- 作业讲解
  - TJ第8章练习6、7、8、9、11、13、18、19、21、22、23

TJ第8章练习18

- 如果不全为0：
  - 其中为0的codeword构成非空集合S0
  - 其中为1的codeword构成非空集合S1
- 欲证S0和S1之间存在双射：
  任取S1中元素z，f(x)=x+z: S0→S1
  - 单射：f(x1)=f(x2) → x1+z=x2+z → x1=x2
  - 满射：任取S1中元素y，构造x=y+z
    → x在S0中（因为：x是codeword，且第i位为0）
    且f(x)=f(y+z)=y+z+z=y

- 教材讨论
  - TC第32章

问题1：naive

- 你能够基于naïve算法高效地解决这个问题吗？

Suppose we allow the pattern P to contain occurrences of a *gap character* ◇ that can match an *arbitrary* string of characters (even one of zero length). For example, the pattern ab◇ba◇c occurs in the text cabccbacbacab as

```
c ab cc ba cba c ab
  ab ◇ ba ◇ c
```

and as

```
c ab ccbac ba c ab .
  ab ◇ ba ◇ c
```

Note that the gap character may occur an arbitrary number of times in the pattern but not at all in the text. Give a polynomial-time algorithm to determine whether such a pattern P occurs in a given text T, and analyze the running time of your algorithm.

- 分段匹配

问题1： naïve (续)

- 我们稍稍改一改问题，你还能够高效地解决吗？
  - P occurs in a given text T → P matches T（即必须与整个T匹配）
  - ◊ → ?和*

- 动态规划
  - if: P[i]==T[j] || P[i]=='?'&&T[j]!=EMPTY
    - ans[i-1,j-1]
  - if: P[i]=='*'
    - ans[i,j-1] | ans[i-1,j] | ans[i-1,j-1]

问题2：Rabin-Karp

- 这是对naïve和Rabin-Karp的另一种叙述方式，你理解了吗？

```
1 function NaiveSearch(string s[1..n], string pattern[1..m])
2    for i from 1 to n-m+1
3       for j from 1 to m
4          if s[i+j-1] ≠ pattern[j]
5             jump to next iteration of outer loop
6       return i
7    return not found
```

```
1 function RabinKarp(string s[1..n], string pattern[1..m])
2 hpattern := hash(pattern[1..m]);  hs := hash(s[1..m])
3 for i from 1 to n-m+1
4   if hs = hpattern
5     if s[i..i+m-1] = pattern[1..m]
6       return i
7   hs := hash(s[i+1..i+m])
8 return not found
```

- 按照这种叙述，Rabin-Karp的效率一定比naïve高吗？
- 使Rabin-Karp的效率高于naïve的原因是什么？
  - rolling hash

问题2： Rabin-Karp (续)

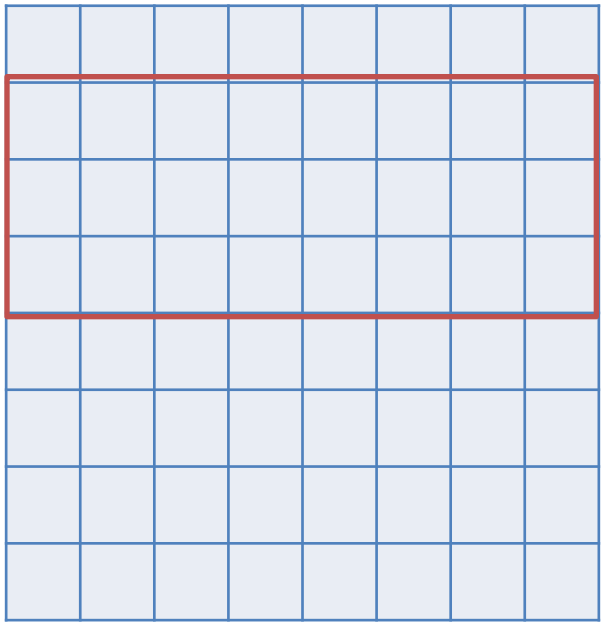
- How would you extend the Rabin-Karp method to the problem of searching a text string for an occurrence of any one of a given set of k patterns? Start by assuming that all k patterns have the same length. Then generalize your solution to allow the patterns to have different lengths.

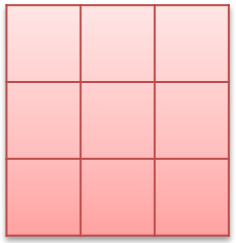
- hash table

问题2： Rabin-Karp (续)

- Show how to extend the Rabin-Karp method to handle the problem of looking for a given $m \times m$ pattern in an $n \times n$ array of characters. (The pattern may be shifted vertically and horizontally, but it may not be rotated.)

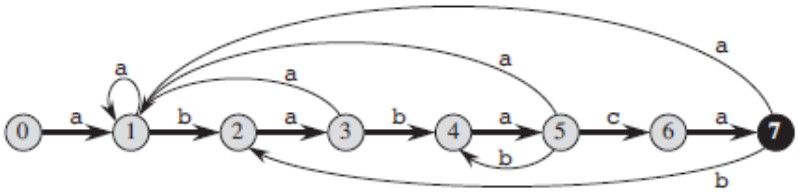
- mxn和mxm → 1xn和1xm → string matching (2维rolling hash, 先列后行)





问题3：automaton

- 自动机的5个组成部分是什么？

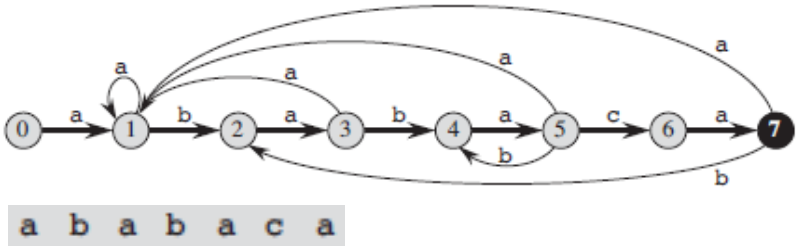


- Q is a finite set of ***states***,
- $q_0 \in Q$ is the ***start state***,
- $A \subseteq Q$ is a distinguished set of ***accepting states***,
- Σ is a finite ***input alphabet***,
- δ is a function from $Q \times \Sigma$ into Q, called the ***transition function*** of M.

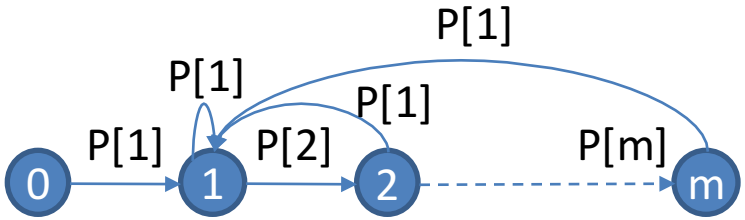
- 粗线和细线分别表示什么意思？

问题3：automaton (续)

- 你理解这个transition function了吗？ $\delta(q,a) = \sigma(P_q a)$
- 这个自动机始终维护的invariant是什么含义？ $\phi(T_i) = \sigma(T_i)$



- We call a pattern P ***nonoverlappable*** if $P_k \sqsupset P_q$ implies $k = 0$ or $k = q$. Describe the state-transition diagram of the string-matching automaton for a nonoverlappable pattern.
  - 任选一些nonoverlappable pattern，给出其string-matching automaton
  - 你能总结出规律吗？



其它情况都回到0