文字列間の極大共通部分系列を求める線形時間アルゴリズム

3T−8

木田 和寿　中西 通雄　橋本 昭洋
大阪大学基礎工学部情報工学科 (kida@ics.es.osaka-u.ac.jp)

1 はじめに

Σを、2つ以上の文字からなる有限アルファベットとする。Σ 上の文字からなる有限長系列を文字列という。文字列 $s = a_1a_2\ldots a_n$に対して、文字列 $t = a_{i_1}a_{i_2}\ldots a_{i_j}(1 \le i_1 < i_2 < \ldots < i_j \le n)$ を s の部分系列という。例えば、文字列 $abcba$ に対して $abb, bba, abcba$ 等は部分系列である。Sを文字列の有限集合とする。もし、文字列 t が Sに属するすべての文字列に対して部分系列となっている時、t を Sの共通部分系列という。Sの共通部分系列のうち、長さが最大のものを Sの最長共通部分系列という。また、文字列 t が Sの極大共通部分系列であるとは、t が Sの共通部分系列であり、かつ、Sのどの共通部分系列も t を真の部分系列として含まないことをいう。例えば、文字列集合 $S = \{abac, aabc, abca\}$ に対して $a, b, c, aa, ab, ac, bc, abc$ は共通部分系列であり、aa, abc は極大共通部分系列、abc は最長共通部分系列である。

マルチプルアライメントとは、与えられた複数の有限長文字列を類似した文字パタンがなるべく揃うように配置する操作をいう。このマルチプルアライメントは、(1) 生物の進化の推定 (2) 蛋白質や遺伝子における構造と機能との関連調査などの生物学の研究において利用される重要な技術である。マルチプルアライメントを行なう際に、共通部分系列を求めることで類似した文字パタンを揃えやすくなる。しかし、任意の文字列集合 Sの最長共通部分系列を求めることはNP 困難である [1]。本稿では、先に示した Sの極大共通部分系列を1つ求める $O(l\cdot\|S\|)$ 時間の手続き [2] を $O(k\cdot\|S\|)$ に改良した手法を示す。ここで、l は Sに属する最短の系列の長さで、kはΣに属する文字の数、$\|S\|$ は Sの記述長である。

2 極大共通部分系列を求める手続き

以下の2つの手続きにおいて、入力は文字列の有限集合 $S = \{s_1, s_2, \ldots, s_n\}$ であり、出力は極大共通部分系列の1つである。

文献 [2] で示された、極大共通部分系列 p を求める $O(l\cdot\|S\|)$ 時間の手続きは、以下の通りである。

[手続き 1]

1. $p \leftarrow null$, $q \leftarrow null$
2. for *each* $c \in \Sigma$ do
   (a) $w \leftarrow 1$
   (b) $w-1$ 番目または w番目の文字が *non-null* の間、又は $p = null$の時、以下の操作列を実行する。
      i. if $p = null$ then $q \leftarrow c$
         else $q \leftarrow p$ の左から w番目の文字の左 (w番目の文字が $null$ならば、$w-1$ 番目の文字の右) に c を1つ挿入した文字列
      ii. qが Sの共通部分系列ならば、$p \leftarrow q$
      iii. $w \leftarrow w+1$ □

各文字列 s_iに対して、$k\cdot|s_i|$ のサイズの配列を準備し、s_iの先頭から任意の位置までに出現する各文字の個数を記憶させることにより、$O(k\cdot\|S\|)$ 時間の手続き2が得られる。ここで $|s_i|$ は s_iの長さを表す。

[手続き 2]

1. for $i = 1$ to n do
   $T_i[1..|s_i|] \leftarrow s_i$, $T_i[0] \leftarrow null$, $T_i[|s_i|+1] \leftarrow null$,
   $L_i \leftarrow 0$, $R_i \leftarrow |s_i|+1$, 前述のデータ構造を初期化する。
2. Σに属する各文字 c に対して、次の操作列を実行する。
   (a) while $R_i \neq |s_i|+1$ do
      · 副手続き search(c) を実行する。
   (b) 副手続き search(c) を実行する。
   (c) 副手続き remake_link を実行する。
3. T_1について、先頭から順にリンク†の張られている位置の文字を出力する。

副手続き: search(c)

以下のステップ2〜6は、$i = 1, \ldots, n$ に対する操作とする。

1. $M \leftarrow \min_{1\le i\le n}\{\, T_i[L_i+1\,..\,R_i-1]$ の中の c の個数 $\}$
2. $T_i[L_i+1\,..\,R_i-1]$ に対して、文字 c について左詰めになるように M個のリンクを張る。
3. 2で張ったリンクのうち最も右にあるリンクの位置 (2でリンクが1つも張られなかった時は L_iの値) を $nearR_i$とする。
4. R_iの位置にリンクがあれば、そのリンクを消去する。
5. if $R_i = |s_i|+1$ then $L_i \leftarrow nearR_i$
   else $T_i[nearR_i+1\,..\,R_i]$ に対して、$T_i[R_i]$ の文字について $nearR_i$に最も近い位置に1つリンクを張り、そのリンクの位置を新たに L_iとする。
6. if $R_i \neq |s_i|+1$ then
      if R_iの右側にリンクがない then $R_i \leftarrow |s_i|+1$
      else $R_i \leftarrow R_i$の右側で最も左にあるリンクの位置

副手続き: remake_link

$L_i \neq 0$ の間、以下の操作列を $i = 1, \ldots, n$ に対して実行する。

(a) L_iの位置にあるリンクを消去する。
(b) $T_i[L_i\,..\,R_i-1]$ に対して、$T_i[L_i]$ の文字について、R_iに最も近い位置に1つリンクを張り、そのリンクの位置を新たに R_iとする。
(c) if L_iの左側にリンクがない then $L_i \leftarrow 0$
   else $L_i \leftarrow L_i$の左側で最も右にあるリンクの位置 □

手続き2により極大共通部分系列が求まることは、手続き1のステップ2と手続き2のステップ2が等価であることを示すことにより証明できる。また、手続き2が $O(k\cdot\|S\|)$ であることの証明は、スペースの都合により省略する。これらの詳細は、文献 [3] を参照のこと。

一般に、同じ入力 Sに対しても、手続き2のステップ2の文字の選択順序により、得られる極大共通部分系列は異なる。

A Linear Time Algorithm for Computing a Maximal Common Subsequence Among Strings
Kazutoshi Kida, Michio Nakanishi, Akihiro Hashimoto
Osaka University, Toyonaka, Osaka 560, Japan

† n 本の文字列間で、文字 c どうしを $n-1$ 本の線で結んだもの全体を、文字 c の1つのリンクという。リンクの具体例は図2を参照。

FAST ALIGN の実行例　　(実行時間 0.1 秒)

```
S1 HYAMKILDKQKVVKLKQIEHTLNEKGGEMFSHL-PH-DLIYRDLKPENLLID-QQGYIQVTDFGFAKRVK-CAVDWWALGVLIYEMAAGYPPFFADQPIQIYEKIVSG-
S2 HYAMKILNKQKVVKMKQVEHILNEKGE-MFSFSEPH-DLIHRDLKPENLLID-QQGYLQVTDFGFAKRVK-CAVDWWALGVLIYEMAVGFPPFYADQPIQIYEKIVSGR
S3 DYAMKILDKQKVVKLKQVEHTLNEKR--MFSHL-PHSDLIYRDLKPENLLIDSQ-GYLKVTDFGFAKRVKGC-VDWWALGVLVYEMAAGYPPFFADQPIQIYEKIVSG-
    ****** ****** ** ** ****   ***    ** *** ********** * **  ********** * ********* **** * *** **************

S1 --K------VS-----------------HFSSDLKDLLRNLLQVDLTKRFGNLKNGVNDIKNHKWF
S2 VSKLRSLLQVDLTKRFGNLRNGVGDIKNH-----K---------------------------WF
S3 --K------VS-----------------HFGSDLKDLLRNLLQVDLTKRYGNLKAGVNDIKNQKWF
     *      *                   *     *                          **
```

CLUSTAL V の実行例　　(実行時間 1.1 秒)

```
S1 HYAMKILDKQKVVKLKQIEHTLNEKGGEMFSHLPH-DLIYRDLKPENLLIDQQGYIQVTDFGFAKRVKCAVDWWALGVLIYEMAAGYPPFFADQPIQIYEKIVSGKVSH
S2 HYAMKILNKQKVVKMKQVEHILNEKG-EMFSFSEPHDLIHRDLKPENLLIDQQGYLQVTDFGFAKRVKCAVDWWALGVLIYEMAVGFPPFYADQPIQIYEKIVSGRVSK
S3 DYAMKILDKQKVVKLKQVEHTLNEK--RMFSHLPHSDLIYRDLKPENLLIDSQGYLKVTDFGFAKRVKGCVDWWALGVLVYEMAAGYPPFFADQPIQIYEKIVSGKVSH
   .******.******.**.** ****   ***     *** ********** ***..**********  *********.**** *.***.**************.**

S1 FSSDLKDLLRNLLQVDLTKRFGNLKNGVNDIKNHKWF
S2 --------LRSLLQVDLTKRFGNLRNGVGDIKNHKWF
S3 FGSDLKDLLRNLLQVDLTKRYGNLKAGVNDIKNQKWF
          **.*********.***. ** ****.***
```

図 8: FAST ALIGN と CLUSTAL V の実行例

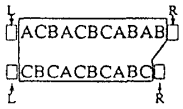

図 1: ステップ 1 での初期化後

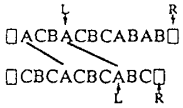

図 2: 文字 A に対するステップ 2-(b) の終了時

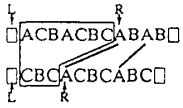

図 3: 文字 A に対するステップ 2-(c) の終了時

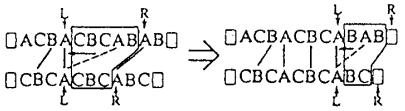

図 4: 文字 B に対するステップ 2-(a) における
副手続き search の繰り返し

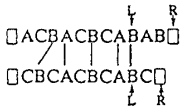

図 5: 文字 B に対するステップ 2-(b) の終了時

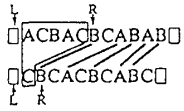

図 6: 文字 B に対するステップ 2-(c) の終了時

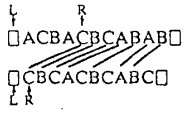

図 7: 文字 C に対するステップ 2-(c) の終了時

[例 1] S = {ACBACBCABAB, CBCACBCABC} に対する手続き 2 の実行例を図 1〜7 に示す。ここで、太い実線は新たに張られたリンクを表し、細い実線はすでに張られていたリンクを表す。また、点線は矢印の先にリンクが張り直されたことを表し、囲みは次に見る範囲を表す。図 7 より、極大共通部分系列として CBCABAB が得られる。 □

3　評価

3 本の文字列に対して、手続き 2 を用いたアライメントプログラム FAST ALIGN と、既存のアライメントツール CLUSTAL V[4] を、SUN SPARC 2 上で実行した結果を図 8 に示す。ここで、'−' はアライメントにより挿入されたギャップを表し、'*' はすべての文字列でその位置にある文字が揃っていることを表す。文字列集合のサイズが大きい場合に対しても、表 1に示すように、FAST ALIGN の実行速度は高速である。

しかし、図 8 を見てもわかるように、FAST ALIGN では、文字列の後ろに近い部分のアライメントはうまくできていない。これを改良することが、今後の課題である。

表 1: CLUSTAL V と FAST ALIGN の実行時間 ($k = 20$)

文字列数	1 本の平均長	CLUSTAL V	FAST ALIGN
12 本	270	14.1 秒	0.6 秒
20 本	280	31.1 秒	0.9 秒
50 本	280	108.7 秒	2.1 秒

参考文献

[1] D. Maier: "The complexity of some problems on subsequences and supersequences", JACM, vol. 25, No. 2, pp.322-366, 1978.

[2] 清水 邦保、伊藤 実、橋本 昭洋: "系列集合に対する極大共通部分系列を求める多項式時間アルゴリズム", 信学技法 COMP92-32, pp.29-36, 1992.

[3] 木田 和寿: "文字列間の極大共通部分系列を求めるアルゴリズムの高速化とそのデータ構造", 大阪大学卒業論文, 1993.

[4] D. Higgins, A. Bleasby, and R. Fuchs: "CLUSTAL V: improved software for multiple sequence alignment", CABIOS, vol. 8, No. 2, pp.189-191, 1992.