Counting Incomplete Databases

Marcelo Arenas, Pablo Barceló, Mikaël Monet
Millennium Institute for Foundational Research on Data

Millennium Institute Foundational Research on Data

Incomplete databases

- Most common way of dealing with missing values in relational databases:

Room allocation

Marcelo	NULL	07-08-19
Jorge	1003	08-08-19
Isabella	502	07-08-19
Pablo	NULL	07-08-19
Sofía	502	NULL
	⋮	

Room view

100	Parking
502	Sea
1004	Sea
1005	Parking
⋮	

→ In general, it is hard to reason about uncertain values because these might define an exponential number of possible complete databases
- Decision problems have been studied already (*certainty*, *possibility*, etc.)

→ What if we want *quantitative* information?

Our problems

- Relational databases with named nulls, and finite domains for each null [1]

$$D = \begin{array}{cc} \multicolumn{2}{c}{S} \\ \hline a & b \\ \mathrm{NULL}_1 & a \\ a & \mathrm{NULL}_2 \end{array} + \begin{array}{l} \mathrm{dom}(\mathrm{NULL}_1) = \{a,b,c\} \\ \mathrm{dom}(\mathrm{NULL}_2) = \{a,b\} \end{array}$$

- A *valuation* ν of D assigns a constant $\nu(\mathrm{NULL}_i) \in \mathrm{dom}(\mathrm{NULL}_i)$ to every null. Each valuation ν defines a *completion* of D, written $\nu(D)$:

$\nu(\mathrm{NULL}_1), \nu(\mathrm{NULL}_2)$	(a,a)	(a,b)	(b,a)	(b,b)	(c,a)	(c,b)
$\nu(D)$	S: a b; a a	S: a b; a a	S: a b; b a; a a	S: a b; b a	S: a b; c a; a a	S: a b; c a

- Let q be a *Boolean query*, i.e., a query that a complete database can either *satisfy* or *violate*
- We consider the following two *counting problems*:
  1. $\mathrm{CountVals}(q)$: INPUT: an incomplete database D. OUTPUT: the number of valuations ν of D such that $\nu(D)$ satisfies q
  2. $\mathrm{CountVals}(q)$: INPUT: an incomplete database D. OUTPUT: the number of completions of D that satisfy q
- Example: let q be the Boolean conjunctive query $q = \exists x\, S(x,x)$. Given as input the incomplete database above, $\mathrm{CountVals}(q)$ answers **4** and $\mathrm{CountCompl}(q)$ answers **3**.

Objectives

Study the *data complexity* of $\mathrm{CountVals}(q)$ and $\mathrm{CountCompl}(q)$ for diverse classes of Boolean queries (self-join–free CQs, CQs, UCQs, FO, SO, etc.). When is it tractable? When is it not? When can we approximate?

Relevant complexity classes and results

- Class FP: function problems that can be solved in polynomial time
- Class #P: count the number of accepting computation paths of a nondeterministic Turing machine running in polynomial time
- Class Span-P: count the number of distinct outputs of a nondeterministic transducer running in polynomial time
- Class Span-L: count the number of distinct outputs of an NL transducer
- Fully Polynomial-time Randomized Approximation Scheme (FPRAS): a randomized algorithm to efficiently approximate a counting problem
  - → **Theorem**: every function in Span-L admits a FPRAS [2]
  - → **Theorem**: counting the number of independent sets in a graph (#IS) has no FPRAS unless NP=RP [3]

First observations

1. If there are a bounded number of nulls, then $\mathrm{CountVals}(q)$ and $\mathrm{CountCompl}(q)$ are PTIME equivalent to the model checking problem for q (written $\mathrm{MC}(q)$)
2. If $\mathrm{MC}(q)$ is in P then $\mathrm{CountVals}(q)$ is in #P
3. If $\mathrm{MC}(q)$ is in NP then both $\mathrm{CountVals}(q)$ and $\mathrm{CountCompl}(q)$ are in Span-P
4. If q is *monotone*, has the *bounded models property*, and $\mathrm{MC}(q)$ is in nondeterministic linear space, then $\mathrm{CountVals}(q)$ is in Span-L
   - → **Proposition**: $\mathrm{CountVals}(q)$ is in Span-L (hence has a FPRAS) for any UCQ

Some results

- A dichotomy of $\mathrm{CountVals}(q)$ when q is a self-join–free conjunctive query:
  - → **Proposition**: if there is a variable that occurs at least twice in q then $\mathrm{CountVals}(q)$ is #P-complete. Otherwise $\mathrm{CountVals}(q)$ is in FP
  - ▷ The simplest hard queries: $\exists x\, R(x,x)$ and $\exists x\, R(x), S(x)$
- Counting the number completions is harder than counting valuations!
  - → **Proposition**: counting the number of completions of a unary table is #P-hard, and has no FPRAS unless NP=RP
  - ▷ Parsimonious reduction from #IS
- A query for which our problems are Span-P–complete:
  - → **Proposition**: there exists a query q with $\mathrm{MC}(q)$ in NP such that $\mathrm{CountVals}(q)$ and $\mathrm{CountCompl}(q)$ are Span-P–complete
  - ▷ Reduction from counting the number of Hamiltonian subgraphs of a graph

Work in progress

- Dichotomies for CQs? (This is usually much harder to obtain)
- A query q with $\mathrm{MC}(q)$ in P such that $\mathrm{CountCompl}(q)$ is Span-P–complete?
- Study *uniform* variants of our problems, where all the nulls share the same domain
  - → For instance, here counting the completions of a unary table is in FP!

Related problems

- Decision problems for incomplete databases (*membership*, *possibility*, *certainty*, etc.) [1]
- Block-independent probabilistic databases [4]
- Counting database repairs under primary keys [5]

References

[1] Tomasz Imieliński and Witold Lipski, Jr.
Incomplete Information in Relational Databases.
J. ACM, 31(4):761–791, 1984.

[2] Marcelo Arenas, Luis Alberto Croquevielle, Rajesh Jayaram, and Cristian Riveros.
Efficient Logspace Classes for Enumeration, Counting, and Uniform Generation.
In *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 59–73. ACM, 2019.

[3] Martin Dyer, Alan Frieze, and Mark Jerrum.
On counting independent sets in sparse graphs.
SIAM Journal on Computing, 31(5):1527–1541, 2002.

[4] Nilesh Dalvi, Christopher Re, and Dan Suciu.
Queries and materialized views on probabilistic databases.
Journal of Computer and System Sciences, 77(3):473–490, 2011.

[5] Dany Maslowski and Jef Wijsen.
Counting Database Repairs that Satisfy Conjunctive Queries with Self-Joins.
In *ICDT*, pages 155–164, 2014.