アルゴリズムとデータ構造

第25週目

担当　情報システム部門　徳光政弘
2025年12月16日

今日の内容

- 文字列の照合アルゴリズム　ホールスプール法
- 文字列の照合アルゴリズム　ボイヤー・ムーア法

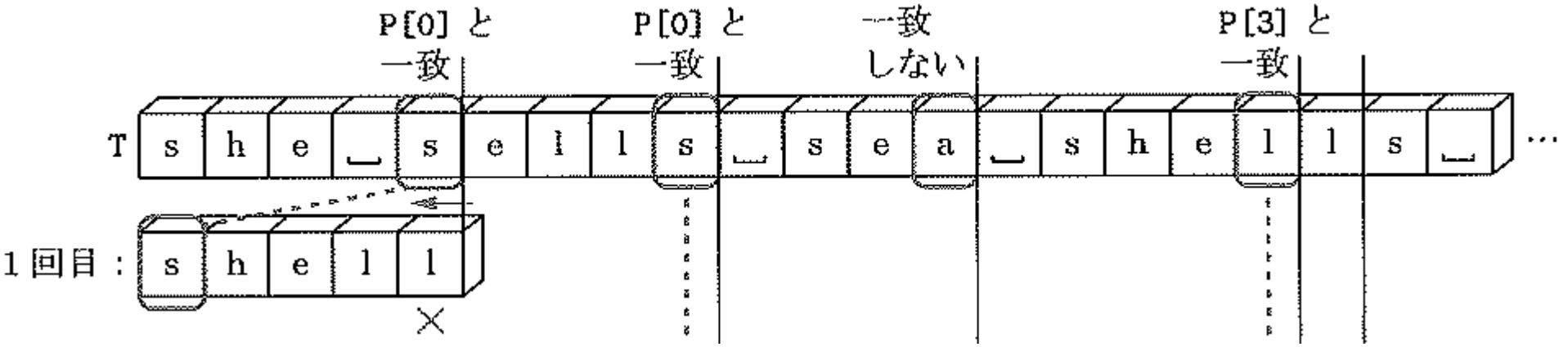
用語の定義

- テキスト　検索対象の文字列
- パターン　テキストから検索したい文字列
- 文字列の照合　テキストからパターンに一致する文字列を探し出し、照合する部分の先頭の添字を返す

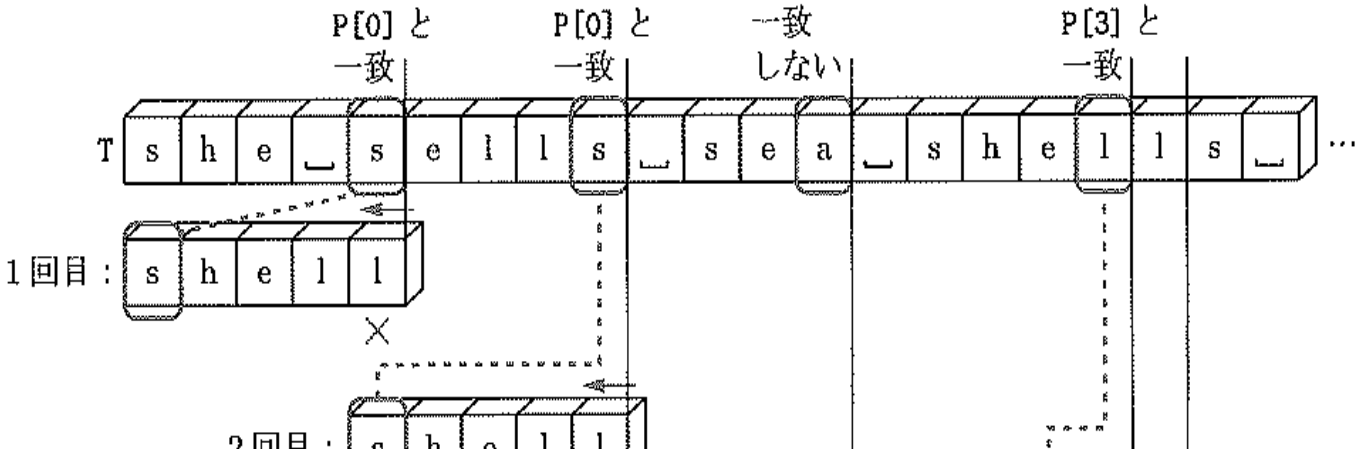
パターンはテキストのどこに含まれているか？

ホールスプール法

- 左から右に向かって照合し、不一致が起こった場合は、比較を起こったテキストの右端の文字の情報を利用する（パターン文字列を構成する文字の情報と比較する）



ホールスプール法



ホールスプール法

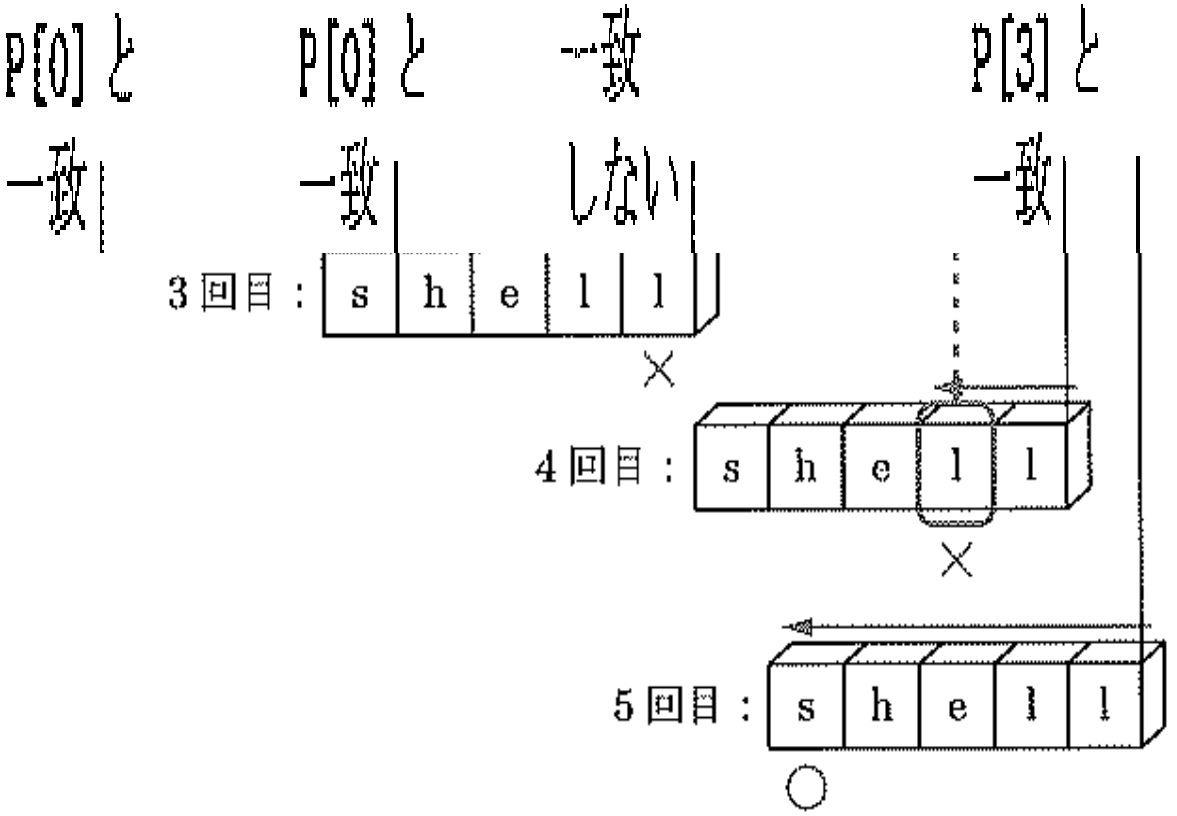


図 12.4　ホールスプールのアルゴリズムのアイデアによる文字列照合の実行例

文字列のずらし方

- 文字 α がパターン中に含まれない場合：文字 α はパターンと一致することはないので，ずらす量は m となり，S[α]=m とする[1)]．
- 文字 α がパターン中に含まれる場合：P[i]=α の場合，ずらす量は $m-1-i$ なので，S[α]=m-1-i とする．ただし，α がパターン中に複数回現れる場合は，もっとも右に現れる P[i]=α となる P[i] に対して S[α]=m-1-i とする．また，α がパターンの末尾 P[m-1] にしか現れない場合のずらす量は m であり，S[α]=m とする．

文字列のずらし方

例： 図 12.2 の場合，テキスト中の文字は，“a”，“b”，“e”，“h”，“l”，“o”，“r”，“s”，“t”，“y”，“.”，“␣ (空白)” の 12 種類である．これらの文字に対して，入力パターンが “shell” であるとき，P[0]=s, P[1]=h, P[2]=e, P[3]=l, P[4]=l なので，配列 S の値は以下のようになる．

$$\mathtt{S[a]} = 5, \quad \mathtt{S[b]} = 5, \quad \mathtt{S[e]} = 2, \quad \mathtt{S[h]} = 3, \quad \mathtt{S[l]} = 1, \quad \mathtt{S[o]} = 5,$$
$$\mathtt{S[r]} = 5, \quad \mathtt{S[s]} = 4, \quad \mathtt{S[t]} = 5, \quad \mathtt{S[y]} = 5, \quad \mathtt{S[.]} = 5, \quad \mathtt{S[␣]} = 5$$

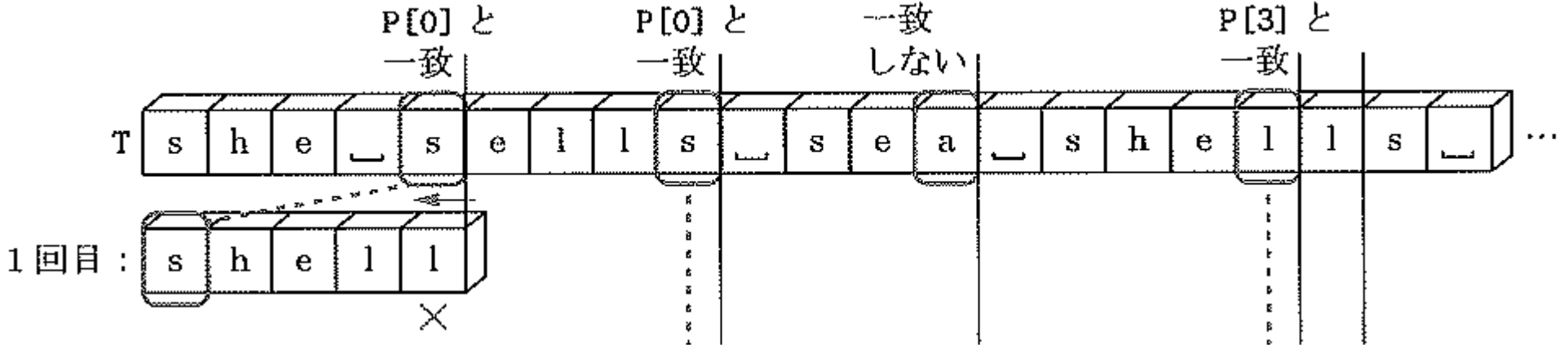
文字列のずらし方

例： 図 12.4 の 1 回目の照合では α=“s” であり，照合位置を S[s]=4 だけ右にずらす．また，2 回目以降の照合における比較を行ったテキストの右端の文字 α とずらす量は以下のようになる．

2 回目：α =“s” であり，照合位置を S[s]=4 だけ右にずらす．

3 回目：α =“a” であり，照合位置を S[a]=5 だけ右にずらす．

4 回目：α =“l” であり，照合位置を S[l]=1 だけ右にずらす．



ホールスプール法

アルゴリズム 12.2　文字列照合を行うホールスプールのアルゴリズム

```
入力：テキストを表す配列 T[0], T[1], ..., T[n-1], およびパターンを表す配列
      P[0], P[1], ..., P[m-1]
for (i=0, α=テキスト中の文字 ; i<テキスト中の文字の種類数; i=i+1)
     {S[α]=m; }
            // テキスト中のすべての文字について配列Sの値をmに初期化

for (i=0; i<m-1; i=i+1)  { S[P[i]]=m-1-i; }
                    //パターンの各文字P[i]についてS[P[i]]の値を計算
```

ずらす量の計算

ホールスプール法

```
while (i<n-m+1)  {
  j=m-1;                  //jはパターンに対して照合を行う位置を表す
  α=T[i+j];               //比較を行うテキストの右端の文字αを記録
  while ((T[i+j]==P[j])かつ(j>=0))  { j=j-1; }
  if (j==-1) { iを出力しアルゴリズム終了; }
  else { i=i+S[α]; }
}
"一致しない"と出力;
```

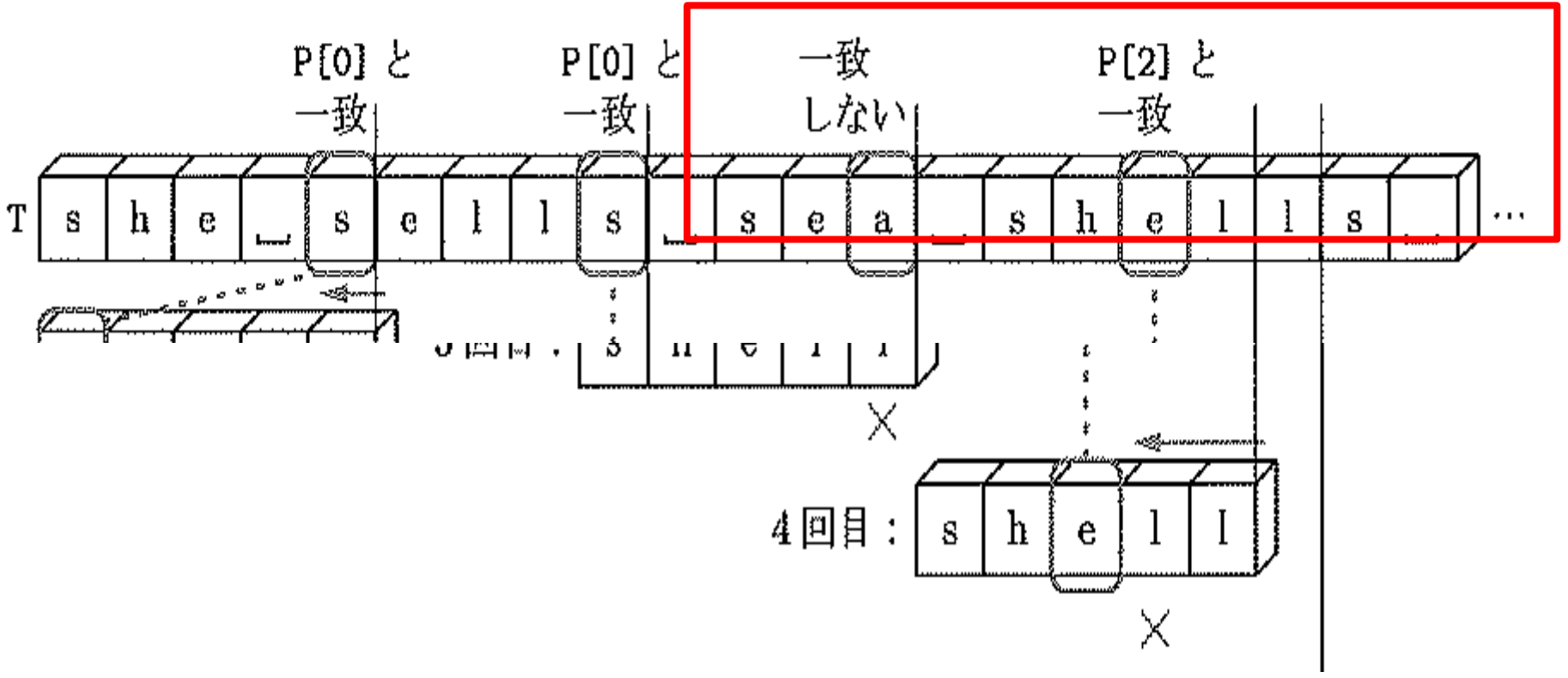
テキストの照合

ボイヤー・ムーア法

- 実用性が高い文字列照合アルゴリズム
- 2個のアイデア(失敗に対するパターン)で構成されている

ボイヤー・ムーア法(アイデア1)

- 不一致が起こったテキスト中の文字の利用



ボイヤー・ムーア法(アイデア1)

① 入力テキスト中で用いられている各文字について，その文字がパターン中の何文字目にでてくるかを計算する．ここでは，文字 α がパターンの P[k] と等しい場合，配列 L を用いて $L[\alpha]=k$ とする．なお，文字 α がパターン中にでてこない場合は $L[\alpha]=-1$ とし，また，文字 α がパターン中に複数回でてくる場合は，パターン中でもっとも右に現れる $P[k]=\alpha$ となる P[k] に対して $L[\alpha]=k$ とする．

ボイヤー・ムーア法(アイデア1)

例：図 12.5 の場合，テキスト中の文字は，“a”，“b”，“e”，“h”，“l”，“o”，“r”，“s”，“t”，“y”，“.”，“␣(空白)”の 12 種類である．これらの文字に対して，入力パターンが “shell” であるとき，P[0]=s，P[1]=h，P[2]=e，P[3]=l，P[4]=l なので，配列 L の値は以下のようになる．

$$L[a]=-1,\quad L[b]=-1,\quad L[e]=2,\quad L[h]=1,$$
$$L[l]=4,\quad L[o]=-1,\quad L[r]=-1,\quad L[s]=0,$$
$$L[t]=-1,\quad L[y]=-1,\quad L[.]=-1,\quad L[␣]=-1$$

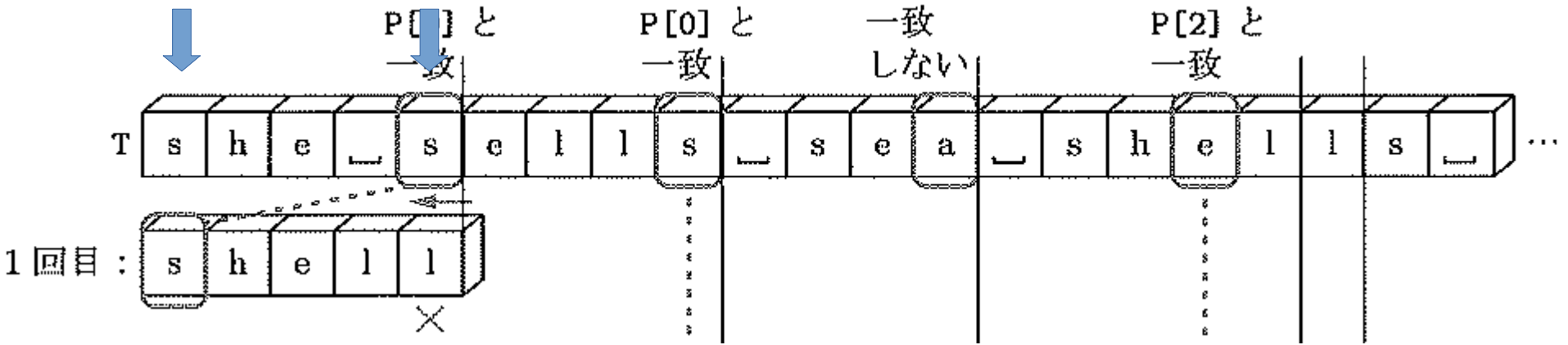
入力パターン shell

s	h	e	l	l
0	1	2	3	4

ボイヤー・ムーア法(アイデア1)

② パターンの右から左に向かって照合を行う．パターン全体とテキストが一致すればアルゴリズムを終了する．一致しない場合は，パターンの何文字目で不一致が起こったかと照合位置のテキストの文字 α を記録する．ここでは，テキストの T[i+j] $=\alpha$ とパターンの P[j] で不一致が起こったと仮定する．

例： 図 12.5 の 1 回目の照合では，$i=0$，$j=4$，$\alpha=$“ s ”である．



ボイヤー・ムーア法(アイデア1)

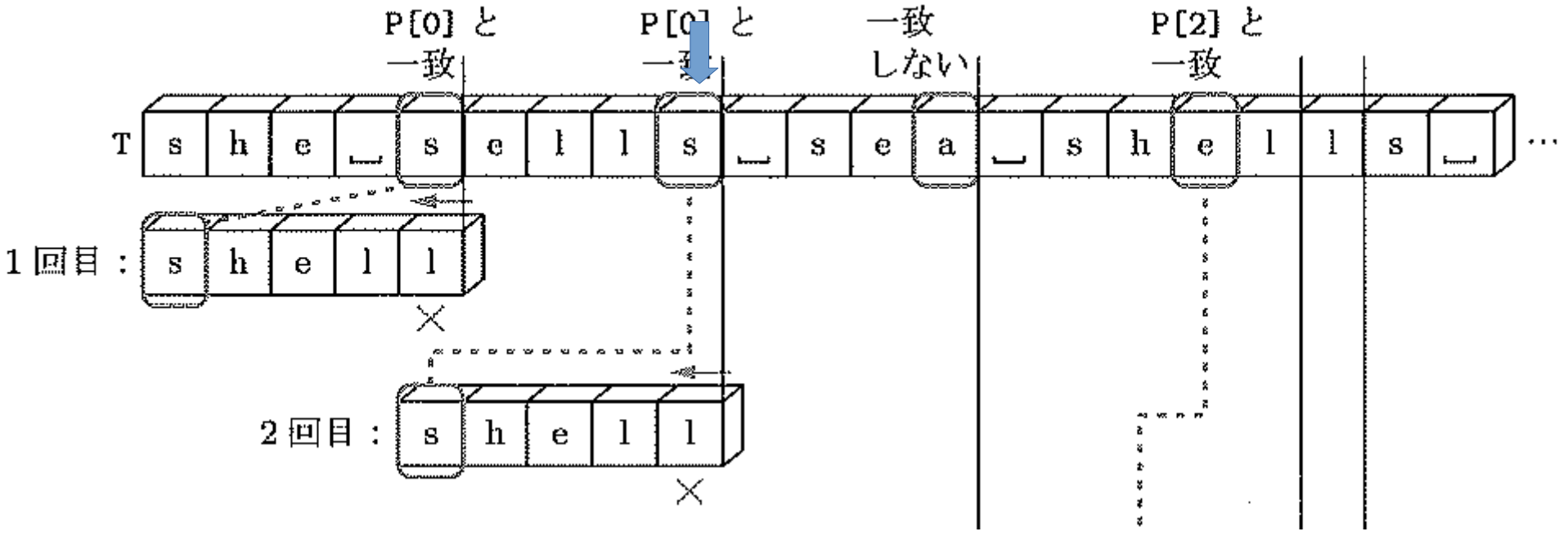
③ ②で記録したjとαに対して，L[α]$<j$の場合に照合位置を右に$j-$L[α]だけずらし，L[α]$\geq j$の場合は照合位置を1つだけ右にずらす．この処理を行った後に②に戻る．

例： 図12.5の1回目の照合ではL[s]=0であり，L[s]$<j$が成り立つので照合位置を$4-0=4$だけ右にずらす．また，2回目以降の照合におけるi, j, αの値と，照合位置を右にずらす量の計算は，以下のようになる．

2回目： $i=4, j=4, \alpha =$“s”であり，L[s]$<j$が成り立つので照合位置を$4-0=4$だけ右にずらす．

ボイヤー・ムーア法(アイデア1)

を4−0＝4だけ右にずらす．　ここではL[s] = 0になっている



ボイヤー・ムーア法(アイデア1)

アルゴリズム 12.3

ボイヤー-ムーア法の1つ目のアイデアによる文字列照合アルゴリズム

```
入力：テキストを表す配列 T[0], T[1], ..., T[n-1], およびパターンを表す配列 P[0],
      P[1], ..., P[m-1]
for (i=0, α=テキスト中の文字 ; i<テキスト中の文字の種類数; i=i+1) {
     L[α]=-1; }
                  //テキスト中のすべての文字について配列Lの値を-1に初期化する
for (i=0; i<m; i=i+1)  { L[P[i]]=i; }
```

ボイヤー・ムーア法(アイデア1)

```
while (i<n-m+1)  {
  j=m-1;                    //jはパターンに対して照合を行う位置を表す
  while ((T[i+j]==P[j])かつ(j>=0))  { j=j-1; }
  if (j==-1) { iを出力しアルゴリズム終了; }
  else if (L[T[i+j]]<j) { i=i+(j-L[T[i+j]]); }   //L[α]<jの場合の処理
  else { i=i+1; }                                  //上記以外の場合の処理
}
"一致しない"と出力;
```

ボイヤー・ムーア法(アイデア2)

- パターンとテキストが一致した部分を利用
- 例
  - パターン　shell
  - テキスト　sells
  - 「ell」を利用して照合を省略する

ボイヤー・ムーア法(アイデア2)

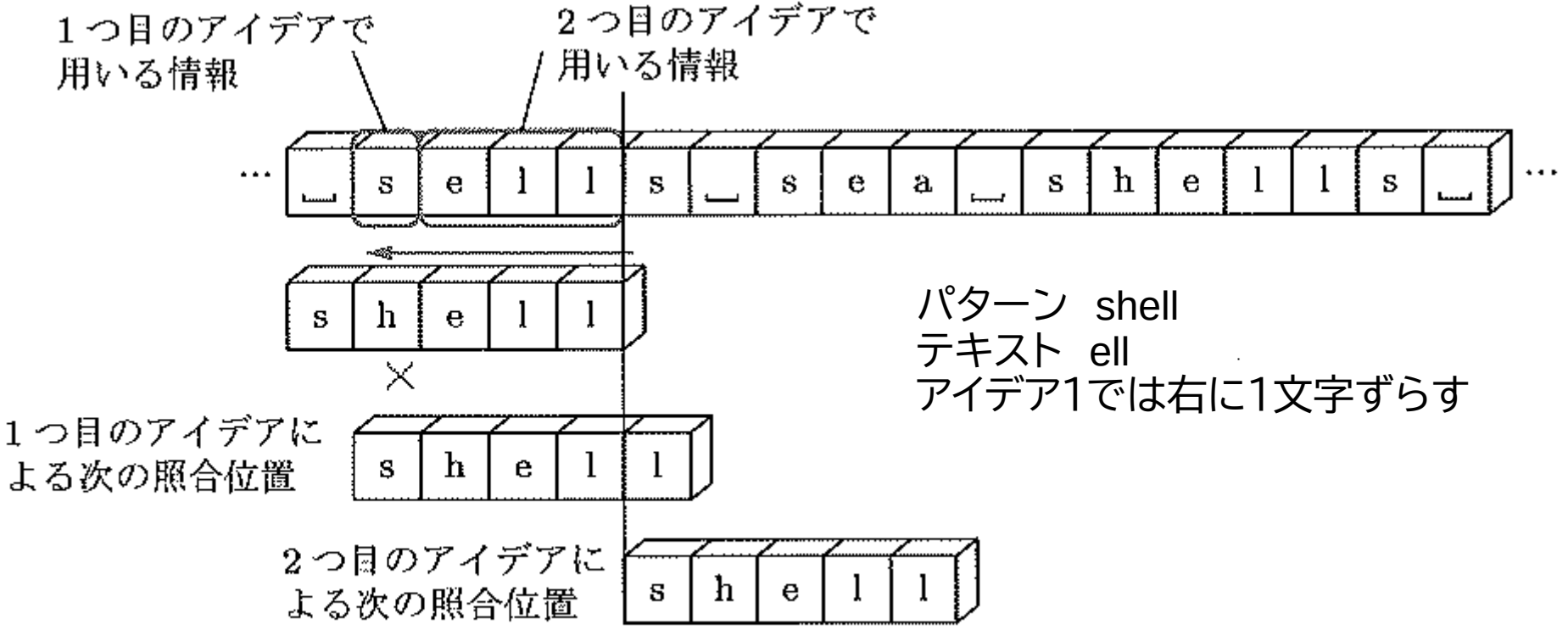


図 12.6 1つ目のアイデアと2つ目のアイデアの比較

ボイヤー・ムーア法(アイデア2)

場合分け

(A) 照合で一致する部分がない場合

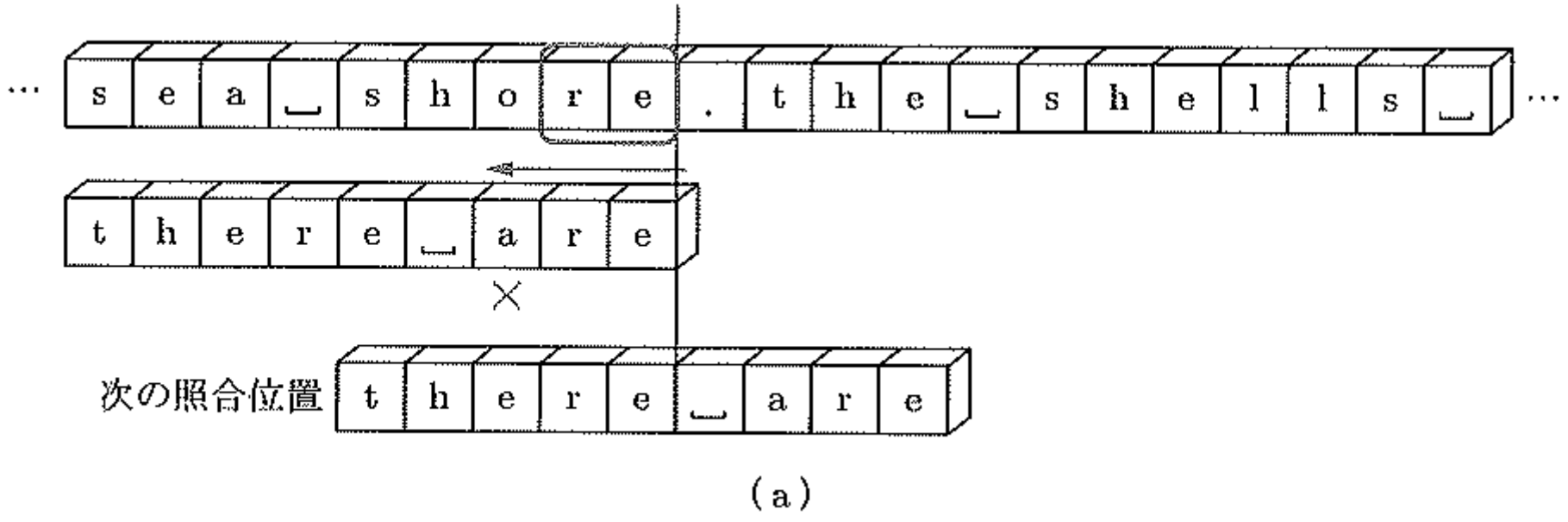
(B) 照合で一致した部分がパターン中に現れる場合

(C) 照合で一致した部分の最後尾がパターンの先頭に一致する場合

(D) 1～3に該当しない場合

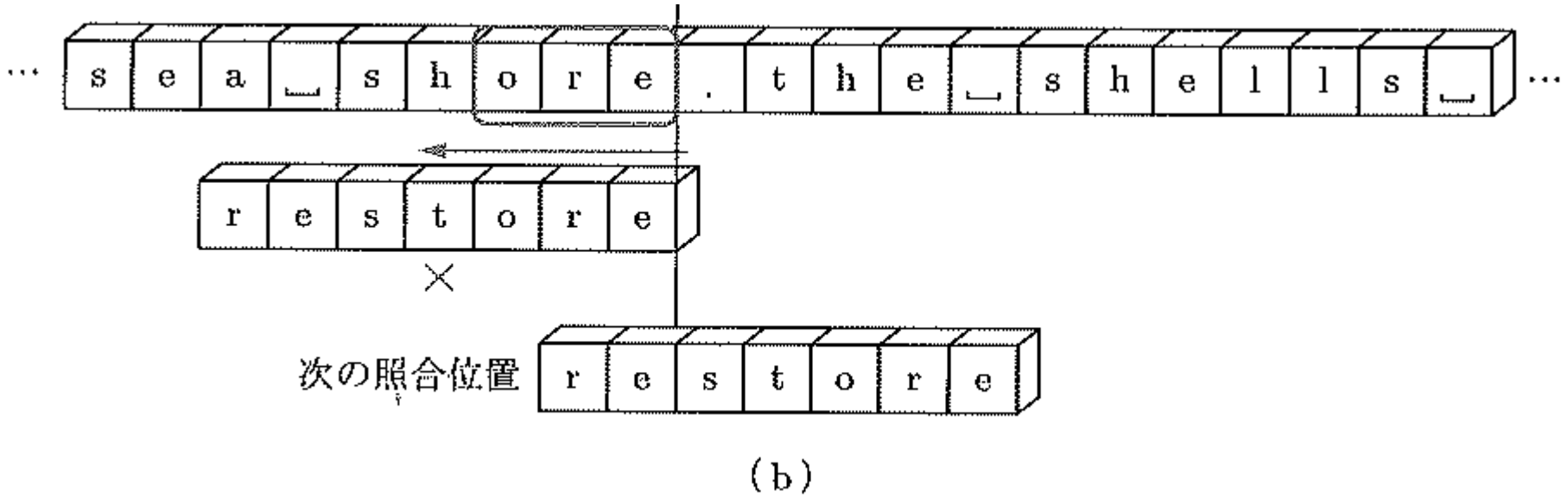
ボイヤー・ムーア法(アイデア2)

場合分け2 「re」が一致



ボイヤー・ムーア法(アイデア2)

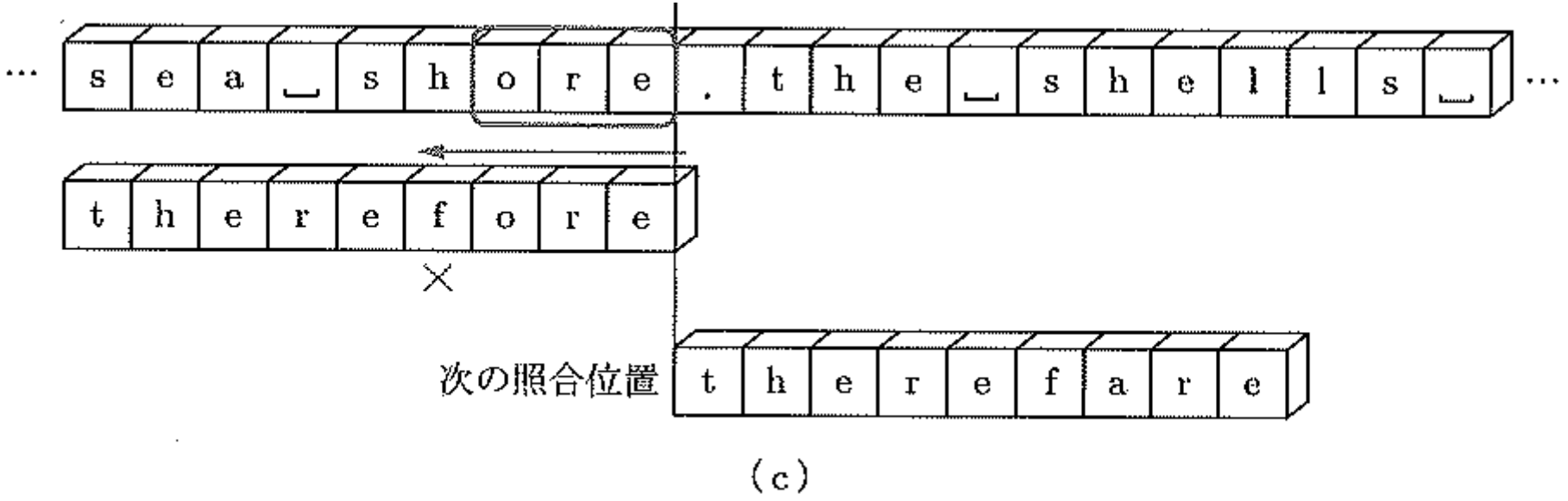
場合分け3 「re」が一致

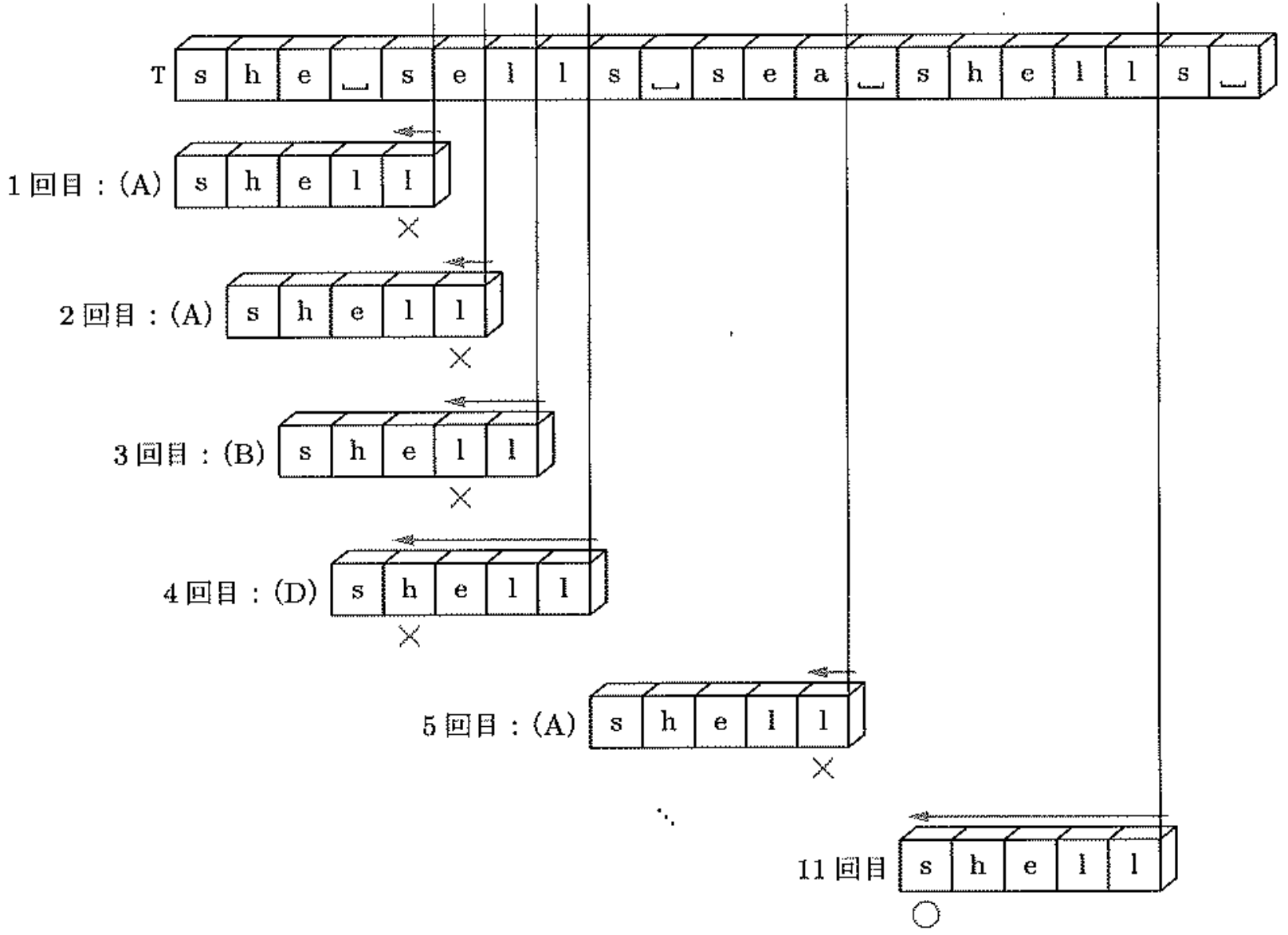


(b)

ボイヤー・ムーア法(アイデア2)

場合分け4 「re」が一致

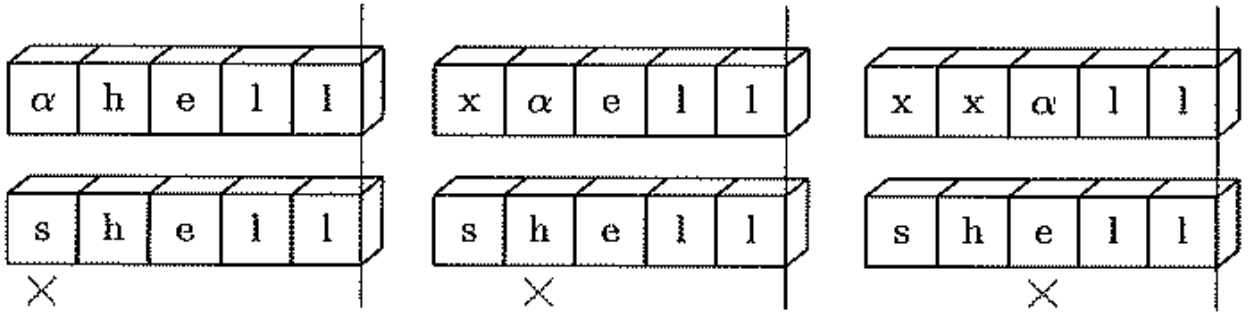



T s h e ␣ s e l l s ␣ s e a ␣ s h e l l s ␣
1回目：(A) s h e l l
2回目：(A) s h e l l
3回目：(B) s h e l l
4回目：(D) s h e l l
5回目：(A) s h e l l
11回目 s h e l l

S[0], S[1], S[2]：それぞれ，図 12.9 (a)，(b)，(c)のような場合であり，すべて(D)の場合があてはまるので，S[0]=S[1]=S[2]=5.

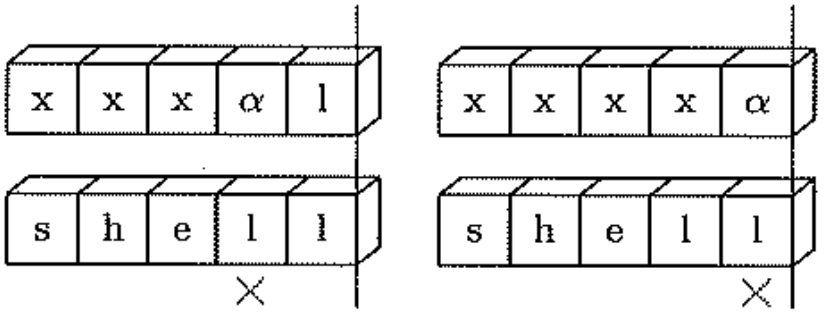
S[3]：図 12.9 (d)の場合であり，(B)の場合があてはまるので S[3]=1.

S[4]：図 12.9 (e)の場合であり，(A)の場合があてはまるので S[4]=1.



(a)　(b)　(c)

文字のずらし量



(d)　(e)

アルゴリズム 12.4

ボイヤー－ムーア法の2つ目のアイデアによる文字列照合アルゴリズム

入力：テキストを表す配列 T[0], T[1], ..., T[n-1], およびパターンを表す配列 P[0], P[1], ..., P[m-1]

```
for (i=0; i<=m-1; i=i+1) { 各文字P[i]についてS[i]を計算; }
i=0;                        //iはテキストに対して照合を行う位置の先頭を表す
while (i<n-m+1)  {
  j=m-1;                    //jはパターンに対して照合を行う位置を表す
  while ((T[i+j]==P[j])かつ(j>=0))  { j=j-1; }
  if (j==-1) { iを出力しアルゴリズム終了; }
  else { i=i+S[j]; }  //照合位置をずらす
}
"一致しない"と出力;
```

ふたつのアイデアの統合

- アイデア1とアイデア2の文字のずらす量の大きい方を選び、照合を効率化する。
- 教科書はアイデアそれぞれの紹介にとどめている。

補足

- 世の中の実装例は、簡易型としてアイデア1の実装例が多い。
- 基本情報技術者試験の午後問題(平成27年秋期、今で言う科目B)にも出題されたことがある。つまり、今後は出題されにくい？、ネタ切れなら出題がある？