



本节主题:

串的模式匹配(Brute-Force算法)

串的模式匹配

❏ 模式匹配

- ❏ 设有主串s和子串t，**子串t的定位**就是要在主串s中找到一个与子串t相等的子串。

s: This is his hat. It is historical.

t: his

- ❏ 通常把主串 s 称为目标串，把子串t称为模式串，定位也称作**模式匹配**。

❏ 模式匹配的结果

- ❏ 模式匹配成功：在目标串s中找到一个模式串t；
- ❏ 模式匹配不成功：目标串s中不存在模式串t。

模式匹配有大用！

2002 年 6 月
第 25 卷第 6 期

重庆大学学报 (自然科学版)
Journal of Chongqing University(Natural Science Edition)

Vol.25 No.6

Jun.2002

文章编号:1000-582X(2002)06-0027-05

一种基于点模式匹配的指纹识别方法

王崇文¹, 李见为¹, 郑治伟², 林国清¹,

2007年2月
第24卷第1期

沈阳航空工业学院学报
Journal of Shenyang Institute of Aeronautical Engineering

Feb. 2007
Vol. 24 No. 1

文章编号:1007-1385(2007)01-0038-03

基于模式匹配的中文问答系统

NM_001081414.2	CTTGGT.TTAGAAA....AAAAAAAGACCTCTGTAATGCTGAGCCTCCATAATTTTCTACAAATTCAGAT
NM_001143834.1	CTTGGT.TTAGAAA....AAAAAAAGACCTCTGTAATGCTGAGCCTCCATAATTTTCTACAAATTCAGAT
NM_009932.3	GCTGCTGCTAGCAACTGTGACCTGTGGGACTCTGCT..CAGAGCTCTTGGGGGGTGTGAAAGAGTTGGA
Consensus	tg t tag aa a a gac ctg t agc tc t t a aa t ga
NM_001081414.2	CGAAACTGACCT...AGTATATCCAGCT.GTTTGTGTC..ACATATGATCATAAATTCAGTTATGAATA
NM_001143834.1	CGAAACTGACCT...AGTATATCCAGCT.GTTTGTGTC..ACATATGATCATAAATTCAGTTATGAATA
NM_009932.3	TGGAGGGAGAGACTGCACTGGGGGTTGCCAGTGCACCGGACAAAGGAGCAAGGGGTC.AGCCGGGGCA
Consensus	c ga a ct agt gc gt t cc a a a ga ca tc ag a g a
NM_001081414.2	TATGGATGCACTCTAACTGACACTCTCTGGGGACCTTGTTATTTTATAGAGAAATTTCTAAATTAATGCC
NM_001143834.1	TATGGATGCACTCTAACTGACACTCTCTGGGGACCTTGTTATTTTATAGAGAAATTTCTAAATTAATGCC
NM_009932.3	ONGGGTACA.....ATGGTCCCCAGGG...TTGCAAGGATCCAGG..A...CTACAGGGGCGCA
Consensus	a gg t ca a c c c ggg ttg a tt ag a cta a c gc

s: This is his hat. It is historical.

t: his

自然语言处理：信息检索

生物信息处理：DNA匹配

语音识别：输入的语音符号化

网络安全：病毒入侵检测

Brute-Force 算法

Brute-Force, BF算法, 亦称简单匹配算法

基本思路

从目标串 $s = "s_0s_1...s_{n-1}"$ 的第一个字符开始和模式串 $t = "t_0t_1...t_{m-1}"$ 中的第一个字符比较, 若相等, 则继续逐个比较后续字符

否则从目标串 s 的第二个字符开始重新与模式串 t 的第一个字符进行比较

依次类推.....

若从目标串 s 的第 i 个字符开始, 每个字符依次和模式串 t 中的对应字符相等, 则匹配成功, 返回 i ; 否则, 匹配失败, 函数返回-1。

s: a a a a a b a
t: a a a b



s: a a a a a b a
t: a a a b



s: a a a a a b a
t: a a a b



B-F算法的一般过程

$$\begin{array}{ccccccc}
 t & = & t_0 & t_1 & \dots & t_{j-1} & t_j & t_{j+1} & \dots & t_{m-1} \\
 & & | & | & & | & \diagdown & & & \\
 s & = & s_0 & s_1 & \dots & s_{j-1} & s_j & s_{j+1} & \dots & s_{n-1} \\
 & & | & & & & & & & \\
 & & t & = & t_0 & t_1 & \dots & t_{j-1} & t_j & t_{j+1} & \dots & t_{m-1}
 \end{array}$$

$$\begin{array}{ccccccc}
 t & = & t_0 & t_1 & \dots & t_{j-1} & t_j & t_{j+1} & \dots & t_{m-1} \\
 & & | & | & & | & \diagdown & & & \\
 s & = & s_0 & s_1 & \dots & s_{i-j} & \boxed{s_{i-j+1}} & \dots & s_{i-1} & s_i & s_{i+1} & \dots & s_{n-1} \\
 & & & & & | & & & & & & & \\
 & & & & & t & = & t_0 & t_1 & \dots & t_{j-1} & t_j & t_{j+1} & \dots & t_{m-1}
 \end{array}$$

```

int index(SqString s,SqString t)
{
    int i=0,j=0;
    while (i<s.length && j<t.length)
    {
        if (s.data[i]==t.data[j])
        {
            i++;
            j++;
        }
        else
        {
            i=i-j+1;
            j=0;
        }
    }
    if (j>=t.length)
        return(i-t.length);
    else
        return(-1);
}
    
```

算法"走一走"

↓ i=0
s: a b a b c a b c a c b a b
t: a b c a c
↑ j=0

↓ i=2
s: a b a b c a b c a c b a b
t: a b c a c
↑ j=2

↓ i=1
s: a b a b c a b c a c b a b
t: a b c a c
↑ j=0

```
int index(SqString s,SqString t)
{
    int i=0,j=0;
    while (i<s.length && j<t.length)
    {
        if (s.data[i]==t.data[j])
        {
            i++;
            j++;
        }
        else
        {
            i=i-j+1;
            j=0;
        }
    }
    if (j>=t.length)
        return(i-t.length);
    else
        return(-1);
}
```

算法分析

📁 最好情况复杂度： $O(m)$

📁 目标串的前 m 个字符正好等于模式串的 m 个字符。

📁 最坏情况复杂度： $O(n*m)$

📁 每次到最后一个字符才发现不匹配，这时再“倒回去”进行比较

📁 计数关系： $m*(n-m)$

📁 算法评价

📁 这个算法简单，易于理解，但效率不算高。

📁 当 m 小于5时，几乎是最好的。

s: a a a a a b a
t: a a a b

s: a a a a a b a
t: a a a b

s: a a a a a b a
t: a a a b