Data Generation using Declarative Constraints
Scalable and Dynamic Regeneration of Big Data Volumes

Data Generation using Declarative Constraints

Arvind Arasu
Microsoft Research
Redmond, WA
arvinda@microsoft.com

Raghav Kaushik
Microsoft Research
Redmond, WA
skaushi@microsoft.com

Jian Li
University of Maryland
College Park, MD
lijian@cs.umd.edu

SIGMOD 11’

Scalable and Dynamic Regeneration of Big Data Volumes

Anupam Sanghi, Raghav Sood, Jayant Haritsa
Indian Institute of Science
Bangalore, India
{anupam,raghav,haritsa}@dsl.cds.iisc.ac.in

Srikanta Tirthapura
Iowa State University
Ames, USA
snt@iastate.edu

EDBT 18’

目录

CONTENTS

DataSynth Motivation

Motivation：1. DBMS testing：测试新组件的正确性和性能

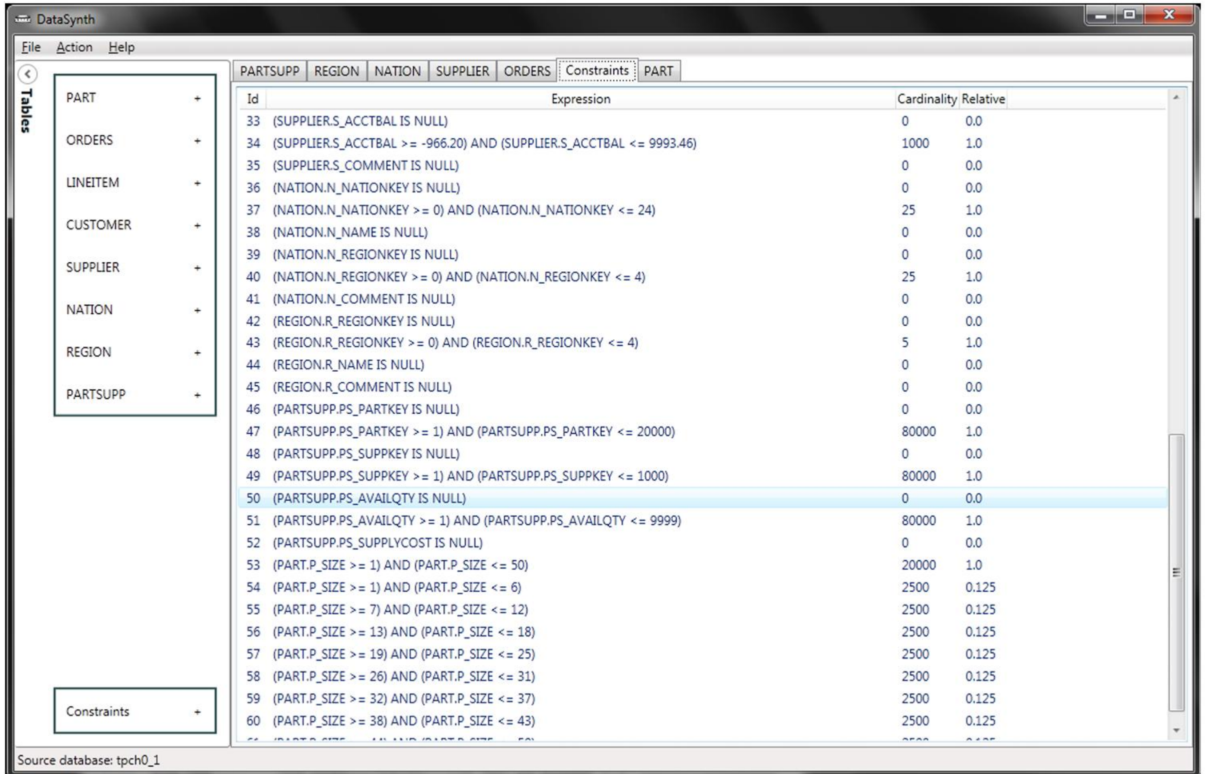
2. 规范：先前工作QAGen工作不够规范，数据特征是non-declarative的

3. 全面：先前工作使用约束仅描述了单一的数据特征，如查询基数

Contributions：1. 使用声明式的方法，即使用基数约束来捕捉更多的数据特征，从而生成符合各种复杂要求的合成数据库。2. 将基数约束转为线性规划问题，并使用区间化、概率图模型来优化

Data Generation Problem (DGP): Given a data base **schema** and a collection of cardinality constraints **C1, . . . , Cm**, generate a database instance conforming to the **schema** that satisfies all the **constraints**.

做法：**单表单属性**线性规划求解、**单表多属性**概率图模型减小LP的大小(指数大小)、**多表**借助视图拆解多表连接为单视图求解



DataSynth 单表单属性

前提假设： D = Dom(A)是非负整数, A is attribute，[D] = {1, . . . , D} ，约束C1, . . . , Cm，|R| = N

选择算子： $A \in [l, h]$; 等值选择即l = h，规范形式：$|\sigma_{lj \le A < hj}(R)| = kj$

单表单属性算法：对 any i ∈ [D] , x_i 表示 i 在 R 中的频数，则有：$\sum_{i=l_j}^{h_j-1} x_i = k_j \quad \text{for} \quad j = 1, \ldots, m$

整数线性规划ILP 求解复杂度为O(2^n) 而线性规划则可以在多项式时间O(n^3)

因为Ax= b 且 A是unimodularity时，可以松弛为线性规划求解出来依然为整数

但是该LP的变量数量为D，约束数量为m，求解复杂度O(m*D)

从求解单个取值的频数转变为求解区间的频数=>引入**Intervalization**减少变量数量:

将1~D+1分成L-1个区间，Let v1 = 1, v2, . . . , vl = D + 1

每个区间表示为[vi, vi+1)， $x_{[vi,vi+1)}$表示R(A)在该区间 $\sum_{i=p}^{q-1} x_{[v_i, v_{i+1})} = k_j$

对Cj : |σlj≤A<hj (R)| = kj，存在$v_p = l_j$ and $v_q = r_j$,则有

现在变量数则减少到了L (最多为2*m)

DataSynth 单表单属性 例子

Example 1.

Consider a DGP instance with three constraints $|\sigma_{20\leq A<60}(R)| = 30$, $|_{\sigma 40\leq A<101}(R)| = 40$, and $|R| = 50$ and assume a domain size D = 100. There are 4 basic intervals: [1, 20), [20, 40), [40, 60), [60, 101). The corresponding linear program consists of the three equations:

$$
\begin{aligned}
x_{[1,20)} + x_{[20,40)} + x_{[40,60)} + x_{[60,101)} &= 50 \\
x_{[20,40)} + x_{[40,60)} &= 30 \\
x_{[40,60)} + x_{[60,101)} &= 40
\end{aligned}
$$

One solution to the LP is x[1,20) = 2, x[20,40) = 8, x[40,60) = 22, and x[60,101) = 18. To generate R(A), we pick 2 values (e.g., at random) from [1, 20), 8 values from [20, 40), and so on.

DataSynth 单表多属性

n个属性的约束可以写作：$|\sigma_{Pj\,(R)}| = kj$，对每个元组 $t \in Dom(A1) \times \cdots \times Dom(An)$，$x_t$是tuple t在R中的频数，则有 $\sum_{t:P_j(t)=\text{true}} x_t = k_j$

Baseline算法LPA_{LG}：利用上述约束，直接求解出x_t

但变量数为：L^n 仍为指数级（即使间隔化，有L个取值区间，n个属性，L^n 种可能的tuple）

使用更有效的算法---概率图模型. 建模：每个属性Ai，令$Xi \in Dom(Ai)$， $X = \{X1, . . . , Xn\}$，则有联合分布概率p(X1, . . . , Xn)=p(X)；**Generative Distribution**：对于Cj=<Pj, kj>，Pj对按照p(X)采样元组为真的概率为kj/N (|R| = N) 故先求p(X)，再Sample N times from p(X) to generate R

问题在于：拆分复杂的多列联合分布为单列分布+少量的联合分布，减少变量个数（隐含假设：列之间独立，只有有约束才相关，才需要求联合分布概率）

p(X) 可以拆解为求多个子分布概率，即：$p(\mathcal{X}) = \prod_{\mathcal{X}_i:\exists C_j s.t. \mathcal{X}_i=\mathcal{X}(C_j)} f_i(\mathcal{X}_i)$

例如：Attrs(C1) = {A1, A2}(意为：C1约束涉及A1 A2两列)，其他约束Cj (j != 1)有 |Attrs(Cj)| = 1, 则p(X)= f1(X1, X2)f3(X3) · · · fn(Xn)

DataSynth 单表多属性 概率求解

将多列属性和约束建模为**Graphical Models**，通过Chordal graphs或Markov Blankets求解p(X)

Markov network G = (X , E)，顶点X1, . . . , Xn，如果{Xi, Xj} ⊆ X (Cj) 则加入一条边 (Xi, Xj)

马尔科夫网络是一种无向图模型，用于表示变量之间的依赖关系，最大团是图中一个完全连通的子图每个最大团对应于一个因子而联合分布可以通过这些因子来分解可以将联合分布分解为各个最大团的边缘分布

(a)

最大团因子分解

$$p(X_1, X_2, X_3, X_4) = p(X_1, X_2)p(X_3|X_1, X_2)p(X_4|X_1, X_2, X_3)$$

弦图性质

$$= p(X_1, X_2)p(X_3|X_2)p(X_4|X_3)$$

条件概率

$$= p(X_1, X_2)\frac{p(X_2, X_3)}{p(X_2)}\frac{p(X_3, X_4)}{p(X_3)}$$

求解最大团的边缘分布，根据约束：

$$\sum_{\mathbf{x} \in Dom(\mathcal{X}_{ci})} p_{\mathcal{X}_{ci}}(\mathbf{x}) = 1 \qquad 1 \leq i \leq l$$

$$\sum_{\mathbf{x} \in Dom(\mathcal{X}_{ci}): P_j(\mathbf{x}) = \text{true}} p_{\mathcal{X}_{ci}}(\mathbf{x}) = k_j/N \qquad \mathcal{X}(C_j) \subseteq \mathcal{X}_{ci}$$

INPUT: A data generation problem involving $R(A_1, \ldots, A_n)$ and constraints $C_1, \ldots, C_m$ and $|R| = N$

OUTPUT: A generative probability distribution $p(\mathcal{X})$

1. Construct the Markov network $G = (\mathcal{X}, E)$
2. Add edges to G to get a chordal graph $G_c = (\mathcal{X}, E_c)$
3. Identify maximal cliques $\mathcal{X}_{c1}, \ldots, \mathcal{X}_{cl}$ in G_c
4. Solve for marginal distributions $p(\mathcal{X}_{c1}), \ldots, p(\mathcal{X}_{cl})$
5. Use chordal graph property to construct $p(\mathcal{X})$ from the marginals $p(\mathcal{X}_{c1}), \ldots, p(\mathcal{X}_{cl})$

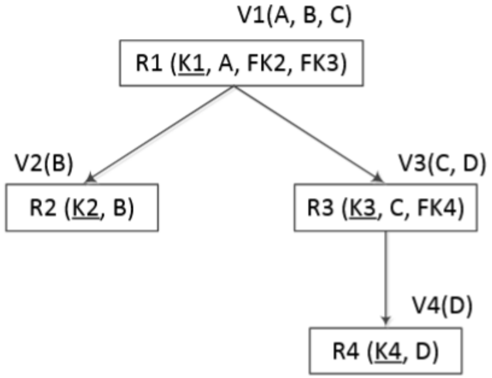
Figure 4: Indentifying a generative distribution using Chordal graphs

对于具有10 个属性且域大小D = 10 的类似的图，可以证明CLPAlg 使用$9 \cdot 10^2$ = 900 个变量，而LPAlg 使用10^{10} 个变量。

多表约束Cj: $|\sigma_{P_j}(R_{i_1} \bowtie \cdots \bowtie R_{i_s})| = k_j$ 通过创建视图将连接约束转化为单表的问题 $|\sigma_P(V_i)| = k$

表之间的关系建模为树形结构，边表示外键连接；每个表有一个由子表连接后生成的视图Vi

约束方程只涉及单表（视图）的列分布求解，使用前面的算法求解



Example 11. Consider two relations R1(K1, A, FK2) and R2(K2, B) and four constraints。区间化处理之后约束如右图：

$$|\sigma_{B\in[1,5)}(R_2)| = 2 \qquad |\sigma_{B\in[1,2)}(R_1 \bowtie R_2)| = 2$$
$$|\sigma_{B\in[5,10)}(R_2)| = 3 \qquad |\sigma_{B\in[2,10)}(R_1 \bowtie R_2)| = 2$$

$$\longrightarrow$$

$$x_{[1,10)[1,2)} = 2 \qquad y_{[1,2)} + y_{[2,5)} = 2$$
$$x_{[1,10)[2,5)} + x_{[1,10)[5,10)} = 2 \qquad y_{[5,10)} = 3$$

求解得到：

x[1,10)[1,2) = 2, x[1,10)[2,5) = 1, x[1,10)[5,10) = 1

y[1,2) = 0, y[2,5) = 2, y[5,10) = 3.

缺点：算法设计依赖于雪花模式（类树形结构）

有明确的父子关系，便于从根节点递归生成数据

A	B
1	1
1	1
1	2
1	5

$V_1(A, B)$

B
2
2
5
5
5
1

$V_2(B)$

K_1	A	FK_2
1	1	6
2	1	6
3	1	1
4	1	4

$R_1(K_1, A, FK_2)$

K_2	B
1	2
2	2
3	5
4	5
5	5
6	1

$R_2(K_2, B)$

DataSynth Projection

对于投影约束，提出了在单表单属性情况的算法：

1. 定义表和约束：设 R(A) 表示正在生成的表，A是该表的属性集。

$|\pi A(\sigma A \in [lj,hj](R))| = kj$：表示在选择谓词 σ 的限制下，投影 A 的基数为 kj。

2. 区间化：对于每个区间[vi, vi+1)，引入两个变量：x[vi, vi+1)：表示在该区间内的总元组数。y[vi, vi+1)：表示在该区间内的不同（去重）元组数。

3. 线性方程：$y[vi, vi+1) \le x[vi, vi+1)$：确保去重的元组数不会超过总元组数。

$y[vi, vi+1) \le (vi+1 - vi)$：确保区间中的不同元组数不超过区间的大小。

4. 线性规划求解

5. 随机舍入：当某些约束的结果是$q \le y[vi, vi+1) \le x[vi, vi+1) < (q + 1)$时，需要保证随即舍入后满足约束。做法：对于每个变量 x[vi, vi+1) 和 y[vi, vi+1)，如果它们的值大于 r，则舍入为 q + 1，否则舍入为 q。

DataSynth Experiments

支持TPC-H 8/23条sql，不支持 非等值过滤和LIKE；最大TPC-H 1GB data

Input可以任意改变查询参数[SEGMENT] [DATE] (查询参数和数据分布解耦)

算法整体的三个阶段：①分析输入约束来建立线性规划②使用LP求解器求解线性规划

③从每个视图的概率分布中采样生成数据库

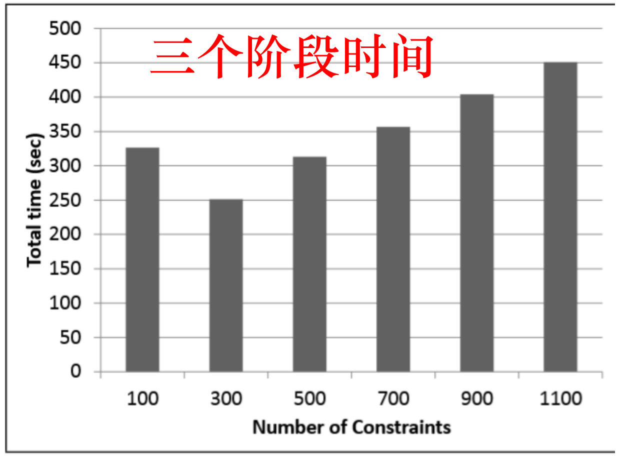


Figure 8: Total runtime

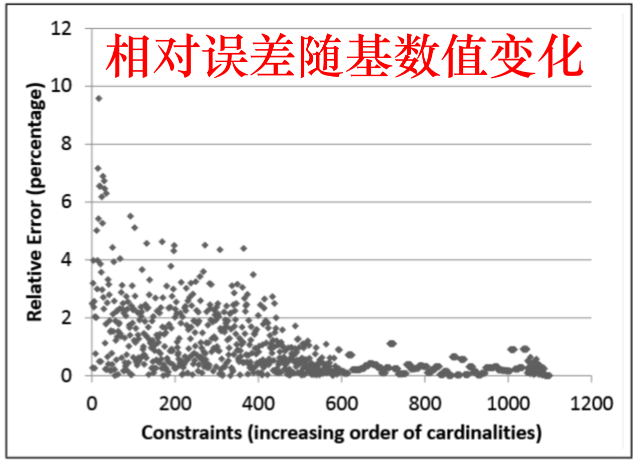


Figure 9: Relative errors in constraints

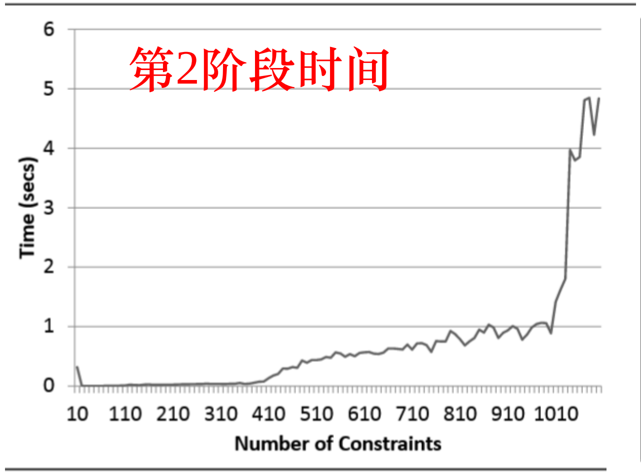


Figure 14: Time to solve the LP

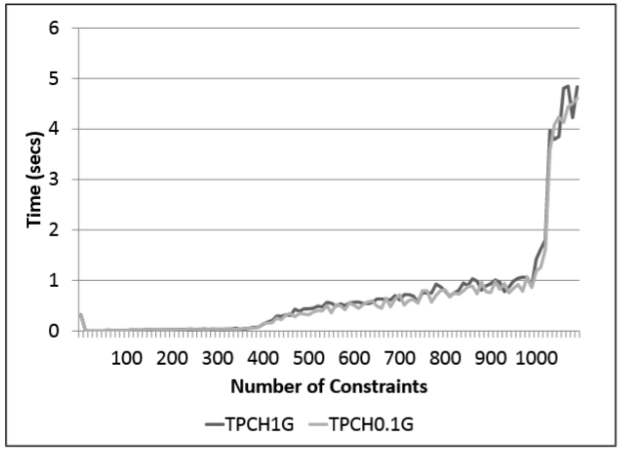


Figure 15: Time to solve the LP (1G vs 0.1G)

HYDRA Motivation

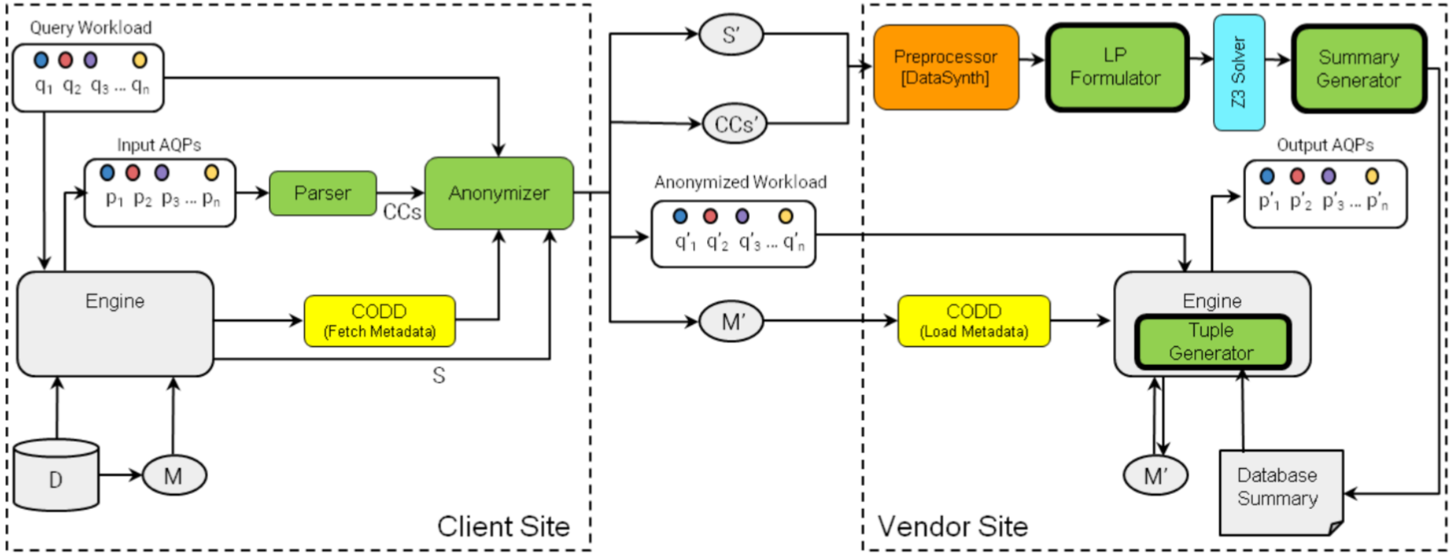
Motivation：1. 场景上同DataSynth 2. 先前工作扩展到大型查询工作负载和大数据量的能力不明确

Contributions：1. **区域划分优化**：Region-Partitioning的方法优于其他划分策略，大大降低了 LP 的复杂性，使其能够比以往的Grid-Partitioning的方法更高效地处理丰富的查询工作负载和大数据量。

2. **动态再生**： HYDRA 引入了动态再生(regeneration)的概念，通过创建一个database summary，可以在查询执行过程中动态生成数据，而无需物化整个数据集。

3. **确定性对齐算法**：替代 DataSynth 的采样方法，直接对数据库摘要执行确定性操作。

4. **全面评估**：在TPC-DS进行实验，有100多个查询负载；JOB(IMDB)



HYDRA Partitioning

原方法：基于固定网格划分，分区数与全局维度相关。

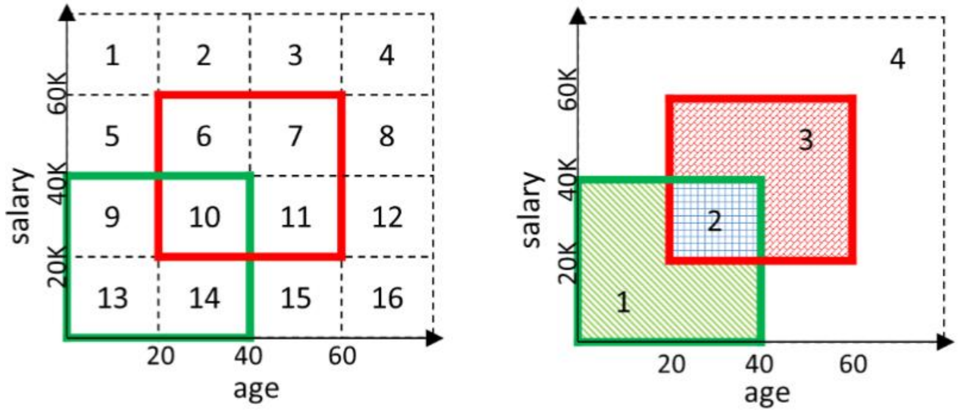
改进后：基于约束条件动态划分，分区数与约束复杂度相关。

数据库包含 t 个表，每个表有 pi个属性（对于第 i个表），约束的数量为 m，且每个约束 Cj 涉及dj个属性（dj << pi）

DataSynth：每个属性 Aj在分区时被划分为 r个区间，对于每个表 Ri中的 pi个属性，变量个数为 r^{pi}，t个表：$O(r^{\sum_{i=1}^{t} p_i})$

Hydra：$O\left(r^{\sum_{j=1}^{m} d_j}\right)$

|age < 40 ∧ salary < 40K (Person)| = 1000
|20 ≤ age < 60 ∧ 20K ≤ salary < 60K (Person)| = 2000
|Person| = 8000



(a) Grid-Partitioning

(b) Region-Partitioning

$$x_9 + x_{10} + x_{13} + x_{14} = 1000$$
$$x_6 + x_7 + x_{10} + x_{11} = 2000$$
$$x_1 + x_2 + \ldots + x_{16} = 8000$$

(a) Grid-Partitioning

$$y_1 + y_2 = 1000$$
$$y_2 + y_3 = 2000$$
$$y_1 + y_2 + y_3 + y_4 = 8000$$

(b) Region-Partitioning

HYDRA Summary Generator

原方法：视图(A, B, C) 分成子视图(A, B) and (B, C), 计算Prob(A, B) and Prob(C |B)，然后依次采样生成tuple。问题是计算开销巨大和采样带来的误差

改进后：按照拓扑顺序合并视图，生成数据库摘要，用summary取代概率实例化避免采样误差

构建Summary Generator过程：

(1) Constructing a solution for complete views

(2) Instantiating view summaries

(3) Making view summaries consistent wrt each other

(4) Extracting relation summaries from view summaries

A	B
1	1
1	1
1	2
1	5

B
2
2
5
5
5
1

DataSynth

$V_1(A, B)$ $V_2(B)$

R		
R_pk	S_fk	T_fk
1 – 30K	321	1
30001 – 50K	621	601
50001 – 60K	71	601
60001 – 70K	121	1
70001 – 80K	1	1

S		
S_pk	A	B
1–100	0	15
101–250	20	15
251–500	20	10
501–700	0	5

T	
T_pk	C
1–600	0
601–1500	2

Hydra

Figure 5: Example Database Summary

A	B	NumTuples
[60, inf)	[0,inf)	30K
[40, 60)	[0, 5) U [15, inf)	20K
[40, 60)	[5, 15)	10K
[20, 40)	[5, 10)	10K
[20, 40)	[15, inf)	10K

A	C	NumTuples
[60, inf)	[0, inf)	30K
[40, 60)	[2, 3)	30K
[20, 40)	[0, 2) U [3, inf)	20K

(a) Sub-view Solution

A	B	NumTuples
[60, inf)	[0,inf)	30K
[40, 60)	[0, 5) U [15, inf)	20K
[40, 60)	[5, 15)	10K
[20, 40)	[5, 10)	10K
[20, 40)	[15, inf)	10K

A	C	NumTuples
[60, inf)	[0, inf)	30K
[40, 60)	[2, 3)	20K
[40, 60)	[2, 3)	10K
[20, 40)	[0, 2) U [3, inf)	10K
[20, 40)	[0, 2) U [3, inf)	10K

(b) View Alignment

A	B	C	NumTuples
[60, inf)	[0,inf)	[0, inf)	30K
[40, 60)	[0, 5) U [15, inf)	[2, 3)	20K
[40, 60)	[5, 15)	[2, 3)	10K
[20, 40)	[5, 10)	[0, 2) U [3, inf)	10K
[20, 40)	[15, inf)	[0, 2) U [3, inf)	10K

(c) Merged View Solution

Figure 8: Align and Merge Example

HYDRA Tuple Generator

原方法：物化数据，生成到磁盘。

改进后：利用数据库摘要在查询执行时动态生成(查询)数据

K_1	A	FK_2
1	1	6
2	1	6
3	1	1
4	1	4

$R_1(K_1, A, FK_2)$

K_2	B
1	2
2	2
3	5
4	5
5	5
6	1

$R_2(K_2, B)$

R		
R_pk	S_fk	T_fk
1 – 30K	321	1
30001 – 50K	621	601
50001 – 60K	71	601
60001 – 70K	121	1
70001 – 80K	1	1

S		
S_pk	A	B
1–100	0	15
101–250	20	15
251–500	20	10
501–700	0	5

T	
T_pk	C
1–600	0
601–1500	2

Figure 5: Example Database Summary

HYDRA Experiments

复杂负载：100GB TPC-DS，131条不同的查询； Datasynth变量数: billion

简单负载：99条查询，只涉及non-key filter predicates 和 PK-FK joins; Datasynth变量数: million

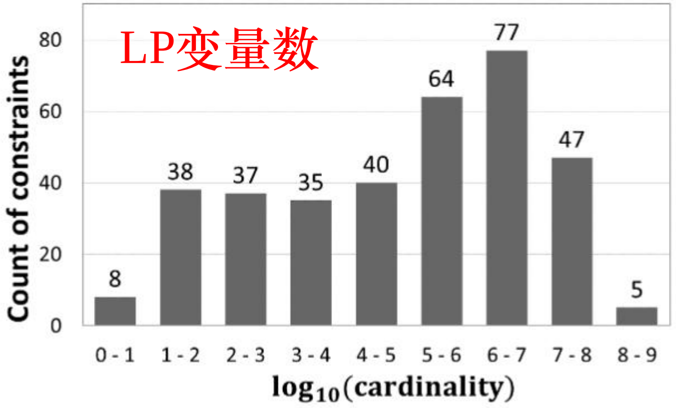


Figure 9: Distribution of Cardinality in CCs (*WL_c*)

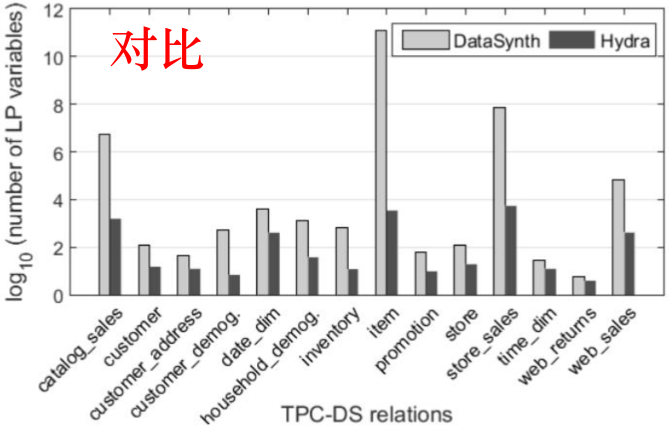


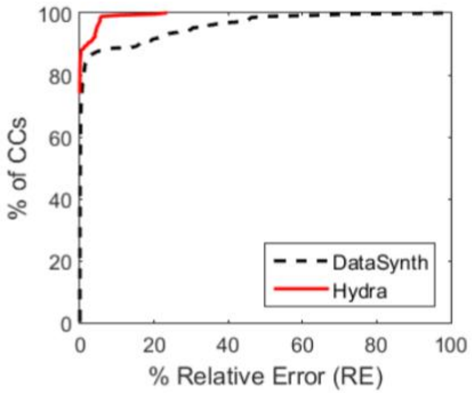
Figure 12: Number of variables in the LP (WL_c)

Complex Workload (WL_c)		Simple Workload (WL_s)	
DataSynth	**Hydra**	**DataSynth**	**Hydra**
crash	58 sec	50 min	13 sec

Figure 13: LP Processing Time

Size (in GB)	DataSynth	Hydra
10	4 hours	2 min
100	42 hours	11 min
1000	> 1 week	1.6 hours

Figure 14: Data Materialization Time



误差

Figure 10: Quality of Volumetric Similarity (*WL_c*)

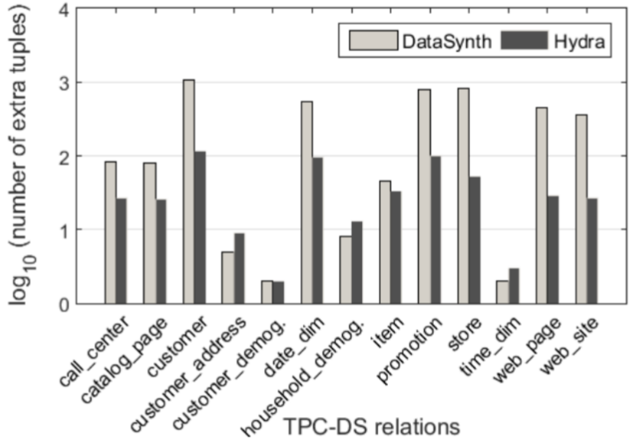


Figure 11: Extra tuples for Referential Integrity (*WL_c*)

Rel. Name	Size (in GB)	Row count (in millions)	Scan time (secs)	
			Disk	**Dynamic**
store_returns	3	29	16	8
web_sales	10	72	43	25
inventory	19	399	107	74
catalog_sales	20	144	46	48
store_sales	34	288	168	87

Figure 15: Data Supply Times

动态生成元组

DataSynth & HYDRA Summary

DCGen类工作

优点： ① 变量数量和约束数量相关，性能较好 ② 保真度高，90%+满足基数约束，其余误差也在10%以内 ③约束支持灵活(可以把其他任务转成约束加进来)

缺点： ①没直接把连接建模为LP，而是依赖于视图转化多表连接为单表问题，需要从父表到子表的单向依赖关系 ②高重复率 ③不能处理非线性的约束，故不支持算术选择算子

维度	DataSynth	HYDRA
算子支持	SELECT, JOIN, PROJECTION	SELECT, JOIN, PROJECTION
Fidelity（数据一致性）	受采样误差影响，可能存在偏差	精确控制，无采样误差，数据一致性高
Efficiency（效率）	LP 求解优化，但在复杂约束下效率降低	LP 优化与最大团分解相结合，效率更高
TB 数据生成	支持，但生成过程受存储和计算瓶颈限制	高效，能生成 TB 级数据且扩展性更好
Flexibility（灵活性）	支持自定义约束和扩展性，但受限于 LP 的规模	支持动态适应复杂模式和多种约束
Scalability（可扩展性）	高维和复杂约束下扩展性受限，变量数量快速增长	可扩展性更好，变量数量优化，适应高维数据
Theoretical Error（理论误差）	存在采样误差和完整性误差	基于确定性生成，理论误差更低