LA

k ミスマッチデカルト木問題に関する研究

東北大学 大学院情報科学研究科
システム情報科学専攻 篠原・吉仲研究室
修士課程 1 年
小瀧 由浩

2025 年 1 月 27 日

目次

第 1 章

はじめに

1.1 背景

近年，時系列データと呼ばれるデータへの解析が盛んになっている．時系列データとは，時間的な順序を伴いながら観測されるデータのことである．分かりやすい例だと気温や株価などのデータであるが，近年では自身の体重や 1 日で歩いた距離などの身近な記録さえも時系列データとして収集できる時代になり，時系列データの解析に注目が集まっている．これら時系列データに対して解析をする上で基本的な技術の一つとして厳密文字列照合がある．

厳密文字列照合 [1] とは，アルファベット上のテキスト T と呼ばれる文字列の中からパターン P と呼ばれる探したい文字列と完全に一致している文字列の位置を全て探す照合である．例えば，時系列データでの厳密文字列照合では，$T = (20, 10, 50, 40, 60)$, $P = (20, 10, 50)$ とすると，テキストの先頭にパターンと一致する文字列を見つけることができる．しかし，時系列データ分析では，パターン P と完全に一致している部分文字列 $T[1:3]$ を探すだけではなく，P と “似ている” 部分文字列 $T[3:5] = (50, 40, 60)$ を探すことも意味がある．そのため，パターンと “似ている” 文字列をテキストから探す照合が様々あり，それらの一つとして順序保存照合やデカルト木照合がある．

順序保存照合 [2] とは，全順序アルファベット上のテキスト文字列 T の中からパターン文字列 P の要素の順位と一致する部分文字列の位置を全て探す照合で，例えば $T = (33, 25, 36, 18, 45, 30, 49, 26)$, $P = (17, 10, 19, 6, 24, 15, 27)$ が与えられた時，P の各要素の順位の列は $(4, 2, 5, 1, 6, 3, 7)$ であり，これと同じ順位の列である $T[1:7] = (33, 25, 36, 18, 45, 30, 49)$ を見つける．一方，デカルト木照合 [3] とは，テキスト文字列 T

の中からパターン文字列 P のデカルト木と同じデカルト木となる文字列を全て探す照合である．デカルト木とは，文字列を表す木構造であり，文字列 $S_1 = (17, 10, 19, 6, 24, 15, 27)$ のデカルト木は以下の図**??**のようになる．例えば，文字列 $S_1 = (17, 10, 19, 6, 24, 15, 27)$ と文字列 $S_2 = (29, 9, 21, 4, 23, 14, 27)$ がデカルト木照合でマッチする文字列同士か判断する場合，以下の図**??**より文字列 S_1 と S_2 はともに同じデカルト木となる文字列なので，デカルト木照合でマッチする．しかし，順序保存照合の場合，文字列 S_1 と S_2 の各要素の順位の列は一致しないので，順序保存照合ではマッチしない．このように，デカルト木照合では，順位の列が完全に一致していない文字列でも，文字列のデカルト木の形が同じでさえあれば検出できる照合なので，順序保存照合よりも寛容な照合であり，より直感的に形が似ている文字列を探すことができる照合である．

本論文で紹介する，k ミスマッチデカルト木問題

本論文では，k ミスマッチデカルト木問題を定義し，$O(m^2k^2)$ 時間で解くアルゴリズムを提案した．

1.2 構成

本論文の構成は次のとおりである．まず，第 2 章において表記法，k ミスマッチデカルト木照合の定義を述べる．第 3 章において提案手法を述べる．第 4 章においてまとめと今後の課題を述べる．

第 2 章

準備

2.1 表記

整数アルファベットを Σ とし，Σ 上の全ての系列からなる集合を Σ^* と表し，Σ^* 上の要素を文字列と呼ぶ．文字列 S の長さを $|S|$ と表す．文字列 S の i 番目 $(1 \leq i \leq |S|)$ の文字を $S[i]$ と表し，文字列 S の i 番目から j 番目までの部分文字列を $S[i:j]$ と表す．特に，$1 \leq i \leq j \leq |S|$ とし，$i > j$ の時は空文字列とする．集合 $L_{[1:m]}$ を，任意の整数 t $(0 \leq t \leq m)$ において，$1 \leq i_1 < \cdots < i_t \leq m$ を満たす添え字列 $I = (i_1, \cdots, i_t) \in \{1, \cdots, m\}^t$ からなる集合とする．添え字列 $I \in L_{[1:m]}$ において，I に対応する文字列 S の部分列を $S_I = (S[i_1], \cdots, S[i_t])$ と表す．デカルト木 $CT(S)$ の根ノードの位置を $CTR(S)$ と表す．以下の図 2.1 の文字列 S の場合，$CTR(S) = 2$ となる．$TopK(S,k)$ を文字列 S 内の要素を値の昇順に並べた時の 1 番目から k 番目までの要素に対応する位置からなる集合とする．以下の図 2.1 の文字列 S の場合，$TopK(S,3) = \{2,4,6\}$ となる．

2.2 デカルト木照合問題

定義 1 (デカルト木 [3])**.** 文字列 $S[1:m]$ に対応したデカルト木 $CT(S)$ とは以下の規則に従って再帰的に定義される二分木である．

$$CT(S) = \begin{cases} i \text{ に対応する節点を根}, CT(S[1:i-1]) \text{ を左部分木}, \\ \qquad CT(S[i+1:m]) \text{ を右部分木とした木} & (S \text{ は空文字列ではない}) \\ \text{空の木} & (S \text{ は空文字列}) \end{cases}$$

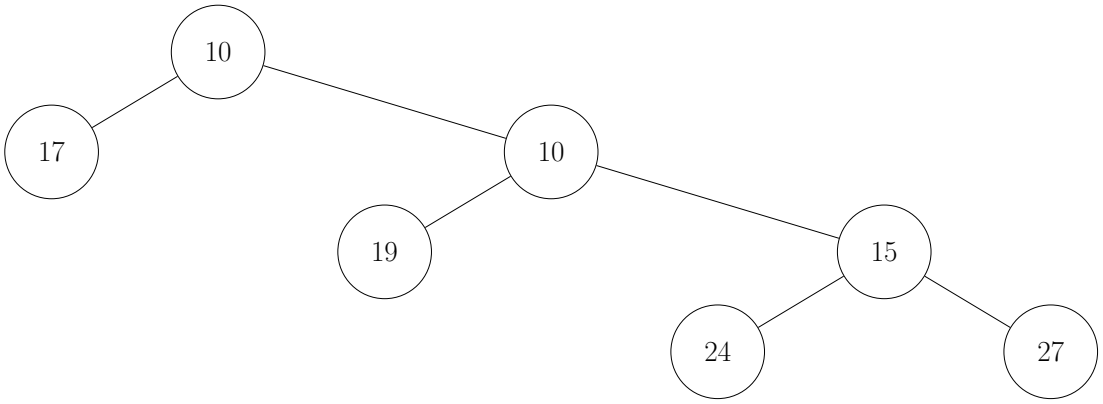

図 2.1: 文字列 $S = (17, 10, 19, 10, 24, 15, 27)$ のデカルト木

特に，i は S 中の最小値の最左のインデックスである．

例 1. 文字列 $S = (17, 10, 19, 10, 24, 15, 27)$ のデカルト木は以下の図 2.1 のようになる．

定義 2 (デカルト木同型)**.** 同じ長さの文字列 S_1 と S_2 において，

$$CT(S_1) = CT(S_2)$$

ならば，S_1 と S_2 はデカルト木同型であると言い，$S_1 \approx S_2$ と表記する．

定義 3 (デカルト木照合問題 [3])**.** デカルト木照合問題とは，長さ n のテキスト T と，長さ m のパターン P の 2 つの文字列が与えられた時に，

$$P[1 : m] \approx T[i : i + m - 1]$$

を満たす全ての i $(1 \leq i \leq n - m + 1)$ を出力する問題.

2.3 デカルト木ミスマッチ数の定義と問題定義

同じ長さの文字列 S と P において，S と P のデカルト木が同型となるための最小のミスマッチ数をデカルト木ミスマッチ数と呼び，以下のように定義する．

定義 4 (デカルト木ミスマッチ数)**.** 長さ m の文字列 P と S において，P と S のデカル

ト木ミスマッチ数は $CTMiss(S,P)$ と書き，以下のように定義される．

$$CTMiss(S,P) = \min_{I \in L_{[1:m]}} \{m - |I| \mid CT(S_I) = CT(P_I)\}$$

例 2. 文字列 $P = (17, 25, 21, 11, 27, 15)$ と $S = (21, 60, 30, 40, 50, 10)$ において，$CTMiss(S,P) = 1$ である．

問題 1 (k ミスマッチデカルト木決定問題)**.** 同じ長さ m の文字列 P と S，許容ミスマッチ数 k $(0 \leq k \leq m)$ が与えられた時に，$CTMiss(S,P) \leq k$ かどうかを判定する問題.

第 3 章

提案手法

k ミスマッチデカルト木決定問題を動的計画法を用いて，$O(m^2k^2)$ 時間で解けることを示す．

3.1 素朴なアルゴリズム

あとで書く

3.2 動的計画法

$DP(i,j,v)$ を以下のように定義する．整数 i,j $(1 \leq i \leq j \leq m)$，整数 v $(i \leq v \leq j)$ に対して，

$$DP(i,j,v) = \min(\{j-i+1-|I| \mid I \in L_{[1:m]} \ \wedge \ CT(S_I) = CT(P_I) \ \wedge \\ v \in I \ \wedge \ CTR(P_I) = CTR(S_I) = v\} \cup \{j-i+1\})$$

とする．
次の補題を用いて，$DP(i,j,v)$ を計算する．

補題 1. $DP(i,j,v)$ は以下の漸化式を満たす.

$$DP(i,j,v) = leftmiss + rightmiss$$

$$leftmiss = \begin{cases} \min\{DP(i,v-1,l) \mid l \in A\} & (\text{集合 } A = \\ & \{l \in \{i,\cdots,v-1\} \mid P[l] > P[v] \wedge S[l] > S[v]\}) \\ v-i & (\text{A が空集合の場合}) \end{cases}$$

$$rightmiss = \begin{cases} \min\{DP(v+1,j,r) \mid r \in B\} & (\text{集合 } B = \\ & \{l \in \{v+1,\cdots,j\} \mid P[l] \geq P[v] \wedge S[l] \geq S[v]\}) \\ j-v & (\text{B が空集合の場合}) \end{cases}$$

証明. あとで書く □

以下の補題 2 は，k ミスマッチデカルト木決定問題を解くためには，$DP(1,m,v)$ において，全ての v $(1 \leq v \leq m)$ を調べる必要はなく，$v \in TopK(S[1:m], k+1)$ だけを調べればよいことを示す.

補題 2.

$$CTMiss(S,P) \leq k \Leftrightarrow \min\{DP(1,m,v) \mid v \in TopK(S[1:m], k+1)\} \leq k$$

証明. あとで書く □

3.3 提案手法の時間計算量

補題 2 より，集合 A と B の要素 l と r はそれぞれ集合 $TopK(S[i:v-1], k+1)$ と集合 $TopK(S[v+1:j], k+1)$ 内の要素からなる．任意の i,j $(1 \leq i \leq j \leq m)$ における $TopK(S[i:j], k+1)$ に対しては，$O(m^2k)$ 時間で前処理計算する．これにより，求める必要がある $DP(i,j,v)$ は $1 \leq i \leq j \leq m$, $v \in TopK(S[i:j], k+1)$ と $l \in TopK(S[i:j], k+1)$ の範囲をループすれば求めることができるので，全体の時間計算量は $O(m^2k^2)$ 時間となる．提案手法の疑似コードを以下の Algorithm1 に示す.

Algorithm 1: 提案手法のアルゴリズム

Input: 文字列 $S[1:m]$，文字列 $P[1:m]$，許容ミスマッチ数 k

Output: $CTMiss(S,P) \leq k$ かどうか True/False

```
|S| × |S| × k 次元の配列を |S| − 1 で初期化;
CTMiss ← |S| − 1;
任意の i, j (1 ≤ i ≤ j ≤ m) において，集合 TopK(S[i : j], k + 1) を前処理で計算する;
for i ← 1 to m do
    for j ← i + 1 to m do
        for v ∈ TopK(S[i : j], k + 1) do
            leftmiss ← v − i;
            if v ≥ 2 then
                for l ∈ TopK(S[i : v − 1], k + 1) do
                    if P[l] > P[v] ∧ S[l] > S[v] then
                        leftmiss ← min{DP(i, v − 1, l), leftmiss};
                    end
                end
            end
            rightmiss ← j − v;
            if v ≤ m − 1 then
                for r ∈ TopK(S[v + 1 : j], k + 1) do
                    if P[r] ≥ P[v] ∧ S[r] ≥ S[v] then
                        rightmiss ← min{DP(v + 1, j, r), rightmiss};
                    end
                end
            end
            DP(i, j, v) ← min{DP(i, j, v), leftmiss + rightmiss};
            CTMiss ← min{CTMiss, DP(1, m, v)};
        end
    end
end
return CTMiss ≤ k
```

第 4 章

まとめ

適当に書く

参考文献

[1] Donald Ervin Knuth, James H. Morris, and Vaughan R. Pratt. Fast pattern matching in strings. *SIAM J. Comput.*, Vol. 6, pp. 323–350, 1977.

[2] Jinil Kim, Peter Eades, Rudolf Fleischer, Seok-Hee Hong, Costas S. Iliopoulos, Kunsoo Park, Simon J. Puglisi, and Takeshi Tokuyama. Order preserving matching. *CoRR*, Vol. abs/1302.4064, , 2013.

[3] Sung Gwan Park, Amihood Amir, Gad M. Landau, and Kunsoo Park. Cartesian tree matching and indexing. *CoRR*, Vol. abs/1905.08974, , 2019.

謝辞

適当に書く