Secure Sampling for Approximate Multi-party Query Processing

QIYAO LUO, Hong Kong University of Science and Technology, Hong Kong SAR, China
YILEI WANG, Alibaba Group, Hangzhou, China
KE YI, Hong Kong University of Science and Technology, Hong Kong SAR, China
SHENG WANG, Alibaba Group, Hangzhou, China
FEIFEI LI, Alibaba Group, Hangzhou, China

SIGMOD 2023

Secure Multi-Party Computation (MPC)

保险公司需要估计2023年各种疾病的理赔预算

医院

$R(\text{person}, \text{year}, \text{disease})$

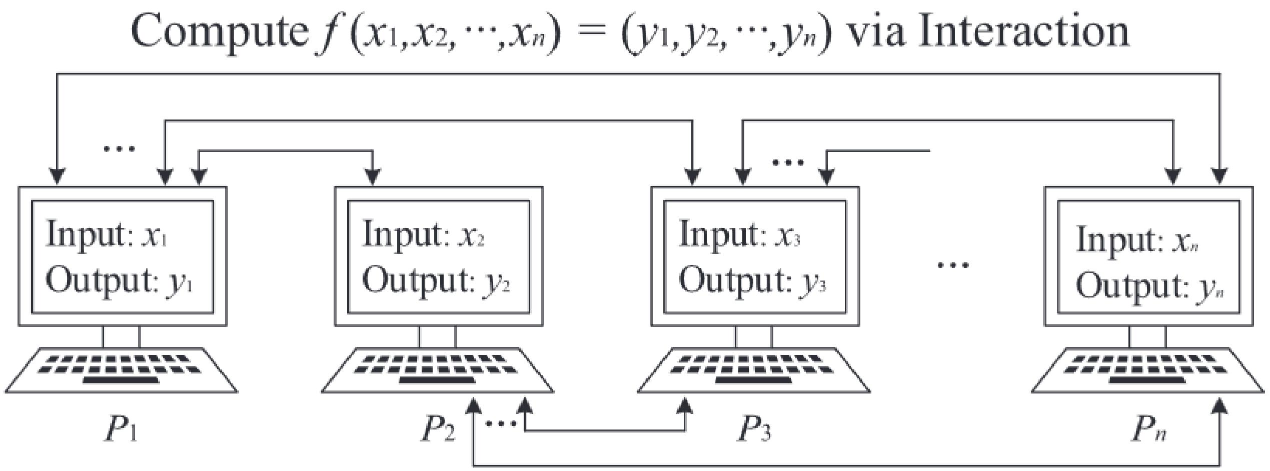
保险公司

$S(\text{person}, \text{cost}, \text{rate})$

```
SELECT disease, SUM(cost*rate)
FROM R, S
WHERE R.person = S.person AND R.year = 2023
GROUP BY disease
```

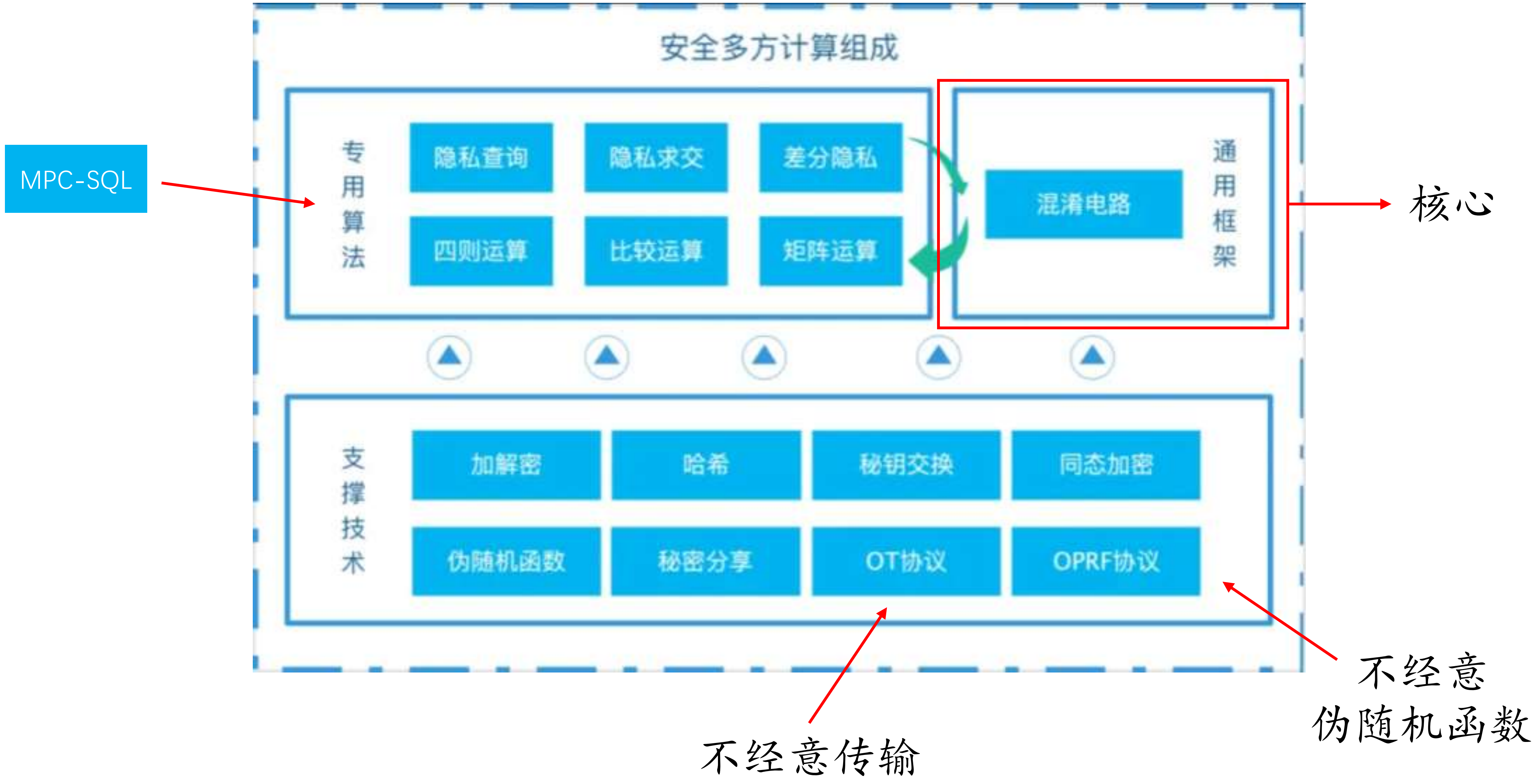
包含**敏感数据**的多方协同计算

Secure Multi-Party Computation (MPC)



安全多方计算于1986 年由**姚期智**院士通过姚氏百万富翁问题提出：两个富翁街头邂逅，想**比一比谁更有钱**，但是出于隐私，都**不想让对方知道自己到底拥有多少财富**，如何在不借助第三方的情况下，让他们知道谁更有钱。姚氏“百万富翁问题”后经发展，成为现代密码学中非常活跃的研究领域，即安全多方计算。

Secure Multi-Party Computation (MPC)



现有MPC-SQL系统效率低

- 现有的MPC-SQL的系统比明文计算慢**1000+倍**
  - SMCQL上执行涉及200条元组的查询就需要**1000秒**
- 现有工作主要通过**降低安全标准**或**针对特定类型查询**来提升效率
  - Conclave(2019)：依赖可信的**第三方**
  - Scape(2022)：会**透露**Join中间结果的大小
  - Shrinkwrap(2018)：通过差分隐私保护中间结果规模，但$\Omega(n^2)$开销
  - Secure Yannakakis(2021)：仅面向**特定类型**查询

Approximate Query Processing (AQP)

```
SELECT disease, SUM(cost*rate)
FROM R, S
WHERE R.person = S.person AND R.year = 2023
GROUP BY disease
```

正态分布下，*s*个样本的估计误差正比于 $\frac{1}{\sqrt{s}}$

```
SELECT disease, SUM(cost * rate)
FROM SAMPLE OF
        SELECT disease, cost*rate
        FROM R, S
        WHERE R.person = S.person AND R.year = 2023
GROUP BY disease
```

由于MPC协议的高开销，基于MPC的AQP技术比基于明文的AQP技术有更强的需求。

Approximate Query Processing (AQP)

- SAQE(2020)：首个MPC-AQP系统
  - 时间和通信成本均为$\boldsymbol{O(n\log n)}$，无论采样数s有多少
- 坏消息：安全采样算法有$\boldsymbol{\Omega(n)}$**的时间下界**
- 本文的解决方法：**Batch Sampling**（预处理+查询均摊）
  - 预处理出$\boldsymbol{n/s}$组独立采样，每组$\boldsymbol{s}$个样本
  - 查询时每次返回一组样本
  - 均摊后，采样s个样本只需要$\boldsymbol{\tilde{O}(s)}$时间
- 对于每次查询样本数不确定的情况
  - 分别对$\boldsymbol{s=1,2,4,\ldots,\frac{n}{2},n}$执行采样预处理，查询时拼凑
  - 只适用于**WR sampling**和**Shuffle sampling**

本文技术路线

Shuffle Sampling
WR Sampling
WoR Sampling
Stratified Sampling

改写

采样电路

混淆

混淆电路

two-server model

MASQUE

基础电路

- Uniform Random Number Generator
- Sorting
- Prefix-sum
- Compaction
- Expansion
- Primary Key Join

差分隐私
电路优化等

基础电路

- **Uniform Random Number Generator**
  - URNG($\boldsymbol{x}$)生成$\{1, \ldots, x\}$中的随机数
- **Sorting**
  - Bitonic sorter：$O(n\log^2 n)$-size，$O(\log^2 n)$-depth（本文选用）
  - AKS network：$O(n\log n)$-size，$O(\log n)$-depth，但常数巨大
- **Prefix-sum**
  - 输入$(x_1, x_2, \ldots, x_n)$，输出$(x_1, x_1 \oplus x_2, \ldots, x_1 \oplus x_2 \oplus \cdots \oplus x_n)$
  - Segmented prefix-sum：分为若干段，每段内求前缀和
  - 若$\oplus$可被常数大小的电路执行，则(segmented) prefix-sum可被 $O(n)$-size，$O(\log n)$-depth的电路执行

基础电路

- **Compaction**
  - 输入：序列(3,1,2,7,5,4,6)和掩码(0,0,1,1,0,1,0)
  - 输出：(2,7,4,3,1,5,6)
  - $O(n\log n)$-size，$O(\log n)$-depth
- **Expansion**
  - 输入：$(x_1, x_2, \dots, x_n), (d_1, d_2, \dots, d_n)$
  - 输出：$(\underbrace{x_1, \dots, x_1}_{d_1}, \underbrace{x_2, \dots, x_2}_{d_2}, \dots, \underbrace{x_n, \dots, x_n}_{d_n})$
  - $O(m\log m)$-size，$O(\log m)$-depth，其中$m = \sum_{i=1}^{n} d_i$

基础电路

- **Primary Key Join**
  - 输入：关系$R(x,y),S(y,z)$，其中y是R的主键
  - 输出：$R \bowtie S = \{(x,y,z)|(x,y) \in R \wedge (y,z) \in S\}$
  - $O(n\log^2 n)$-size，$O(\log^2 n)$-depth

Methods	Independence	Sampling error	Privacy amplification	Circuit depth	Circuit size
Shuffle sampling	No	$O\left(\frac{1}{\sqrt{s}} \cdot \sqrt{\frac{n-s}{n}}\right)$	$(\varepsilon, 0)^*$	$O(\log^2 n)$	$O(n\log^2 n)$
WR sampling	Yes	$O\left(\frac{1}{\sqrt{s}}\right)$	$\left(\frac{s}{n} \cdot \varepsilon, O\left(\frac{s^2}{n^2} \cdot \varepsilon\right)\right)^\star$	$O(\log^2 n)$	$O(n\log^2 n)$
WoR sampling	Yes	$O\left(\frac{1}{\sqrt{s}} \cdot \sqrt{\frac{n-s}{n}}\right)$	$\left(\frac{s}{n} \cdot \varepsilon, 0\right)^\dagger$	$O(\log^2 n \log \sigma)$	$O(n\log^2 n \log \sigma)$
Stratified sampling	Yes	$O\left(\frac{1}{\sqrt{k_i}} \cdot \sqrt{\frac{d_i-k_i}{d_i}}\right)^\ddagger$	$\left(\frac{k_i}{d_i} \cdot \varepsilon, 0\right)$	$O(\log^2 n \log \sigma)$	$O(n\log^2 n \log \sigma)$

*Shuffle sampling provides no privacy amplification;

*WR sampling has $\varepsilon' = \log((1-(1-\frac{1}{n})^s) \cdot (e^\varepsilon - 1) + 1) \approx s/n \cdot \varepsilon$ and $\delta' \leq \sum_{k=1}^{s} \binom{s}{2} \left(\frac{1}{n}\right)^k \left(1-\frac{1}{n}\right)^{s-k} \left(\frac{\varepsilon}{2} - \frac{\varepsilon}{2k}\right) = O\left(\frac{s^2}{n^2} \cdot \varepsilon\right)$;

†WoR sampling has $\varepsilon' = \log(s/n \cdot (e^\varepsilon - 1) + 1) \approx s/n \cdot \varepsilon$ and keeps pure-DP;

‡Each stratum, which takes k_i WoR samples from d_i data, has the same sampling error and privacy amplification as WoR sampling.

Shuffle Sampling

- **Shuffle Sampling**
  - 输入：$(x_1, x_2, \dots x_n)$
  - 为每个元素生成一个足够长的随机数，使其唯一
    - 即生成$R(x, e) = \{(x_i, \mathrm{URNG}(N)) | 1 \le i \le n\}$
  - 将R按属性e排序
  - 返回$R.x$序列
- 与排序电路相同的规模：$O(n\log^2 n)$-size，$O(\log^2 n)$-depth

WR Sampling

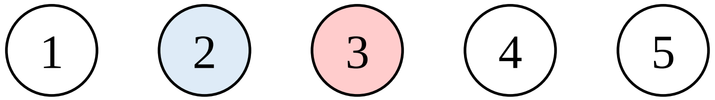
- **WR Sampling**
  - 输入：$(x_1, x_2, \dots x_n)$
  - 令$R(x, eid) = \{(x_i, i) | 1 \le i \le n\}$
  - 令$S(eid, sid) = \{(\mathrm{URNG}(n), \lceil i/s \rceil) | 1 \le i \le n\}$
  - $T \leftarrow R \bowtie S$
  - 对T按属性eid排序
  - 返回$T.x$序列
- 电路规模：$O(n\log^2 n)$-size，$O(\log^2 n)$-depth

WoR Sampling

Floyd's Algorithm

- For $n - s < b \leq n$:
  - 随机选$a \in \{1,2,\ldots,b\}$
  - 若$a \notin S$：$S \leftarrow S \cup \{a\}$
  - 若$a \in S$：$S \leftarrow S \cup \{b\}$
- 返回S

例：5个元素中采样3个



1. $b = 3, a = 2, S = \{\}$

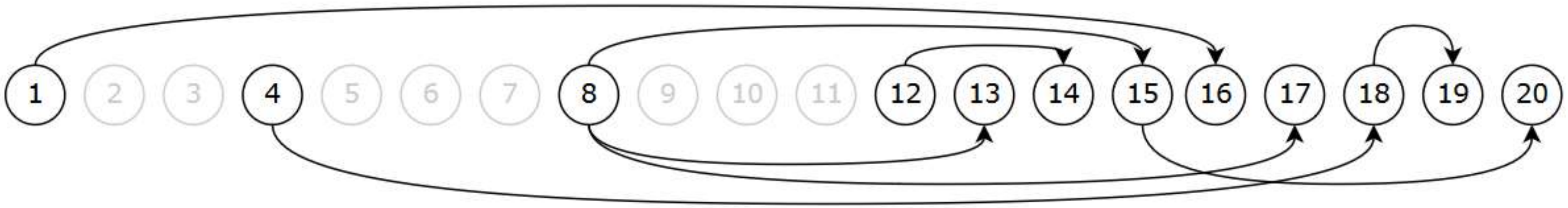
2. $b = 4, a = 2, S = \{2\}$

3. $b = 5, a = 4, S = \{2,4\}$

4. $S = \{2,4,5\}$

依赖图

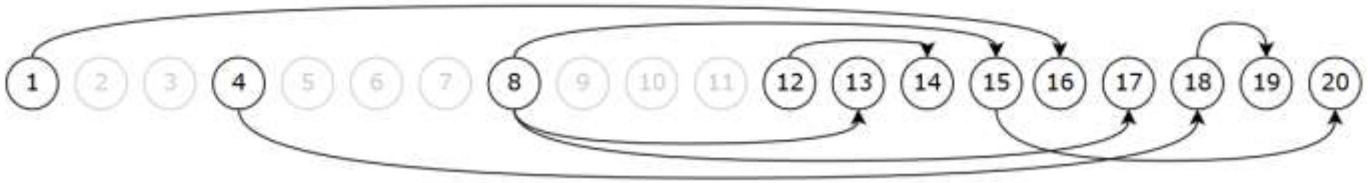
- 电路无法执行**集合成员性判断**
- 将每一步随机选择的过程转变为**依赖图$\boldsymbol{G(V,E)}$**
  - $V=\{1,2,3,\dots,n\}$
  - 若存在某一步选取了a,b，则$(\boldsymbol{a},\boldsymbol{b})\in\boldsymbol{E}$

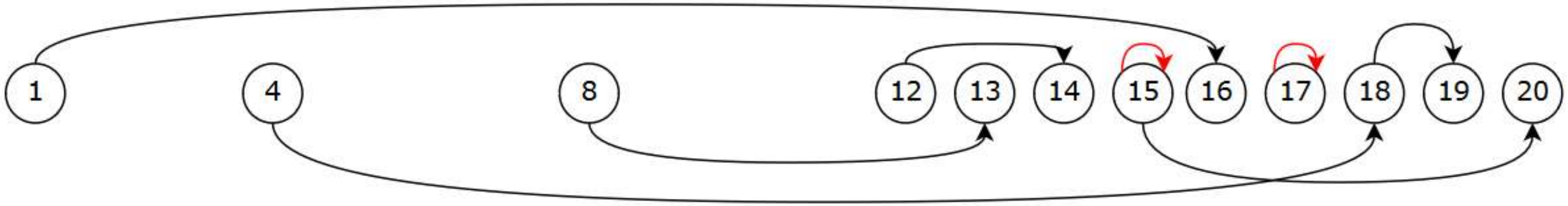


The original graph G with edges $E=\{(8,13),(12,14),(8,15),(1,16),(8,17),(4,18),(18,19),(15,20)\}$.

依赖图约简

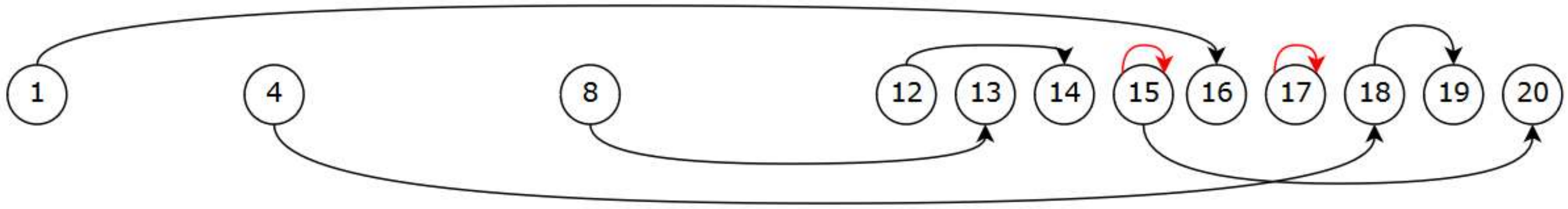
- 依赖图约简
  - 去掉没有连边的点
  - 若存在边$(a, b_1), (a, b_2), \ldots, (a, b_k) \in E$
    - 去除边$(a, b_2), \ldots, (a, b_k)$
    - 添加自环边$(b_2, b_2), \ldots, (b_k, b_k)$
  - 很容易通过电路实现





The reduced graph G' of G, where the edges $(8, 15)$ and $(8, 17)$ are replaced by the self-loops (marked in red) $(15, 15)$ and $(17, 17)$, respectively.

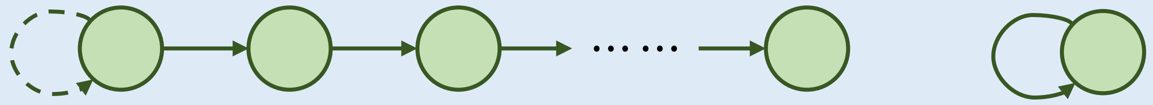
依赖图约简



The reduced graph G' of G, where the edges $(8, 15)$ and $(8, 17)$ are replaced by the self-loops (marked in red) $(15, 15)$ and $(17, 17)$, respectively.

引理

约简后的图由形如

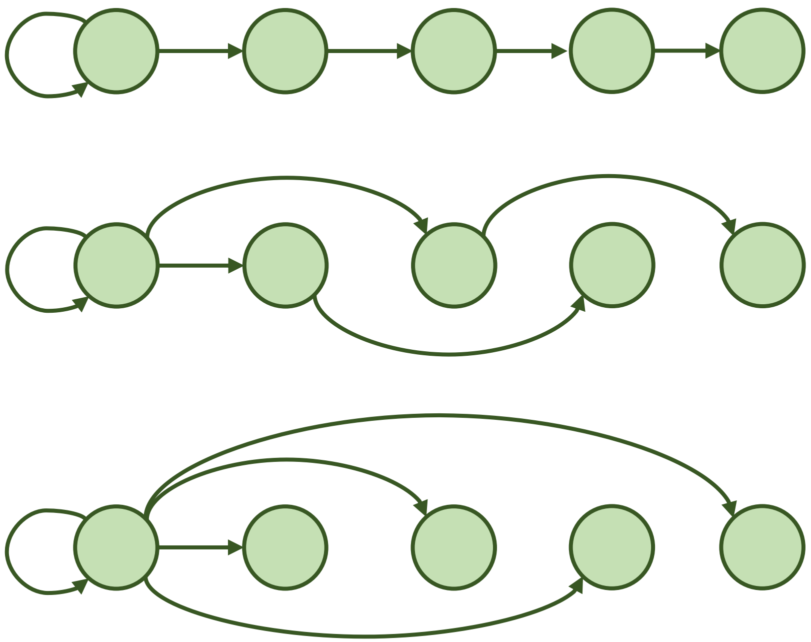


的链或自环点（虚线边可以不存在）构成，且任意节点不属于采样集S当且仅当**它是某个以$\boldsymbol{r \leq n - s}$为根的链的尾节点。**

Pointer Jumping

Pointer Jumplin

- Repeat h times
  - For each $(a, b), (b, c) \in E$
    - $E \leftarrow E \setminus \{(b, c)\}$
    - $E \leftarrow E \cup \{(a, c)\}$
- h至多为$O(\log s)$
  - 高概率为$O(\log\log n)$



使用电路快速找到链头

Stratified Sampling

- 样本空间被分为g个层
  - 第$i(1 \le i \le g)$层的大小为d_i
  - 第i层的采样数为k_i
- 确定$(k_1, \dots, k_g)$的方法
  - **Individualized sample sizes**：$k_i = F(i, d_i)$，如$k_i = d_i \cdot \frac{s}{n}$
  - **Threshold policy**：$k_i = \max(k, d_j)$
    - 其中，k为满足$\sum_{i=1}^{g} \max(k, d_j) \le s$的最大的$k$
- 难点：$(\boldsymbol{d_1}, \boldsymbol{d_2}, \dots, \boldsymbol{d_g})$在计算中需要被保护，不能泄露

Individualized sample sizes

- 输入：$R(eid, gid)$
- 对R按属性gid排序
- $(d_1, \dots, d_n) \leftarrow$对$(1,1,\dots,1)$以$R.gid$为分段的前缀和
- $(d_1, \dots, d_g) \leftarrow$将$(d_1, \dots, d_n)$只保留每组的最后一个值并压缩
- 分别计算$(k_1, \dots, k_g) \leftarrow \left(F(1, d_1), \dots, F(g, d_g)\right)$
- 返回$(d_1, \dots, d_g), (k_1, \dots, k_g)$

例：

$R.gid$	1	2	2	2	3	3	4
d	1	1	2	3	1	2	1
d'	1	3	2	1	null	null	null

Threshold Policy

- 输入$(d_1, \dots, d_g)$
- 对$(d_1, \dots, d_g)$按升序排序
- 令$t_1 \leftarrow d_1 \cdot g$
- 对于$2 \le i \le g$，令$t_i \leftarrow t_{i-1} + (d_i - d_{i-1}) \cdot (g - j + 1)$
- $l \leftarrow$满足$s \ge t_l$最大的l
- 返回$k \leftarrow \lfloor (s - t_l)/(g - l) \rfloor + d_l$

例：

d	1	1	2	3
t	4	4	6	7

Summary

Index	Income ($\mathcal{X}$)	Nation (gid)
1	20,000	Canada
2	5,000	China
3	12,000	China
4	3,500	China
5	100,000	China
6	10,000	USA
7	13,000	USA
8	7,000	USA
9	350,000	USA
10	70,000	USA
11	31,000	USA

Compute strata and sample size

Strata size (d)	Sample size (k)
1	1
4	2
6	2
⊥	⊥
⊥	⊥

Expansion (Algorithm 9 line 1-7)

gid	D	K	P	O
1	1	1	0	1
2	4	2	1	1
2	4	2	1	2
3	6	2	5	1
3	6	2	5	2

Generate random edges E (line 8-12)

a	b
1	1
3	4
4	5
9	10
9	11

Compute reduced graph, execute pointer jumping, and sort the table (line 13-15)

a	b
1	1
3	4
3	5
9	10
11	11

Get sample indices (line 17-25)

S
1
4
3
9
11

Methods	Independence	Sampling error	Privacy amplification	Circuit depth	Circuit size
Shuffle sampling	No	$O\left(\frac{1}{\sqrt{s}} \cdot \sqrt{\frac{n-s}{n}}\right)$	$(\varepsilon, 0)^*$	$O(\log^2 n)$	$O(n \log^2 n)$
WR sampling	Yes	$O\left(\frac{1}{\sqrt{s}}\right)$	$\left(\frac{s}{n} \cdot \varepsilon, O\left(\frac{s^2}{n^2} \cdot \varepsilon\right)\right)^\star$	$O(\log^2 n)$	$O(n \log^2 n)$
WoR sampling	Yes	$O\left(\frac{1}{\sqrt{s}} \cdot \sqrt{\frac{n-s}{n}}\right)$	$\left(\frac{s}{n} \cdot \varepsilon, 0\right)^\dagger$	$O(\log^2 n \log \sigma)$	$O(n \log^2 n \log \sigma)$
Stratified sampling	Yes	$O\left(\frac{1}{\sqrt{k_i}} \cdot \sqrt{\frac{d_i-k_i}{d_i}}\right)^\ddagger$	$\left(\frac{k_i}{d_i} \cdot \varepsilon, 0\right)$	$O(\log^2 n \log \sigma)$	$O(n \log^2 n \log \sigma)$