数理解析研究所講究録
第1849巻 2013年 18-21

部分文字列最大密度索引

Indexing maximum densities of characters in substrings of a sparse string

酒井義文*
東北大学大学院農学研究科

Yoshifumi Sakai
Graduate School of Agricultural Science, Tohoku University,
Sendai 981-8555, Japan

概要

S を記号 σ が k 回現れる長さ n の文字列とする．S 中の各 σ には n や k に依存しない整数の重みが与えられているとする．1 から n までの各区間幅 w に対する S の幅 w の連続領域における σ 重みの総和の最大値をすべて求めるアルゴリズムを提案する．このアルゴリズムの実行時間は，σ の重みがすべて 1 である場合，$O(n+\min(k^2/\log k,(n-k)^2/\log(n-k)))$，$\sigma$ の重みがすべて自然数である場合，$O(n+k^2/\log k)$，σ の重みがすべて整数である場合，$O(n+k^2\log\log k)$ である．

1 はじめに

S を記号 σ が k 回現れる長さ n の任意の文字列とする．S に現れる各 σ には n や k に依存しない任意の整数の重みが与えられているとする．1 から n までの各区間幅 w に対して，$W_{S,\sigma}(w)$ で，S の幅 w の連続領域における σ の重みの総和の最大値を表す．n 要素配列 $[W_{S,\sigma}(1),W_{S,\sigma}(2),\ldots,W_{S,\sigma}(n)]$ を，$W_{S,\sigma}$ 索引とよぶ．本稿では，与えられた文字列 S に対して $W_{S,\sigma}$ 索引を求める問題について考える．この問題を解くアルゴリズムは，たとえば区間幅に依存する閾値よりも大きな重みつき出現頻度で特定の記号が現れる最大区間幅をもつ連続領域を見つけ出すことに利用できるため，データマイニングやパターン認識、分子生物学などの様々な分野において多くの応用をもつと考えられる．たとえば，図 1 を見られたい．

最近，Moosa と Rahman [3] は，2 値アルファベット文字列の置換マッチング問題を解くための前処理アルゴリズムとして，$W_{S,\sigma}$ 索引を求める $O(n^2/\log n)$ 時間アルゴリズムを提案した．このアルゴリズムは，再帰的に 2 つの部分問題と 1 つの和最小値畳込み計算に分割することで問題を解く．ここで，2 つの m 要素ベクトル $\langle a_0,a_1,\ldots,a_{m-1}\rangle$，$\langle b_0,b_1,\ldots,b_{m-1}\rangle$ に対する和最小値畳込みは，$c_i=\min_{0\le j\le i}(a_j+b_{i-j})$ である m 要素ベクトル $\langle c_0,c_1,\ldots,c_{m-1}\rangle$ であり，Bremner ら [1] の方法を用いると $O(n^2/\log n)$ 時間で得ることができる．本稿では，ベクトルの要素の順番を入れ替えたり添え字を変更することによって，m 要素ベクトル $\langle a_1,a_2,\ldots,a_m\rangle$，$\langle b_1,b_2,\ldots,b_m\rangle$ に対する和最小値畳込みを，$c_i=\min_{i\le j\le m}(a_j+b_{j-i+1})$ である m 要素ベクトル $\langle c_1,c_2,\ldots,c_m\rangle$ と扱う．また，$c_i=\max_{i\le j\le m}(a_j+b_{j-i+1})$，$c_i=\min_{i\le j\le m}(a_j-b_{j-i+1})$，$c_i=\max_{i\le j\le m}(a_j-b_{j-i+1})$ である m 要素ベクトル $\langle c_1,c_2,\ldots,c_m\rangle$ をそれぞれ，和最大値畳込み，差最小値畳込み，差最大値畳込みとよぶ．ただし，これらを求める計算を一括して和最小値畳込み計算とよび，Bremner ら [1] の方法を用いて $O(m^2/\log m)$ 時間で実行されるものと仮定する．

本稿では，$W_{S,\sigma}$ 索引を求める単純なアルゴリズムを提案する．このアルゴリズムは，Moosa と Rahman[3] のアルゴリズムと同様に Bremner ら [1] の方法を用いるものの，和最小値畳込み計算を再帰的に何度も実行するのではなく，高々 1 回しか実行しない．提案するアルゴリズムの漸近的な実行時間は，σ の重みがすべて 1 である場合，$O(n+\min(k^2/\log k,(n-k)^2/\log(n-k)))$，$\sigma$ の

* Email: sakai@biochem.tohoku.ac.jp

```
   1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
A  A G T T A C T G G C A A A G T G A C A T G A C G T G
B  A T T T G C T G G T C T A G T T A C T T C G C G A G
S  A * T T * C T G G * * * A G T * A C * T * * C G * G
```

図1　2本の文字列 A, B のコンセンサス文字列 S の例．ただし，A と B の対応する位置の記号が不一致であることを S では記号 $\star$ で表す．もしも S において区間幅 w に対して $w^{0.75}$ を超える個数の $\star$ を含む最大区間幅の連続領域を A と B の非類似領域として求めたいならば，見つけなければならないのは下カッコで示された 10 番目の文字から 22 番目の文字までの区間幅 13 の領域である．2 番目に広い区間幅をもつ領域は，9 番目，10 番目，19 番目の文字を左端とする幅 4 の 3 つの領域である．記号 $\star$ に対する重みをすべて 1 とした場合の $W_{S,\star}$ 索引が利用可能ならば，これらの領域を S の長さの線形時間で求めることができる．

重みがすべて自然数である場合，$O(n+k^2/\log k)$，σ の重みがすべて整数である場合，$O(n+k^2\log\log k)$ である．したがって，S に σ が疎にしか現れない場合（あるいは，σ の重みがすべて 1 であるときは，S に σ が密に現れる場合においても），提案するアルゴリズムを用いることで，Moosa と Rahman[3] のアルゴリズムと比較して高速に $W_{S,\sigma}$ 索引を求めることができる．

2　アルゴリズム

2.1　単純な $O(n^2/\log n)$ 時間アルゴリズム

$W_{S,\sigma}$ 索引は，以下の補題より，n 要素ベクトルに対する 1 回の和最小値畳込み計算により得られる．

補題 1　$1 \le i \le n$ である任意の添え字 i に対して，a_i が S の長さ i である接頭辞における σ の重みの総和に等しく，b_i が S の長さ $i-1$ である接頭辞における σ の重みの総和に等しい 2 つの n 要素ベクトル $\langle a_1, a_2, \ldots, a_n\rangle$，$\langle b_1, b_2, \ldots, b_n\rangle$ の差最大値畳込みは，$\langle W_{S,\sigma}(1), W_{S,\sigma}(2), \ldots, W_{S,\sigma}(n)\rangle$ に等しい．

証明　S の区間幅が w である任意の連続領域に対して，この領域が S の $j-w+1$ 文字目から j 文字目までの範囲ならば，j は $w \le j \le n$ を満たす．また，この領域は S の長さ j の接頭辞から長さ $j-w$ の接頭辞を削除することで得られるから，この領域に現れる σ の重みの総和は $a_j - b_{j-w+1}$ に等しい．したがって，補題は成立する．□

2.2　単位重みの場合

記号 σ の重みをすべて 1 とする．$W_{S,\sigma}$ 索引を求める $O(n+k^2/\log k)$ 時間アルゴリズムと $O(n+(n-k)^2/\log(n-k))$ 時間アルゴリズムを与える．これらのアルゴリズムは以下の補題に基づく．

補題 2　$1 \le w \le n$ である任意の区間幅 w に対して，$W_{S,\sigma}(w)$ は $W_{S,\sigma}(w-1)$ または $W_{S,\sigma}(w-1)+1$ のどちらかに等しい．ただし，$W_{S,\sigma}(0)=0$ とする．

証明　長さがちょうど 1 だけ異なり一方がもう一方の接頭辞あるいは接尾辞である任意の 2 つの文字列において，σ が現れる回数の差は 0 また 1 である．したがって，補題は成立する．□

この補題と $W_{S,\sigma}(n)=k$ より，$W_{S,\sigma}$ 索引は k 要素配列 $[w(1), w(2), \ldots, w(k)]$ として表せる．ただし，$w(i)$ は $W_{S,\sigma}(w)=i$ である最小の w である．この k 要素配列を n 要素配列である $W_{S,\sigma}$ 索引に $O(n)$ 時間で変換することは容易である．よって，次の補題より，$W_{S,\sigma}$ 索引を求める $O(n+k^2/\log k)$ 時間アルゴリズムを直ちに得ることができる．

補題 3　S 中の k 個の σ を先頭から順に $\sigma_1, \sigma_2, \ldots, \sigma_k$ とする．$1 \le i \le k$ である任意の添え字 i に対して，a_i と b_i+1 がどちらも σ_i の出現位置に等しい 2 つの k 要素ベクトル $\langle a_1, a_2, \ldots, a_k\rangle$，$\langle b_1, b_2, \ldots, b_k\rangle$ の差最小値畳込みは，$\langle w(1), w(2), \ldots, w(k)\rangle$ に等しい．

証明　$w(i)$ は，S において σ がちょうど i 回現れる連続領域の最小区間幅に等しい．$i \le j \le k$ である任意の添え字 j に対して，S における σ_{j-i+1} から σ_j までのちょうど i 個の σ を含む最小区間幅の連続領域は，S の長さ a_j の接頭辞から長さ b_{j-i+1} の接頭辞を削除することで得られる．この連続領域の区間幅は $a_j - b_{j-i+1}$ に等しい．よって補題は成立する．□

同様に，$W_{S,\sigma}$ 索引は，$n-k$ 個の要素からなる配列 $[w'(1), w'(2), \ldots, w'(n-k)]$ として表せる．ただし，$w'(i)$ は $w - W_{S,\sigma}(w) = i-1$ である最大の w である．この $n-k$ 要素配列を $W_{S,\sigma}$ 索引に $O(n)$ 時

間で変換することは容易である．よって，次の補題より，$W_{S,\sigma}$ 索引を求める $O(n+(n-k)^2/\log(n-k))$ 時間アルゴリズムが直ちに得られる．

補題 4 S 中の σ 以外の記号をすべて同一の記号 τ とみなし，先頭から順に $\tau_1, \tau_2, \ldots, \tau_{n-k}$ とする．$1 \leq i \leq n-k+1$ である任意の添え字 i に対して，$a_i + 1$ が τ_i の出現位置に等しい $n-k+1$ 要素ベクトル $\langle a_1, a_2, \ldots, a_{n-k+1} \rangle$ と，b_i が τ_{i-1} の出現位置に等しい $n-k+1$ 要素ベクトル $\langle b_1, b_2, \ldots, b_{n-k+1} \rangle$ の差最大値畳込みは，$\langle w'(1), w'(2), \ldots, w'(n-k), n \rangle$ に等しい．ただし，便宜的に，存在しない記号 τ_0，τ_{n-k+1} の出現位置をそれぞれ，$0, n+1$ とする．

証明 $w'(i)$ は，S において τ がちょうど $i-1$ 回現れる連続領域の最大区間幅に等しい．$i \leq j \leq n-k+1$ である任意の添え字 j に対して，S における τ_{j-i+1} から τ_{j-1} までのちょうど $i-1$ 個の τ を含む最大区間幅の連続領域は，S の長さ a_j の接頭辞から長さ b_{j-i+1} の接頭辞を削除することで得られる．この連続領域の区間幅は $a_j - b_{j-i+1}$ に等しい．よって補題は成立する．□

2.3 自然数重みの場合

S 中の各 σ の重みを n や k に依存しない任意の自然数である場合に $W_{S,\sigma}$ 索引を求めることのできる $O(n + k^2/\log k)$ 時間アルゴリズムを与える．

σ の重みが正であることから，$0 \leq W_{S,\sigma}(1) \leq W_{S,\sigma}(2) \leq \cdots \leq W_{S,\sigma}(n) = O(k)$ が成立する．このことから，$l = W_{S,\sigma}(n)$ とすると，$W_{S,\sigma}$ 索引は，l 要素配列 $[w(1), w(2), \ldots, w(l)]$ として表せる．ここで，$w(i)$ は，S において σ の重みの総和が i 以上である連続領域の最小区間幅である．この l 要素配列は，前節で導入した k 要素配列 $[w(1), w(2), \ldots, w(k)]$ の自然な拡張になっている．ただし，$w(i-1) \leq w(i)$ ではあるものの，前節の場合と異なり，必ずしも $w(i-1) < w(i)$ が成立するとは限らない．もしも $w(i-1) = w(i)$ ならば，$W_{S,\sigma}(w) = i-1$ を満たす区間幅 w は存在しない．しかし，$w(i-1) < w(i)$ を満たす添え字 i にのみ注目することで，この l 要素配列を $W_{S,\sigma}$ 索引に $O(n)$ 時間で変換することも容易にできる．よって，次の補題より，$W_{S,\sigma}$ 索引を求める $O(n + k^2/\log k)$ 時間アルゴリズムを直ちに得ることができる．

補題 5 $1 \leq i \leq l$ である任意の添え字 i に対して，a_i が S において σ の重みの総和が i 以上である接頭辞の最小区間幅に等しい l 要素ベクトル $\langle a_1, a_2, \ldots, a_l \rangle$ と，b_i が S において σ の重みの総和が i 未満である接頭辞の最大区間幅に等しい l 要素ベクトル $\langle b_1, b_2, \ldots, b_l \rangle$ の差最小値畳込みは，$\langle w(1), w(2), \ldots, w(l) \rangle$ に等しい．

証明 S 中の k 個の σ を先頭から順に $\sigma_1, \sigma_2, \ldots, \sigma_k$ とする．$1 \leq i \leq l$ である任意の添え字 i に対して，a_i の定義より，S の a_i 文字目の記号は σ である．この σ を $\sigma_{x(i)}$ とする．このとき，$1 = x(1) \leq x(2) \leq \cdots \leq x(l) = k$ が成立する．また，$2 \leq i \leq l$ である任意の添え字 i に対して，$a_i = a_{i-1}$ ならば，$x(i) = x(i-1)$，そうでないならば，$x(i) = x(i-1) + 1$ である．b_i の定義より，S の $b_i + 1$ 文字目の記号も σ である．この σ を $\sigma_{y(i)}$ とする．

i を，$1 \leq i \leq l$ である任意の添え字とする．$w(i)$ は S において σ の重みの総和が i 以上である連続領域の最小区間幅であるから，$1 \leq y \leq x \leq k$ である添え字 y，x が存在して，S における σ_y から σ_x までの連続領域の区間幅は $w(i)$ であり，この領域における σ の重みの総和は i 以上である．j を $x(j) = x$ である最大の添え字とする．したがって，S の $\sigma_{x(j)}$ を末尾とする接頭辞における σ の重みの総和はちょうど j である．$y(j-i+1) = y$ であることを以下に示す．定義より，S の $b_{j-i+1} + 1$ 文字目の記号は $\sigma_{y(j-1+1)}$ である．また，b_{j-i+1} は，S において σ の重みの総和が $j-i$ 以下である接頭辞の最大区間幅に等しい．したがって，S の $\sigma_{y(j-i+1)}$ から $\sigma_{x(j)}$ までの連続領域における σ の重みの総和は $j-(j-i) = i$ 以上である．一方，S の $\sigma_{y(j-i+1)+1}$ から $\sigma_{x(j)}$ までの連続領域における σ の重みの総和が i 以上であると仮定すると，S の $\sigma_{y(j-i+1)}$ を末尾とする接頭辞における σ の重みの総和は $j-i$ 以下であるから，b_{j-i+1} の定義に矛盾する．したがって，S の $\sigma_{y(j-i+1)+1}$ から $\sigma_{x(j)}$ までの連続領域における σ の重みの総和は i よって，$y(j-i+1) = y$ が成立する．このことは，$a_j - b_{j-i+1} = w(i)$ であることを意味する．

$w(1) \leq w(2) \leq \cdots \leq w(l)$ より，$i \leq j \leq l$ である任意の添え字 j に対して，$a_j - b_{j-i+1} \geq w(i)$ であることも容易に確認できる．よって，補題は成立する．□

2.4 整数重みの場合

S 中の各 σ の重みを n や k に依存しない任意の整数である場合に $W_{S,\sigma}$ 索引を求めることのできる $O(n + k^2 \log\log k)$ 時間アルゴリズムを与える．単純な $O(nk)$ 時間アルゴリズムをはじめに与え，後

に，これを $O(n+k^2 \log\log k)$ 時間で動作するように変形する．

S 中の k 個の σ を先頭から順に $\sigma_1, \sigma_2, \ldots, \sigma_k$ とする．σ_i は S において a_i 文字目に現れるとする．$a_0 = 0$，$a_{k+1} = n+1$ とする．S の連続領域を，以下のような $k(k+1)$ 個のグループに分類する．$0 \le i \le k$ である任意の添え字 i と，$1 \le j \le k$ である任意の添え字 j に対して，$\mathcal{S}(i,j)$ で，S の連続領域で σ_{i+1} から σ_j までのちょうど $j-i$ 個の σ を含むすべてからなる集合を表す．$W(i,j)$ で，$\sigma_{i+1}, \sigma_{i+2}, \ldots, \sigma_j$ の重みの総和を表す．また，$u(i,j) = \max(1, a_i - (a_{j+1} - 1))$，$v(i,j) = (a_{i+1} - 1) - a_j$ とする．この定義より，$\mathcal{S}(i,j)$ に含まれる S の各連続領域における σ の重みの総和はすべて $W(i,j)$ に等しい．更に，$\mathcal{S}(i,j)$ に含まれる S の連続領域の区間幅は $u(i,j)$ 以上かつ $v(i,j)$ 以下であり，$\mathcal{S}(i,j)$ 中にこの範囲の任意の区間幅をもつ S の連続領域が少なくとも 1 つ以上存在する．S の任意の連続領域はいずれかの $\mathcal{S}(i,j)$ に属するから，$W_{S,\sigma}$ 索引を $[W(1), W(2), \ldots, W(n)]$ として求める以下の単純な $O(nk)$ 時間アルゴリズムが直ちに得られる．

```
1: For each w from 1 to n, W(w) ← 0;
2: for each j from 1 to n,
3:     for each i from 0 to n,
4:         for each w from u(i,j) to v(i,j),
5:             W(w) ← max(W(w), W(i,j)).
```

このアルゴリズムを高速化するため，2，3 行目で指定される添え字対 (i,j) の順序を，$W(i,j)$ の昇順に変更する．この変更により，5 行目では $W(w)$ と $W(i,j)$ の比較をすることなしに，常に $W(i,j)$ を $W(w)$ に代入することができる．このことは，もしもアルゴリズムが $[W(1), W(2), \ldots, W(n)]$ を $W(w) \neq W(w+1)$ である（$W(w)$ をラベル付けられた）w すべてからなる集合 $\mathcal{W}$ としてもつならば，4，5 行目の実行を $\mathcal{W}$ の要素に関する以下のならし定数回の操作により完了することができることを意味する．

1. $\mathcal{W}$ 中に $u(i,j)$ 以上かつ $v(i,j)+1$ 以下である要素が存在するならば，そのような最大の w を $\mathcal{W}$ から削除した後に，$W(w) < W(i,j)$ ならば，$W(w)$ をラベル付けられた $v(i,j)+1$ を $\mathcal{W}$ に新しい要素として加える．
2. $\mathcal{W}$ 中の $u(i,j)$ 以上かつ $v(i,j)$ 以下の要素を，最小のもの（あるいは最大のもの）から順にすべて削除する．
3. $\mathcal{W}$ 中の $u(i,j)$ 未満の最大の要素 w を探し，そのような w が存在しない，あるいは $W(w) < W(i,j)$ ならば，$W(i,j)$ をラベル付けされた $u(i,j)$ を $\mathcal{W}$ に新しい要素として加える．

上記のアルゴリズムの実行時間を以下に与える．S を走査することで，1 から k までのすべての添え字 i に対する a_i と，$\sigma_1, \sigma_2, \ldots, \sigma_i$ の重みの総和 $W(0,i)$ を $O(n)$ 時間で求めることができる．アルゴリズムは，これらの値を保持し，任意の添え字対 (i,j) に対する $W(i,j)$，$u(i,j)$，$v(i,j)$ を定数時間で求めるために用いる．Han[2] の整数ソートアルゴリズムを用いることで，$k(k+1)$ 個の添え字対 (i,j) すべてに対する $2k(k+1)$ 個の整数 $u(i,j)$，$v(i,j)$ を $O(k^2 \log\log k)$ 時間で昇順に並べることができる．この $2k(k+1)$ 個の整数のみを要素としてもつことのできる van Emde Boas[4] のデータ構造を用いて $\mathcal{W}$ を実装することで，先に述べた各添え字対 (i,j) に対する $\mathcal{W}$ の要素に関するならし定数回の操作を，ならし $O(\log\log k)$ 時間で実行できる．また，Han[2] の整数ソートアルゴリズムを用いることで，$k(k+1)$ 個の添え字対 (i,j) を $O(k^2 \log\log k)$ 時間で $W(i,j)$ の昇順に並べることができる．$\mathcal{W}$ を $O(n+k^2 \log\log k)$ 時間で $W_{S,\sigma}$ 索引に変換することも容易である．

以上より，提案したアルゴリズムを用いることで，$W_{S,\sigma}$ 索引を $O(n+k^2 \log\log k)$ 時間で求めることができる．

参考文献

[1] D. Bremner, T.M. Chan, E.D. Demaine, J. Erickson, F. Hurtado, J. Iacono, S. Langerman, P. Taslakian, Necklaces, convolutions, and $X + Y$, in: Y. Azar, T. Erlebach (Eds.), ESA 2006, LNCS 4168, pp. 160–171, 2006.

[2] Y. Han, Deterministic sorting in $O(n \log\log n)$ time and linear space, J. Algorithms 50 (2004) 96–105.

[3] T.M. Moosa, M.S. Rahman, Indexing permutations for binary strings, Inform. Process. Lett. 110 (2010) 795–798.

[4] P. van Emde Boas, Preserving order in a forest in less than logarithmic time and linear space, Inform. Process. Lett. 6 (1977) 80–82.