4C-4

パターンストリームにおけるデータ検索

山口信† 島田 諭† 三浦 孝夫‡
法政大学† 法政大学† 法政大学‡

1.前書

情報検索の分野では従来様々な手法によりデータから特定のパターンを発見する手法が提案されている。SuffixArray や SuffixTree は予めテキストデータに索引付けをしておくことにより、高速な検索を実現するアルゴリズムである。しかしながら大量のストリームデータにおいて、このようなデータ構造を構築、維持することは、空間的速度的に非常に高価であり、現実的でない。一方、KMP 法や BM 法といったそのアルゴリズムによる手法は、構造のないデータ検索に適しているといえる。またストリームパターンに対応する方法はあまり知られていない。しかし検索パターンが変動する中で検索を行わねばならない状況は十分に考えられる。例えば、株価は常に変動しているが、その変化の特徴を逐次過去のデータから検索することは、株主にとって有用なことである。本研究手法では KMP 法において、検索パターンが連続的に増加すると仮定し、増加分による検索を、差分情報を用いて高速に文字列検索するアルゴリズムを提案する。

2.KMP 法と BM 法

パターン中の部分文字列の接頭辞と接尾辞の最長一致長から瞬時に SKIP 量を計算し、冗長な検索を省略することができるアルゴリズムが KMP 法または BM 法である。skip 量をまとめた skip 表の構築時間はパターンの線形時間であり、一般にパターンはデータに比べ小さいので、この構築時間は大きくなりにくい。KMP 法では文字列検索はパターンの先頭から行い、BM 法は一般的に文字列検索をパターンの末尾から行う。末尾から比較し不一致点でスキップすることによりテキスト上で一度も比較しない文字が生じ、計算量を抑えることが出来る。ただし BM 法は日本語のように全角文字を用いる文化圏では空間的に高価な可能性がある。

3.パターンストリーム

KMP 法は検索時の一致長内の接頭辞と接尾辞の最長一致がパターンの幅に関与する。そこでその接尾辞と接頭辞の最長一致を MPL として予めテーブルに保持しておけばそれが skip 表の役割を果たすことになる。

MPL の定義

pattern[0,1,2,...,m]のとき
MPL(m)
=MAX{k|[pattern[0...k]=pattern[(m-k),...,m]], 0<k<m}

接頭辞と接尾辞の一部が重なっている場合、そのパターンは、接頭辞の重複を除く接頭辞部分の繰り返し文字列になっている。ただし最後の繰り返し部分は完全な接頭辞になっているとは限らない。また接頭辞と接尾辞の一部が重なっていない場合は以下のようになる、pattern[m],K=MPL(m)とするとき、

$$k<\frac{m}{2}$$

特に k=0 の時は pattern(m) ≠pattern(1)
かつ pattern(m-i,...,m) ≠pattern(0,..,i) である。
次に、差分的に求める MPL 長を MPL(m+1)とすると、MPL(m+1)は MPL(m)より 1 文字長くなるか、MPL(m)より短くなることを示す。αを MPL(m)で一致する文字列、z を接頭辞αの次の文字、x を MPL(m+1)で追加された文字とする。

α	z	…	α	x

まず x=z のときは、αZ=αx は明らかなので MPL(m+1)=MPL(m)+1 はひとまず成り立つが、実はこれ以上長い MPL(m+1)は存在しない。仮に存在すると仮定すると、より長い MPL(m+1)=”β”が存在して、下のようになる。

α	x	…	x	…	α	x
β				…	β	

接尾辞ベータの末尾は x ゆえ、接頭辞βは
β＝αx…x の並びになっている。ここでαx…＝β’とおいて接頭辞β＝β’x としておく。一方接尾辞βはβ＝…αx ＝β’x の並びになっている。つまり“…α”はβ’というより長い MPL(m)を保持していたことになる。この矛盾から MPL(m+1)=MPL(m)+1 が成り立つ。次に X≠z のときは MPL(m+1)≦MPL(m)となる。仮に MPL(m+1)＞

MPL(m)と仮定すると、MPL(m+1)の接頭辞βは、x≠z であるから、この長さは MPL(m)+1 以上になる。

α	z	…	x	…	α	x
β				…	β	

接尾辞βに着目するとβ＝…αx＝β'x となっている。ここで接頭辞βはβ＝αz…x であるからαz…x＝β'x これでは、MPL(m)はより長いものが既に存在していたことになり矛盾する。よってMPL(m+1)≦MPL(m)となる。

4.提案手法

直前の MPL(m)から MPL(m+1)を差分的に構築する。この時第 3 章で述べたように追加文字が pattern(MPL(m)+1)と一致した場合は MPL(m+1)=MPL(m)+1 となるが、そうでないない場合 MPL(m+1)は MPL(m)よりも短い範囲から探せば良いことがわかっている。その際のテーブルは、追加された文字 pattern[i]の直前の出現位置へのポインタが格納されている。こうすることにより比較回数が $\frac{1}{|\Sigma|}$ になることが期待できる。ここで|Σ|は半角文字の総数である。このリンク表を skip2 と名付け、リンクした位置から先頭まで一致する MPL が MPL(m+1)となる。MPL(M+1)を求めるこの手法の計算量は次の通りである。

$$O(1+|\alpha|+|\alpha|\times\frac{|\alpha|}{|\Sigma|}) = O(\frac{|\alpha|^2}{|\Sigma|})$$

ここでαは MPL(m)のことである。MPL(m+1)求める際、改めて KMP 法や BM 法を用いることでαの線形時間にすることは可能だが、新たに skip 表を作成するコストと、実際にはαは非常に小さいということ考慮し、今回は２乗のオーダのままにしてある。

5.実験

実験で用いるコーパスはロイター誌から 1996 年の記事 2586 種を抽出する。テキストデータはこの 2586 種類の記事からランダムに 2586 回抽出して結合したものとした。検索パターンは 2568 種類の記事から一つを選択したものである。

実験結果としてパターンの違いにおける性能評価結果を表 1 に示す。skip 表作成時比較コストは MPL(m+1)を構築する際の文字列比較回数と、表への挿入回数で、skip 表作成時挿入コストは skip 表への値の挿入回数を示し、差分法の場合は 2 つの表への挿入回数の合計となっている。skip 表作成コストは上記 2 つの合計を表し、それと検索時の比較回数を表す検索コストとの合計をとったものが総コストである。

表 1 パターンお違いにおける性能差

	差分平均	従来平均
skip 表作成時比較コスト	2150.4	2614346
skip 表作成時挿入コスト	4300.4	2611465
skip 表作成コスト	6450.8	5225811
検索コスト	2490802	2489882
総コスト	2497253	7715693

全ての実験結果で優秀な結果となった。KMP 法の skip 表は動的に作成すればほとんどコストが掛からず平均 800 倍の速度となった。しかしながら、データ検索のコストを含めると、3 倍程度の成果にとどまった。KMP 法では検索速度に課題が残る。また今回はパターンにロイターコーパスを用いたため、英語の文章では大文字と小文字の区別が存在してしまい、MPL は 2 以上になることは極めて稀な小規模なものとなった。そのため差分構築において 2 回以上の比較を行うことはまず無いと言える。同じ文字列が繰り返しでてこないパターンでは差分構築のコストを安く済ませられる反面、検索の効率が上がらず、また今回提案したリンク表である skip2 を十分に活用することができない。

6. 結論

パターンの先頭から MPL をとる KMP 法は、差分構築においてとても優秀な結果を出した。BM 法を用いればより高速な検索が期待できるが、パターン増加に対する差分構築は複雑であり、構築と検索の双方の観点から比較する必要がある。今後は異なる特性を示すコーパスを用いることにより、一層の有用性を示せると考える。

文献

[1] 石畑　清：アルゴリズムとデータ構造

Information retrieval in the pattern stream

Makoto Yamaguchi・Hosei
Satoshi Shimada ・HOsei
Miura Takao ・Hosei