論 文 内 容 の 要 旨

論文題目

Development of Basic Algorithms for Genome Mapping of Short Reads

（短配列ゲノム・マッピングの基本アルゴリズムの開発）

氏名　　木 村　宏 一

本論文では、大量の短配列データに対して、参照ゲノム配列と比較照合して類似する部分配列の位置を特定する「ゲノム・マッピング問題」を、高速かつ高精度に解くために新たに開発した幾つかの基本アルゴリズムについて述べる。対象とする短配列データは、長さ 30～100 塩基程度の数百万本以上の DNA 塩基配列データであり、対象とする参照ゲノム配列は、ヒトなどの高等生物の数十億塩基対にも及ぶ大型のゲノム塩基配列である。

このようなゲノム・マッピング問題は、2000 年代後半に「次世代型 DNA シーケンサ」と称される超並列型 DNA シーケンサが登場したことを背景として、そこから得られる大量の DNA 配列データを解析して生物学的に有用な情報を得るための最初の重要なステップの一つとして、近年注目されるようになってきた。

第 1 章のイントロダクションでは、このような問題背景について手短に紹介し、コンピュータ・サイエンスの分野で関連する幾つかの技術について言及したのち、以後の各章で取り上げる話題の位置付けを述べる。

第 2 章では、Burrows-Wheeler 変換を用いて高速なゲノム・マッピングを行う際に基礎となる、ランク関数とセレクト関数の新しい高速計算法を提案した。これらの関数は、指定された部分文字列内に含まれる A や C などの特定の文字の出現回数をカウントする関数である。ゲノム・マッピング問題は、Burrows-Wheeler 変換されたゲノム配列上でランク関数やセレクト関数を計算する問題に帰着させることができ、それにより効率的に解くことができる。これらの関数の計算に適したデータ構造とし階層的バイナリ文字列（Hierarchical Binary String）を提案した。これは、バイナリ文字列に対して、その先頭から各文字位置までの部分和のビット表示を 8 ビットを 1 桁として分解し、各桁を上位桁に上がるごとに 256 分の 1 に間引きその差分ビットを並べた列を上位桁のバイナリ文字列として、階層構造を与えたものである。この構造化に伴うメモリ量のオーバヘッドの割合は、文字列長に依存せず常に約 3.5%と少ない。長さ n のバイナリ文字列には、h を $\log_2 n / 8$ 以上の最小の整数として、h 階層の構造が与えられる。上記の関数は階層数 h に比例した時間で高速に計算でき、例えば 64 ギガビットのランダム文字列の場合であれば、8 ギガバイト強のメモリと 3 ギガヘルツの CPU で 1 マイクロ秒未満で計算できる。これらを従来から知られている他の方法と比較すると、メモリ量のオーバヘッドの割合は約 20 分の 1 に抑えられ、計算速度は少なくとも同等以上になると評価された。また、これらをマッピング問題に応用して評価した。長さ 24～41 塩基の短配列データに対して、クエリー配列当たり高々1 個以下の不一致（置換、挿入、または、欠失）を許す条件を課して、上記 CPU で毎秒 2,000～4,800 本程度をマッピングできた。これは、短配列用の最も高速なマッピング・ツールの一つである ELAND より 3～7 倍程度高速であった。

第 3 章では、ペアエンド配列のゲノム・マッピング問題を扱う。次世代型 DNA シーケンサでは、配列長が比較的短く、マッピング位置を特定するための情報が不足しがちである。そこで、この短所を補うため、ペアエンド方式とよばれるシーケンシング法が用いられる。この方式では、長さを調整された配列断片の両端の末端配列をペアとしてシーケンシングする。従って、ペアエンド配列のゲノム・マッピング問題では、ペアをなす配列を、互いにほぼ一定距離だけ離れた位置にマッピングすることが求められる。即ち、この場合、単に文字列照合するだけでなく、位置の制約を考慮する必要がある。最も素朴な解法は、最初に各配列

を独立にゲノム・マッピングした後で、位置制約を満たすものを選ぶという方法である。この方法では、ペアエンド配列がリピート領域に由来する場合、最初に膨大なヒットを生じて、処理効率の低下を招きやすい。一方、サフィックス・アレイや Burrows-Wheeler 変換などの従来の方法では、辞書式順にデータをソートすることにより、文字列照合は効率的に行えるが、そのとき同時に位置の制約を考慮することは困難であった。そこで、文字列照合と位置制約考慮を同時に効率的に行える新しいデータ構造として、局所化サフィックス・アレイ（Localized Suffix Array）を提案した。局所化サフィックス・アレイは、大まかな位置情報を表わす上位ビットと、そこで局所的にソートされた辞書式順序情報を表わす下位ビットから構成され、これら 2 種類の情報が混在した情報を扱えるという特徴をもち、実装上は、通常のサフィックス・アレイの内部のビットを入れ替えたものとして実現できる。局所化サフィックス・アレイを用いることにより、リピート領域に由来する配列の場合は、多数のマッピング候補位置を徐々に詳細化しながら、制約を満たさない候補位置は一括して排除することが可能となる。これらの計算はランク関数の計算に帰着され、第 2 章で提案したアルゴリズムを用いて高速に計算される。それにより候補位置が 2,000 箇所以上ある場合に平均 10 倍以上の高速化の効果が得られることを、36 塩基長のペアエンド配列の実データを用いて確認した。

第 4 章では、2 塩基符号化（Two-Base Encoding）方式で得られたシーケンシング・エラー率が比較的高い配列データに対するゲノム・マッピング問題を扱う。2 塩基符号化方式では隣り合った 2 塩基の 4×4 通りの組み合わせを 4 色の蛍光で検出するため、一次配列データとしては 4 色からなるカラー配列が得られ、それを塩基配列に復号化する。この方式のメリットの一つは、カラー配列と参照ゲノム配列のアラインメントが得られれば、符号化の冗長性を利用してシーケンシング・エラーを訂正することができ、最終的に高品質の配列データが得られる点にある。そのためには、多数のエラーを含んだ状態で、カラー配列と参照ゲノム配列とのアラインメントを求める必要がある。即ち、多数の不一致を許容するゲノム・マッピングを求める問題を解く必要がある。このような問題、或いは、一般に高感度なホモロジー・サーチに適した方法として、穴開きシード（spaced seeds）を利用する方法が従来から知られている。しかしながら、高感度化のためには複数のマスクパターン（穴開きのパターン）を利用する必要が生じてインデクス・データの肥大化を招きやすい。また、穴開きシードを Burrows-Wheeler 変換で効率的に扱うことは難しく、それらのインデクス・データを大幅に圧縮することは望めない。そこで、穴開きシードを用いずに、通常の連続シードのみを用いて、Burrows-Wheeler 変換によるコンパクトなインデクス・データとメモリ容量で、高速・高感度にゲノム・マッピングを行う方法を検討した。但し、高頻度の不一致に対応するためには、シード内部にも若干の不一致を許容する必要があり、また、比較的短いシードを利用する必要がある。このようなシードは計算コストの上昇を招きやすいため、高感度で効率的なシードのセットを構築するための新たな方法を提案した。シードの役割は、広範囲なゲノム全体の中から候補位置を絞り込むこと、即ち、候補位置の情報を知らせることにある。その情報量は、逆に、候補位置がどの程度曖昧か、即ち、候補位置がどれだけ残っているかによって捉えることができる。例えば、リピート領域に由来するシードは、多数の候補位置を提示し、その情報量は少ないと考えられる。また、一般に、マッピング処理全体の計算コストは、全てのシードに対する候補位置の数の総和に概ね比例する。従って、候補位置の数はシードの特性を表わす重要な指標である。また、その対数をとったものは、情報理論の観点からは候補位置の曖昧性を測るエントロピーとも解釈できる。そこで、エントロピーの観点から高感度で効率的なシードのセットを求める方法として、等エントロピー分割法（Equi-Entropy Partitioning, EEP）を提案した。この方法では、クエリー配列を同程度のエントロピーをもつ断片配列に分割し、それらの断片配列を組合わせて各シードを構成する。それにより、例えば、リピート領域では長めのシードが採用され、逆に一意性が高い領域では短めのシードが採用され、高精度と高速性を両立させることが可能になる。HLA（Human Leukocyte Antigen）領域をシーケンシングした一千万本の 50 塩基長のシングル・エンド配列の実データを用いて評価を行い、穴開きシードを利用した短配列用の高感度なゲノム・マッピング・ツールである BFAST と比較して、ほぼ同程度の精度を達成でき、約 1.3 倍高速になるという結果を得た。そのとき、ヒトゲノムのインデクス・データのサイズは約 30 分の 1（約 4 ギガ・バイト）に、必要なメモリ量は約 4 分の 1（約 4 ギガ・バイト）に抑えられた。

最後の第 5 章では結言を述べる。

以上のように、本論文では、次世代型 DNA シーケンサから得られる大量の短配列データから生物学的知見を得るために有用な情報を提供するゲノム・マッピング処理を、高速・高精度に行うための幾つかの基本アルゴリズムを提案し、その有効性を実データを用いて確認した。