ (問い合わせ配列) を s 文字ずらしで K-mer に区切り，これを key とする．この様子を**図 2** に示す．

ここでクエリでの key の出現位置を縦軸，DB での key の出現位置を横軸として，クエリと DB 間で，同じ key が出現する箇所に印をつけていく．クエリと DB 間で高い一致率がある領域ではこの印がいくつか同一斜線上付近に並ぶ．これを利用して DB を r 文字の大きさの領域で区切り，同領域と右隣の領域内の key 数の和がある閾値 t 以上となった領域をマッピング候補とする．**図 3** に閾値 t が 2 の場合を示す．

3.1.3 アラインメントによるスコア計算

マッピング候補探索によって得られたマッピング候補位置の周辺を Smith-Waterman アルゴリズムによって最適なアラインメントとそのスコアを計算し，マッピング候補位置のスコアとする．マッピング候補探索によってマッピング位置が大まかに分かっているため，アラインメントを計算する領域の幅は**図 4** のように m+2r+2e とした．ここで m はクエリの長さ，r は探索のときに用いた r の値，e はアラインメントの計算を行う領域の幅を拡張する長さである．こうすることで少なくとも r+e 個ギャップがあったとしても正しいアラインメントが計算できる．

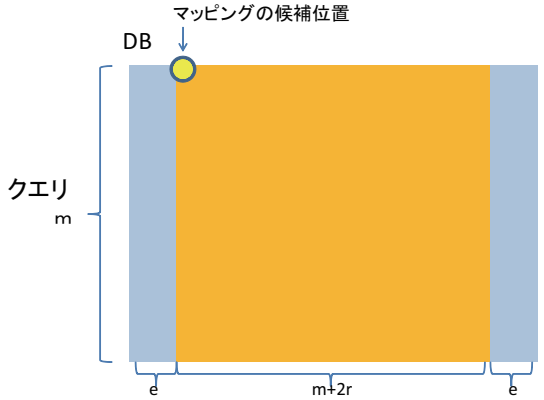

図 4 アラインメントを計算する領域
Fig. 4 Calculating an alignment on region

3.2 GPU による実装

3.1 のアルゴリズムを CPU だけで実行した場合，マッピング候補探索とアラインメントによるスコア計算の箇所で多くの計算時間が必要となることが判った．このため，本研究ではこれらの 2 か所を GPU によって処理することとした．

3.2.1 マッピング候補探索の GPU 化

マッピング候補探索では予めどれほどの個数のマッピング候補位置が出力されるかわからない．このため，一度ですべてのマッピング候補位置を計算しようとするとマッピング候補位置の個数が GPU の global memory の大きさの限界を超える可能性がある．このため，本研究では以下のように 2 段階に分けてマッピング候補探索を行うようにした．

(1) クエリ毎にマッピング候補の数をカウント
(2) GPU の global memory を超えない数ずつマッピング候補位置を計算し，すべての候補位置を計算するまで繰り返す

このようにすることで GPU 上でも問題なくマッピング候補探索を実行することができる．

3.2.2 アラインメントによるスコア計算の GPU 化

Smith-Waterman アルゴリズムの GPU 化は既に SW-CUDA[6] 等が提案されている．しかし，ほとんどの場合，ある程度長い配列同士のアラインメントを複数の thread を同期させながら計算していくという手法である．一方で，今回のように短い配列のアラインメントでは同期の時間の割合が増えてしまって計算速度が出ない．このため，本研究ではマッピング候補位置毎に 1 つの thread を割り当てて，計算を実行するようにした．また，スコア行列はすべての thread が何度もアクセスするため本研究では texture memory に配置した．こうすることで cache の効果により global memory に配置するよりも高速にアクセスが可能となっている．

4. 実験結果

本研究で提案した DNA 断片配列を DB 内のタンパク質配列にマッピングする新システムの実験を行った．実験に使用した CPU は Opteron 2224 SE (3.2GHz)，GPU は Tesla S1070(1.44 GHz) である．以下に実験の手順と結果について述べる．

4.1 使用データ

本研究では既知タンパク質配列の DB としては京都大学 KEGG[7][8][9] の Web サイトから 2009 年 4 月 30 日にダウンロードした genes.pep というタンパク質配列データを使用した．この DB の中の配列の本数は約 423 万本，全配列の合計長は約 15 億残基である．また，クエリとなる DNA 断片配列は東京工業大学 大学院生命理工学研究科 生命情報専攻の黒川顕教授より 60 塩基の配列 10 万本を頂き，速度比較用としてこれをそのまま使用した．また既存法 (BLAST) との精度比較用として 10 万本の中からランダムに選んだ 2,000 本を使用した．

4.2 新システムで使用した値

簡単な試行錯誤を経た結果，今回の実験で使用した変数 (3 章参照) の値を示す．

- K-mer の K の値は 3，4，5
- クエリを K-mer で区切る際に何文字ずらしに区切っていくかを指定する s の値は 2
- マッピング候補探索の際に，いくつ key があれば候補にするかを指定する閾値 t の値は 2
- マッピング候補探索の際に，DB を区切る領域の大きさ r の値は 8
- アラインメントのスコア計算の際，アラインメント領域の拡張幅 e の値は 2

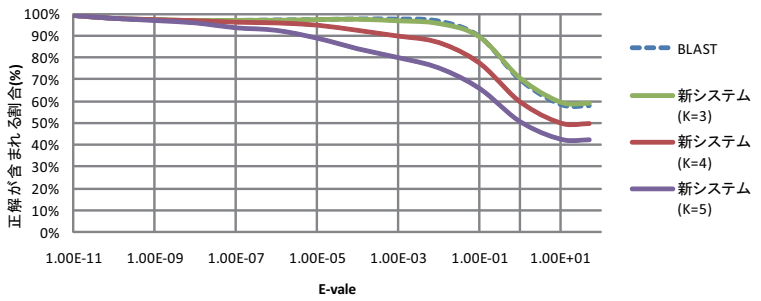

図 5 E-value を変化させた場合の正解が含まれる割合（%）
Fig. 5 Correct alignment rate vs. E-value

また，アラインメントのスコア行列は PAM30，ギャップのペナルティは-8 を使用した．

4.3 既存法（BLAST）のオプション

今回使用した BLAST のオプションは以下の通りであり，ギャップペナルティ等は新システムと同じものを用いている．

%blastall -p blastx -G 8 -E 8 -g T -F F -e 50 -M PAM30

4.4 マッピング精度の比較

マッピング精度を比較するために，既存の近似計算手法 (BLAST) と，提案手法との間で，クエリ毎にマッピング位置とアラインメントの正確さを評価した．正解には，厳密な動的計画法を行う SSEARCH の結果を用いて，両手法の結果に SSEARCH が与える正解が含まれる割合を比較した．**図 5** に結果を示す．結果として，K=3 の時は新システムは BLAST と同等の結果となった．K=4 の場合は新システムの方が若干精度が落ちるものの，E-value が十分に低い領域では BLAST とほぼ同等の精度が得られた．一方，K=5 の場合は BLAST と比べると E-value が低い値であっても精度の低下がみられた．

4.5 計算速度比較

速度比較には 60 塩基の DNA 配列 10 万本を用い，計算時間を測定した．測定の対象は，新システムの K=3，4，5，それぞれの CPU のみを使用した場合，1 GPU の場合，4 GPU の場合，及び既存法 (BLAST) の 1 thread の場合と 4 thread の場合の計 11 通りとし，それら全ての計算時間を測定した．BLAST の 1 thread の場合の計算速度を 1 とした場合の速度向上比を**図 6** に示す．結果として，GPU を使用した場合はどの場合も BLAST の 1thread

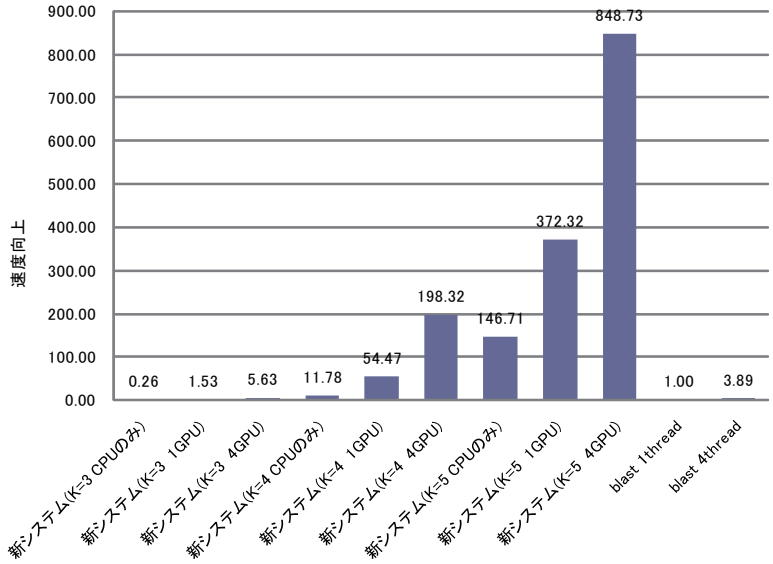

図 6 計算速度向上比（BLAST の 1thread を 1 とした場合）
Fig. 6 Speed up ratio (Compared to blast 1 thread)

の時よりも高速化出来た．特に BLAST の 1 thread と比べると K=4 の時，1 GPU では約 50 倍，4 GPU では約 200 倍，K=5 の時には 1 GPU では約 370 倍，4 GPU では約 850 倍の高速化が達成できた．

4.6 考　　察

4.6.1 マッピング精度について

K=3 の時には新システムと BLAST との結果はほぼ一致しているものの，K=4，5 と大きくするにつれて徐々に精度が落ちていく．これは K が大きくなるにつれてマッピング候補の探索時に，より長い部分文字列の一致を使用し，粗い探索を行なうためである．しかし，実際は E-value が高いヒットは偶然でも得られる可能性があるため，大量データ解析の場合にはあまり使われない．このため，新システムの K=4 でも実用上，十分な精度が出ていると考えられる．一方，K=5 の場合は BLAST と比べると E-value が低い値であっても精度の低下がみられた．

4.6.2 計算速度について

まず，BLAST と新システムについて比較する．GPU を使用した場合はどの場合も BLAST の 1thread の時よりも高速となっている．また，K が大きくなるにつれて BLAST との速度向上の値が大きくなっている．これは K が大きくなるにつれてマッピング候補の探索時に，より長い部分文字列の一致を使用し，粗い探索を行なうため，高速に計算出来ていると考えられる．次に新システムの CPU のみの場合と GPU を使用した場合について比較する．CPU のみの場合と 1 GPU の場合では 1 GPU 場合の方が約 3〜6 倍高速化出来た．これは計算に多くの時間を要するマッピング候補探索の部分とマッピング候補毎のスコア計算の部分が GPU によって高速化出来たためである．次に新システムの 1 GPU の場合と 4 GPU の場合を比較する．K=5 の場合には約 2 倍となっている．これは GPU が計算している以外の時間の割合が大きくなったためである．一方，マッピング候補探索とスコア計算に時間がかかっている K=3，4 の場合には約 4 倍となっており，並列化の効果が出ていると考えられる．

5. 結　　論

5.1 本研究の成果

本研究ではメタゲノム解析にも対応可能な曖昧さを許容し，GPU を用いることで高速に実行可能となる配列マッピングのアルゴリズムを提案し，Tesla 上でこの実装を行った．実装した新システムでは K=4 の場合は BLAST と比べると若干落ちるものの，E-value の低い領域ではほぼ同等の精度が得られ，実用上，十分な精度を持っていることが示された．計算速度においては GPU を使用した場合はどの場合も BLAST の 1 thread よりも高速化出来た．特に BLAST の 1 thread と比べると K=4 の時，1 GPU では約 50 倍，4 GPU では約 200 倍，K=5 の時には 1 GPU では約 370 倍，4 GPU では約 850 倍の高速化が達成できた．

5.2 今後の課題

今後，DNA シーケンサーから出力されるデータは技術の発展により，さらに増大すると考えられ，また，読み取られる DNA 断片の長さも長くなっていくと考えられる．このため今よりもさらにマッピングを高速化する必要がある．また，現在の新システムではシーケンサーが読み取った短い断片配列を対象としているが，これをさらに長い断片配列でも同様に高速に計算できるようにすることが将来的には求められる．

Vol.2010-BIO-21 No.30
2010/6/19

6. 謝　　辞

本稿で貴重なデータを使わせ頂き，また終始ご助言を頂いた東京工業大学 大学院生命理工学研究科 生命情報専攻の黒川顕教授に御礼申し上げます.

参 考 文 献

1) Andrew D Smith,Zhenyu Xuan,Michael Q Zhang: “Using quality scores and longer reads improves accuracy of Soleza read mapping”, *BMC Bionfomartics*, 9: 128, 2008.
2) Langmead B, Trapnell C, Pop M, Salzberg SL: “Ultrafast and memory-efficient alignment of short DNA sequences to the human genome”, *Genome biology*, 10,25, 2009.
3) Smith TF, Waterman MS: “Identification of Common Molecular Subsequence”, *Journal of Molecular Biology*, 17(2): 147: 195-197, 1981.
4) Pearson WR., Searching protein sequence libraries: “comparison of the sensitivity and selectivity of the Smith-Waterman and FASTA algorithms”, *Genomics*, 3:635-650, 1991.
5) Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ: “Basic local alignment search tool”, *Journal of Molecular Biology*, 215: 403-410, 1990.
6) Manavski SA, Valle G: “CUDA compatible GPU cards as efficient hardware accelerators for Smith-Waterman sequence alignment”, *BMC Bioinformatics*, 9:26, 2008.
7) Kanehisa, M., Goto, S., Furumichi, M., Tanabe, M., and Hirakawa, M.: “KEGG for representation and analysis of molecular networks involving diseases and drugs”, *Nucleic Acids Res*, 38, 355-360, 2010.
8) Kanehisa, M., Goto, S., Hattori, M., Aoki-Kinoshita, K.F., Itoh, M., Kawashima, S., Katayama, T., Araki, M., and Hirakawa, M.: “From genomics to chemical genomics: new developments in KEGG”,*Nucleic Acids Res*, 34, 354-357, 2006.
9) Kanehisa, M. and Goto, S.: “KEGG: Kyoto Encyclopedia of Genes and Genomes”, *Nucleic Acids Res*, 28, 27-30, 2000.