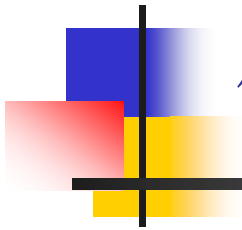
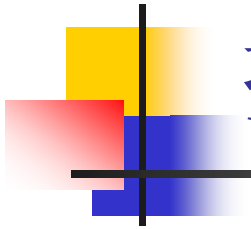


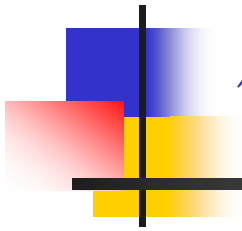
信息过滤

刘挺
哈工大信息检索研究室
2004年秋



提纲

- 信息过滤概述 (概念)
- 模式匹配 (方法)
- 垃圾邮件过滤 (应用)



信息过滤概述

信息过滤概述
——基本概念

Google推出新闻过滤： https://www.google.com/accounts/Login

tliu@ir.hit.edu.cn | 设置 |

Google 快讯

管理快讯

创建 Google 快讯　　发送 HTML 格式的电子邮件。改换为文

搜索字词	类型	频率
	新闻	一天一次

您的 Google 快讯

搜索字词	类型	频率
信息检索 哈尔滨工业大学	新闻	一天一次
哈尔滨工业大学	新闻	一天一次
马祖光	新闻	一天一次

发件人: Google 快讯 [googlealerts-noreply@google.com] 发送时间: 2004-9-23 (星期四)
收件人: tliu@ir.hit.edu.cn
抄送:
主题: Google 快讯 - 哈尔滨工业大学

Google 快讯 - 哈尔滨工业大学

为“夕阳”续晚情

人民铁道报(需订阅) - Beijing,China

这趟旅游专列运行1万多公里，行程14天。游客将参观兵马俑、嘉峪关长城、天池、莫高窟、龙门石窟等景点。加旅游者以文化层次较高的老年游客居多，哈尔滨工业大学等黑龙江省内几所高校均有教授级的老人参团旅游。

拟提职干部公示名单

黑龙江日报 - Ha'erbin,Heilongjiang,China

... 曹殿富，现任齐齐哈尔市总工会主席、党组书记。男，汉族，中共党员，1948年7月生，1969年12月参加工哈尔滨工业大学经济管理专业大学毕业（函授），经济师。曾任建华机械厂工人、生产科调度员、调度组长，生服务公司副科长 ...

安徽新兴袁立:工大高新股价超跌严重

上海证券报 - Shanghai,China

工大高新(600701):控股股东哈尔滨工业大学高新技术开发公司背景雄厚,具有很强的科研开发实力,近年来公司在股东支持下大举向网络、高科技生物制药、基因工程等领域拓展。公司近年来还积极向生物制药和基因工程发展

发件人: Google 快讯 [googlealerts-noreply@google.com] 发送时间: 2004-9-24 (星期五)
收件人: tliu@ir.hit.edu.cn
抄送:
主题: Google 快讯 - 哈尔滨工业大学

Google 快讯 - 哈尔滨工业大学

工大首创:低价超跌科技股一飞冲天

金融界 - Beijing,China

工大首创该股流通盘9150万股，目前股价仅5元多，是两市科技股中价位最低的品种之一。该公司是计算机软件业的一只生力军，哈尔滨工业大学直属公司哈工大八达集团入主之后，把其控股97.75%的优质资产哈尔滨工业软件工程有限公司 ...

查看此主题的所有报道

许继用先进技术打造客户满意度

新浪网 - Beijing,China

... 迫切需要调试、配网自动化产品，便与日立、东芝等公司合作，引进其成熟的配网自动化和调度自动化产品，时满足了客户的需求。近年来，许继先后与清华大学、哈尔滨工业大学、西安交大、华北电力大学、合肥工业大学 ...

查看此主题的所有报道

中国政治核心世代交替江下胡上

新浪网 - Beijing,China

... 但之前二十馀年都在基层部队工作。徐才厚从哈尔滨军事工程学院电子工程系毕业后，先后在东北某农场劳动炼，任过副指导员、干部处干事、副处长、省军区政治部副主任等职。李继耐毕业于哈尔滨工业大学工程力学系

发件人: Google 快讯 [googlealerts-noreply@google.com] 发送时间: 2004-9-25 (星期六)
收件人: tliu@ir.hit.edu.cn
抄送:
主题: Google 快讯 - 哈尔滨工业大学

Google 快讯 - 哈尔滨工业大学

马祖光：名利面前一盏灯

文摘报 - Beijing,China

马祖光是中国科学院院士、哈尔滨工业大学博士生导师，曾被黑龙江省政府授予特等劳动模范称号，获全国“五一”劳动奖章。2003年7月15日，马祖光不幸辞世。 1996年，科学家王大珩院士来到哈尔滨 ...

高校助学贷款调整实施机制

黑龙江日报 - Ha'erbin,Heilongjiang,China

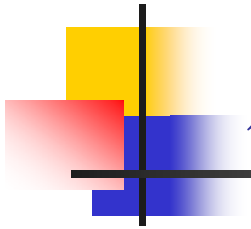
本报23日讯今天，中国银行黑龙江省分行与哈尔滨工业大学等3所在哈中央部属院校就国家助学贷款工程全面实达成协议，这标志着由国家多个部委共同实施的全国116所中央部属高校国家助学贷款工作在我省正式启动。 .

查看此主题的所有报道

华西村：共同富裕的典范

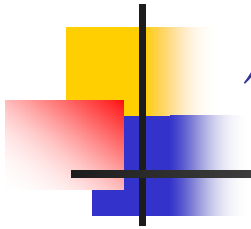
新华网 - Beijing,China

... 而是华西所培养、锻炼出的一大批各类人才。多年来，不仅吸引了海内外的２０００多名外地人才，本村的大生也无一例外地回到了故乡。今年刚大学毕业的２４岁的梅振华放弃了已考取的哈尔滨工业大学研究生录取和主等企业的邀请 ...



定义

- 什么是信息过滤？
  - 是指计算机根据用户提供的一个过滤需求(user Profile)，从动态变化的信息流(比如Web, e-mail)中自动检索出满足用户个性化需求的信息。
  - Profile：一组对用户过滤需求的描述，这种“profile”描述了用户长期的、稳定的兴趣爱好
- 近义术语
  - 信息的选择分发(Selective Dissemination of Information, SDI)，来自图书馆领域
  - 分流 (Routing)，来自Message Understanding
  - Current Awareness, 来自数据挖掘

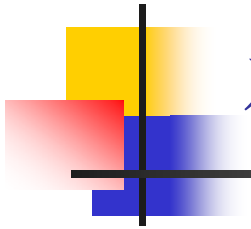


信息过滤的主要特点

- 无结构的或半结构化的数据
  - 电子邮件是典型的半结构化数据
    - 结构化的邮件头
    - 无结构的邮件正文
- 文本数据
- 对用户profile的描述
- 既可以用来屏蔽有害信息，也可以用来收集有益信息

信息检索和信息过滤

	信息检索 (IR)	信息过滤 (IF)
用户需求	“query”	“user profile”
信息流	静态	动态
需求	动态变化	静态
需要了解用户的情况	否	是

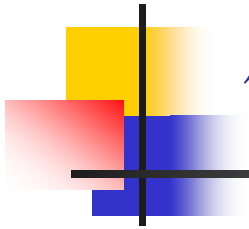


和其它概念的区别

- 和文本分类(Categorization)的区别
  - 分类系统中的类不会经常改变。
  - 相对而言，User Profile会动态变化
- 和信息抽取(Information Extraction,IE)
  - IF关心相关性
  - IE只关心抽取的那些部分，不管相关性

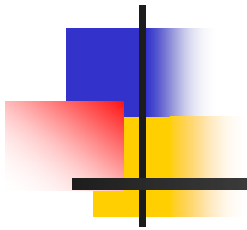
信息过滤的应用

- 克服重复查询
  - 网络信息是动态变化的，用户时常关心这种变化
  - 而在搜索引擎中，用户只能不断地在网络上查询同样的内容，以获得变化的信息，这花费了用户大量的时间
- 提供个性化信息服务
  - 对不同的用户采取不同的服务策略，提供不同的服务内容。
  - 实现“主动服务”，“信息找人”
- 实现有害信息的过滤
  - 反动言论，保护国家安全
  - 谣言，保护社会稳定
  - 色情内容，保护青少年身心健康



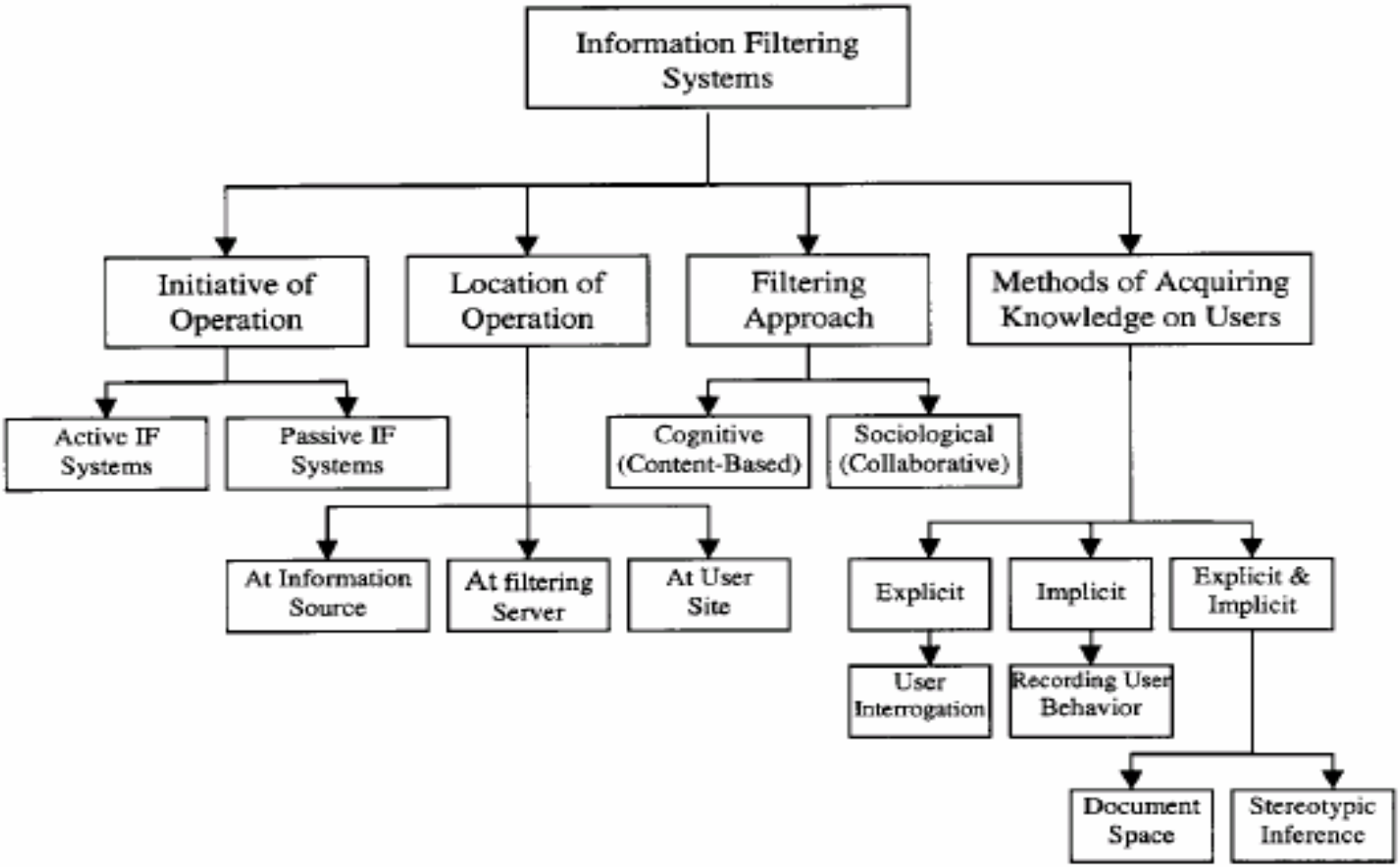
信息过滤的应用（续）

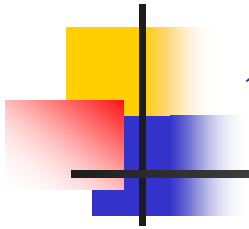
- 垃圾信息过滤
  - 垃圾邮件
  - 垃圾短信
- 推荐Recommendation
  - 根据不同用户之间需求的相关性推荐信息



信息过滤概述
——分类体系

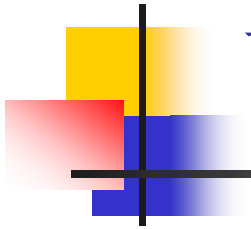
信息过滤系统分类示意图





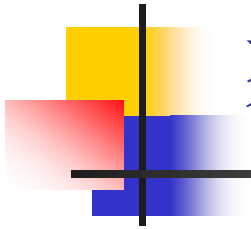
分类

- 主动，还是被动
  - 主动过滤
    - 主动向用户推送相关信息
  - 被动过滤
    - 比如垃圾邮件过滤
- 过滤操作的位置
  - 在信息源
  - 在过滤服务器上
  - 在客户端
    - 如：Outlook邮件过滤



两种主要的过滤方法

- 基于内容的信息过滤
  - 用户需求文档的形成及相关度的计算仅依靠信息的内容
- 协作信息过滤
  - 合作式信息过滤被定义为“通过掌握一个用户群体的诸个体间的相互联系及组织关系来实现的信息过滤方法。”
  - 许多人将合作式信息过滤的方法解释为“‘相似’用户之间相互合作的过程。”

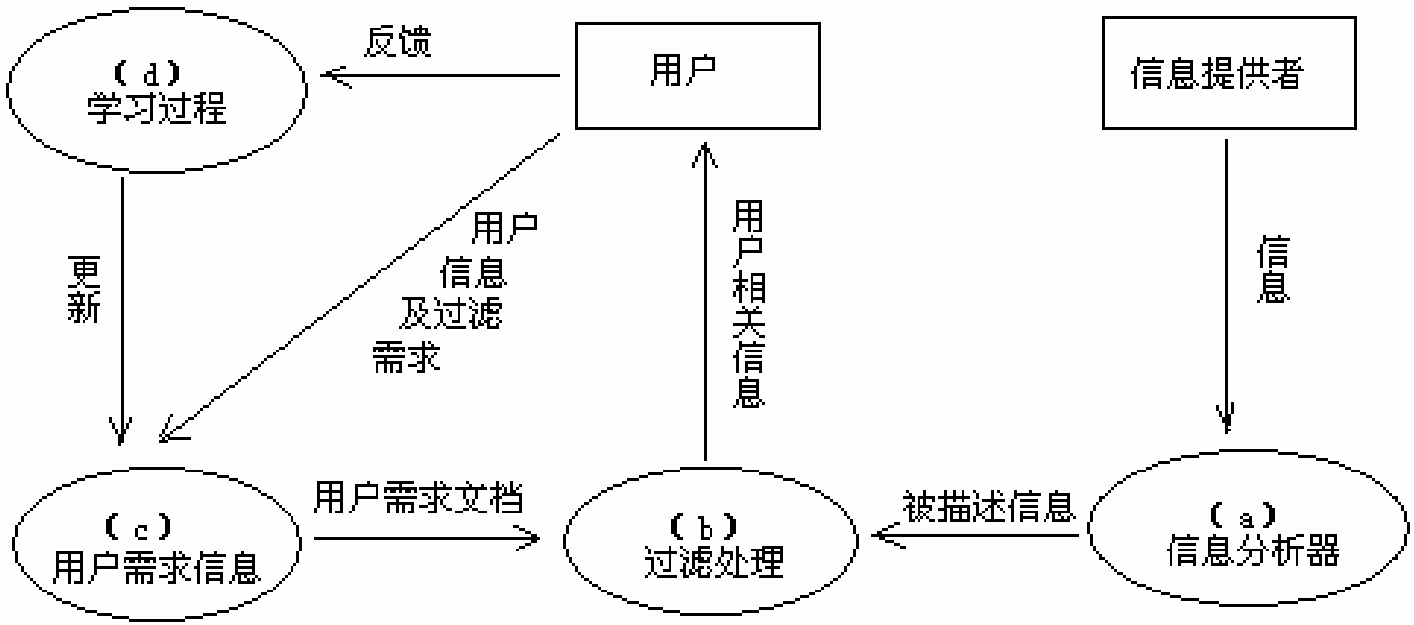


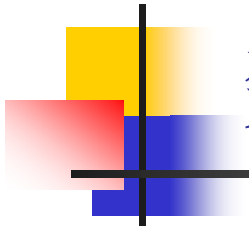
获取用户信息

- 显式获取用户信息
  - 用户直接填表
  - 用关键词表达用户过滤需求
  - 用文档集表达用户过滤需求
- 隐式获取用户信息
  - 无需用户直接参与，通过观察用户的动作行为判断用户需求
  - 用户阅读文档的时间可以作为衡量该文档相关度的一个指标。
  - 其他的一些用户行为——诸如用户是否保存、删除或是打印某篇文档也可以作为度量文档相关度的一个指标。

信息过滤概述
——信息过滤系统的组成

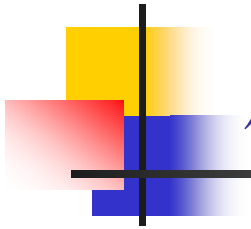
一般组成


（d）
学习过程
反馈
用户
信息提供者
更新
用户
信息
及过滤
需求
用户
相关
信息
信息
（c）
用户需求信息
用户需求文档
（b）
过滤处理
被描述信息
（a）
信息分析器



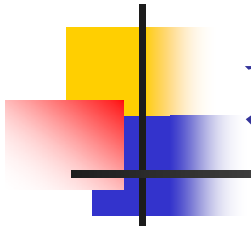
数据信息分析部分

- 靠近信息提供者
- 从信息提供者处获得或者搜集数据
- 文档分析或表示 (例如：Boolean Model, VSM, etc)
- 把这种表示传给过滤部分



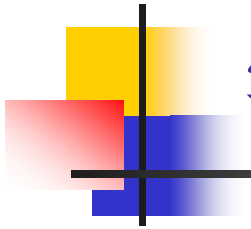
用户模型部分

- 收集用户信息 (显式或者隐式的)
- 构造用户profiles或者其它的用户模型 (rules, VSM, documents center)
- 把用户模型也传给过滤部分
- 用户模型必须适应文档表示



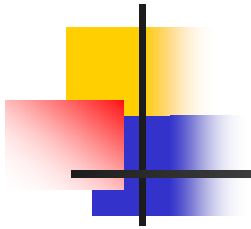
过滤部分

- 信息过滤系统 (IF system) 的核心
- 将用户的profiles和数据项的表示相匹配
- 最终决策可能是二元的或者一个概率值 (可以排序)
- 相关信息被送到学习部分 (feedback)



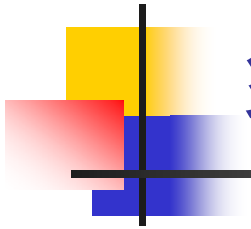
学习部分

- 为了提高过滤系统的性能
- 侦测用户兴趣的变化
- 更新用户模型



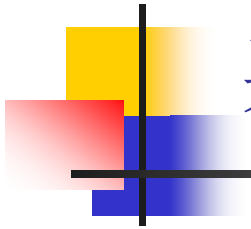
IF系统中的两种方法

- 基于统计的方法
- 基于知识的方法



统计的方法

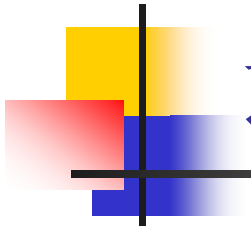
- 用户模型部分：
  - Profile是一个term的加权向量 (such as: VSM, LSI)
- 过滤部分：
  - 相互关系, Cosine measure
  - (naïve) Bayesian 分类器
- 学习部分：
  - 反馈



基于知识的方法

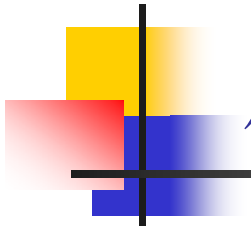
- 基于规则的方法和语义网方法
  - 知识 (if…then…)
  - WordNet (HowNet, etc)

信息过滤概述
——国内外发展现状和趋势



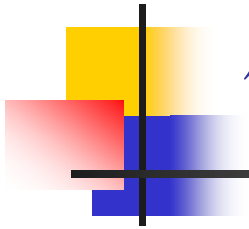
过滤模型

- 布尔模型
- 向量空间模型
- 概率推理模型
- 隐性语义标引



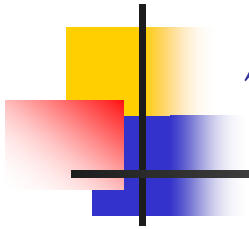
用户需求文档的学习和更新

- 用户的两类兴趣变化
  - 渐进式
  - 突发式
- 用户过滤需求的更新方法
  - 观察法
  - 用户提供相关反馈



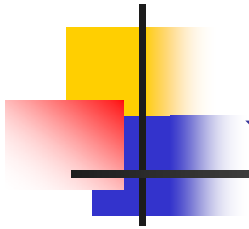
信息过滤的三个子任务(TREC)

- 分流（routing）
  - 用户需求固定、训练文本充足、无需给定相关度
- 批过滤（batch filtering）
  - 用户需求固定、训练文本充足、需要给定相关度
- 自适应过滤（adaptive filtering）
  - 用户需求变化、训练文本很少、不断调整相关度



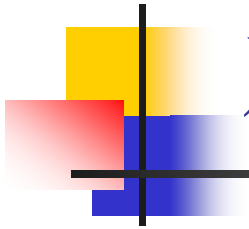
例子：

- 基于向量空间模型的文本过滤系统
  - 复旦大学
  - 2000年TREC-9
  - 自适应过滤平均准确率：26.5%；排名：3。
  - 批过滤平均准确率：31.7%；排名：1。



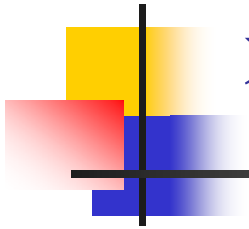
主要技术特点

- 向量空间模型
  - 训练、过滤两个主要过程
- 形成初始用户模式
  - 主题向量、正例特征向量、反例特征向量
- 自适应的阈值调整
- 自适应的模版修改
  - 主题向量、正例特征向量、反例特征向量



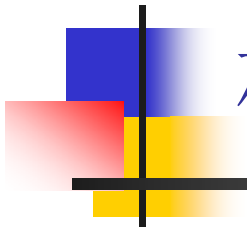
对信息过滤系统的评价

- 借鉴信息检索系统的一些评价方法
  - 准确率和召回率
- 尚未找到对信息过滤系统公认的、有效的评价方法
  - 信息检索领域中存在着对准确率和召回率这两种基本评价方法的质疑
  - 在自适应信息过滤系统该如何评价这一问题上，人们尚未达成共识
  - 从信息检索领域借鉴过来的评价方法只能对基于内容的信息过滤系统进行评价，无法处理其它的用户参数

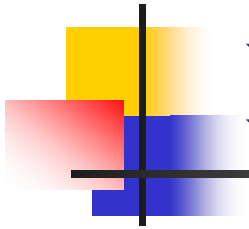


有IRLab特色的信息过滤

- 自然语言描述的“user profile”
- 词义消歧
- 指代消解
- 转述（paraphrasing）
- 融合丰富的NLP信息



模式匹配

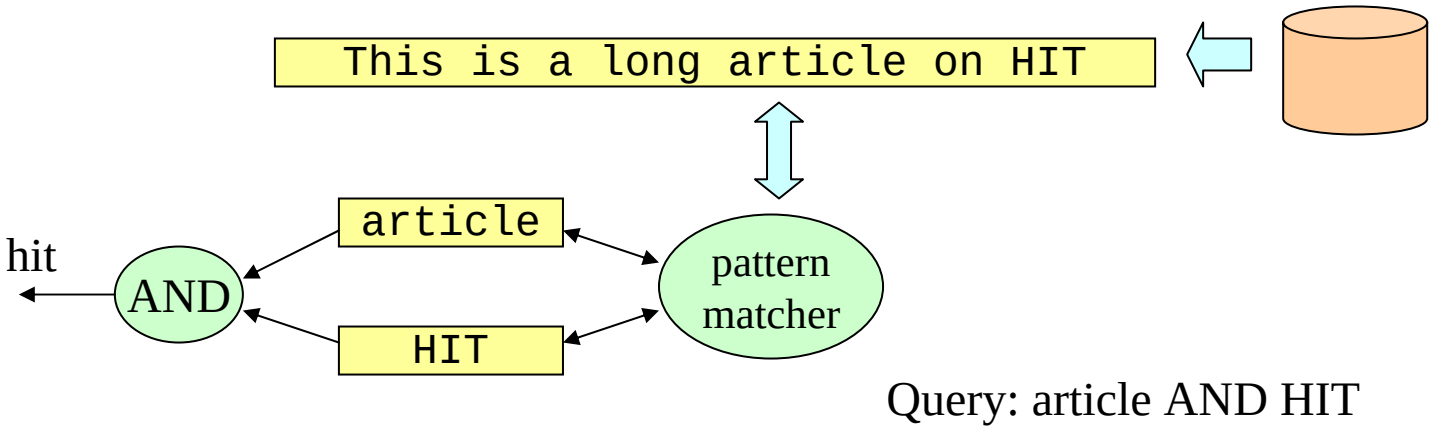


重提倒排文件 (Inverted File)

- 索引(倒排文件)导致了太多的存储和处理开销
- Posting表的后处理(合并和插入)开销越来越大
- 倒排索引对于以下情况比较好:
  - 相对静态的数据库
  - 检索速度要求非常高的大型数据库
  - 复杂的布尔查询, 邻接(proximity)搜索, stemming等复杂操作

全文扫描(Full Text Scanning)

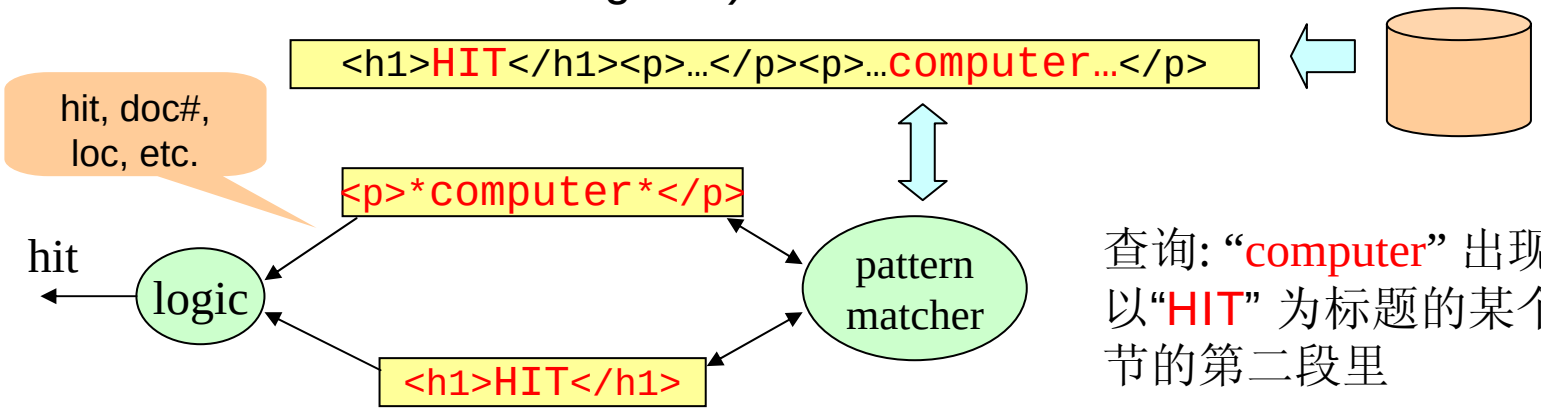
- 不维护索引表
- 直接在文本上进行搜索
- 需要模式匹配和逻辑组合来处理布尔条件



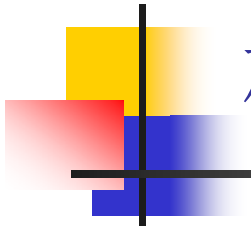
Query: article AND HIT

全文扫描

- 优点:
  - 不在索引方面花费时空开销
  - 适用于文本频繁产生和更新的动态环境
  - 在原始文本上完成搜索任务
    - 理论上，文本中的任何信息都可以被找到 (例如, 不需要停用词和Stemming操作)



- 缺点: 搜索速度慢，但对于小文档集合来说是可以接受的。 (例如个人文档集合)



模式匹配算法

- 基本的文本扫描操作 -- 字符串搜索:
  - 给定一个单独的模式*p* (搜索串)和一个输入串*s*, 如果*p* 是*s* 的子串就回答yes，否则回答no
- Brute force串匹配算法
- 快速串匹配算法
  - KMP算法
  - BM算法

模式匹配
——Brute Force

Brute Force算法

- 搜索串 (模式): $p_1p_2...p_m$
- 文本串: $s_1s_2...s_n$　(通常 $n >> m$)
- 将模式和m个字符的字串$s_ks_{k+1}...s_{k+m-1}$进行匹配，k从1到$n - m + 1$.
- 模式要么和子串匹配，要么找到一个位置发现二者不匹配

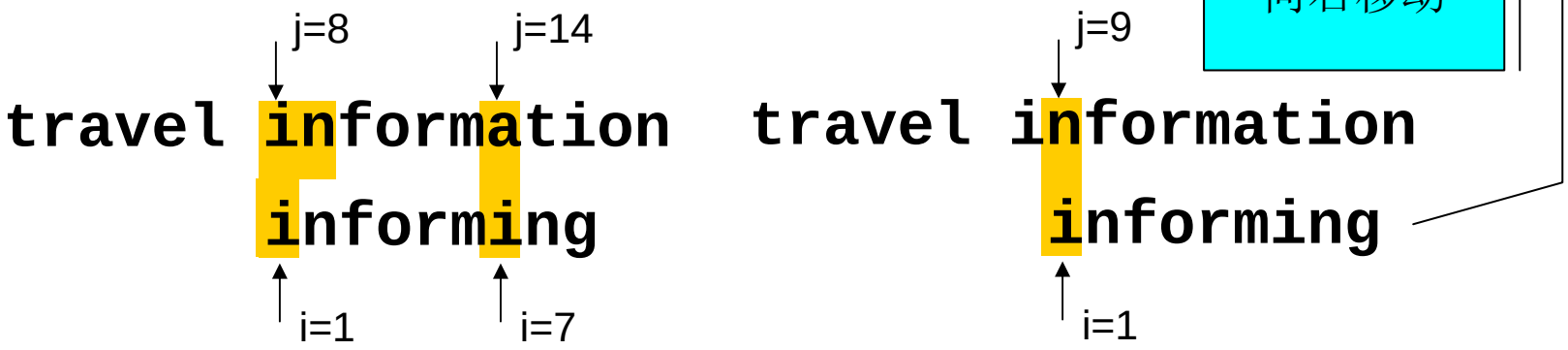
$$\begin{array}{c} \downarrow \\ p_1 \cdots p_i \cdots p_m \\ s_1 \cdots s_k \cdots s_j \cdots s_{k+m-1} \cdots \\ \uparrow \end{array}$$

Brute Force算法(2)

- **begin**
  - $i := 1$
  - $j := 1$
  - **while** $i \le m$ **and** $j \le n$ **do**
    - **if** $pi = sj$ **then begin** $i := i + 1$; $j := j + 1$ **end**
    - **else begin** $j := j - i + 2$; i :=1 **end**
  - **if** $i > m$ **then return** “yes **else return** “no”
- **end**

此循环可以更早地终止: $j \le n - m + 1$

最多 $n \times m$ 次



Brute Force算法—性能

	worst case	Best case	Between
text	aaaaaaa	abcdefg	abababa
keyword	aab	*xab*	abc
comparisons	*(n-m+1) * m*	*n-m+1*	
	15	5	11

在模式的末尾发现不匹配

在模式的开头发现不匹配

模式匹配
——KMP

KMP(Knuth-Morris-Pratt) 匹配算法

- Brute force算法较慢

Position	1	2	3	4	5	6	7	8
Text String	a	b	d	a	d	e	f	g
Pattern	a	b	d	f				

a b d f ←——— Brute force: 移动一个字符

a b d f ←—— 聪明的做法：移动三个字符

- 在位置4出现不匹配的现象，Brute force算法只移动了一位
- 由于前三个位置(a b d)都匹配成功了，移动一个位置不可能找到“a”，因为它已经被识别为 “b”

- 在文本的前缀被匹配成功后，你已经对文本串有了一些了解

KMP算法-基本思想

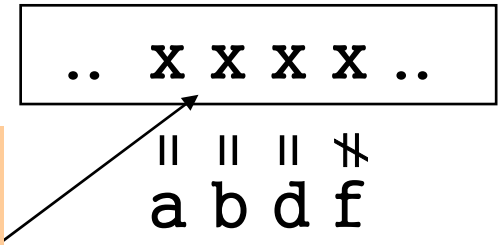
- 充分利用在一次失败匹配过程中获得的知识，避免重复比较已经知道的字符

位置:	1	2	3	4	5	6	7	8
文本串	a	b	d	a	d	e	f	g
模式	a	b	d	f				

- 关于文本的知识: a b d ？
- 需要在文本串中找到另一个 “a”
- 我已经知道 “a”不可能出现在下两个字符中, 因此可以向右移动三个字符

需要实现对模式(pattern)中的字符进行分析

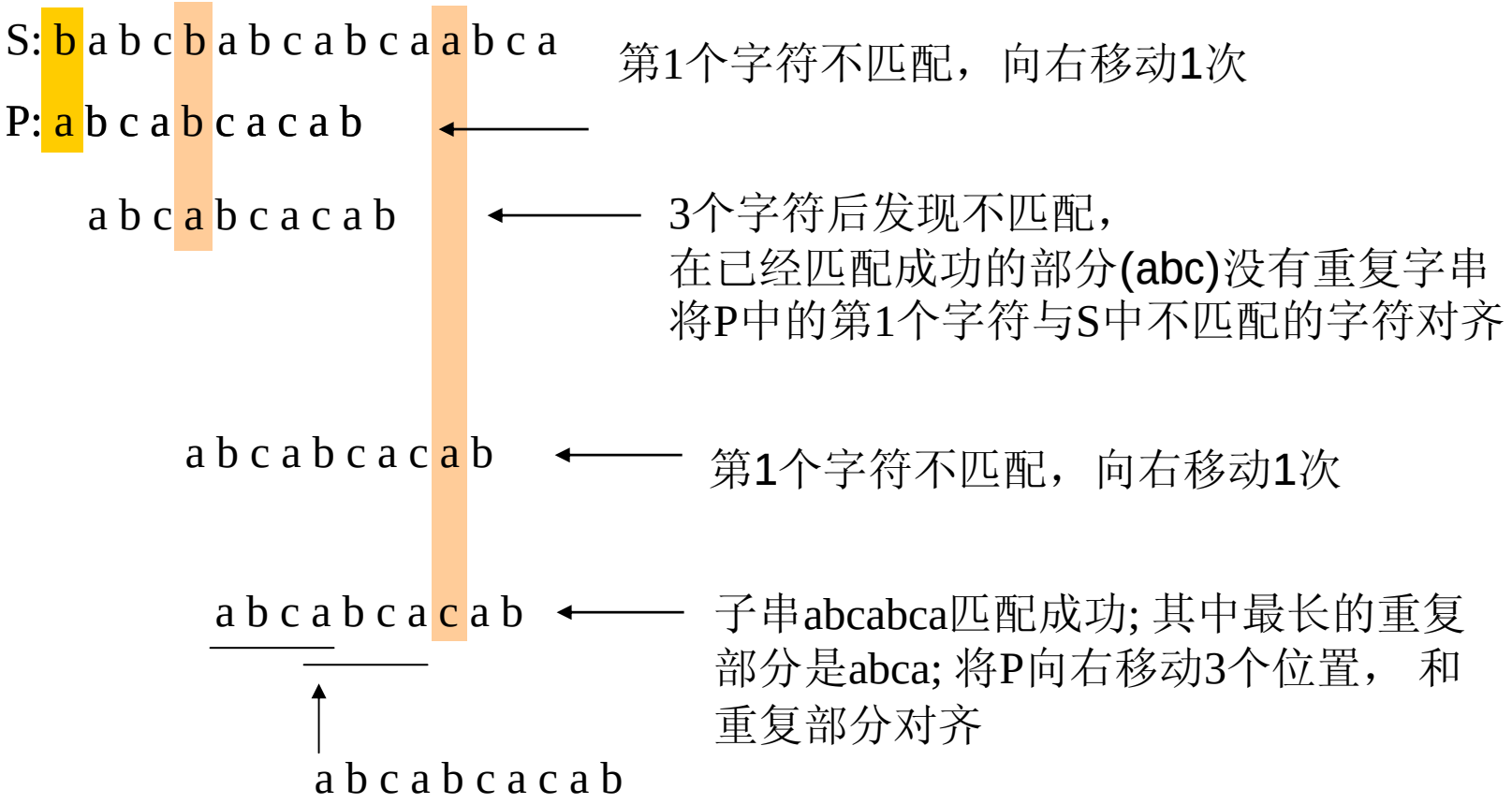
KMP算法 – 基本思想



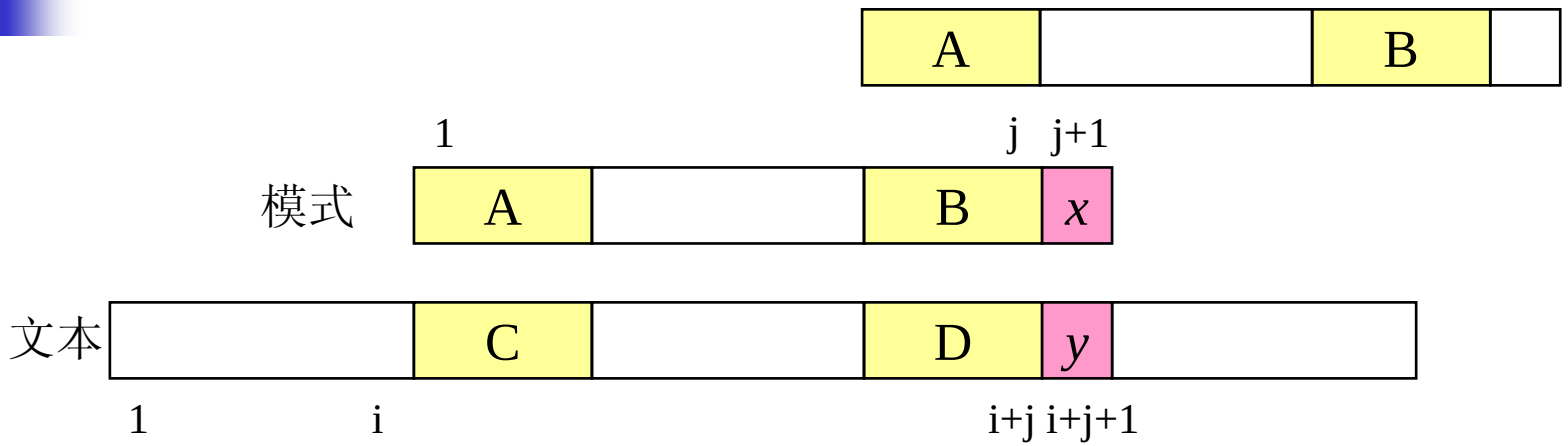
如果模式向右移动一个位置，它能够匹配成功吗？
通过检查模式的前缀a b d，就能知道肯定无法匹配，因为如果x和b匹配成功，就不可能和a匹配成功

要点是：我们能够通过预先对模式的分析获得知识：
如果 (在模式的位置1或2匹配失败) 则移动1个位置
如果 (在模式的位置3匹配失败) 则移动2个位置
如果 (在模式的位置4匹配失败) 则移动3个位置

KMP匹配算法 – 举例


S: b a b c b a b c a b c a a b c a
P: a b c a b c a c a b
第1个字符不匹配，向右移动1次
a b c a b c a c a b
3个字符后发现不匹配，
在已经匹配成功的部分(abc)没有重复字串
将P中的第1个字符与S中不匹配的字符对齐
a b c a b c a c a b
第1个字符不匹配，向右移动1次
a b c a b c a c a b
子串abcabca匹配成功; 其中最长的重复部分是abca; 将P向右移动3个位置， 和重复部分对齐
a b c a b c a c a b

KMP – 一般原则



- 从匹配成功的子模式中找出“能够相互匹配的最长的前缀和后缀”
- 通过对模式的分析，我们知道A=B，在匹配过程中知道A=C, B=D;
- 移动模式，让A和D对齐，从位置i+j+1处开始匹配

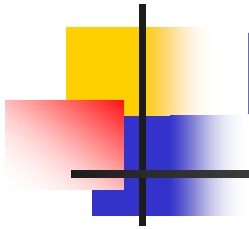
- 如果在模式 [1..*j*]中没有重复模式, 则可以直接移动 *j*个字符
- 有重复子模式的模式需要更多的时间去匹配，例如：aaaaaab

KMP算法 – Shift表

- 匹配失败时，决定移动多少个位置

关键词 字符	匹配的长度 重复子串	跳过的字符数
a	0	1
b	0	1
c	0	2
a	0	3
b	1	3
c	2	3
a	3	3
c	4	3
a	0	8
b	1	8

因为
abcab≠abcac
所以，找不到
“重复子串”



KMP算法 – 性能

- Shift表可以在O*(m)* 的时间复杂度下，通过对模式的分析而获得
  - m是模式长度
  - 对每个模式字符，需要计算当匹配失效发生时应该跳过几个字符
- KMP算法花费 O(*m+n*)的时间来解决此问题

模式匹配
——Boyer-Moore

Boyer-Moore算法

- 在理论上和实践上，BM都比KMP更快
- 它跳过了输入串中对匹配不可能有贡献的点
  - KMP对文本中的每个字符都进行匹配
  - 一般来说，BM匹配很少的字符

- 基本想法: 从右到左匹配输入串
- 我们比较 p_m 和 s_m，如果s_m 不在关键词中出现，那么从s的前m个字符中任何一个字符开始的字符串都不可能和p相比配，我们可以因此安全地向右滑动m个字符，从而避免m-1次不必要的匹配。

BM算法 -- Δ_1 Shift

- S: B A N A N A ^ C R E A M
- P: C R E A M

$N(s_m)$ 和$M(p_m)$不匹配，且N不在P中出现; 向右移动 $m=5$ 个位置, $\Delta_1 = 5$

- S: B A N A N A ^ C R E A M
- P: C R E A M

E和M不匹配，E (s_m) 在p中出现; 移动P使之相互对齐，$\Delta_1 = 2$

- S: B A N A N A ^ C R E A M
- P: C R E A M

如果有两个E，应该和最右边的E对齐，不能移动得太快

Δ_1 Shift表

- 如果模式的长度为 p 并且字母表的大小为q (对小写字母来说q=26), 我们需要$p \times q$的 Δ_1 shift表

α	1	2	3	4	5
M	1	2	3	4	Φ
A	1	2	3	Φ	1
E	1	2	Φ	1	2
R	1	Φ	1	2	3
C	Φ	1	2	3	4

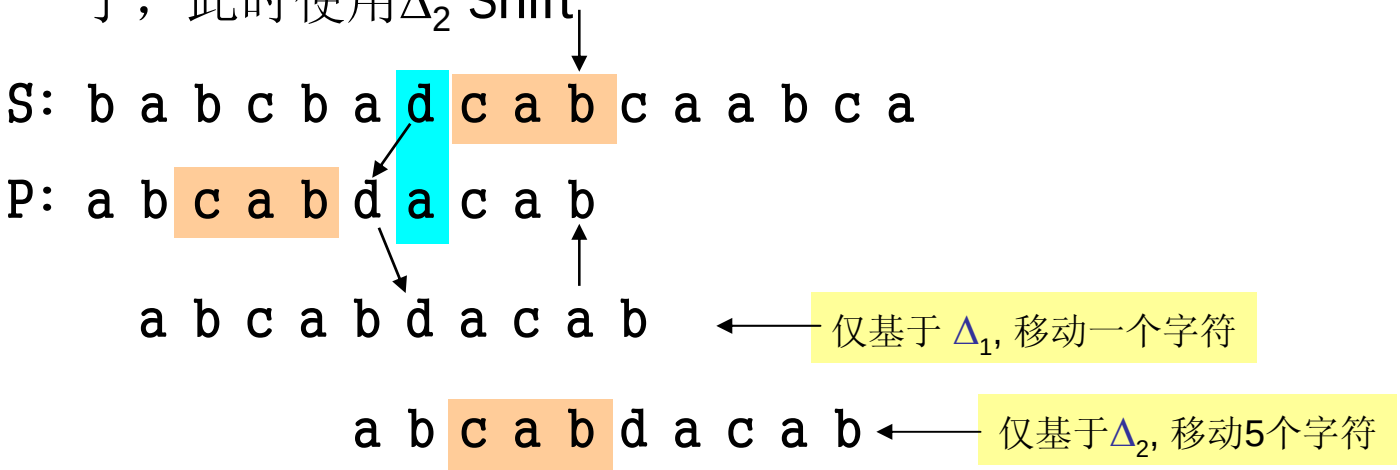
最右不匹配的
字母位置编号

α：字母表 – {C, R, E, A, M}

模式P=CREAM

BM算法 -- Δ_2 Shift

- p的尾部已经和s的某些子串相匹配，但再向左移动一位就不匹配了，此时使用Δ_2 Shift



- 如果应用Δ_1, 只能移动一个字符
- 和KMP相似, 我们可以知道已经匹配成功的模式后缀cab在模式的前面已经出现过, 因此移动1个字符肯定无法匹配成功
- 我们应该移动模式，用前面已经出现的模式模式和S中相应的文本对齐 $\Delta_2 = 5$
  - 问题: 如果 cab 不在模式中重复出现，怎么办？

Δ_2 Shift 的实现

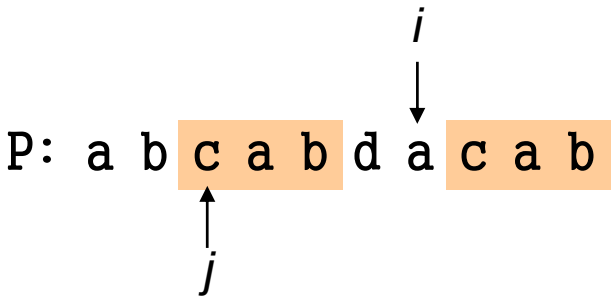
If mismatch at p[*i*]

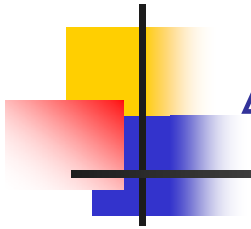
len = *m* - *i*

find largest *j* such that p [*j* .. (*j* + len - 1)]

= p [(*i*+1) .. *m*]

$shift_2[i] = i - j + 1$





Δ_1 和 Δ_2 的对比

- 考虑如下字符串

S: b a b c b a ***d*** **c a b** c a a b c a
P: a b **c a b c *a* c a b**

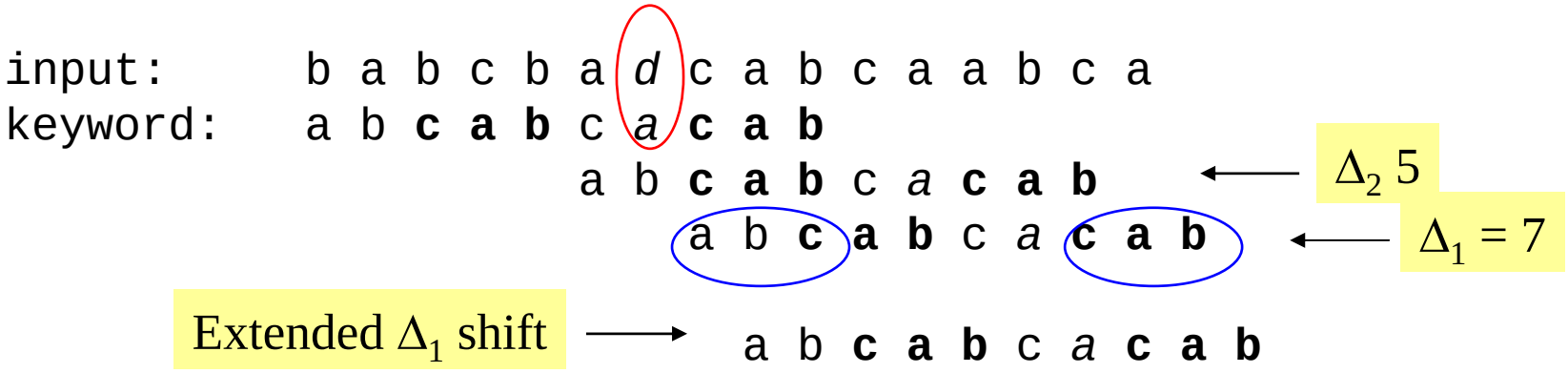
a b **c a b c *a* c a b** $\Delta_1 = 7$

a b **c a b c *a* c a b** $\Delta_2 = 5$

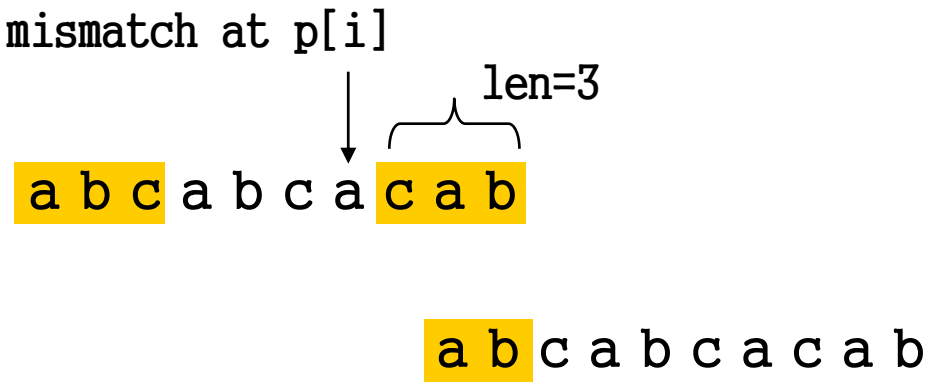
- $\Delta_1 = 7$, 因为不匹配的字符 d 不在剩余的模式中出现
- 如果基于重复子模式来考虑，则$\Delta_2 = 5$
- 我们需要同时计算出 Δ_1 和 Δ_2 ，并使用最大的shift

BM算法 – 扩展的 Δ_1 Shift

- BM算法使用两个shift中最大的shift来决定最终移动的位数
- 如果应用 Δ_1 shift, 发生不匹配, 进一步移动到模式P的头和尾恰巧匹配的位置



扩展 Δ_1 Shift算法



If mismatch at p[*i*] and Δ_1 is used

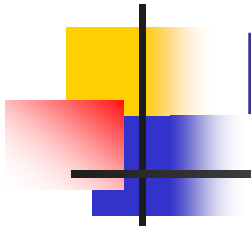
len = *m* – *i*

if p[1 .. len] = p[*i*+1 .. *m*] then Δ_1 unchanged

else find *j* such that p[1 .. *j*] = p[*m*-*j*+1 .. *m*]

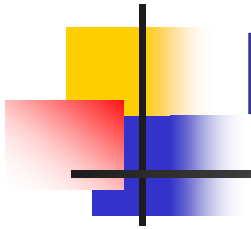
$\Delta_1 = \Delta_1$ + len - *j*

longest prefix in p[1..len] that matches the suffix of p[*m*-len+1..*m*]



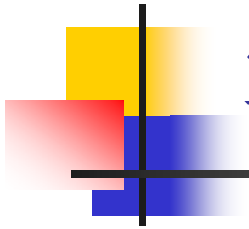
BM算法 – 小结

- 算法维护两个shift 表 (Δ_1 和 Δ_2 shift):
  - Δ_1-shift 表根据字母表和模式计算
  - Δ_2-shift 表根据模式计算
- 对于每一次匹配失效，从两张表中找出最长的shift应用
- 扩展的Δ_1-shift



BM算法 – 性能

- 当模式中重复部分很少时，效率最高 (类似KMP)
- 需要比较的字符数比输入字符串的长度*n*要小
- 很难应用于多关键词匹配 (KMP也一样)



算法选择: 启发式

- 这里介绍了一组经典的文本搜索算法，文献中还有更多的介绍
- 如果关键词的长度很小(1-3 characters), 可以使用Brute force算法
- 如果字母表很大，KMP算法是一个合适的选择
- 否则，特别是对于长文本来说, BM 是最佳选择

基本KMP

- BM通过检查不匹配的文本字符而提高了查找速度
- 相似的想法可以被用于基本的KMP方法中，进一步提高它的速度 (例如：将模式移动的距离最大化)

基本KMP:

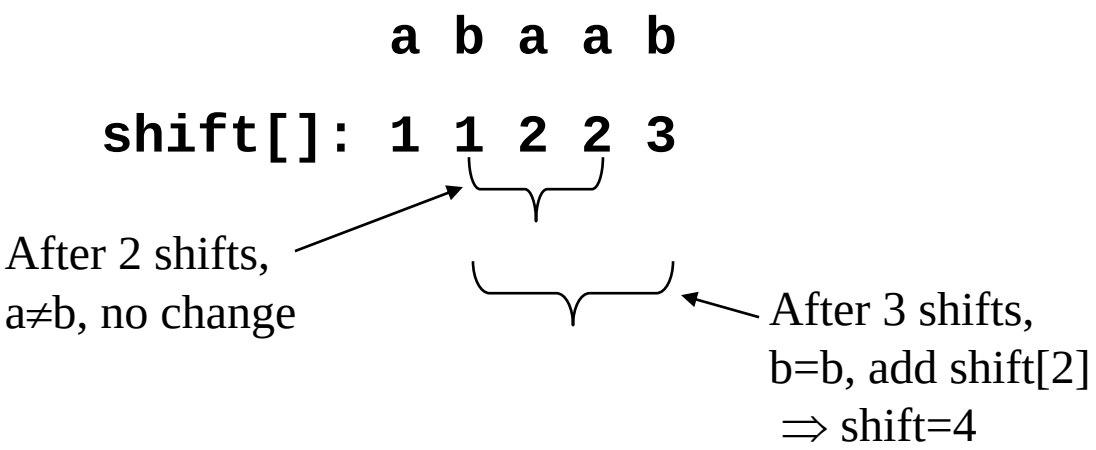
```
    a b a a b
shift[]: 1 1 2 2 3
```

```
. . a b x y z . . .
    a b a a b
      a b a a b
```

- 移动2个字符肯定不能匹配
- 在不匹配的x=a上叠代地应用KMP
- 再移动一个字符

一个实例

Basic KMP:

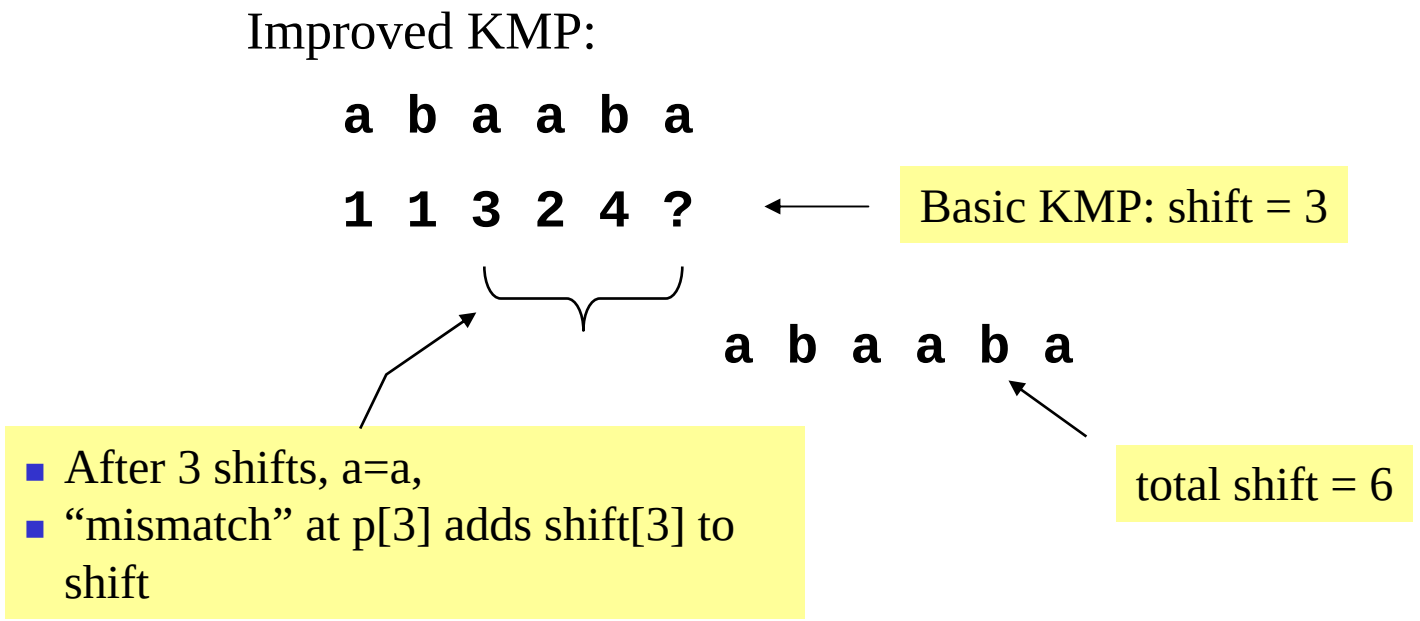


Improved KMP:

```
a b a a b
1 1 3 2 4
```

叠代Shifts的方法

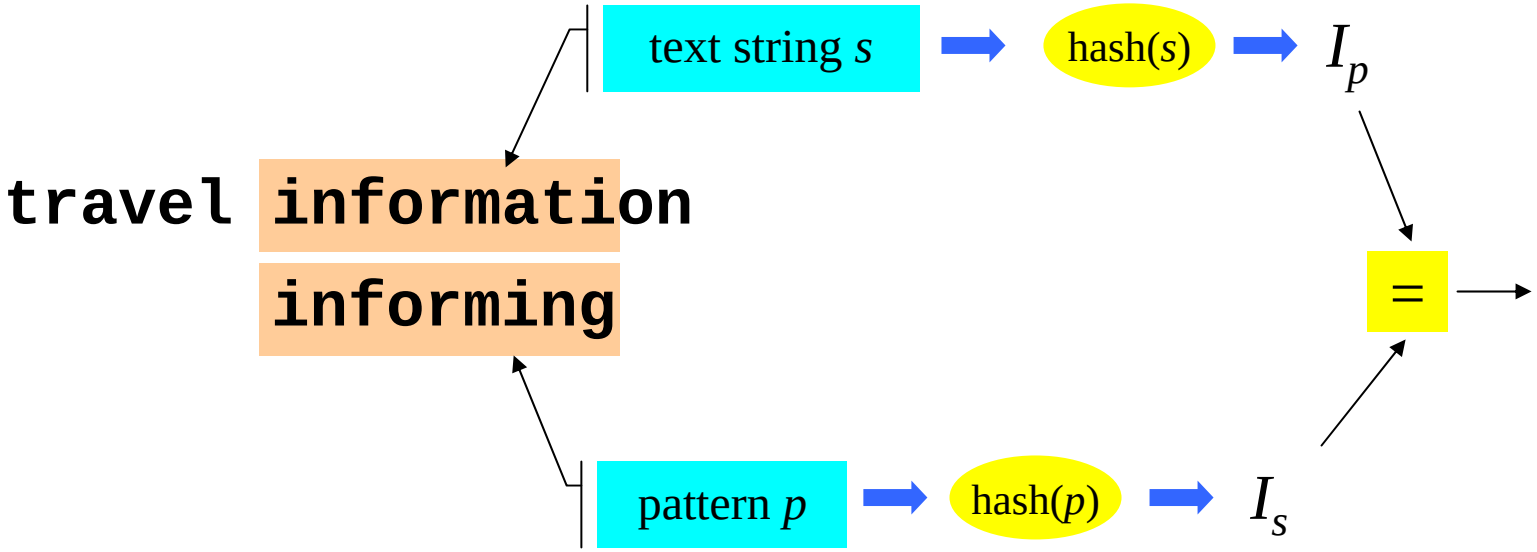
- 叠代的增加移动数目



模式匹配
——Karp-Rabin

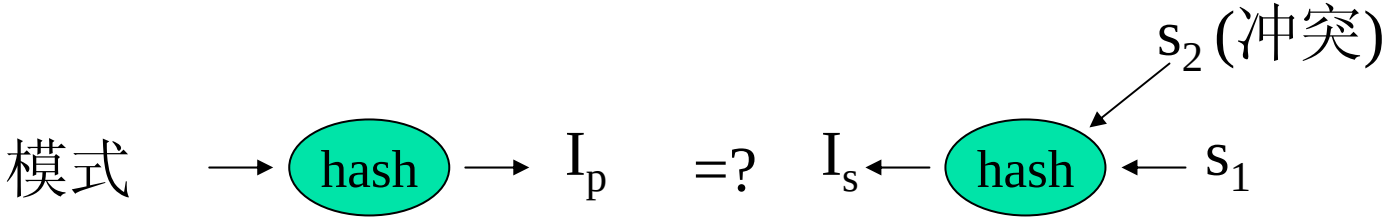
Karp-Rabin算法

- Karp-Rabin算法使用一个hash函数来降低将关键词和每一个m字符的字串相对比的开销



Karp-Rabin算法

- *h* 是一个hash函数，将 *m*个字符的字符串影射为一个整数
  - 如果$h(p_1p_2\ldots p_m) \neq h(s_ks_{k+1}\ldots s_{k+m-1})$，那么我们可以确信$p_1p_2\ldots p_m$不可能和$s_ks_{k+1}\ldots s_{k+m-1}$相互匹配
  - 如果$h(p_1p_2\ldots p_m) = h(s_ks_{k+1}\ldots s_{k+m-1})$，那么我们必须继续将 $p_1p_2\ldots p_m$和$s_ks_{k+1}\ldots s_{k+m-1}$.. 在字符级逐个匹配，从而保证没有误匹配发生



Karp-Rabin算法 (2)

- Hash函数将 m 个字符的比较降低为单个整数的比较, 但是需要一个快速的散列函数
- Karp和Rabin建议使用函数 $h(x) = x \bmod q$ ， q 是一个适当大的质数
- 对 m 个字符的字符串 $s_k s_{k+1} \ldots s_{k+m-1}$, 计算

  $x_k = s_k b^{m-1} + s_{k+1} b^{m-2} + \ldots + s_{k+m-1}$

  $x_{k+1} = (x_k - s_k b^{m-1}) b + s_{k+m}$

string = abcde
ascii(a) = 97
m = 4

hash("abcd") = $(97*2^3 + 98*2^2 + 99*2^1 + 100*2^0) \% q = 1466 \% q$

hash("bcde") = $(98*2^3 + 99*2^2 + 100*2^1 + 101*2^0) \% q = 1481 \% q$

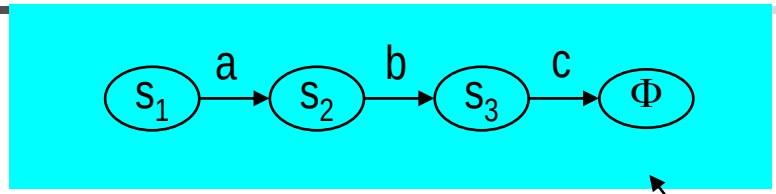
ascii(e)

hash("bcde") = $[(1466 - 97*2^3) * 2 + 101] \% q$

- 如果模式的hash值和文本的hans值每次都匹配，这是最坏的情况，时间复杂度为 $O(mn)$ ，几乎不可能出现
- 期望的时间复杂度是： $O(m+n)$

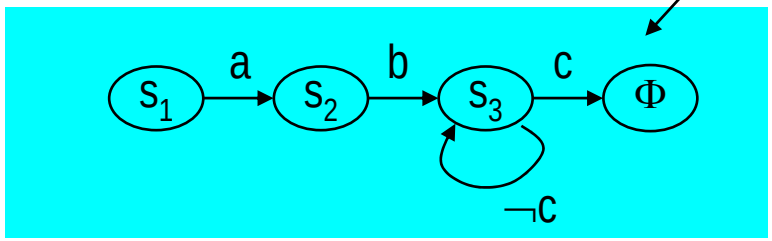
有限状态自动机

- abc



FSA

- ab*c

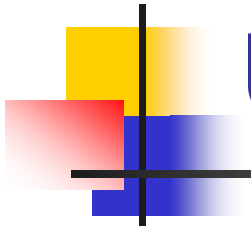


transition table

s_1	a	s_2
s_2	b	s_3
s_3	c	Φ

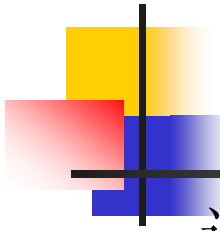
s_1	a	s_2
s_2	b	s_3
s_3	c	Φ
s_3	¬c	s_3

- What about a (b|c|d) e
  or a (b|c|d)* e ?
- 自动将正则表达式转换为状态转移表，并压缩状态转移表



UNIX grep 命令

- fgrep
  - 匹配固定字符串，使用类似KMP/BM算法
- grep
  - 匹配有限的正则表达式
  - .　　匹配任何字符
  - [a..c]　　匹配 a, b, 或者 c
- egrep
  - 匹配完备的正则表达式
  - p+　　p出现一次或多次
  - p?　　p出现0次或一次
  - p | q　　p 或者 q
  - ( )　　例如：(p | q)+



有效地模式匹配是否有用？

- 认为没有用的观点:
  - 如果文本存储在磁盘上，全文扫描需要将文件读入主存，并执行模式匹配: 读盘操作决定了时间开销
  - 倒排文档不需要模式匹配
- 认为有用的观点
  - 文本可以在主存中被缓存 (例如：今天的新闻)
  - 即时使用倒排文件，模式匹配仍然有用:
    - 加亮匹配部分
    - 如果倒排文件不包括词的位置，就需要利用模式匹配技术确认词的位置
  - 信息推送: 大量的用户profiles (patterns/keywords) 被用来匹配少量的新到的文本 (e.g., emails)

模式匹配
——WM(多模式匹配)

算法基本思想 - WM算法

1994年， Sun Wu 和 Udi Manber 提出的快速的多模式匹配算法

通过SHIFT, HASH, PREFIX和POINTER四个表格完成匹配

字符集为**{S H E R}**

SHEE HERS

对于字符集中任意两个字符的组合，通过一定规则，散列成一个整数**h**，作为其在**SHIFT**表中的入口

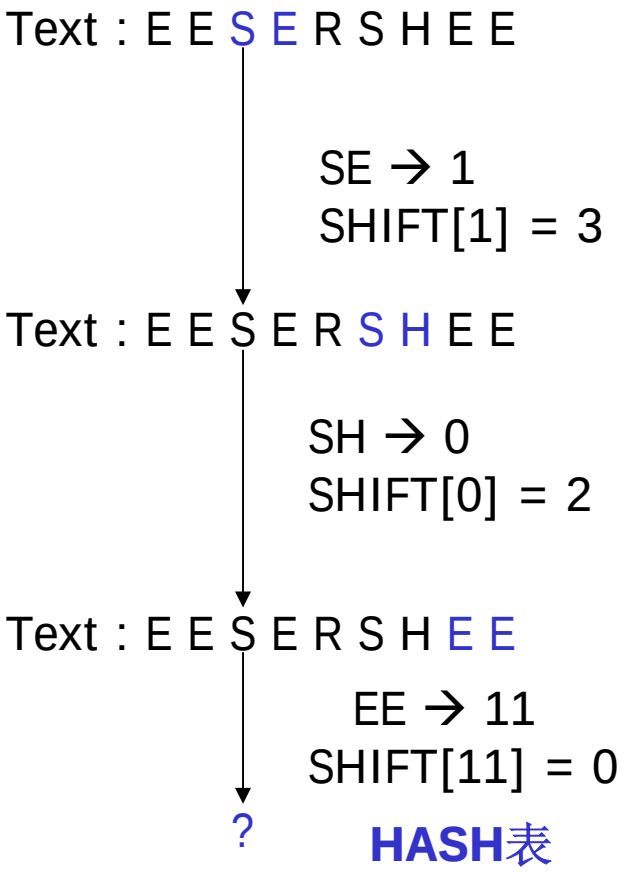
SHIFT表

可能的字符组合	移动距离
SH(0)	2
SE(1)	3
SR(2)	3
SS(3)	3
HS(4)	3
HE(5)	1
HR(6)	3
HH(7)	3
ES(8)	3
EH(9)	3
ER(10)	1
EE(11)	0
RS(12)	0
RH(13)	3
RE(14)	3
RR(15)	3

算法基本思想(2) - WM算法

SHIFT表

可能的字符组合	移动距离
SH(0)	2
SE(1)	3
SR(2)	3
SS(3)	3
HS(4)	3
HE(5)	1
HR(6)	3
HH(7)	3
ES(8)	3
EH(9)	3
ER(10)	1
EE(11)	0
RS(12)	0
RH(13)	3
RE(14)	3
RR(15)	3



算法基本思想(3) - WM算法

HASH表

可能的字符组合	HASH 值
SH(0)	0
SE(1)	0
SR(2)	0
SS(3)	0
HS(4)	0
HE(5)	0
HR(6)	0
HH(7)	0
ES(8)	0
EH(9)	0
ER(10)	0
EE(11)	0
RS(12)	1
RH(13)	2
RE(14)	2
RR(15)	2

POINTER表

0	SHEE
1	HERS

PREFIX表

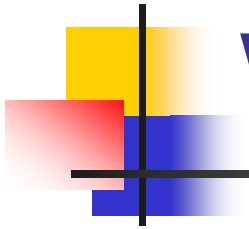
PATTERN	前缀
SHEE(0)	SH
HERS(1)	HE

Text : E E S E R S H E E

计算前缀**text_prefix = SH**的散列值

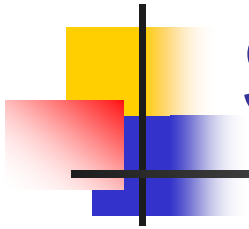
EE → h(11)

SHIFT[11] = 0 ⟶ HASH[11] = 0



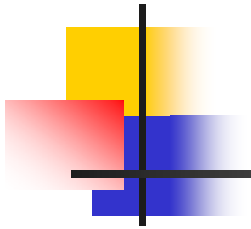
WM

- 多模式匹配
- m=最短模式的长度
- B=被考察的“块”的大小

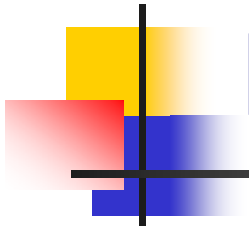


Shift表的构建

- 大小：B×字母表大小
- 字母表中每B个字符构成的字符串被散列为一个整数h，h就是SHIFT表的入口
- SHIFT中每一项的值决定在文本中出现某B个字符组成的字符串时pattern的移动距离
- 两种情况
  - X不在任何一个pattern中出现，SHIFT[h] =m-B+1
  - X在某些pattern中出现，且这些pattern中出现X的最右位置是q，则SHIFT[h]=m-q



- 如当前所考察的字符串在SHIFT表中映射的值为0，这意味着我们现在考察的text中的子串已经和某个或某些pattern的最后B个字符匹配。
- 但是我们并不知道是哪个或哪些pattern匹配了，为了避免将text中的子串和每一个pattern比较，我们采用散列的技术解决这个问题。



Hash表

- HASH[h]中存放着一个指针，这个指针同时成为POINTER表的入口

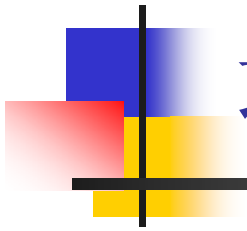
扫描文本寻找匹配的主要过程

1、根据文本当前正考察的B个字符计算其散列值h（从$T_{m-B+1}\cdots T_m$开始）

2、检查SHIFT[h]的值，如果SHIFT[h] > 0，那么移动SHIFT[h]个字符，返回第一步，否则，进入第三步。

3、计算文本中对应“前缀”的散列值，记为text_prefix.

4、对符合HASH[h] ≤ p < HASH[h+1]的每一个p值，检验是否存在PREFIX[p] = text_prefix.如果他们相等，就对text和pattern进行直接检查，pattern由POINTER[p]获得。



垃圾邮件过滤

垃圾邮件的定义

- 对于垃圾邮件(Spam)的定义，世界各国的技术专家和反垃圾邮件组织十分一致:
  - 批量发送的未经过收信人同意的电子邮件，发件人几乎不为每封邮件付出代价，而每个收件人为此要花费大量时间和金钱去处理这些邮件。从事此类活动的人员叫做垃圾邮件制造者（Spammer）。

2002年中国互联网协会对垃圾邮件给出了更加明确的定义

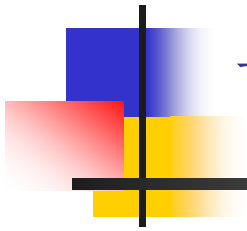
- 收件人事先没有提出要求或者同意接收的广告、电子刊物、各种形式的宣传品等宣传性质的电子邮件
- 收件人无法拒收的电子邮件
- 隐藏发件人身份、地址、标题等信息的电子邮件
- 含有虚假的信息源、发件人、路由等信息的电子邮件

垃圾邮件过滤技术面临的挑战

- 宁肯放过一千，不能错杀一个！
- Osterman Research最近一份研究显示，只有25%的用户对他们的垃圾过滤产品的判断能力表示满意

技术的现状及发展趋势

- 文本分类技术
- 多模式匹配技术
- 使用不同过滤技术的组合，如关键字、域和IP阻挡、黑名单、白名单或收信账户等



谢谢！

本课件参考了中科院计算所王斌研究员的相关文章和胶片