有限状态机与最小编辑距离

01

有限状态机

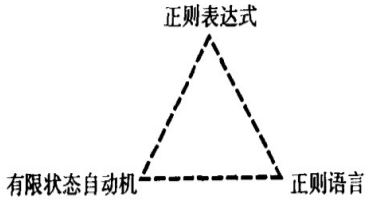
有限状态机

1. 概述
2. 应用

stefanie

有限状态机的概述

- 任何正则表达式都可以用有限状态自动机来实现(除了使用存储特性的那些正则表达式之外）
- 任何有限状态自动机都可以用正则表达式来描述。有限状态自动机和正则表达式彼此对称。
- 正则表达式是用来刻画正则语言（ regularlanguage)的一种方法
- 正则语言是一种特别的形式语言。



stefanie

有限状态机的概述

- 如何识别羊的叫声
  - baa!
  - baaa!
  - baaaa!
  - baaaaa!
  - baaaaaa!..

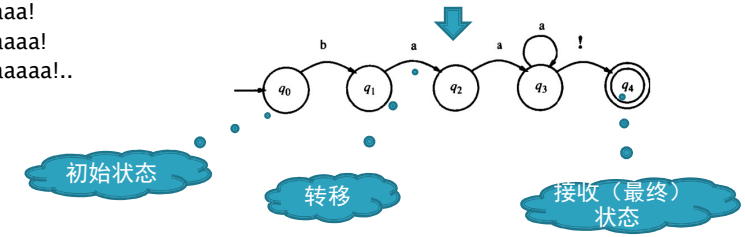
问题： 正则表达式？

stefanie

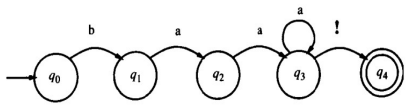
有限状态机的概述

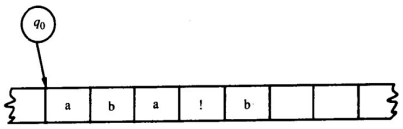
- 如何识别羊的叫声
  - baa!
  - baaa!
  - baaaa!
  - baaaaa!
  - baaaaaa!..

有限状态机:(finite-state machine，缩写：**FSM**）又称**有限状态自动机**，简称**状态机**，是表示有限个状态以及在这些状态之间的转移和动作等行为的数学模型。



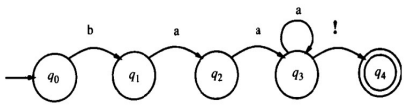
有限状态机的概述





- 从形式上说，一个有限自动机可以用下面5个参数来定义:
  - **Q:**N中状态q_0，… …, q_w的有限集合
  - **Σ:**有限的输人符号字母表
  - **q_0** :初始状态
  - **F:**终极状态的集合，F⊆Q
  - **δ(q, i):**状态之间的转移函数或转移矩阵。给定一个状态q ∈Q和一个输人符号i ∈ ∑. δ(q, i)返回一个新的状态q’，因此， δ(q, i)是从Q× ∑到Q的一个关系。

有限状态机的概述



- 上例中，
  Q=?
  Σ = ? ,
  F =?

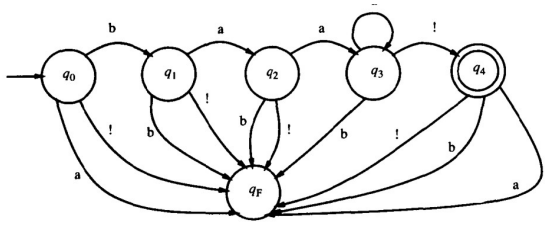
δ(q, i)用转移表确定：

	输出		
状态	b	a	!
0	1	0	0
1	0	2	0
2	0	3	0
3	0	3	4
4:	0	0	0

stefanie

有限状态机的概述

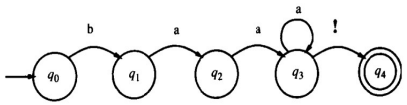
- 增加一个失败状态：



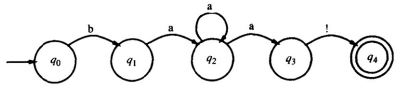
stefanie

有限状态机的概述

- 确定性的的自动机（Deterministic FSA或DFAS),



- 非确定的有限自动机(non-deterministic FSA或NFSA)



stefanie

有限状态机的应用

- 地址识别

广东省深圳市腾讯大厦

广东省 518057 深圳市南山区科技园腾讯大厦

深圳市 518057 科技园腾讯大厦

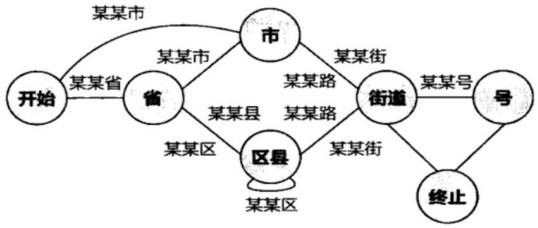
深圳市南山区科技园腾讯公司

深圳市南山区科技园腾讯总部 518000（估计不知道准确的邮编）

广东省深圳市科技园中一路腾讯公司

……

有限状态机的应用

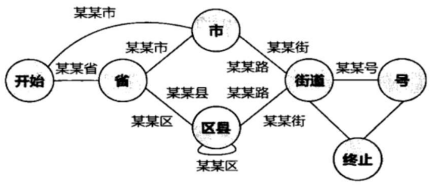


识别地址的有限状态机

如果一个地址能从状态机的起始状态，经过若干中间状态，走到最终状态，那么这条地址就是有效的

stefanie

- 思考：根据有限转态机右下图的哪些地址可以识别？



识别地址的有限状态机

广东省深圳市腾讯大厦

广东省 518057 深圳市南山区科技园腾讯大厦

深圳市 518057 科技园腾讯大厦

深圳市南山区科技园腾讯公司

深圳市南山区科技园腾讯总部 518000（估计不知道准确的邮编）

广东省深圳市科技园中一路腾讯公司

.....

stefanie

有限状态机的应用

- 有限自动机用于英语单词形态分析[Allen, 1995]

英语单词形态变化非常普遍，例如：

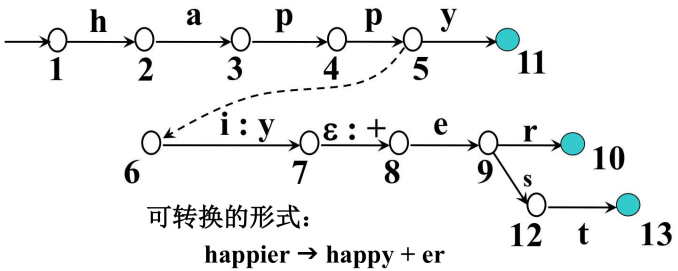
eat: eats, eating, ate, eaten

happy: happier, happiest

stefanie

有限状态机的应用

- 任务：识别单词:happy, happier, happiest



可转换的形式：

happier → happy + er

happiest →happy + est

stefanie

02

最小编辑距离

最小编辑距离

1. 引言
2. 最小编辑距离
3. 最小编辑距离-动态规划
4. 其他应用

stefanie

引言

- 拼写纠错

涛宝 → 淘宝

happy → hoppy

在英文查询串中，有Spelling Error的查询串占到了总查询串的10%~15% (Cucerzan & Brill 2004)，而中文有输入错误的查询串保守估计也要占到所有查询串的5%~10%左右

纠错算法主要是列出语料库词典中与原查询词具有最小编辑距离的词条，将其作为候选词

最小编辑距离

- 最小编辑距离
  - 将一个错误拼写的单词被纠正成正确单词的最小编辑次数，每一次编辑只能改变一个字母。
  - 俄罗斯科学家 Vladimir Levenshtein 在1965年提出来的，所以编辑距离又称为Levenshtein距离

acress→actress （插入操作）：1
acress→access （替换操作）：2
acress→cress （删除操作）：1

stefanie

最小编辑距离

- 原始串s o t
- 目标串s t o p

s o t → s t o t → s t o p

编辑距离：3

s o t → s t t → s t o → s t o p

编辑距离：5

stefanie

最小编辑距离的动态规划

$$substituteCost \begin{cases} = 0 & \text{if target[i] = source[j]} \\ = 2 & \text{otherwise} \end{cases}$$

insertCost = 1
deleteCost = 1

D(0, 0) = 0
D(i, 0) = insertCost * i
D(0, j) = deleteCost * j

i, 目标串字符位置

j, 原始串字符位置

$$D(i, j) = \min \begin{cases} D(i-1, j) + insertCost(target_i) \\ D(i-1, j-1) + substituteCost(source_j, target_i) \\ D(i, j-1) + deleteCost(source_j) \end{cases}$$

Termination: D(N,M) is distance

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j						
3	t					
2	o					
1	s					
0	#	s	t	o	p	
#	0	1	2	3	4	i

n = length (target)

m = length (source)

Create matrix d [n, m];

i=0 j=0

d[0,0] = 0;
d[0,1] = 1; … ; d[0,m] = m;
d[1,0] = 1; … ; d[n,0] = n;

最小编辑距离的动态规划

- 原始串s：s o t
- 目标串t: s t o p

j						
3	t					
2	o					
1	s	**0**				
0	#	s	t	o	p	
#	0	1	2	3	4	i

i=1 j=1

$$d[1,1] = \min \begin{cases} d[0,1] + insert(t[1]) = 2 \\ d[0,0] + substitute(s[1],t[1]) = 0 \\ d[1,0] + delete(s[1]) = 2 \end{cases} = 0$$

stefanie

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j						
3	t					
2	o	1				
1	s	**0**				
0	#	s	t	o	p	
#	0	1	2	3	4	i

i=1 j=2

$$d[1,2] = \min \begin{cases} d[0,2] + insert(t[1]) = 3 \\ d[0,1] + substitute(s[2],t[1]) = 3 \\ d[1,1] + delete(s[2]) = 1 \end{cases} = 1$$

stefanie

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j						
3	t	2				
2	o	**1**				
1	s	**0**				
0	#	s	t	o	p	
#	0	1	2	3	4	i

i=1 j=3

$$d[1,3] = \min \begin{cases} d[0,3]+\text{insert}(t[1]) = 4 \\ d[0,2]+\text{substitute}(s[3],t[1]) = 4 \\ d[1,2]+\text{delete}(s[3]) = 2 \end{cases} = 2$$

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j						
3	t	2				
2	o	**1**				
1	s	**0**	1			
0	#	s	t	o	p	
#	0	1	2	3	4	i

i=2 j=1

$$d[2,1] = \min \begin{cases} d[1,1]+\text{insert}(t[2]) = 1 \\ d[1,0]+\text{substitute}(s[1],t[2]) = 3 \\ d[2,0]+\text{delete}(s[1]) = 3 \end{cases} = 1$$

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j						
3	t	2				
2	o	**1**	2			
1	s	**0**	**1**			
0	#	s	t	o	p	
#	0	1	2	3	4	i

i=2 j=2

$$d[2,2] = \min \begin{cases} d[1,2]+\text{insert}(t[2]) = 2 \\ d[1,1]+\text{substitute}(s[2],t[2]) = 2 \\ d[2,1]+\text{delete}(s[2]) = 2 \end{cases} = 2$$

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j						
3	t	2	1			
2	o	**1**	2			
1	s	**0**	**1**			
0	#	s	t	o	p	
#	0	1	2	3	4	i

i=2 j=3

$$d[2,3] = \min \begin{cases} d[1,3]+\text{insert}(t[2])=3 \\ d[1,2]+\text{substitute}(s[3],t[2])=1 \\ d[2,2]+\text{delete}(s[3])=3 \end{cases} = 1$$

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j\i		t				
3	t	2	1			
2	o	**1**	2			
1	s	**0**	**1**	2		
0	#	s	t	o	p	
#	0	1	2	3	4	

i=3 j=1

d[3,1] = min { d[2,1]+insert(t[3])=2, d[2,0]+substitute(s[1],t[3])=4, d[3,0]+delete(s[1])=4 } = 2

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j\i		t				
3	t	2	1			
2	o	**1**	2	1		
1	s	**0**	**1**	2		
0	#	s	t	o	p	
#	0	1	2	3	4	

i=3 j=2

d[3,2] = min { d[2,2]+insert(t[3])=3, d[2,1]+substitute(s[2],t[3])=1, d[3,1]+delete(s[2])=3 } = 1

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j\i		t				
3	t	2	**1**	2		
2	o	**1**	2	**1**		
1	s	**0**	**1**	2		
0	#	s	t	o	p	
#	0	1	2	3	4	

i=3 j=3

d[3,3] = min { d[2,3]+insert(t[3])=2, d[2,2]+substitute(s[3],t[3])=4, d[3,2]+delete(s[3])=2 } = 2

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j\i		t				
3	t	2	**1**	2		
2	o	**1**	2	**1**		
1	s	**0**	**1**	**2**	3	
0	#	s	t	o	p	
#	0	1	2	3	4	

i=4 j=1

d[4,1] = min { d[3,1]+insert(t[4])=3, d[3,0]+substitute(s[1],t[4])=5, d[4,0]+delete(s[1])=5 } = 3

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j\i		t	2	1	2	
	3	t	2	**1**	2	
	2	o	**1**	2	**1**	2
	1	s	**0**	**1**	2	3
	0	#	s	t	o	p
	#	0	1	2	3	4

i

i=4 j=2

d[4,2] = min { d[3,2]+insert(t[4])=2 ; d[3,1]+substitute(s[2],t[4])=4 ; d[4,1]+delete(s[2])=4 } = 2

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j\i		t	2	1	2	3
	3	t	2	**1**	**2**	**3**
	2	o	**1**	2	**1**	**2**
	1	s	**0**	**1**	2	3
	0	#	s	t	o	p
	#	0	1	2	3	4

i

i=4 j=3

d[4,3] = min { d[3,3]+insert(t[4])=3 ; d[3,2]+substitute(s[3],t[4])=3 ; d[4,2]+delete(s[3])=3 } = 3

最小编辑距离

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

s o t 编辑操作①
s t o t (1. 插入t，1分，累计1分)
s t o p (2. t替换p，2分，累计3分)

s o t 编辑操作②
s t o (1. 删除o，1分，累计1分)
s t o p (2. 插入o，1分，累计2分)
(3. 插入p，1分，累计3分)

s o t 编辑操作③
s t o t (1. 插入t，1分，累计1分)
s t o p t (2. 插入p，1分，累计2分)
s t o p (3. 删除t，1分，累计3分)

s o t 编辑操作④
s t o t (1. 插入t，1分，累计1分)
s t o (2. 删除t，1分，累计2分)
s t o p (3. 插入p，1分，累计3分)

最小编辑距离的动态规划

- 原始串s o t
- 目标串s t o p

j\i		t	2	1	2	3
	3	t	2	**1**	**2**	**3**
	2	o	**1**	2	**1**	**2**
	1	s	**0**	**1**	2	3
	0	#	s	t	o	p
	#	0	1	2	3	4

i

① ② ③ ④

最小编辑距离的动态规划

- 练习：
  intention ⟶ execution

最小编辑距离的动态规划

- 练习：
  intention ⟶ execution

```
i n t e n t i o n
↓ ↓ ↓ ↓ ↓
e x e c u t i o n

s s s s s
2 2 2 2 2   = 10
```

```
i n t e n * t i o n
↓ ↓ ↓   ↓ ↓
* e x e c u t i o n

d s s   s i
1 2 2   2 1     = 8
```


其他应用

- 生物信息学中的**的最小编辑距离（相似度）**
- 问题：找到以下两个序列中的对齐序列，其可能是核苷酸或者蛋白质的结构？

```
AGGCTATCACCTGACCTCCAGGCCGATGCCC
TAGCTATCACGACCGCGGTCGATTTGCCCGAC
```

- 要求得到以下对齐序列：

```
-AGGCTATCACCTGACCTCCAGGCCGA--TGCCC---
TAG-CTATCAC--GACCGC--GGTCGATTTGCCCGAC
```

Thank You !