講義「アルゴリズムとデータ構造」

第x回　その他の精選トピックス

大学院情報科学研究科 情報理工学専攻
情報知識ネットワーク研究室
喜田拓也

今日の内容

喜田’sセレクション：

文字列索引データ構造
- 接尾辞木
- 接尾辞配列
- Burrows-Wheeler変換（BWT）

※ 確率と情報の科学 高速文字列解析の世界 ーデータ圧縮・全文検索・テキストマイニングー, 岡之原大輔, 岩波書店, 2012年.

幾何的アルゴリズム
- 点・線・線分
- 符号付面積
- 線分の交差判定
- 凸多角形の判定

※ アルゴリズム・サイエンス シリーズ10 計算幾何 ー理論の基礎から実装までー, 浅野哲夫 著, 共立出版, 2007年.

文字列索引データ構造

パターン照合を行う際，テキスト側のデータをあらかじめ前処理することができるならば，「索引」を作っておけばいいじゃないか！

転置ファイル：　キーワードの逆引き辞書をリストで持つ

接尾辞木：　テキストのすべての接尾辞を木で持つ

接尾辞配列：　接尾辞の位置をソートした配列で持つ

FM-index：　BWT文字列をウェーブレット木で持つ

ブロックソート圧縮法（bzip2など）に用いられている

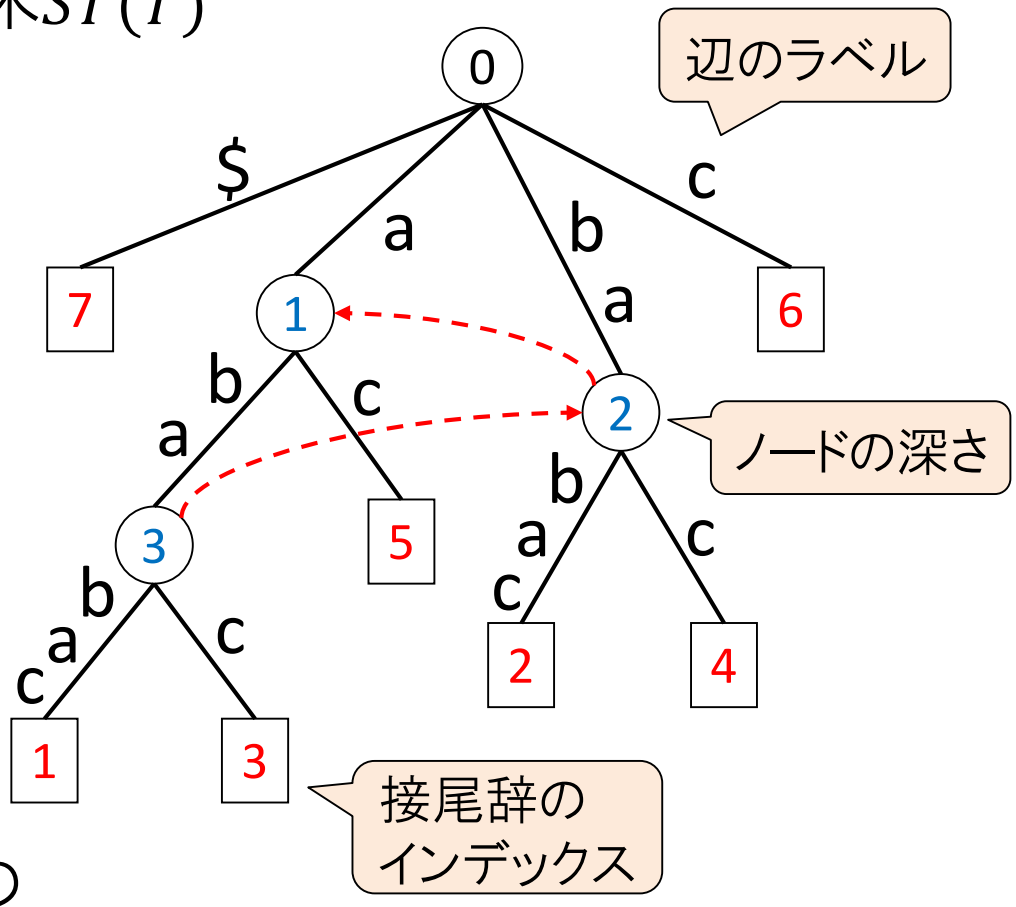
接尾辞木（Suffix Tree）

テキストのすべての接尾辞を木で持つ索引データ構造
索引＝元の文字列＋接尾辞木$ST(T)$
$O(n \log n)$ビットの領域を使用
検索はパターン長mに対して
$O(m \log |\Sigma|)$時間

1 2 3 4 5 6 7

T ＝ a b a b a c $

Suffix link



辺のラベルは，T上の文字列の
（開始位置，終了位置）として持つ

接尾辞配列（Suffix Array）

テキストのすべての接尾辞を辞書順でソートし，その順番に接尾辞の開始位置（インデックス）を配列として保持したもの
索引＝元の文字列＋接尾辞配列$SA_T[i]$
$n \log n + n \log |\Sigma|$ビットの領域を使用
検索はパターン長mに対して$O(m \log n)$時間

1	2	3	4	5	6	7
a	b	a	b	a	c	$

$T =$ a b a b a c $

単純な構築法だと$O(n \log n)$時間かかるが，$O(n)$時間でできるうまい方法が知られている

i	$SA[i]$	
1	7	$
2	1	a b a b a c $
3	3	a b a c $
4	5	a c $
5	2	b a b a c $
6	4	b a c $
7	6	c $

Burrows-Wheeler変換（BWT; BW変換）

テキストTのソートした各接尾辞の
1つ前の文字を並べたもの

$$BW_T[i] = T[SA[i] - 1]$$

テキストTの並び替えになっている
BW_Tから元のTを復元できる
BW_Tは圧縮しやすい
（同じ文字が連続しやすい）

1 2 3 4 5 6 7 8 9

$$T = \mathtt{acagcaggg\$}$$

$$BW_T = \mathtt{g\$ccaggaa}$$

$SA[i]$	$BW_T[i]$	
9	g	$
1	$	a c a g c a g g
3	c	a g c a g g $
6	c	a g g $
2	a	c a g c a g g $
5	g	c a g g $
8	g	g $
4	a	g c a g g $
7	a	g g $

先頭の1つ前
は末尾とする

BW逆変換

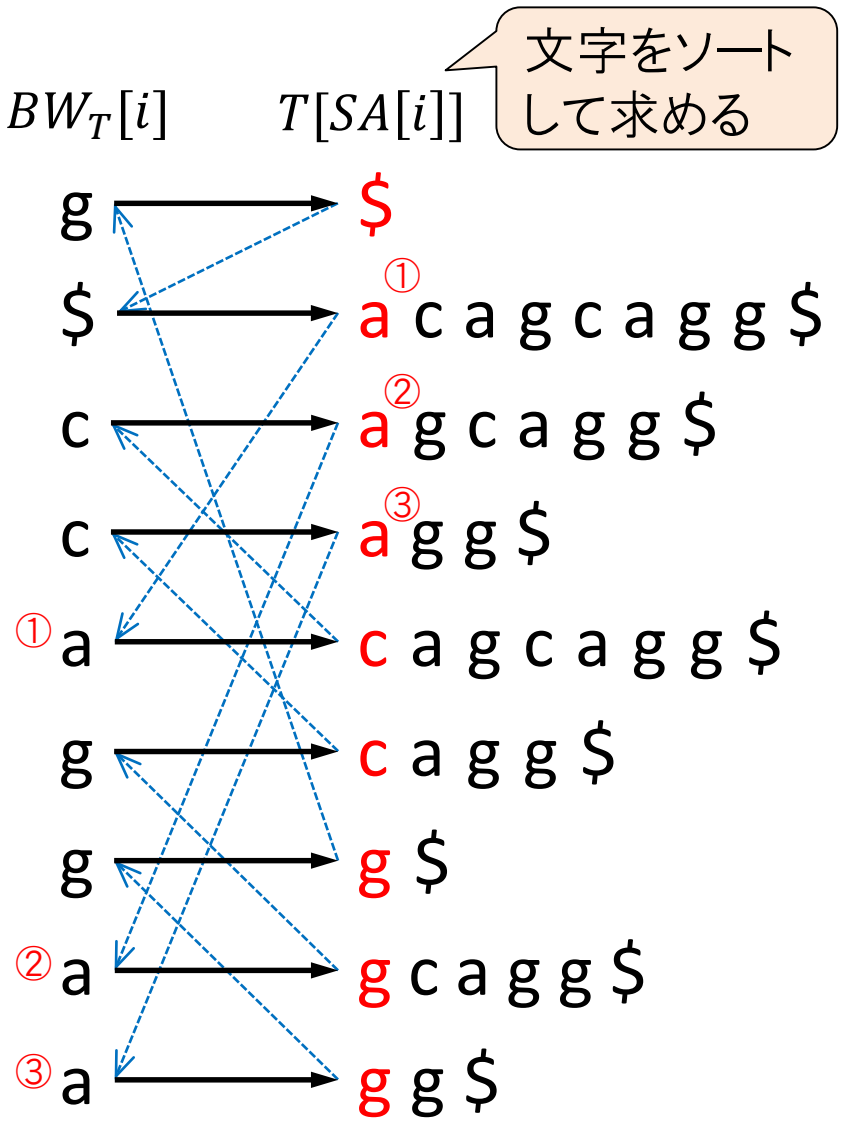
TはBW_Tから復元できる

1. BW_Tの文字の並びをソートして$T[SA[i]]$を得る
2. BW_T上の\$のインデックスkから始めて，対応する$T[SA[k]]$を出力
3. $T[SA[k]]$の文字のBW_T上でのインデックスを求め，次のkとする
4. \$を出力するまで2と3を繰り返す

$$BW_T = \text{g\$ccaggaa}$$

$$T = \text{acagcagg\$}$$

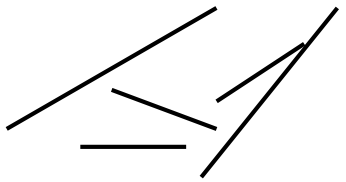
同じようにして$SA[i]$も復元できる！



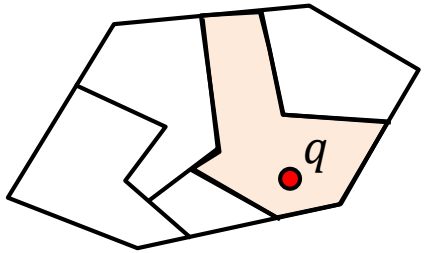
幾何的アルゴリズム

計算機で解きたい幾何問題はいっぱいある！

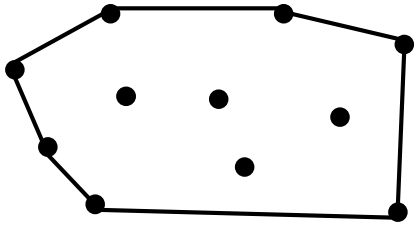
結構，難しい



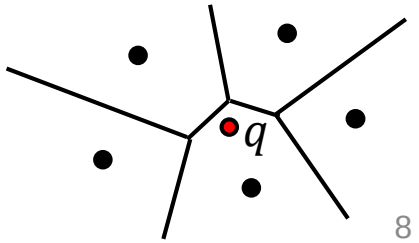
線分の交差判定問題：「平面上にn本の線分が与えられたとき，これらの線分のあいだに交差があるかどうかを判定せよ」



点位置決定問題：「平面上に直線で描かれたサイズnの平面地図に対して，質問点qを含む領域の名前を求めよ」



凸包問題：「n個の点が与えられたとき，これらすべての点を含む最小の凸多角形（凸包，convex hull）を求めよ」



最近点問題：「あらかじめ与えられたn個の点に対し，任意に与えられた質問点にもっとも近い点を効率よく求めよ」

点・直線・線分

点pの表現：　平面の座標値$p(x, y)$

点$p_1(x_1, y_1)$と点$p_2(x_2, y_2)$の距離：　ユークリッド距離

$$\mathrm{dist}(p_1, p_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

直線の表現：　直線の方程式 $ax + by + c = 0$

あるいは，通る2点$p_1(x_1, y_1)$と$p_2(x_2, y_2)$を指定した式を使う.

$$y = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) + y_1$$

ただし，$x_1 \neq x_2$である必要がある．よって一般には，変形した式

$$(y_2 - y_1)x - (x_2 - x_1)y = x_1 y_2 - x_2 y_1$$

を使う．線分は2つの端点を指定する．線分を含む直線の式は上のとおり.

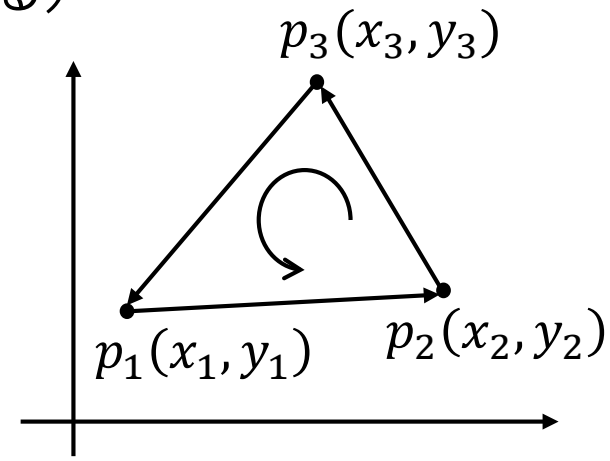
符号付面積

3頂点$p_1(x_1, y_1), p_2(x_2, y_2), p_3(x_3, y_3)$からなる三角形の面積

$$面積S = \frac{1}{2}\left((x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)\right)$$

として求めることができる（通常は絶対値をとる）

実はこの式の符号には重要な意味がある！



$$\Delta(p_1, p_2, p_3) = (x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)$$

とすると，

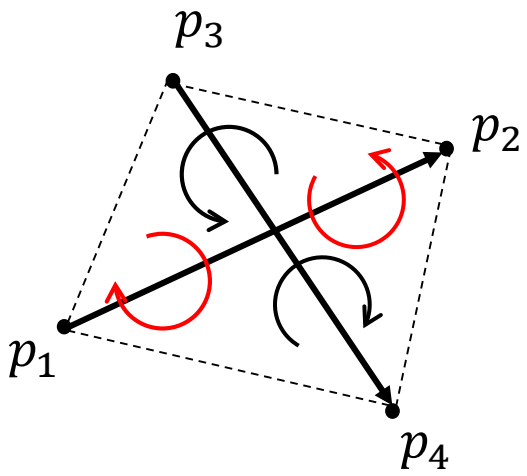
$$\Delta(p_1, p_2, p_3) \begin{cases} > 0 : p_1, p_2, p_3 が反時計回りのとき \\ = 0 : p_1, p_2, p_3 が一直線上にあるとき \\ < 0 : p_1, p_2, p_3 が時計回りのとき \end{cases}$$

ベクトルの外積とほぼ等価な概念

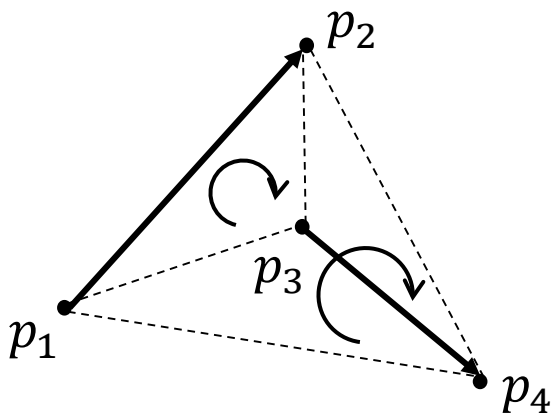
線分の交差判定

2つの線分が与えられたとき，それらが互いに交差するかどうかを判定したい

符号付面積をうまく使うと，簡単に判定できる！



$$\Delta(p_1,p_2,p_3)\times\Delta(p_1,p_2,p_4)<0$$
かつ
$$\Delta(p_3,p_4,p_1)\times\Delta(p_3,p_4,p_2)<0$$



$$\Delta(p_1,p_2,p_3)\times\Delta(p_1,p_2,p_4)>0$$

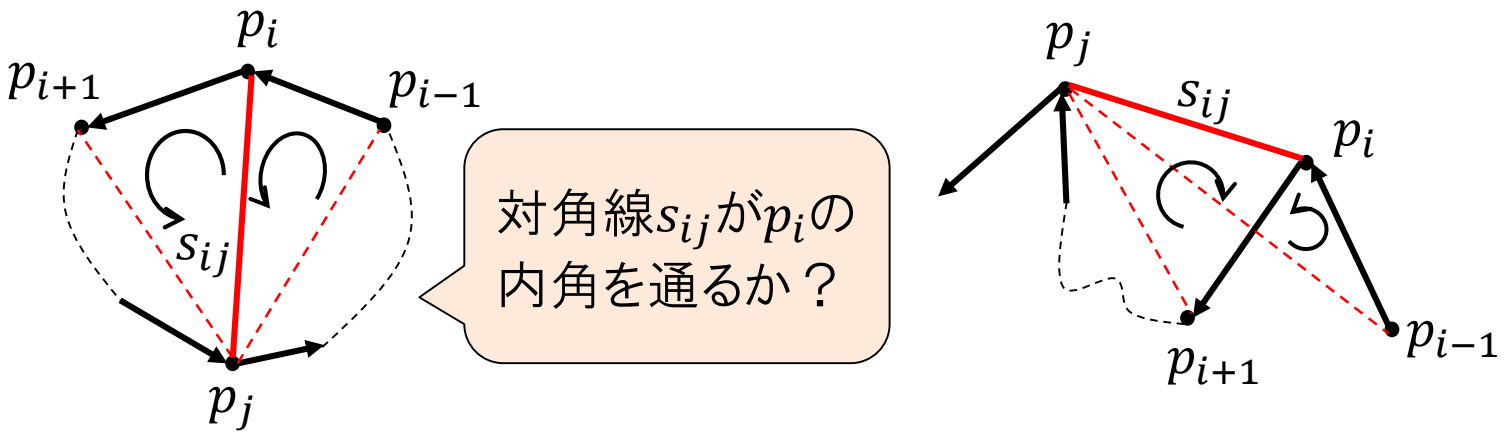
交差しない場合

凸多角形の判定

多角形の頂点列が反時計回りで与えられているとき，この多角形が凸多角形（すべての頂点が凸の多角形）かどうかを判定したい

定理

多角形Pが凸多角形であるための必要十分条件は，Pのすべての対角線がPの内部にあるときである．



対角線の個数は頂点数がnのとき，$n(n-1)/2$個ある $\Rightarrow O(n^2)$