



# 高级算法 Advance Algorithms

骆吉洲  
计算机科学与技术学院



## 第1章 绪论



### 提纲

- 1.1 课程基本信息
- 1.2 随机算法的概念
- 1.3 为什么要学随机算法
- 1.4 课程内容
- 1.5 教学过程中的潜在问题
- \*.\* 实验一



### 1.1 课程基本信息



### Information about Instructor

- Instructor: 骆吉洲
- Room: 致知楼11
- Time: Wednesday, 3-4节  
Friday, 1-2节  
3-6周9-10, 格物楼213, 214 实验
- Office: 科学园科创大厦K1421室
- Email: [luojizhou@hit.edu.cn](mailto:luojizhou@hit.edu.cn)



### Information about Class

- 本科算法课程QQ群: **552377355**
  - 课程通知、须知
  - 课件材料
  - 作业布置和提交
  - 师生交流
- 分数构成 (拷贝=分数均摊)
  - 实验 40% (6个实验)
  - 书面作业 20% (无答案, 只更好, 没最好)
  - 期末考试 40%



群名称: 高级算法  
群号: 552377355



## 1.2 随机算法的概念



### 随机算法

**计算：** 给定计算模型上的可以机械执行的一系列操作步骤

**算法：** 满足确定性、精确性、终止性且具有输入和输出的计算

**随机算法：** 利用概率和统计方法确定算法某些执行步骤的算法



### Max-3-CNF问题

例

**输入：** 合取范式公式  $F = C_1 \wedge C_2 \wedge \dots \wedge C_m$ ,  $|C_i| \leq 3$

如:  $(x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge (x_3 \vee \neg x_1) \wedge (\neg x_1 \vee \neg x_3)$

**输出：** 一组变量赋值, 最大化值为真的析取式个数

#### • 随机算法

Random-Max-3-CNF(CNF)

1. For 对于CNF中的每个变量x Do
2. 随机为x赋值:  $\Pr[x=0]=1/2$ ,  $\Pr[x=1]=1/2$
3. 返回得到的赋值

这样就叫随机算法?  
怎么分析其优缺点呢?



### 随机性从何而来

#### 随机性的来源

- 这个世界上有真正随机的东西吗? 争论中....
- 我们认为投掷匀质硬币的结果是真正的随机源
- 计算机程序能有效模拟匀质硬币投掷吗?
- **伪随机数**-计算机算法产生的大致随机的数
  - 大致服从均匀分布 (基础)
  - 大致服从正态分布
  - ...
  - 大致服从任意分布
- 文献: TAOCP vol2 第3章
- 先修课程《随机计算》相关内容和实验



### 随机算法的特点

#### • 优越性

- 某些问题: 算法简单
- 某些问题: 时间复杂性低
- 某些问题: 同时兼有简单和时间复杂性低

#### • 随机性

- 同一实例上多次执行, 效果可能完全不同
- 时间复杂性的一个随机变量
- 解的正确性和准确性也是随机的



## 1.3 学习随机算法的动机



## 日常性

### 生活中处处有随机算法

#### 例1. 微信红包的随机算法

- 一笔金额为 $total$ 的钱
- 封装成 $num$ 个红包
- 每个红包金额范围 $[min, max]$
- 所有红包金额近似服从正态分布



- 怎么设计符合上述要求的红包分配算法呢?
- 如何论证红包算法满足上述要求呢?
- 要用到哪些工具?



## 简洁高效性

### 随机算法简洁而高效

#### 例2. 文档去重

- 30%的网页是重复
- 搜索引擎返回重复网页无意义
- 抄袭检测

#### ➢ 确定型算法

1. For  $i$ =第1个网页 To 第 $n$ 个网页 Do
2. For  $j$ =第 $i+1$ 个网页 To 第 $n$ 个网页 Do
3. If  $(i, j)$ 重复 Then 删除第 $j$ 个网页

如果 $n=10^{12}$ , 算法要需要多长时间呢?



## 简洁高效性

### 随机算法简洁而高效

#### 例2(续). 文档去重

#### 随机算法

文档: We are studing randomized algorithm...

3-gram: "We", "e a", "are", "re", "e s", ...

均匀产生 $27^3$ 个3-gram的随机排列

随机排列1

随机排列2

...

"We"的位置

"We"的位置

...

"e a"的位置

"e a"的位置

...

...

...

...

前 $b$ 个3-gram: 数字指纹1

数字指纹2

...

文档重复  $\Rightarrow$  数字指纹相同

对指纹建立倒排索引, 仅需比较每个倒排表中的文档是否重复

- 参数 $a, b, q$ 如何影响性能? 做个实验试试?
- 文献: Andrei Z. Broder. Identifying and Filtering Near-Duplicate Documents, 1998



## 必需性

### 简单高效随机算法处理大数据是必需的

#### 例3. 大数据可视化

#### 直观展示大数据

- 属性均值 柱状图
- 属性方差 折线图
- 变化趋势 圆饼图



- 算法1: 扫描数据准确计算统计属性值  
即使是线性时间算法, 也难以接受

- 算法2: 抽取样本, 在样本上计算统计属性  
简单、高效, 足够准确即可接受

随机算法是亚线性大数据算法的必要手段



## 创新必备性

### 重大创新必备知识之一

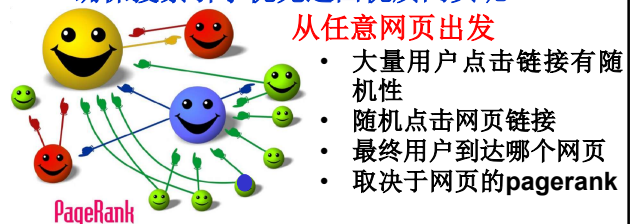
#### 例4. 网页间存在复杂的连接关系, 如何确保搜索引擎优先返回优质网页呢?



## 创新必备性

### 重大创新必备知识之一

#### 例4 (续). 网页间存在复杂的连接关系, 如何确保搜索引擎优先返回优质网页呢?

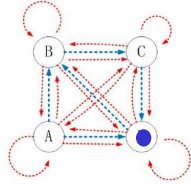




## 创新必备性

### 重大创新必备知识之一

例4 (续). 网页间存在复杂的连接关系, 如何确保搜索引擎优先返回优质网页呢?



网页链接关系用图表示

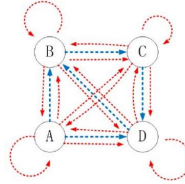
- 网页访问变成随机游走
- 亦即, 一个马尔科夫链
- PageRank是稳定分布  $\pi = \pi P$
- 怎么收集网页链接关系
- 怎么计算稳定分布



## 创新必备性

### 重大创新必备知识之一

例4 (续). 网页间存在复杂的连接关系, 如何确保搜索引擎优先返回优质网页呢?



Google公司的诞生和核心技术

- Larry Page & Sergey Brin
- 1998, Stanford, 博士
- 怎么收集网页
- 怎么高效计算pagerank

- 全网网页进行1次完整计算需要1周以上!
- 能否分布式地算? 理论、技术和方法



## 创新必备性

### 重大创新必备知识之一

例4 (续). 网页间存在复杂的连接关系, 如何确保搜索引擎优先返回优质网页呢?

如此重大的原创性研究工作起源于随机算法

文献: Brin S, Page L (1998) The anatomy of a large-scale hypertextual Web search engine. Comput Netw ISDN Syst 30:107-117

如今的分布式计算仍然在继续依赖于随机算法

文献: Ishii H, Tempo R (2010) Distributed randomized algorithms for the PageRank computation. IEEE Trans Autom Control 55:1987-2002

储备工具千千万, 源头创新也不难



## 广泛性

### 随机算法能应用于众多领域

例5. 数据降维

维数灾难

- 难索引
- 难查询
- 难分类
- 难挖掘
- 难学习
- .....

$(X, d_X)$



保距降维

能找到某个较小的 $\alpha, \beta$ 使得 $\forall x_1, x_2$ 有

$$\alpha D_Y(y_1, y_2) \leq D_X(x_1, x_2) \leq \beta D_Y(y_1, y_2)$$

随机生成矩阵 $A_{m \times n}$

$Y = AX$ 把 $m$ 维数据降维成 $n$ 维数据

保距效果还不错

纳尼?

Johnson-Lindenstrauss定理

在压缩感知(compressed Sensing)中有广泛的应用



## 广泛性

### 随机算法能应用于众多领域

例6. 科学实验

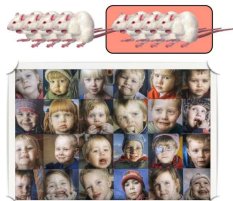
将实验对象分为两组

- 一组用疫苗
- 一组不用疫苗
- 对比发病率
- 检验疫苗是否有效

随机配置算法

- 为每个实验对象均匀随机独立地指定一个分组
- 根本不考虑实验对象的特征

这也行?



## 广泛性

### 随机算法能应用于众多领域

例7. 物理、化学实验

- 在给定的物理、化学条件下
- 特定现象、反应有何规律?



蒙特卡罗算法

- 建立现象、反应的数学模型
- 随机实验, 统计后作出结论



## 结论

### 我们需要学习随机算法

- 日常生活已离不开 (微信红包、区块链...)
- 简洁而高效
- 大数据处理需要随机算法
- 为难解问题求解提供可能性
- 源头创新必备知识之一
- 广泛应用于各个领域
- ...



## 1.4 课程内容



## 从集合的相似性连接谈起

### 集合相似性连接问题(Set Similarity Join)

- 输入: 集族  $\mathcal{A}$  阈值  $\alpha$
- 输出:  $\{(A, B) \mid A, B \in \mathcal{A}, \text{sim}(A, B) \geq \alpha\}$   
 $\triangleright \text{Sim}(A, B) = |A \cap B| / |A \cup B|$

### 应用

- 文档去重
  - $\triangleright$  每个网页视为该网页中的实词集合
  - $\triangleright$  网页相似  $\Leftrightarrow \text{sim}(A, B) \geq \alpha$ ,  $\alpha$  是由应用确定的
- 其他应用 (开动想象力或查阅资料)



## 相似度=概率

- $A = \{a, d\}, B = \{a, c, d\}$   $\text{sim}(A, B) = 2/3$
- $\text{sim}(A, B)$  = 同时取1的行数 / 两列之一取1的行数

	A	B		A	B		A	B		A	B
a	1	1	c	0	1	b	0	0	b	0	0
b	0	0	b	0	0	d	1	1	c	0	1
c	0	1	a	1	1	a	1	1	d	1	1
d	1	1	d	1	1	c	0	1	a	1	1
minHash	0	0		2	0		1	1		2	1

- $\text{minHash}_P(A)$  = 全体的随机排列  $P$  中首个属于  $A$  的行
- $\Pr[\text{minHash}_P(A) = \text{minHash}_P(B)] = \text{sim}(A, B)$
- 均匀产生排列来计算或估计概率也很费事!
- 能用其他方法来准确计算或估计 minHash 吗?



## 哈希模拟随机排列

- 用 hash 函数  $h_i: U \rightarrow \{0:|U|-1\}$  模拟第  $i$  次取定的随机排列

$h_1$	A	B	$h_2$	A	B	$h_3$	A	B	$h_4$	A	B
0	a	1	0	c	0	0	b	0	0	b	0
1	b	0	1	b	0	1	d	1	1	c	0
2	c	0	2	a	1	2	a	1	2	d	1
3	d	1	3	d	1	3	c	0	3	a	1
minHash	0	0		2	0		1	1		2	1

### 问题

1. 若要较精确地估计概率, 哈希函数要满足何种性质
2. 计算过程能否更高效?



## 高效计算 minHash

	a	b	c	d
$h_1$	0	1	2	3
$h_2$	2	1	0	3
$h_3$	2	0	3	1
$h_4$	3	0	1	2

	$A_1$	$A_2$	...	$A_m$	$\in \mathcal{A}$
第1个排列 $h_1$	$\infty$	$\infty$			
第2个排列 $h_2$	$\infty$	$\infty$			
第3个排列 $h_3$	$\infty$	$\infty$			
第4个排列 $h_4$	$\infty$	$\infty$			
.....					
第 $n$ 个排列 $h_n$					

**HIT CS&E**

**高效计算minHash**

	A	B
a	1	1
b	0	0
c	0	1
d	1	1

	A <sub>1</sub>	A <sub>2</sub>	...	A <sub>m</sub> ∈ $\mathcal{H}$
第1个排列 $h_1$	0	0		
第2个排列 $h_2$	∞	∞		
第3个排列 $h_3$	∞	∞		
第4个排列 $h_4$	∞	∞		
.....				
第n个排列 $h_n$				

	a	b	c	d
$h_1$	0	1	2	3
$h_2$	2	1	0	3
$h_3$	2	0	3	1
$h_4$	3	0	1	2

因  $h_1(a)=0$   $a \in A_1$   $a \in A_2$   
 $\minHash(1,1) = \min(0,\infty)=0$   
 $\minHash(1,2) = \min(0,\infty)=0$

因  $h_1(b)=1$   $b \notin A_1$   $b \notin A_2$   
 不做任何修改

因  $h_1(c)=2$   $c \notin A_1$   $c \in A_2$   
 $\minHash(1,2) = \min(2,0)=0$

因  $h_1(d)=3$   $d \in A_1$   $d \in A_2$   
 $\minHash(1,2) = \min(3,0)=0$

**HIT CS&E**

**高效计算minHash**

	A	B
a	1	1
b	0	0
c	0	1
d	1	1

	A <sub>1</sub>	A <sub>2</sub>	...	A <sub>m</sub> ∈ $\mathcal{H}$
第1个排列 $h_1$	0	0		
第2个排列 $h_2$	2	0		
第3个排列 $h_3$	∞	∞		
第4个排列 $h_4$	∞	∞		
.....				
第n个排列 $h_n$				

	a	b	c	d
$h_1$	0	1	2	3
$h_2$	2	1	0	3
$h_3$	2	0	3	1
$h_4$	3	0	1	2

因  $h_2(a)=2$   $a \in A_1$   $a \in A_2$   
 $\minHash(1,1) = \min(2,\infty)=2$   
 $\minHash(1,2) = \min(2,\infty)=2$

因  $h_2(b)=1$   $b \notin A_1$   $b \notin A_2$   
 不做任何修改

因  $h_2(c)=0$   $c \notin A_1$   $c \in A_2$   
 $\minHash(1,2) = \min(2,0)=0$

因  $h_2(d)=3$   $d \in A_1$   $d \in A_2$   
 $\minHash(1,2) = \min(2,3)=2$   
 $\minHash(1,2) = \min(0,3)=0$

**HIT CS&E**

**高效计算minHash**

	A	B
a	1	1
b	0	0
c	0	1
d	1	1

	A <sub>1</sub>	A <sub>2</sub>	...	A <sub>m</sub> ∈ $\mathcal{H}$
第1个排列 $h_1$	0	0		
第2个排列 $h_2$	2	0		
第3个排列 $h_3$	1	1		
第4个排列 $h_4$	∞	∞		
.....				
第n个排列 $h_n$				

	a	b	c	d
$h_1$	0	1	2	3
$h_2$	2	1	0	3
$h_3$	2	0	3	1
$h_4$	3	0	1	2

因  $h_3(a)=2$   $a \in A_1$   $a \in A_2$   
 $\minHash(1,1) = \min(2,\infty)=2$   
 $\minHash(1,2) = \min(2,\infty)=2$

因  $h_3(b)=0$   $b \notin A_1$   $b \notin A_2$   
 不做任何修改

因  $h_3(c)=3$   $c \notin A_1$   $c \in A_2$   
 $\minHash(1,2) = \min(2,3)=2$

因  $h_3(d)=1$   $d \in A_1$   $d \in A_2$   
 $\minHash(1,1) = \min(1,2)=1$   
 $\minHash(1,2) = \min(1,2)=1$

**HIT CS&E**

**高效计算minHash**

	A	B
a	1	1
b	0	0
c	0	1
d	1	1

	A <sub>1</sub>	A <sub>2</sub>	...	A <sub>m</sub> ∈ $\mathcal{H}$
第1个排列 $h_1$	0	0		
第2个排列 $h_2$	2	0		
第3个排列 $h_3$	1	1		
第4个排列 $h_4$	2	1		
.....				
第n个排列 $h_n$				

	a	b	c	d
$h_1$	0	1	2	3
$h_2$	2	1	0	3
$h_3$	2	0	3	1
$h_4$	3	0	1	2

因  $h_3(a)=3$   $a \in A_1$   $a \in A_2$   
 $\minHash(1,1) = \min(3,\infty)=3$   
 $\minHash(1,2) = \min(3,\infty)=3$

因  $h_3(b)=0$   $b \notin A_1$   $b \notin A_2$   
 不做任何修改

因  $h_3(c)=1$   $c \notin A_1$   $c \in A_2$   
 $\minHash(1,2) = \min(1,3)=1$

因  $h_3(d)=2$   $d \in A_1$   $d \in A_2$   
 $\minHash(1,1) = \min(2,3)=2$   
 $\minHash(1,2) = \min(2,1)=1$

**HIT CS&E**

**用minHash判定相似性**

	A	B
a	1	1
b	0	0
c	0	1
d	1	1

2/3

	A <sub>1</sub>	A <sub>2</sub>	...	A <sub>m</sub> ∈ $\mathcal{H}$
第1个排列 $h_1$	0	0		
第2个排列 $h_2$	2	0		
第3个排列 $h_3$	1	1		
第4个排列 $h_4$	2	1		
.....				
第n个排列 $h_n$				

1/2

- $\Pr[\text{simHash}_p(A)=\text{simHash}_p(B)] = \text{sim}(A,B)$
- $\text{sim}(A,B) \approx \text{AB两列相等行数}/n$
- 问题:  $n$ 取何值才能让概率可靠地逼近 $\text{sim}(A,B)$ ?
- 问题: 能否建立false negative与 $n$ 之间的关系?
- 实验一: 用实验得出经验公式来回答上述问题

**HIT CS&E**

**推广**

- minHash的重要特征  
 $\text{sim}(A,B)$  越大, minHash取相同值的概率越高
- 能否推广到高维空间? Locality Sensitive Hash
  - minHash正是LSH的起源
  - 高维空间中点的位置越接近, Hash值相等的概率越高
  - $L_1, L_2, L_p$  Hamming距离的高维空间下具体怎么做?
  - 这种方法能达到的效果有上、下界吗?
- 我们的课程能否直接来回答上面提出的各种问题?
  - 暂时还不能
  - 因为我们需要足够的基础知识
  - 这正是本课程的主要目标



HIT CS&E

## What to Teach

**P问题**

- $O(n^{20})$ 也是高效算法吗?
- 用随机算法怎么样?

**NPC问题**

- 应用中出现NPC怎么办?
- 修改问题设置都可行吗?
- 近似求解都高效可行吗?
- 用随机算法怎么样?

专注于随机算法设计技术和分析工具而非具体领域的随机算法

HIT CS&E

## What to Teach

**NPC问题**

- 很多NPC问题都能找到高效的随机算法
- Full Polynomial Randomized Approximate Schema
- 消除算法中的随机性是不是可能得到NPC问题的P算法呢?
- 随机算法是研究P=? NP的重要工具
- 我们不强调复杂性类, 但会遇到很多研究实例
- 为理解后续课程内容打下基础

HIT CS&E

## 课程的整体目标

- 随机算法的概念和分类
- 随机算法的性能分析工具和方法
  - 基本方法 尾概率界 矩方法 球和箱子模型 鞅...
- 随机算法设计方法
  - 放大 抽样 舍入 骗过对手 去随机化...
- 高效随机数据结构
  - 哈希函数 随机跳表 Bloom Filter Treap....
- 利用随机算法系统求解问题的能力
  - 经典算法积累
  - 分析问题特征
  - 合理利用算法设计技术和实现

HIT CS&E

## 课程大纲

- 第1章. 绪论 (2学时)
- 第2章. 概率算法及其分类 (4学时)
- 第?章. 随机算法时间复杂度下界 (2学时)
- 第3章. 球和箱子模型 (4学时)
- 第4章. Chernoff界 (3学时)
- 第5章. 鞅 (3学时)
- 第?章. 近似算法 (10学时)
- 第6章. 抽样与舍入 (4学时)
- 第7章. 概率方法与去随机化 (4学时)
- 第?章. 代数指纹技术 (3学时)
- 翻转. 论文阅读与专题算法 (3学时)

HIT CS&E

## 参考教材

主要参考4部教材

- 《Randomized Algorithm》
- 《Probability and Computing》
- 《Design and Analysis of Randomized Algorithms》
- 《Randomized algorithms for analysis and control of uncertain systems》

QQ群提供这些教材的电子版

恰当地使用, 不要用作商用目的

HIT CS&E

## 教材(1)

“Randomized Algorithms”

作者: R. Motwani, P. Raghavan

出版社: Cambridge University Press

出版年1995

**Part I: Tools and Techniques**

- Introduction
- Game-Theoretic Techniques
- Probabilistic Methods
- Markov Chains and Random Walks
- Algebraic Techniques

**Part II Applications**

- Data Structures
- Graph Algorithms
- Geometric Algorithms
- Approximate Counting
- On-line Algorithms
- Other advanced Topics

HIT CS&E

## 教材(2)

**“Probability and Computing”**

作者  
Michael Mitzenmacher  
Eli Upfal

出版社  
Cambridge University Press

出版年2007



主要内容:

1. 事件与概率
2. 离散随机变量与数学期望
3. 矩与方差
4. 切尔诺夫界
5. 球、箱子和随机图
6. 概率方法
7. 马尔科夫链与随机游动
8. 连续分布与泊松过程
9. 熵、随机性和信息
10. 蒙特卡罗方法
11. 马尔科夫链的耦合
12. 鞅
13. 两两独立与通用散列函数
14. 平衡配置

HIT CS&E


## 教材(3)

**“Design and Analysis of Randomized algorithms”**

作者  
Juraj Hromkovič

出版社  
Springer

出版年2005



主要内容:

1. 绪论
2. 基础知识
3. 欺骗对手
4. 指纹技术
5. 成功率方法与随机抽样
6. 放弃见证
7. 优化和随机舍入

HIT CS&E

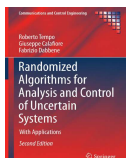
## 教材(4)

**“Randomized algorithms for analysis and control of uncertain systems”**

作者  
Roberto Tempo, Giuseppe Calaisore, and Fabrizio ...

出版社  
Springer

出版年2012



HIT CS&E

## 阅读材料-论文

- 可以选择QQ群给出的论文
  - pdf文档
  - 按章节划分
  - 根据文档标题可以轻松识别论文主题或主要工具
- 也可以自选论文
  - 选择范围: Focs, Stoc, Soda, ....
  - 论文选择要通过老师审查
  - 避免选择过于简单的论文

HIT CS&E

## 实验1:随机算法的优越性

### 实验目的

- 进一步理解随机算法的概念
- 进一步理解随机算法简洁性
- 进一步理解随机算法高效性
- 观察Hash函数个数对算法性能的影响
- 根据实验结果,得出最佳的经验参数设置
- 规范书写实验报告

HIT CS&E

## 实验内容

### 用minHash进行集合相似性连接

- 输入: 集族 $R=\{r_1, \dots, r_n\}$ ,  $c \in (0,1]$
- 输出:  $\{ \langle r, s \rangle \in R \times R \mid \text{similarity}(r, s) \geq c \}$   
 $\text{similarity}(r, s) = |r \cap s| / |r \cup s|$
- 要求: (不要求具体的实现语言)
  1. 实现Naïve算法
  2. 实现minHash算法
  3. 比较两种算法的运行结果是否一致
  4. 设置不同Hash函数个数, 查看运行效率差别
  5. 设置不同Hash函数个数, 比较返回结果差别
  6. 总结得出确保最佳效果的Hash函数个数
  7. 更换数据集, 重复上述结果, 得出一致的结论
  8. 撰写实验报告



HIT CS&E

### 实验数据

id	set
s <sub>1</sub>	{e <sub>1</sub> , e <sub>2</sub> , e <sub>3</sub> , e <sub>5</sub> }
s <sub>2</sub>	{e <sub>1</sub> , e <sub>2</sub> , e <sub>4</sub> }
s <sub>3</sub>	{e <sub>1</sub> , e <sub>3</sub> , e <sub>6</sub> }
s <sub>4</sub>	{e <sub>2</sub> , e <sub>4</sub> , e <sub>5</sub> }

数据的逻辑形式      数据的物理形式  
txt文件

数据的内存形式

HIT CS&E

### 实验步骤-minHash值计算与存储

哈希值      集合编号列表

哈希函数  $h$

链表个数等于值域大小       $i \in L_h(v) \Leftrightarrow h(A_i) = v$

HIT CS&E

### 实验步骤-算法描述

1. 计算minHash并组织计算结果  
/\*找出与A相似的集合,假定使用了H个哈希函数 $h_1, \dots, h_H$ \*/
2. For each set A Do
3.    根据 $h_1(A), \dots, h_H(A)$  定位得到H个链表 $L_1, L_2, \dots, L_H$
4.     $L = \text{Union}(L_1, \dots, L_H)$ ;
5.    For each  $B \in L$  Do
6.         $\text{count}(B) =$  B在 $L_1, L_2, \dots, L_H$ 出现的总次数
7.        if(  $\text{count}(B)/H \geq c$  ) 输出 (A,B)

HIT CS&E

### 实验步骤-4-7步优化

#### T引出的计算问题

合并

递增顺序

查找出现次数  $\geq T = cH$  的元素

HIT CS&E

### 例子

•  $T = 4$

1	10	5	13	15
3	13	7		
5	15	13		
10				
13				

结果: 13

HIT CS&E

### 序列合并算法

HeapMerger      MergeOpt

ScanCount      MergeSkip      DivideSkip

[SK04]  
[LLL08, BK02]

