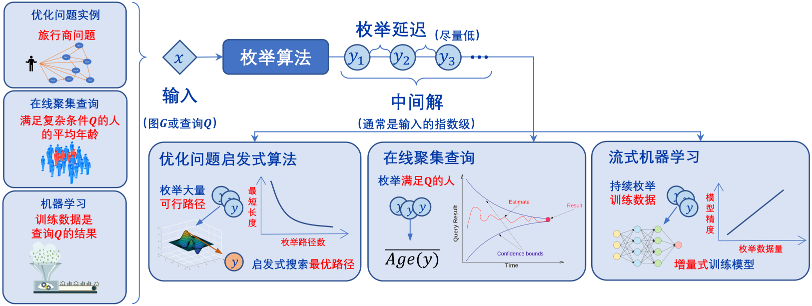
自可约问题的随机枚举算法与应用

哈尔滨工业大学本科毕业设计结题答辩

报告人：陈鹏宇　指导教师：苗东菁、洪晓鹏

2023 年 6 月 2 日

枚举问题背景



定义 1 (枚举问题)

给定一个字母表 Σ，一个枚举问题定义为一个二元组 (X, Sol)，其中 $X \subseteq \Sigma^*$ 是输入集，对于 $\forall x \in X$，$Sol(x) \subseteq \Sigma^*$ 表示输入对应的解集。枚举问题的枚举算法需要对于给定的问题输入，**不重复地**按照一定顺序输出所有解，其输出相邻两个解之间的时间称为枚举算法的**延迟**。

例：正则路径查询

例 2

输入一个边带标签的图 G、两个点 $s,t \in V(G)$、整数 $k \in \mathbb{N}$ 和正则表达式 φ，查询所有**s 到 t 长为 k 且与 φ 匹配的路径**。

- 广泛应用于**数据筛选**、**日志分析**、**路由匹配**等任务中
- 路径枚举 $\Longrightarrow$ 聚集估计、启发式优化...
- Arenas 19：**顺序枚举**、**近似计数**、**均匀采样**[1]

$$\text{枚举延迟}\downarrow \Longrightarrow \text{单位时间数据量}\uparrow \Longrightarrow \begin{cases} \text{优化目标}\uparrow \\ \text{查询精度}\uparrow \\ \text{模型精度}\uparrow \end{cases}$$

[1] Arenas M et al. #NFA Admits an FPRAS: Efficient Enumeration, Counting, and Uniform Generation for Logspace Classes

自可约性

定义 3 (自可约性)

我们称一个枚举问题 (X, Sol) 是自可约的，当且仅当：

- 存在多项式可计算的长度函数 $\zeta : X \to \mathbb{N}$，使得对于 $\forall x \in X$，$\boldsymbol{\forall w \in Sol(x)}$，$\boldsymbol{|w| = \zeta(x)}$，
- $\forall x \in X, \zeta(x) = 0$，可在多项式时间内判断空串是否属于 $Sol(x)$，存在多项式时间可计算的**自归约函数** $\boldsymbol{\Psi : X \times \Sigma^* \to X}$，使得对于 $\forall (x, w) \in X \times \Sigma^*$：
  - $|\Psi(x, w)| \leq |x|$
  - $\zeta(\Psi(x, w)) = \max\{\zeta(x) - |w|, 0\}$
  - $\boldsymbol{Sol(\Psi(x, w)) = \{w' | w \circ w' \in Sol(x)\}}$

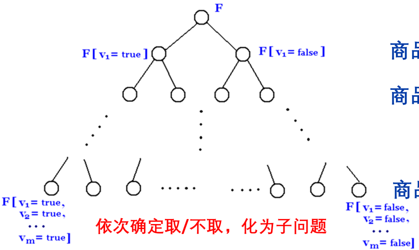
问题 4 (经典背包问题)

给定背包容量和一组物品的体积，要求**枚举出所有可能的装配方案，使得装配物品的总体积不超过背包的容量极限**。形式化地，背包枚举问题可以表示为 (X_{ks}, Sol_{ks})，其中

$$X_{ks} = \{(C, s_1, \ldots, s_n) \mid n \in \mathbb{N}^+, C, s_1, \ldots, s_n \in \mathbb{N}\}$$

$$Sol_{ks}((C, s_1, \ldots, s_n)) = \{S \mid S \subseteq \mathbb{N}[1, n], \sum_{i \in S} s_i \leq C\}.$$

m个商品
在一定限制条件下
枚举所有取法



商品1 取/不取

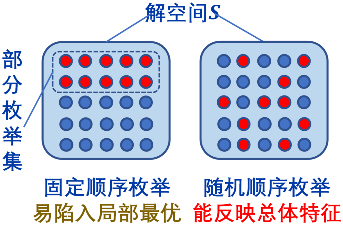
商品2 取/不取

⋮

商品m 取/不取

枚举结果的随机性、多样性

在数据分析中，样本的**多样性**和**随机性**是非常重要的属性，随机枚举的解能够更好地反映解空间的统计特性。



现有的枚举算法大多不关心枚举顺序的随机性：

- 按字典序输出——长公共前缀
- 已生成的答案构造新答案——相近的答案相似
- 通过递归构造子集——相近集合交集大

前置工作

自可约问题
均匀采样

Jerrum et al.

1986

自可约问题
有序枚举

Schmidt

2009

均匀采样构造
随机枚举

Capelli et al.

2019

UCQ随机枚举

Carmeli et al.

2020

更低延迟的
随机枚举

高效并行化

This Work

- 若一个自可约的 NP-枚举问题的计数存在 FPRAS，那么它的解集可以在**多项式时间内高概率地均匀采样**[2]
- 若一个自可约的 NP-枚举问题可以在多项式时间内判断是否有解，那么它的解集可以在**多项式延迟内枚举**[3]
- 可以通过调用多项式时间的均匀采样算法并拒绝重复解来实现**多项式延迟的随机枚举**[4]

[2] Jerrum M R et al. Random generation of combinatorial structures from a uniform distribution

[3] Schmidt J. Enumeration: Algorithms and complexity

[4] Capelli F et al. Incremental delay enumeration: Space and time

主要贡献

可高效(近似)计数（NP问题） $\xrightarrow{\text{自可约}}$ 可高效随机顺序枚举

$$
\text{自可约 NP 问题}
\begin{cases}
\text{精确计数} & \Longrightarrow \text{随机枚举} \\
\text{FPTAS近似计数} & \Longrightarrow \text{Las Vegas随机枚举} \\
\text{FPRAS近似计数} & \Longrightarrow \text{Atlantic City随机枚举}
\end{cases}
$$

XX 随机枚举 $\xrightarrow{\text{并行化}}$ **1.5-最优**的总时间/延迟

多项式精确计数：随机顺序枚举

定理 5

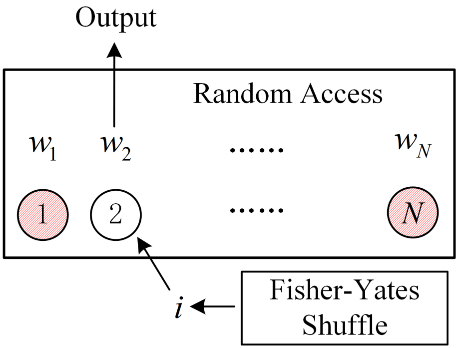
若一个自可约的 *NP*-枚举问题的解**可以被在多项式时间内计数**，给定它的多项式时间的长度函数 ζ、自归约函数 Ψ 和解的计数算法 $\mathcal{A}$，则该问题存在一个延迟为 $O(\zeta(x)(T_\Psi(|x|)+T_\mathcal{A}(|x|)))$ 的随机枚举算法。

- **枚举自然数 $i \to$ 输出字典序下第 i 个解**
- Fisher-Yates Shuffle：$O(1)$ 延迟的自然数枚举
- 自可约性 + 计数算法 $\to$ 解的随机访问

定义 6 (随机顺序枚举)

若一个枚举算法**每一个枚举的解都是在待枚举解集上的均匀采样**，那么这个枚举算法被称为“随机顺序枚举算法”。

多项式精确计数：随机顺序枚举



- **枚举自然数 $i \to$ 输出字典序下第 i 个解**
- Fisher-Yates Shuffle： $O(1)$ 延迟的自然数枚举
- 自可约性 + 计数算法 $\to$ 解的随机访问

FPTAS 近似计数：Las Vegas 随机顺序枚举

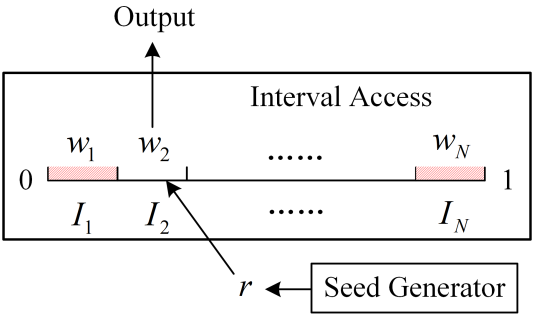
定理 7

若一个自可约的 *NP*-枚举问题的解**可以被 *FPTAS* 近似计数**，给定它的多项式时间的长度函数 ζ、自归约函数 Ψ 和 *FPTAS* $\mathcal{B}$，则该问题存在一个期望延迟为 $O(\zeta(x)(T_{\Psi}(|x|) + T_{\mathcal{B}}(|x|, \zeta(x))))$ 的 *Las Vegas* 随机枚举算法。

定义 8 (Las Vegas 随机顺序枚举)

若一个枚举算法的每一步枚举**以大于某常数 c 的概率成功输出待枚举集上的均匀采样**，且**若失败可以重做该步枚举**，则称该算法为“Las Vegas 随机顺序枚举算法”。

FPTAS 近似计数：Las Vegas 随机顺序枚举



- 为每个解分配一个 $[0,1)$ 上的子区间（互不相交）
- 在**未被访问过的区间上**生成实数种子 r
- 计算**包含 r 的子区间及其对应的解**，概率修正
- 自可约性 + 近似计数算法 $\Rightarrow$ 区间访问

FPRAS 近似计数：Atlantic City 随机顺序枚举

定理 9

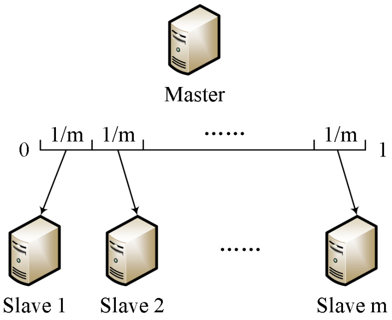
若一个自可约的 *NP*-枚举问题的解**可以被 *FPRAS* 近似计数**，给定它的多项式时间的长度函数 ζ、自归约函数 Ψ 和 *FPRAS* $\mathcal{C}$，则该问题存在一个期望延迟为 $O\left(\zeta(x)\left(T_\Psi(|x|)+T_\mathcal{C}\left(|x|,\zeta(x),\zeta(x)\log\frac{1}{\delta}\right)\right)\right)$ 的 *Atlantic City* 随机枚举算法，使其正确输出随机排列的概率大于 $1-\delta$。

- 采用与区间访问类似的算法
- 近似计数算法 $\mathcal{C}$ 的**错误概率**对算法结果、延迟的影响
- 对相同输入多次运行 $\mathcal{C}$ 结果的**不一致性**

定义 10 (Atlantic City 随机顺序枚举)

若一个随机算法**高概率地成为一个“Las Vegas 随机顺序枚举算法”**，则称该算法为“Atlantic City 随机枚举算法”。

并行枚举算法



- Phase I：Master 将**工作负载**、**初始字典**等发送给各 Slave
- Phase II：Slave 在工作负载区间上随机枚举，并发给 Master
- Phase II：Master 随机选取 Slave 并输出其队列中的一个解

定理 11

令 T 是单台机器枚举全部解所需的时间，则在上述假设下，存在基于 *Master/Slave* 架构的并行化枚举算法使得在 *Slave* 数量为 *m* 时，并行算法枚举全部解的总时间至多为 $\mathbf{1.5} \cdot \frac{\boldsymbol{T}}{\boldsymbol{m}}(1+o(1))$。

定理 12

在上述假设下，对任意 $\alpha \in (0,1)$，存在基于 *Master/Slave* 架构的并行化枚举算法使得在 *Slave* 数量为 m 时，经过 $O((\frac{ms}{\alpha^2}\log\frac{m}{\alpha})(1+o(1)))$ 时间的预处理，并行算法枚举解的延迟在高概率下至多为 $\mathbf{1.5} \cdot \frac{\boldsymbol{s}}{\boldsymbol{m}}(1+\alpha)$。

- 数据集：弗罗里达州立大学的背包问题数据集[5]
- 算法：基于**区间访问**和基于**拒绝采样**的单机和并行枚举算法
- 评价指标：算法的累计运行时间、平均延迟

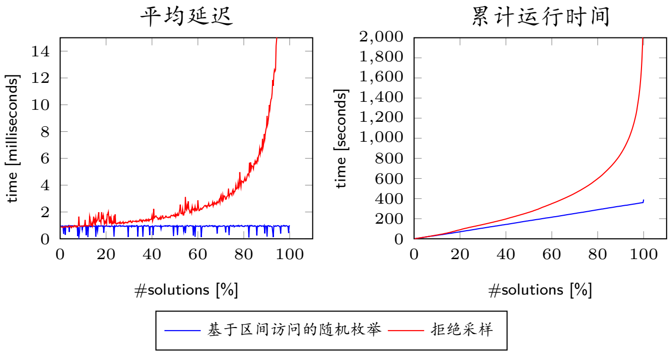


图: 平均延迟 (每 1000 解记录一次) 与累计运行时间

[5] Burkardt J. Data for the 01 Knapsack Problem[J/OL], 2014.

并行实验结果

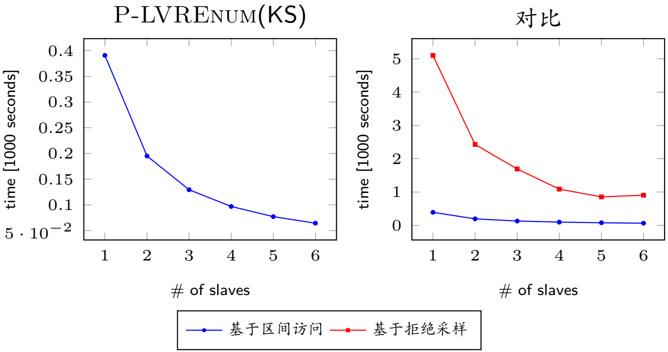


图: 并行算法运行时间及与拒绝抽样的对比

Thanks

感谢各位老师倾听

- 相关论文[6]
  - 数据库理论顶会 PODS 2023 初评 Weak Accept
  - CCF B 类会议 COCOON 2023 在投

[6]Chen P, Miao D, Tong W, et al. Random-Order Enumeration for Self-Reducible NP-Problems[J]. arXiv preprint arXiv:2302.13549, 2023.