アルゴリズム論 11

文字列パターン照合

- 単純法(力まかせ法, 素朴法)
- KMP法
- BM法

文字列パターン照合とは

- **文字列パターン照合**
  - string pattern matching
  - string searching
    - 文書中から**特定の文字列パターンを探索**
- **用語の定義**
  - 探索される側の文字列:**テキスト(text)**
  - 探索する文字列:**パターン(pattern)**

2

文字列パターン照合の応用分野

- ワードプロセッサの検索
- 情報検索(Web)におけるキーワードサーチ

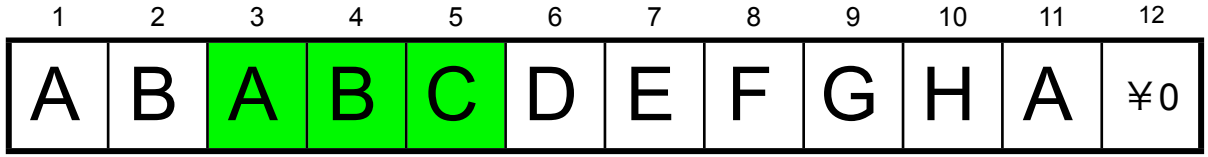
テキストの大容量化、複雑化に伴い

高速な処理が必要

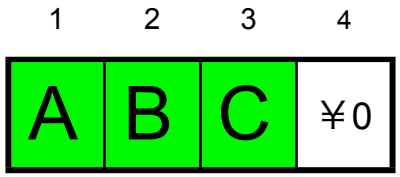
3

文字列パターン照合の例

・**テキスト**



・**パターン**



・**照合結果**

j=3

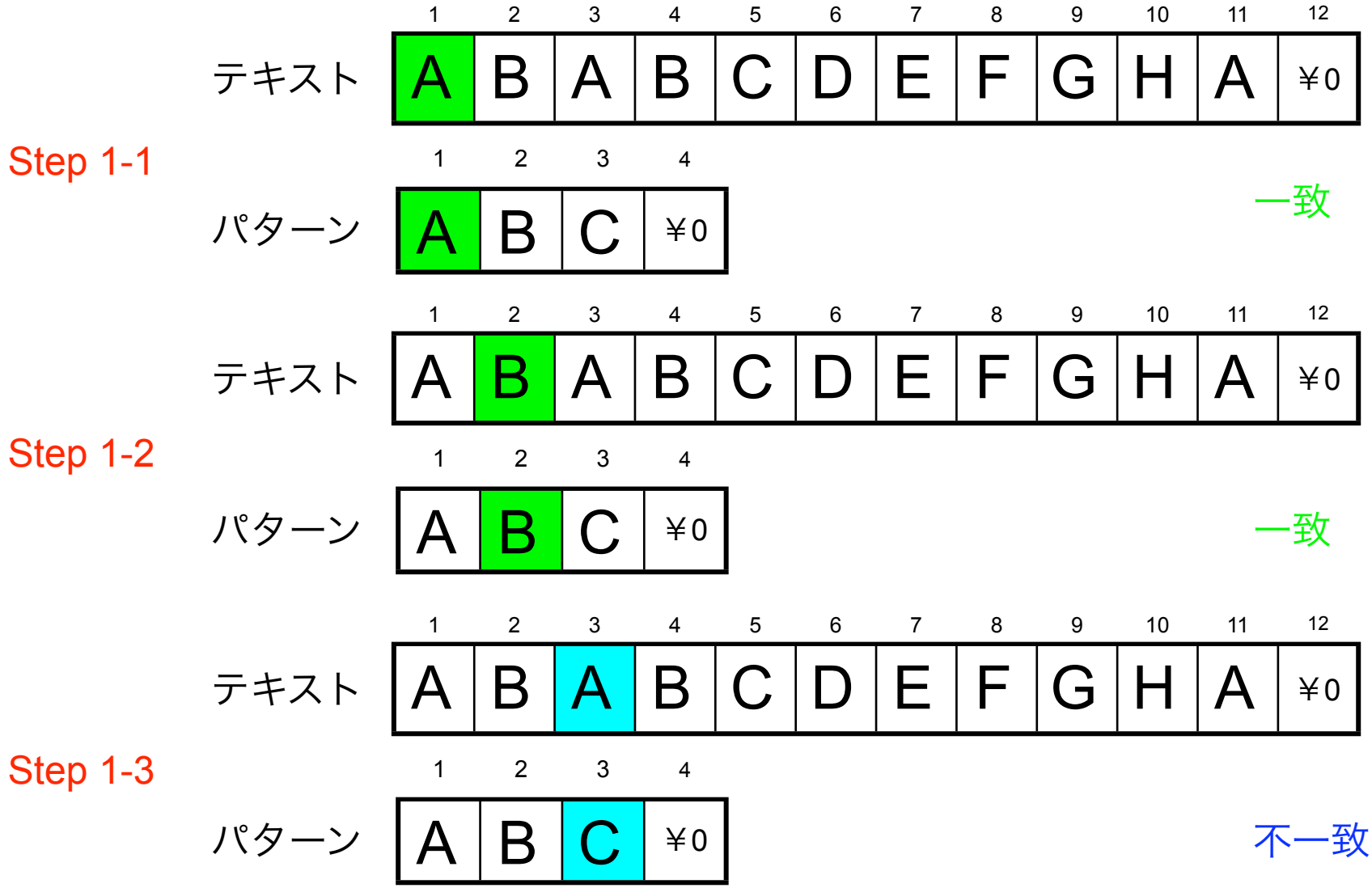
文字列 t[ ] と p[ ] が与えられる場合

t[j]t[j+1]t[j+2]....t[j+m-1]=p[1]p[2]..p[m]　が成立

パターン p が位置 j でテキスト t に照合

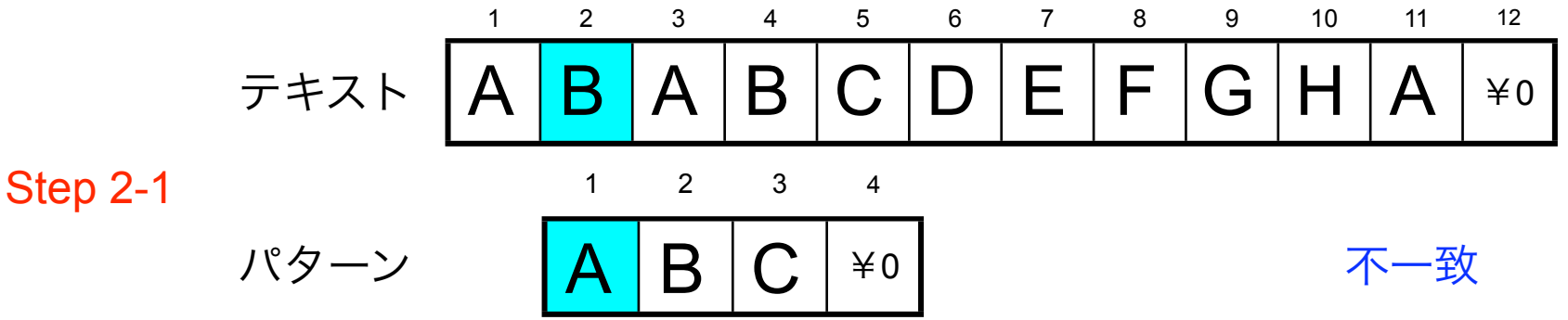
4

単純法（素朴法）


1 2 3 4 5 6 7 8 9 10 11 12
テキスト
A B A B C D E F G H A ¥0
Step 1-1
1 2 3 4
パターン
A B C ¥0
一致
1 2 3 4 5 6 7 8 9 10 11 12
テキスト
A B A B C D E F G H A ¥0
Step 1-2
1 2 3 4
パターン
A B C ¥0
一致
1 2 3 4 5 6 7 8 9 10 11 12
テキスト
A B A B C D E F G H A ¥0
Step 1-3
1 2 3 4
パターン
A B C ¥0
不一致

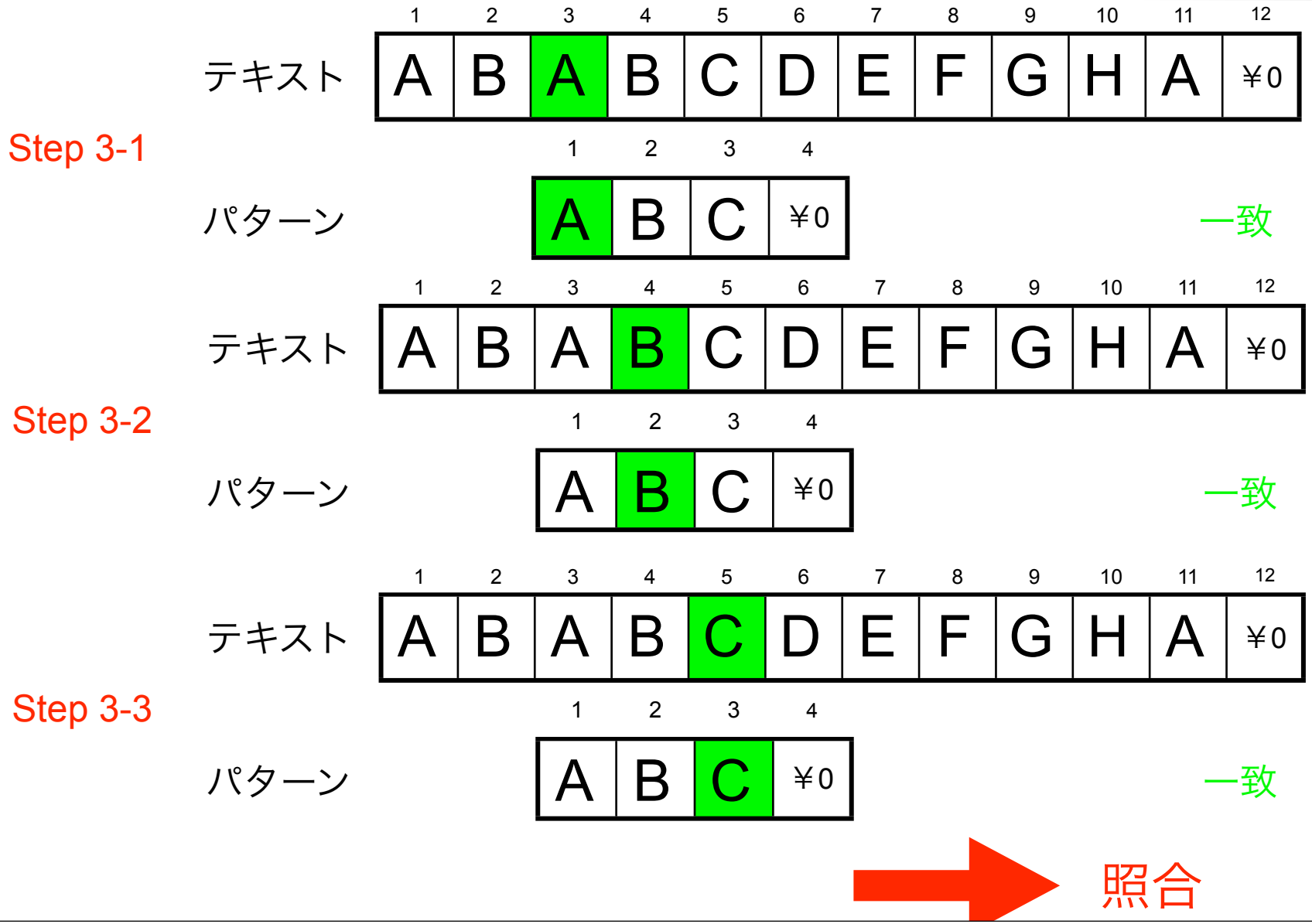
5

単純法(素朴法)


1 2 3 4 5 6 7 8 9 10 11 12
テキスト
A B A B C D E F G H A ¥0
Step 2-1
1 2 3 4
パターン
A B C ¥0
不一致

6

単純法(素朴法)


1 2 3 4 5 6 7 8 9 10 11 12
テキスト
A B A B C D E F G H A ¥0
Step 3-1
1 2 3 4
パターン
A B C ¥0
一致
1 2 3 4 5 6 7 8 9 10 11 12
テキスト
A B A B C D E F G H A ¥0
Step 3-2
1 2 3 4
パターン
A B C ¥0
一致
1 2 3 4 5 6 7 8 9 10 11 12
テキスト
A B A B C D E F G H A ¥0
Step 3-3
1 2 3 4
パターン
A B C ¥0
一致
照合

7

単純法1（メイン）

```
#include <stdio.h>
int count=0;    /* 比較回数カウンタ  */

int simple_match(char txt[], char pat[]);  /* 関数プロトタイプ */

int main(void)
{
        int     idx;        /* 照合位置  */
        char    s1[80];     /* テキスト  */
        char    s2[80];     /* パターン  */

        printf(" Input text :");      /* テキスト入力  */
        scanf("%s",s1);
        printf(" Input pattern :");   /* パターン入力  */
        scanf("%s",s2);
        idx=simple_match(s1,s2);      /* 単純法関数  */

        if (idx==-1)                  /* 結果表示  */
                printf(" No pattern found in the text ¥n");
        else
                printf(" Pattern was found at %d ¥n",idx+1);
        printf(" Number of comparison=%d¥n",count);  /* 比較回数表示  */

        return(0);
}
```


8

単純法2(関数)

```
int simple_match(char txt[], char pat[])
{
        int     pt=0;   /* テキスト　カーソル　*/
        int     pp=0;   /* パターン　カーソル　*/

        while (txt[pt] !='¥0' && pat[pp] != '¥0' )    { /* 照合 */
                if (txt[pt]==pat[pp]) {     /* 一致　*/
                        pt++;
                        pp++;

                } else {                    /* 不一致　*/
                        pt=pt-pp+1;
                        pp=0;

                }
        }
        if (pat[pp]=='¥0')
                return (pt-pp);     /* 戻り値：照合結果　*/
        return(-1);
}
```


9

単純法実行結果

case 2: 教科書p.100のケース

case 1

```
Input text :ABABCDEFGHA
Input pattern :ABC
Pattern was found at 3
Number of comparison=7
```

case 2

```
Input text :ababdababccbdcabcadb
Input pattern :ababc
Pattern was found at 6
Number of comparison=16
```

case 3

```
Input text :ABCABCABCABCABCDABC
Input pattern :ABCABCD
Pattern was found at 10
Number of comparison=34
```

case 4

```
Input text :ABABCDEFGHA
Input pattern :ZZ
No pattern found in the text
Number of comparison=11
```


10

演習問題(講義時間内で実施)

- 文字列パターン照合を行うプログラムのソースコードを入力し実行する
  - メイン
  - 単純法
- テキストおよびパターンの文字列を入力し、実行結果を確認する

11

単純法 の計算量

計算量：テキストn文字、パターンm文字の文字列照合

- 文字の比較回数
  - １回のパターンの比較回数：m回
  - テキスト上のパターンの移動回数：n 回
  - 最悪の場合　m× n 回の比較

オーダ　O(m×n)

12