

数据库系统概论新技术篇

大数据近似算法

魏哲巍

中国人民大学信息学院

2017年4月

大数据近似算法

- ❖ 研究背景与计算模型
- ❖ 随机采样算法
- ❖ 基于计数的近似算法
- ❖ 基于哈希的近似算法
- ❖ 研究成果简介



大数据近似算法

❖ 研究背景与计算模型

❖ 随机采样算法

❖ 基于计数的近似算法

❖ 基于哈希的近似算法

❖ 研究成果简介



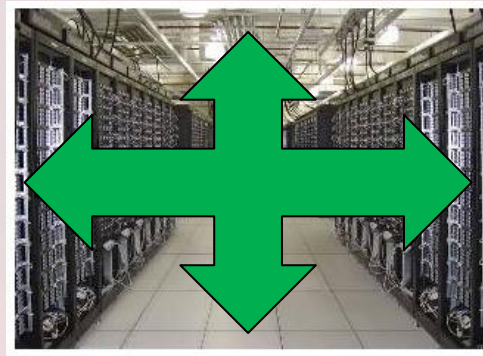
研究背景

❖ 处理大数据的两种思维模式：

- 扩展计算能力：。

❖ 方案一：扩展计算能力

- 超级计算机、分布式系统。。
- 问题：成本昂贵、能源消耗



研究背景

❖ 处理大数据的两种思维模式：

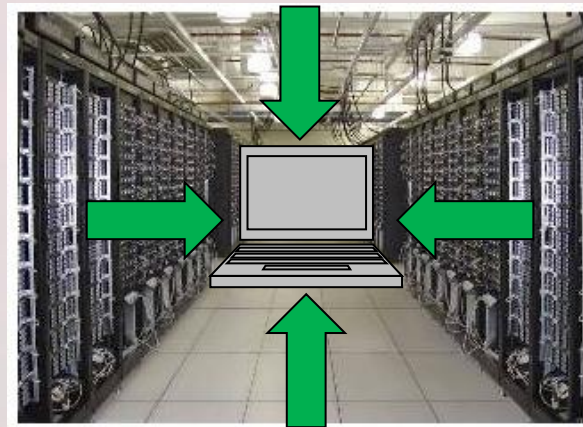
- 扩展计算能力：。

❖ 方案一：扩展计算能力

- 超级计算机、分布式系统。。
- 问题：成本昂贵、能源消耗

❖ 方案二：降低数据规模

- 通过引入近似/允许误差，将大数据变为小数据
- 优点：成本小，可与方案一结合
- 缺点：需针对特定问题设计特定算法



研究背景

- ❖ 大数据近似算法：利用**采样(sampling)**、**略图(sketch)**、**摘要(summary)**等技术，引入可控误差，解决由数据规模扩大带来的时间/空间/通讯量效率问题。
- 大数据通常有冗余，有价值的数据量可能很小；
 - 统计量从宏观上能反映实际问题的特质；
 - 现有的数据采集系统和分析算法也不可避免地会产生误差。



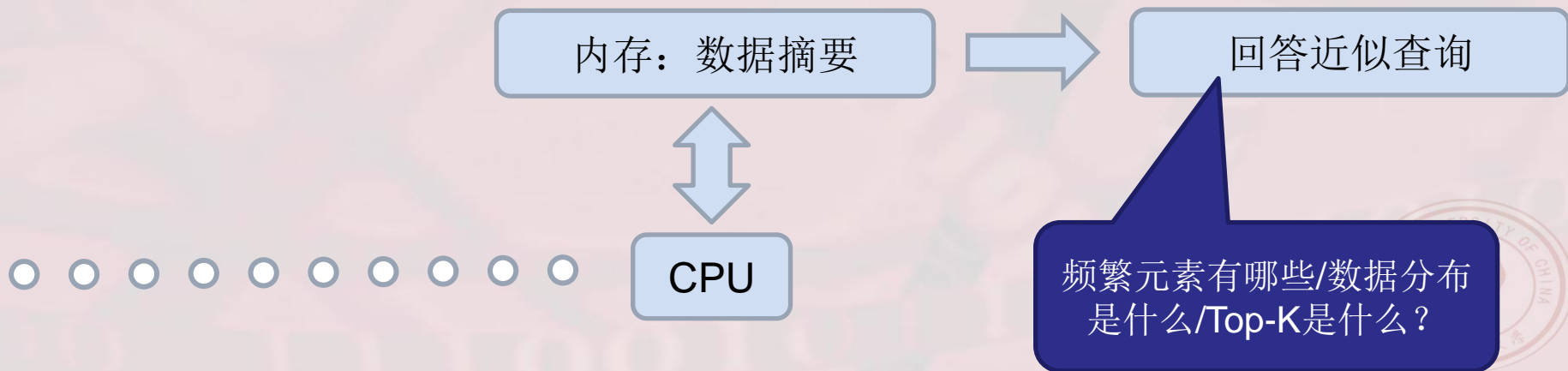
计算模型：数据流模型

❖ 数据流是一个由海量数据组成的数据序列

■ **Single Pass**: 每个数据最多访问一次

■ **Small Space**: 存储空间非常小

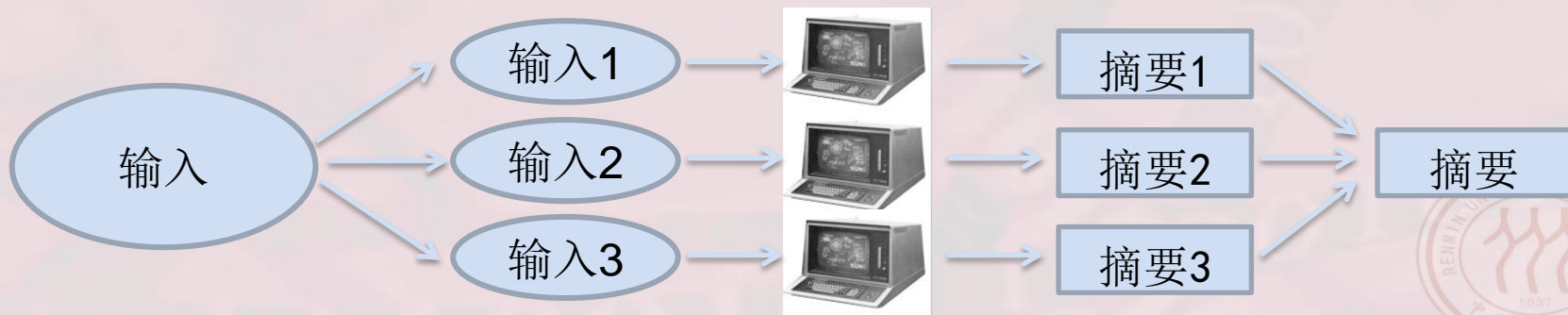
■ **Small time**: 更新(插入删除)速度快



计算模型：分布式模型

❖ 针对MapReduce、Hadoop等分布式计算平台

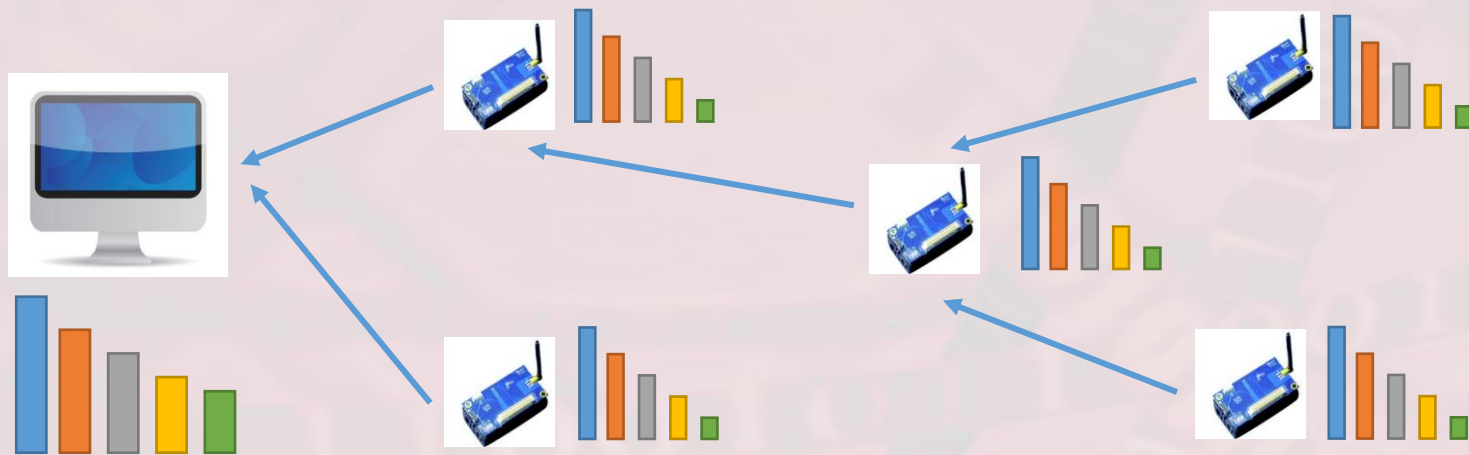
- 输入数据分布在多个节点
- 每个节点基于其数据，独立计算摘要
- 将多个摘要在主节点合并，回答关于原始输入数据的查询



计算模型：分布式模型

❖ 模拟传感器网络中的网络内聚合(In-network aggregation)

- 每个传感器独立观测数据(如湿度、温度、车流量等)，并计算摘要
- 摘要通过通讯依次传输合并
- 最后在主节点合并所有摘要，形成对聚合数据的估计



大数据近似算法

- ❖ 研究背景与计算模型
- ❖ 随机采样算法
- ❖ 基于计数的近似算法
- ❖ 基于哈希的近似算法
- ❖ 研究成果简介



随机采样算法

- ❖ 随机采样与大数定理
- ❖ 随机采样应用实例: **Online Aggregation**
- ❖ 水塘采样算法
- ❖ 随机采样的融合



随机采样

❖ 在元数据中随机选取一些数据作为“代表”

元数据:

9 3 5 2 7 1 6 5 8 4 9 1

$n=12$

采样数据:

9 5 1 8

$s=4$

❖ 回答近似查询:

- 平均值: 近似 $\text{avg}=(9+5+1+8)/4=5.75$, 真实 $\text{avg}=60/12=5$, 误差 $=(5.75-5)/5=15\%$
- 奇数个数: 近似 $\text{count}(\text{odd})=3*12/4=9$, 真实 $\text{count}(\text{odd})=8$, 误差 $=(9-8)/8=12.5\%$



随机采样与大数定理

- ❖ 无偏估计：对于**AVG, COUNT, SUM**等查询，随机采样给出的查询结果的期望就是真实结果
- ❖ 需要多少个采样？
 - 根据大数定理，采样数越多，结果越精确，误差越小
- ❖ **Probably approximately correct (PAC)**，可能近似正确保证：
 - 以**99.99%**的概率/置信度，误差不超过**0.01%**
 - **Chernoff**不等式：当采样数达到 $\frac{1}{\epsilon^2} \log \frac{1}{\delta}$ 时，以 **$1 - \delta$** 的置信度，保证误差不超过 ϵ
 - 大数据->小数据：当误差 ϵ 和置信度 δ 确定后，采样数与元数据大小无关
 - 采样数和误差平方成反比，**1%误差需要10000+个采样。**



随机采样应用: Online Aggregation

```
SELECT SUM(l_extendedprice * (1 - l_discount))  
FROM customer, lineitem, orders, nation, region  
WHERE c_custkey = o_custkey  
      AND l_orderkey = o_orderkey  
      AND l_returnflag = 'R'  
      AND c_nationkey = n_nationkey  
      AND n_regionkey = r_regionkey  
      AND r_name = 'ASIA'
```

该查询返回某公司亚洲地区由于退货导致的利润缩减



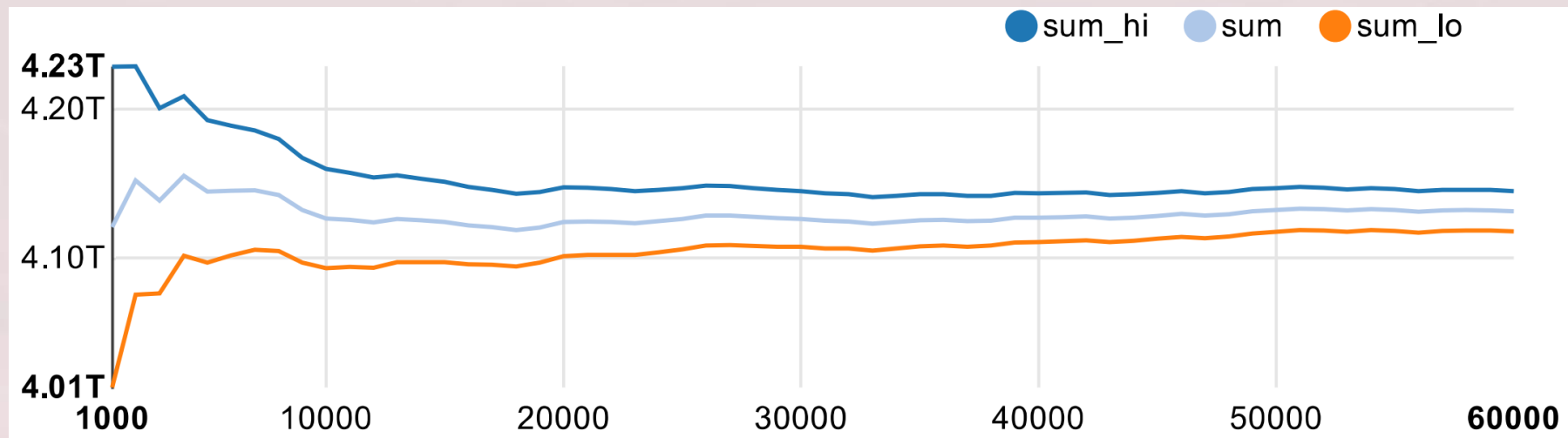
随机采样应用: Online Aggregation

```
SELECT SUM(l_extendedprice * (1 - l_discount))  
FROM customer, lineitem, orders, nation, region  
WHERE c_custkey = o_custkey  
      AND l_orderkey = o_orderkey  
      AND l_returnflag = 'R'  
      AND c_nationkey = n_nationkey  
      AND n_regionkey = r_regionkey  
      AND r_name = 'ASIA'
```

Error 0.01 Confidence 95%



随机采样应用: Online Aggregation

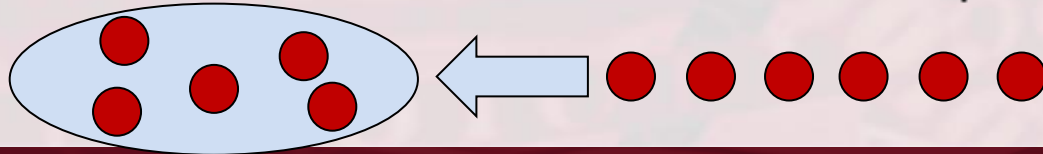


$$\Pr[\underbrace{\tilde{Y} - \varepsilon < Y < \tilde{Y} + \varepsilon}_{\text{置信区间}}] > \underbrace{0.95}_{\text{置信度}}$$



水塘采样算法

- ❖ 给定 N 个元素 $\{a_1, a_2, \dots, a_n\}$ ，希望随机选取 k 个元素，
 - 使得每个元素被选取的概率都是 $\frac{k}{n}$ ？
- ❖ 数据流模型：只能扫描所有元素一遍，可用空间为 k
- ❖ 水塘采样算法(**Reservoir Sampling**)
 - 选取前 k 个元素放入水塘
 - 对于 $i > k$ ，当扫描到第 i 个元素 a_i 时，以 $\frac{k}{i}$ 的概率选取 a_i
 - 若 a_i 被选取，从水塘中随机剔除一个元素，将 a_i 加入水塘

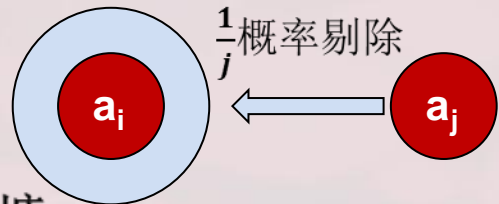


水塘采样算法分析

❖ 考虑 $k=1$ 的简单情况

❖ 第 i 个元素 a_i 被采样的概率是 $\frac{1}{n}$ 吗？

- 首先，在扫描到 a_i 时， a_i 必须被选取，概率为 $\frac{1}{i}$ ；
- 其次，在之后扫描 a_{i+1}, \dots, a_n 时， a_i 没有被剔除出水塘。
 - 扫描第 j 个元素 a_j 时， a_i 被剔除出水塘 $\Leftrightarrow a_j$ 以 $\frac{1}{j}$ 的概率被选取
 - 因此， a_i 仍留在水塘的概率为 $1 - \frac{1}{j} = \frac{j-1}{j}$



❖ 因此 a_i 被采样的概率为： $\frac{1}{i} \times \frac{i}{i+1} \times \frac{i+1}{i+2} \times \dots \times \frac{n-1}{n} = \frac{1}{n}$



水塘采样算法分析

- ❖ 当 $k > 1$ 时，也可以证明采样的随机性。
- ❖ 需要注意的是，不仅仅需要证明 a_i 被选取的概率是 $\frac{k}{n}$
- ❖ 还需要证明任意一个大小为 k 的子集被采样到的概率相等
- ❖ 正确的证明涉及到Fisher-Yates shuffle。

Google reservoir sampling

Web Images Maps More Search tools

About 17,800,000 results (0.21 seconds)

Ad related to reservoir sampling ⓘ

Reservoir Fluid Analysis - slb.com
www.slb.com/Schlumberger
Mercury-Free Reservoir Fluid Sampling and Analysis Services
Contact Us Core and PVT Lab Services
Testing for Life Article TerraTek Services

Reservoir sampling - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Reservoir_sampling
Reservoir sampling is a family of randomized algorithms for randomly choosing a sample of k items from a list S containing n items, where n is either a very large ...
You've visited this page many times. Last visit: 1/31/13

Reservoir Sampling - Department of Computer Science and ...
www.cse.cuhk.edu.hk/~taoyf/course/5705f10/lec7.pdf
Nov 8, 2010 - Lecture 7: Reservoir Sampling. CMSC 5705 Advanced Topics in Database Systems. Yufei Tao. Department of Computer Science and ...

algorithm - Reservoir sampling - Stack Overflow
stackoverflow.com/questions/2612648/reservoir-sampling
Apr 10, 2010 - to retrieve k random numbers from an array of undetermined size we ... I actually did not realize there was a name for this, so I proved and ...
You've visited this page 2 times. Last visit: 1/30/13

Algorithms Every Data Scientist Should Know: Reservoir Sampling ..
blog.cloudera.com/blog/.../hadoop-stratified-random-sampling-algorithm/
Apr 23, 2013 - Data scientists, that peculiar mix of software engineer and statistician, are notoriously difficult to interview. One approach that I've used over the ...

Gregable: Reservoir Sampling - Sampling from a stream of element
gregable.com/2007/10/reservoir-sampling.html
Oct 8, 2007 - If you don't find programming algorithms interesting, stop reading. This post not for you. On the other hand, if you do find algorithms ...

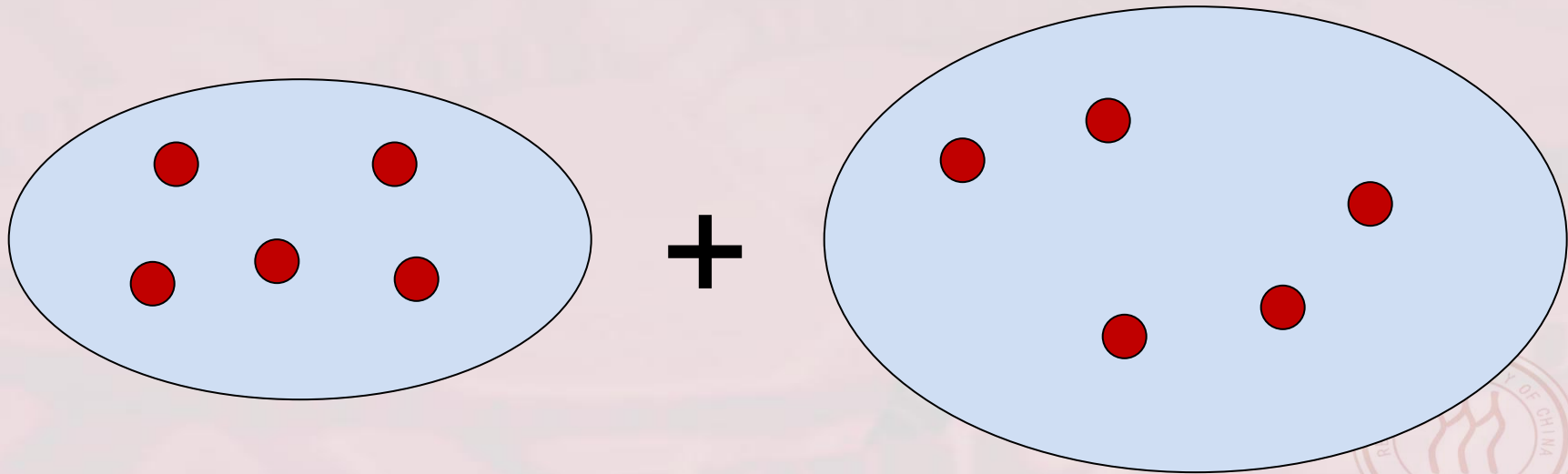
Reservoir Sampling - GeeksforGeeks | GeeksforGeeks
www.geeksforgeeks.org/reservoir-sampling/
Oct 18, 2012 - Reservoir sampling is a family of randomized algorithms for randomly choosing k samples from a list of n items, where n is either a very large or ...

Reservoir Sampling - K samples from an infinite stream
code-slim-jim.blogspot.com/2010/06/reservoir-sampling.html
Jun 17, 2010 - Question: You have a stream of infinite queries (ie Google searches). Describe how you would find K uniformly distributed samples and write ...

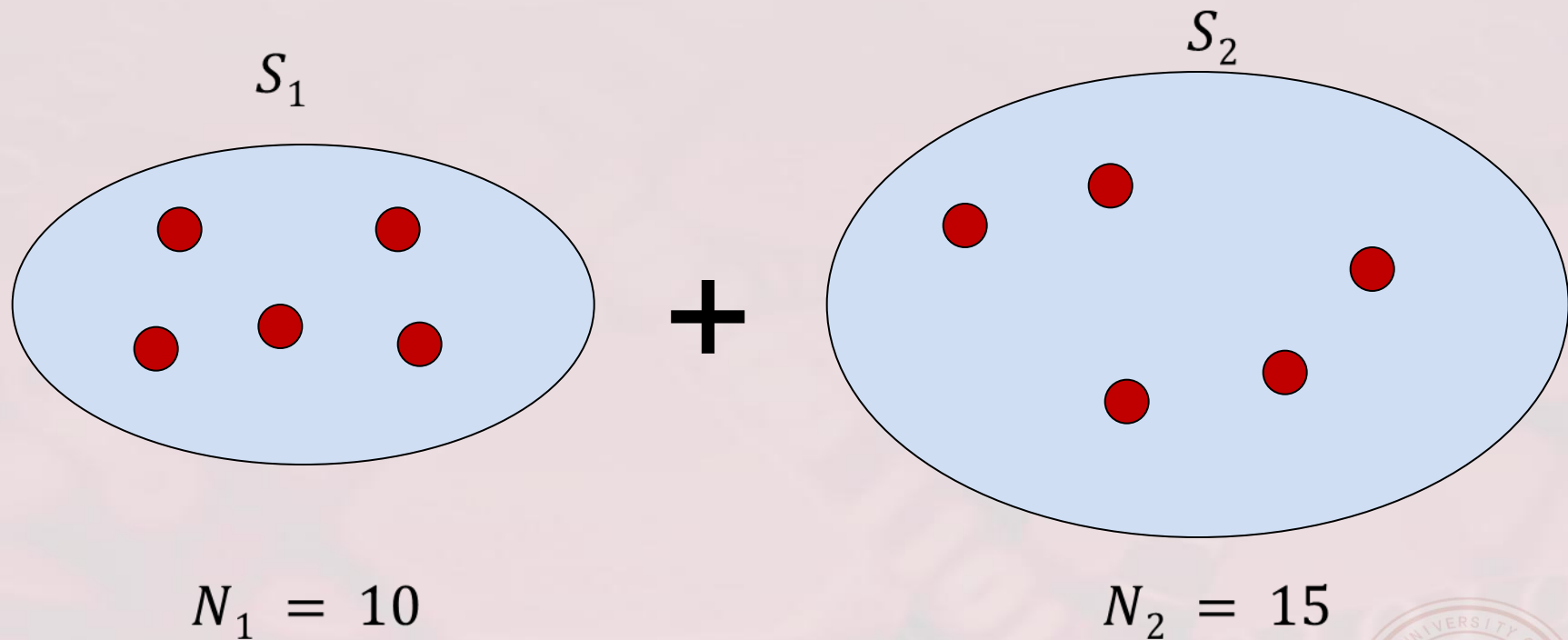
Icons: Green checkmark, Red X, Green checkmark, Red X, Red X, Red X, Red X, Red X

随机采样的合并算法

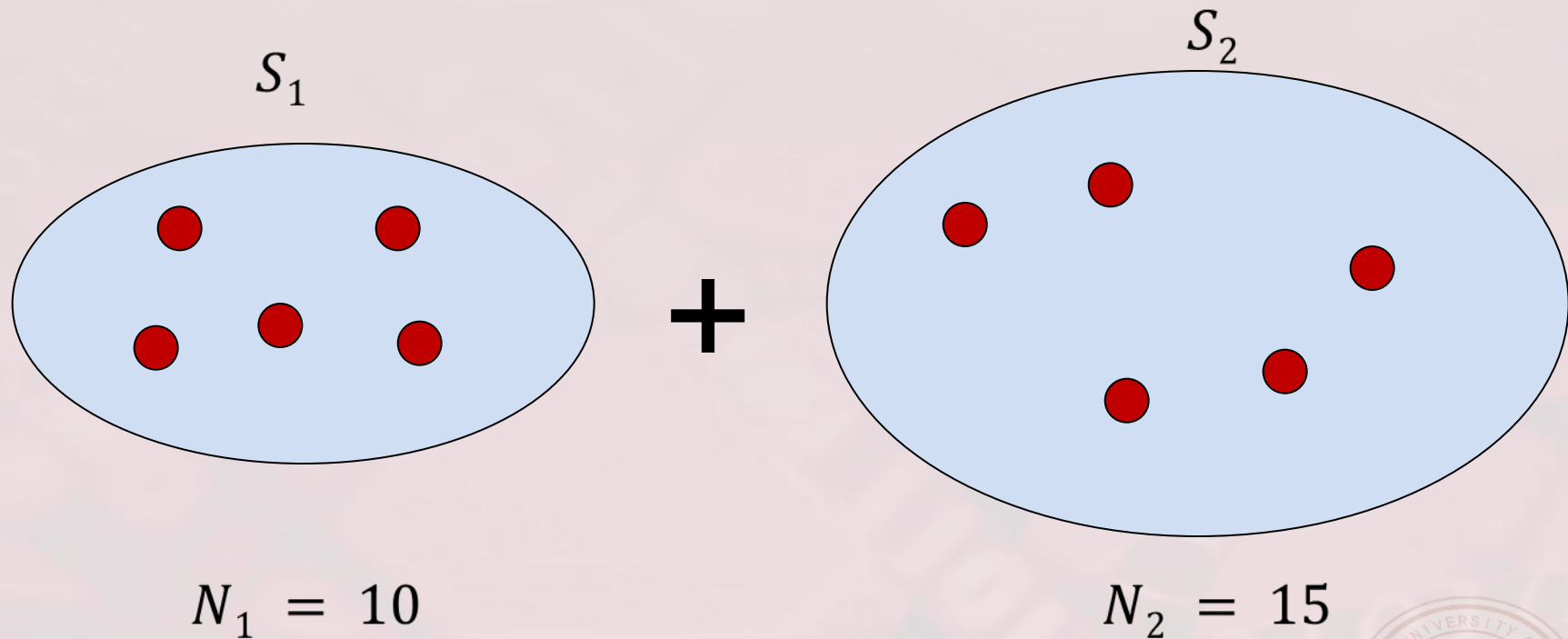
❖ 给定集合 S_1 的 k 个采样与集合 S_2 的 k 个采样，如何在不访问 S_1 和 S_2 的前提下，获得 S_1 与 S_2 合集的采样？



随机采样的合并



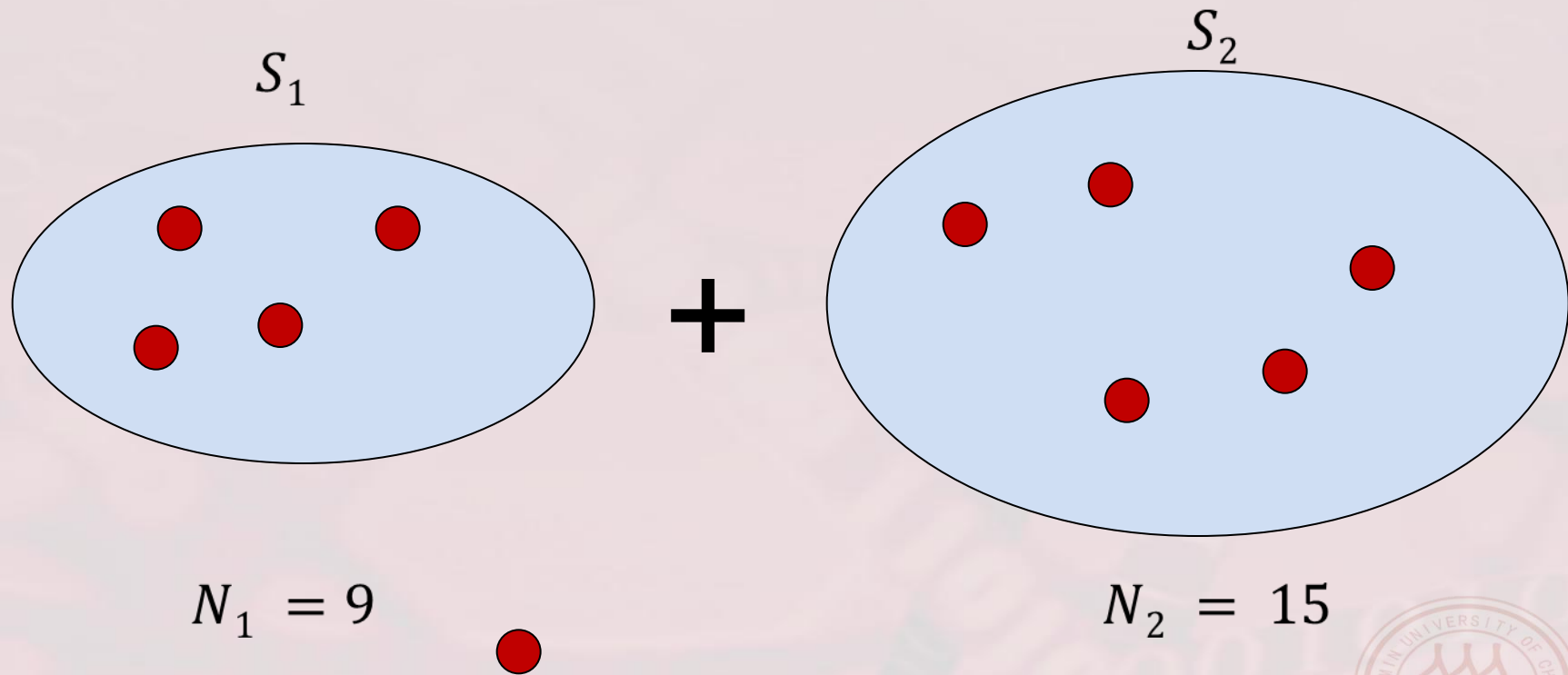
随机采样的合并



以概率 $N_1/(N_1 + N_2)$, 从 S_1 选取一个采样



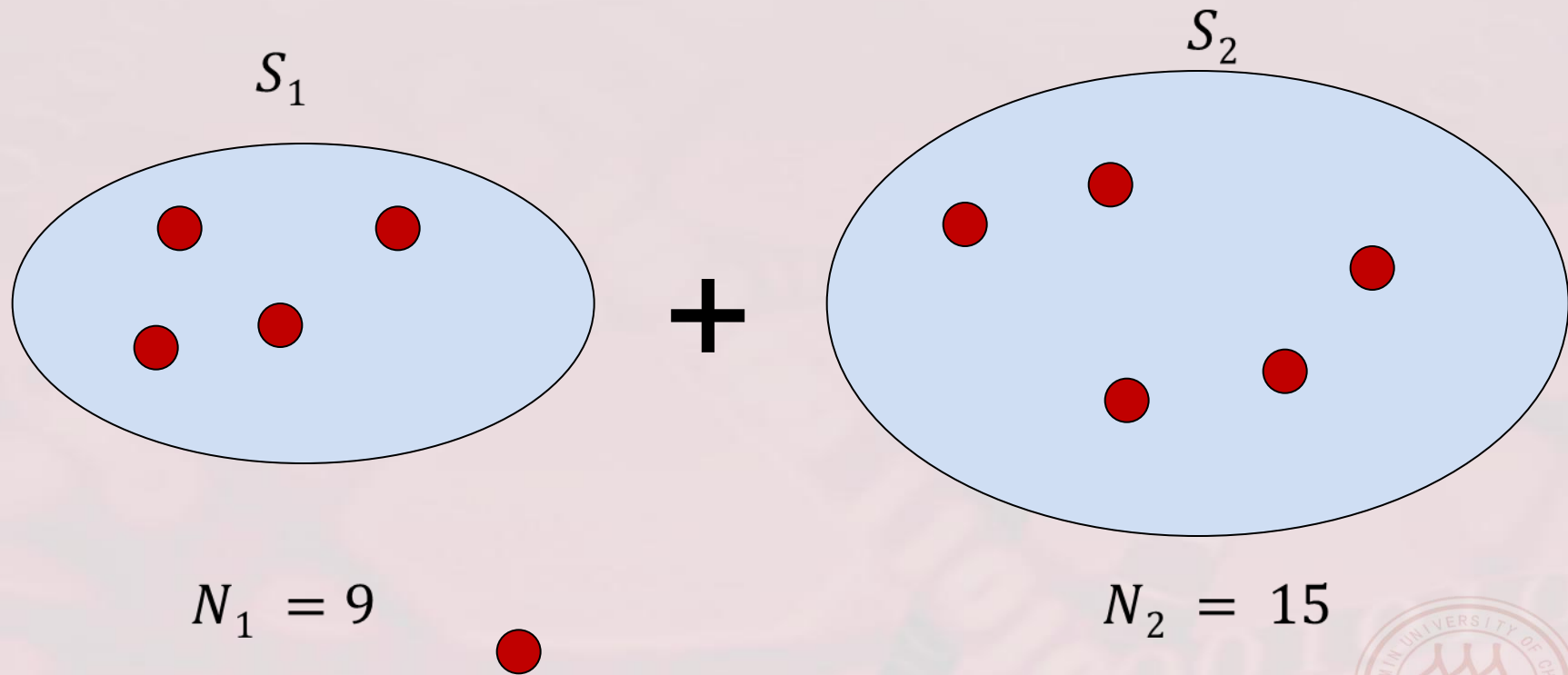
随机采样的合并



以概率 $N_1/(N_1+N_2)$ 从 S_1 选取一个采样



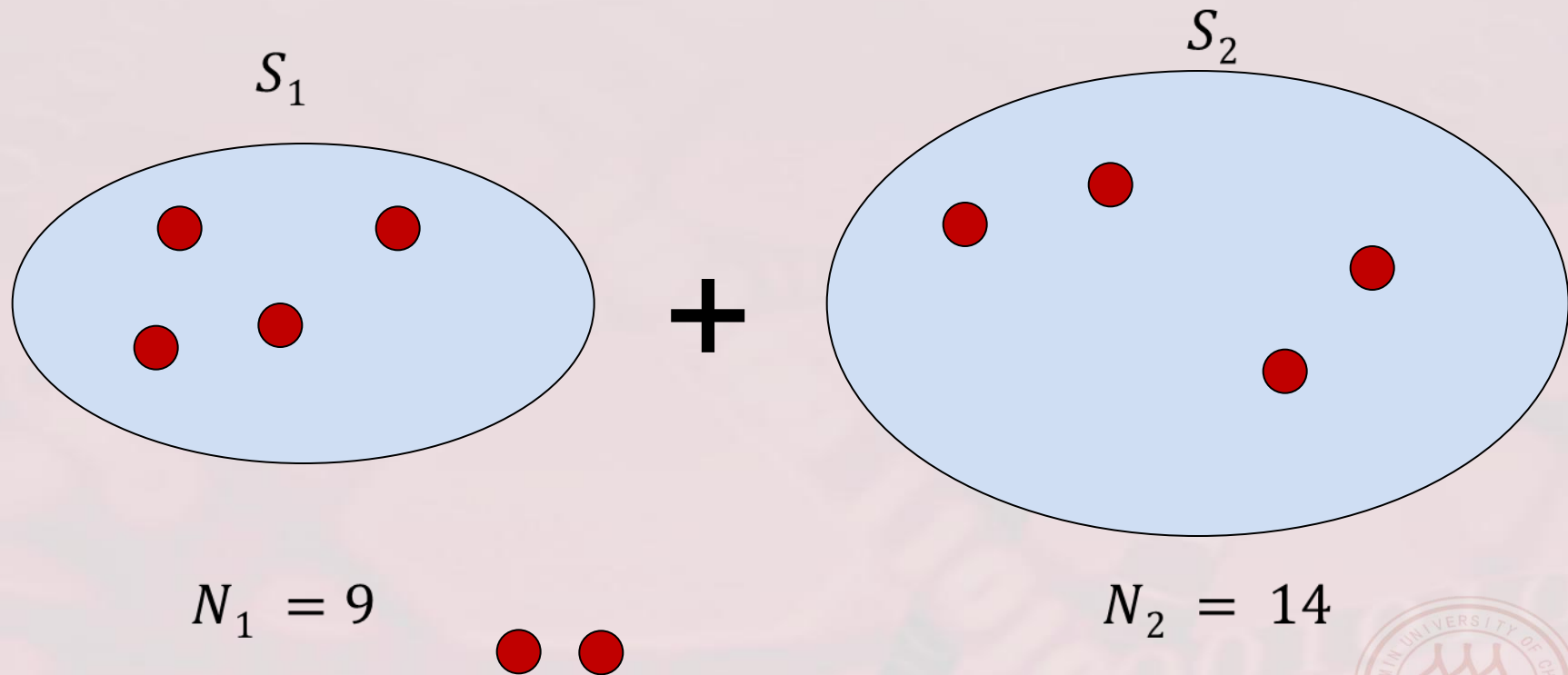
随机采样的合并



以概率 $N_2/(N_1+N_2)$ 从 S_2 选取一个采样



随机采样的合并



随机采样的合并

