Parameterized and Fine-Grained Analysis of Query Evaluation Over Bag PDBs

Su Feng ✉
Illinois Institute of Technology, Chicago, USA

Boris Glavic ✉
Illinois Institute of Technology, USA

Aaron Huber ✉
University at Buffalo, USA

Oliver Kennedy ✉
University at Buffalo, USA

Atri Rudra ✉
University at Buffalo, USA

Abstract

The problem of computing the marginal probability of a tuple in the result of a query over set-probabilistic databases (PDBs) is a fundamental problem in set-PDBs. In this work, we study the analog problem for bag semantics: computing a tuple's expected multiplicity exactly and approximately. We are specifically interested in the fine-grained complexity and how it compares to the complexity of deterministic query evaluation algorithms — if these complexities are comparable, it opens the door to practical deployment of probabilistic databases. Unfortunately, our results imply that computing expected multiplicities for Bag-PDBs based on the results produced by such query evaluation algorithms introduces super-linear overhead (under parameterized complexity hardness assumptions/conjectures). We proceed to study approximation of expected multiplicities of result tuples of positive relational algebra queries ($\mathcal{RA}^+$) over c-TIDBs and for a non-trivial subclass of block-independent databases (BIDBs). We develop a sampling algorithm that computes a $(1 \pm \epsilon)$-approximation of the expected multiplicity of an output tuple in time linear in the runtime of a comparable deterministic query for any $\mathcal{RA}^+$ query.

By def multiplicity implies disjoint property

2012 ACM Subject Classification Information systems → Incomplete data

Keywords and phrases PDB, bags, polynomial, boolean formula, etc.

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

important special case

1 Introduction

a   as follows

This work explores the problem of computing ~~the expectation of~~ a tuple's multiplicity in a ~~bag TIDB. Our analysis specifically considers~~ a restricted form of bag TIDB, which we call a c-TIDB. A c-TIDB, $\mathcal{D} = \left(\{0, \ldots, c\}^D, \mathcal{P}\right)$ encodes a bag of uncertain tuples such that each tuple models c disjoint events, where each such set of disjoint events is itself independent of the others. A tuple in $\mathcal{D}$ has a multiplicity of at most c. The set of all worlds is encoded in $\{0, \ldots, c\}^D$, which is the set of all vectors of length $|D|$ such that each index corresponds to a distinct $t \in D$ storing its multiplicity. $\mathcal{P}$ is ~~the~~ a product distribution over the set of all worlds. A given world $\mathbf{W} = \{0, \ldots, c\}$ can be interpreted for each $\mathbf{W}[t]$ ~~$[i] = j$ as denoting that tuple t_i appears j times in world $\mathbf{W}$ for $j \in [0, c]$~~. The resulting product distribution can then be ~~expressed across the n base tuples of the~~ encoding as $p_{t,j} = Pr\left[W[t] = j\right]$, where each distribution is independent for $t \in D$. Allowing for $\leq c$ multiplicities across all tuples gives rise to having $\leq (c+1)^n$ possible worlds instead of the usual 2^n possible worlds ~~of~~ the traditional set TIDB. In this work, it is natural to be specifically considering bag query

is the mult. of t in

Do we really need this explanation. Just jump to multiplicity?

Better to say that 1-TIDB is the traditional set TIDB.

$t \in D$,

Why is it natural?

42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:64
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

on $T^*(Q,\mathcal{D})$ for

of computing ~~expected~~ mult. of a result tuple

Does this need to be defined since $\mathbf{D}$ is not a set DB anymore

semantics.

We can formally state ~~this~~ our problem as:

▶ **Problem 1.1.** *Given a c-TIDB $\mathcal{D} = \left(\{0,\ldots,c\}^D, \mathcal{P}\right)$, $\mathcal{RA}^+$ query Q, and result tuple t, compute the expected multiplicity of t: $\mathbb{E}_{\mathbf{D}\sim\mathcal{P}}[Q(\mathbf{D})(t)]$.*

Also need to justify why computing expected. mult. is imp/useful

Aaron says: I *think* we use $\mathbf{D}$ to denote something different in one of the proofs. Have to keep an eye open for this to avoid overloading notation.

Re-state this as saying we'll assume $c = O(1)$.

We upperbound the multiplicity of tuples in a c-TIDB since this is what typically seen in practice. Allowing for unbounded c is an interesting open problem.

poly time

Hardness of Set Query Semantics and Bag Query Semantics. Set query evaluation semantics over 1-TIDBs have been studied extensively, and the data complexity of the problem in general has been shown by Dalvi and Suicu to be #P-hard [13]. For our setting, there exists a trivial algorithm to compute problem 1.1 for any query over a c-TIDB due to linearity of expection by Simply perform the probability computations in a 'sum-of-products' fashion. This is made more precise when we discuss polynomial equivalence in the following subsection. Since we can compute problem 1.1 in polynomial time, the interesting question that we explore deals with hardness of computing expectation using fine-grained analysis and parameterized complexity. Specifically, we ask if Problem 1.1 can be solved in time ~~One of the main theoretical points~~ in this work ~~is to discern whether or not bag c-TIDB query semantics are indeed~~ linear in the runtime of an equivalent deterministic query. If this is ~~true, then this would open up the~~ way for deployment of c-TIDBs in practice. Unfortunately, we prove that this is not the case. To analyze this question we denote by $T^*(Q,\mathcal{D})$ the optimal runtime complexity of computing problem 1.1 over c-TIDB $\mathcal{D}$. Let ~~$\overline{D}$ denote the set of tuples in $\mathcal{D}$~~, i.e.,

I don't think so

explain that we are interested in the exponent of poly runtime.

This change needs to be PROPAGATED.

▶ **Definition 1.2** ($\overline{D}$). *Define $\overline{D}$ to be the set of tuples appearing across all the possible worlds of a c-TIDB, formally $\overline{D} = \left\{t_i \mid \forall \mathbf{W} \in \{0,\ldots,c\}^D, \forall i \in |D| : \mathbf{W}[i] > 0\right\}$. When a specific $\mathcal{D} = \left(\{0,\ldots,c\}^D, \mathcal{P}\right)$ is being referred to, we will use D to denote the set of tuples.*

I.e. many of the subsequent D should be say our $\overline{D}$

This should also include c as a param

+PROPAGATE

Let $T^*_{det}(Q,\overline{D})$ be the optimal runtime (with some caveats; discussed in sec. 2.4) of query Q on a comparable deterministic database $\overline{D}$ defined next. ?

At this point say $T^*(Q,\mathcal{D}) \le O(T^*_{det}(Q,\overline{D}))$

~~Table 1 shows our lower bounds for computing problem 1.1 on~~ c-TIDBs.

Our lower bound results. ~~In table 1 we show that~~ Specifically, depending on what hardness

Lower bound on $T^*(Q,\mathcal{D})$	Num. $\mathcal{P}$s	Hardness Assumption
$\Omega\left(\left(T^*_{det}(Q,\overline{D})\right)^{1+\epsilon_0}\right)$ for *some* $\epsilon_0 > 0$	Single	Triangle Detection hypothesis
$\omega\left(\left(T^*_{det}(Q,\overline{D})\right)^{C_0}\right)$ for *all* $C_0 > 0$	Multiple	#W[0] ≠ #W[1]
$\Omega\left(\left(T^*_{det}(Q,\overline{D})\right)^{c_0\cdot k}\right)$ for *some* $c_0 > 0$	Multiple	Conjecture 3.2

for $= (\{0,1\}^D, \mathcal{P})$

Table 1 Our lower bounds for a specific hard query Q parameterized by k. ~~The~~ $\mathcal{D}$ is over the same (family of) $\overline{D}$ and those with 'Multiple' in the second column need the algorithm to be able to handle multiple $\mathcal{P}$ (for a given $\overline{D}$). The last column states the hardness assumptions that imply the lower bounds in the first column (ϵ_o, C_0, c_0 are constants that are independent of k).

result/conjecture we assume, we get various emphatic versions of *no* as an answer to our question. To make some sense of the other lower bounds in Table 1, we note that it is not too hard to show that $T^*(Q,\mathcal{D}) \le O\left(\left(T^*_{det}(Q,\overline{D})\right)^k\right)$, where k is the largest degree of the

there is no notion of "degree" of a query as far as I know.

Spend a sentence or two on the implication of this result.

$$\Phi[\pi_A(Q),\overline{D},t] = \sum_{t':\pi_A(t')=t} \Phi[Q,\overline{D},t'] \qquad \Phi[Q_1 \cup Q_2,\overline{D},t] = \Phi[Q_1,\overline{D},t] + \Phi[Q_2,\overline{D},t]$$

$$\Phi[\sigma_\theta(Q),\overline{D},t] = \begin{cases}\Phi[Q,\overline{D},t] & \text{if } \theta(t)\\ 0 & \text{otherwise.}\end{cases} \qquad \begin{aligned}\Phi[Q_1 \bowtie Q_2,\overline{D},t] =& \Phi[Q_1,\overline{D},\pi_{attr(Q_1)}t]\\ &\cdot \Phi[Q_2,\overline{D},\pi_{attr(Q_2)}t]\\ \Phi[R,\overline{D},t] =& X_t\end{aligned}$$

Figure 1 Construction of the lineage (polynomial) for an $\mathcal{RA}^+$ query over a c-TIDB, where $\mathbf{X}$ consists of all X_t over all R in $\overline{D}$ and t in R. Here $\overline{D}.R$ denotes the instance of relation R in $\overline{D}$. Please note, after we introduct the reduction to 1-BIDB, the base case will be expressed alternatively.

not defined

this is not a standard term I think

query Q (i.e., join width) over all result tuples t (and the parameter that defines our family of hard queries).

What our lower bound in the third row says is that one cannot get more than a polynomial improvement over essentially the trivial algorithm for problem 1.1. However, this result assumes a hardness conjecture that is not as well studied as those in the first two rows of the table (see Sec. 3 for more discussion on the hardness assumptions). Further, we note that existing results already imply the claimed lower bounds if we were to replace the $T^*_{det}(Q,\overline{D})$ by just $|\overline{D}|$ (indeed these results follow from known lower bound for deterministic query processing). Our contribution is to then identify a family of hard queries where deterministic query processing is 'easy' but computing the expected multiplicities is hard.

time

Our upper bound results. We introduce an $(1\pm\epsilon)$-approximation algorithm that computes problem 1.1 in $O_\epsilon(T^*_{det}(Q,D))$. In contrast, known approximation techniques ([38, 30]) in set-PDBs need time $\Omega(|C|^{2k})$ (see Appendix G). Further, we generalize the PDB data model considered by the approximation algorithm to a class of bag-Block Independent Disjoint Databases (see Sec. 2.1.1) (BIDBs).

restate by saying our algo works for a more general notion of bag PDBs beyond c-TIDBs.

1.1 Polynomial Equivalence

Ref to Fig 1 here

A common encoding of probabilistic databases (e.g., in [28, 27, 5, 2] and many others) relies on annotating tuples with lineages, propositional formulas that describe the set of possible worlds that the tuple appears in. The bag semantics analog is a provenance/lineage polynomial $\Phi[Q,\overline{D},t]$ [25], a polynomial with non-zero integer coefficients and exponents, over integer variables $\mathbf{X}$ encoding input tuple multiplicities.

We drop Q, $\overline{D}$, and t from $\Phi[Q,\overline{D},t]$ when they are clear from the context or irrelevant to the discussion. We now specify the problem of computing the expectation of tuple multiplicity in the language of lineage polynomials:

All the $\overline{D}$ should be D here.

should be D

▶ **Problem 1.3** (Expected Multiplicity of Lineage Polynomials). *Given an $\mathcal{RA}^+$ query Q, c-TIDB $\mathcal{D}$ and result tuple t, compute the expected multiplicity of the polynomial $\Phi[Q,\overline{D},t]$ (i.e., $\mathbb{E}_{\mathbf{W}\sim\mathcal{P}}\left[\Phi[Q,\overline{D},t](\mathbf{W})\right]$, where $\mathbf{W} \in \{0,\ldots,c\}$).*

We note that computing Problem 1.1 is equivalent to computing Problem 1.3 (see Proposition 2.1).

1.2 Our ~~Machinery~~ Techniques

1?

Lower Bound Proof Techniques. All of our results rely on working with a *reduced* form of the lineage polynomial Φ. In fact, it turns out that for the 1-c-TIDB case, computing

move this to line 152

Q: Do we need to notion of 1-BIDB to define reduced poly?

Proposal: Expand the example to have general mult maybe have fewer tuples in example. You can define $\widetilde{\Phi}$ directly by defining the distribution over p' over $X_{t,j}$

the expected multiplicity (over bag query semantics) is *exactly* the same as evaluating this reduced polynomial over the probabilities that define the TIDB. This is also true when the query input(s) is a block independent disjoint probabilistice database (with tuple multiplicity of at most 1), which we refer to as a 1-BIDB. For our results to be applicable to c-TIDBs, we introduce the following reduction.

This is hard to follow since 1-BIDB has not been defined.

▶ **Definition 1.4.** *Any c-TIDB $\mathcal{D}$, can be reduced to an equivalent 1-BIDB $\mathcal{D}'$ in the following manner. For each $t_i \in D$, create a block of $c+1$ disjoint BIDB tuples in $\mathcal{D}'$ such that each tuple in the newly formed block is mapped to its own boolean variable $X_{i,j}$ for $i \in |D|$ and $j \in [c+1]$. Then, given $\mathbf{W} \in \{0,\ldots,c\}^D$, the equivalent world in $\mathcal{D}'$ will set each variable $X_{i,j} = 1$ for each $\mathbf{W}[i] = j$, while (for $\ell \neq j$) all other $X_{i,\ell} \in \mathbf{X}$ of $\mathcal{D}'$ are set to 0.*

▶ **Example 1.5.** *Consider the **Route** relation of fig. 2 and query $Q = \pi_{City_1}(\textbf{Route})$. The output relation Q is $\{\langle Chicago, X\rangle, \langle Chicago, Y\rangle\}$ and can be represented as a c-TIDB $Q' = \{\langle Chicago, X', 2\rangle\}$, where the following probabilities are true: $Pr[X' = 0] = Pr[\neg X \wedge \neg Y]$, $Pr[X' = 1] = Pr[(X \vee Y) \wedge (\neg X \vee \neg Y)]$, and $Pr[X' = 2] = Pr[X \wedge Y]$. Q' can then be reduced to a 1-BIDB by creating a block of the following disjoint tuples: $Q'' = \{\langle Chicago, X'_0\rangle, \langle Chicago, X'_1\rangle, \langle Chicago, X'_2\rangle\}$ such that $Pr[X'_i = 1] = Pr[X' = i]$.*

Next, we motivate this reduced polynomial. Consider the query Q defined as follows over the bag relations of Fig. 2:

```
SELECT 1 FROM OnTime a, Route r, OnTime b
WHERE a.city = r.city1 AND b.city = r.city2
```

defining/using BIDB

It can be verified that $\Phi(A, B, C, E, X, Y, Z)$ for the sole result tuple (i.e. the count) of Q is $AXB + BYE + BZC$. Now consider the product query $Q^2 = Q \times Q$. The lineage polynomial for Q^2 is given by $\Phi^2(A, B, C, E, X, Y, Z)$

so this implicitly defines the BIDB but perhaps get by w/o

can without no longer true as if Φ in $\{0..c\}$ instead $\{0,1\}$

$$= A^2X^2B^2 + B^2Y^2E^2 + B^2Z^2C^2 + 2AXB^2YE + 2AXB^2ZC + 2B^2YEZC.$$

The entire example needs to be re-worked to handle mult >1.

By exploiting linearity of expectation, further pushing expectation through independent variables and observing that for any $W \in \{0,1\}$, we have $W^2 = W$, the expectation is $\underset{\mathbf{W}\sim\mathcal{P}}{\mathbb{E}}\left[\Phi^2(\mathbf{W})\right]$ (where W_A is the random variable corresponding to A, distributed by $\mathcal{P}$)

$$\mathbb{E}[W_A]\mathbb{E}[W_X]\mathbb{E}[W_B] + \mathbb{E}[W_B]\mathbb{E}[W_Y]\mathbb{E}[W_E] + \mathbb{E}[W_B]\mathbb{E}[W_Z]\mathbb{E}[W_C] + 2\,\mathbb{E}[W_A]\mathbb{E}[W_X]\mathbb{E}[W_B]\mathbb{E}W_Y\,\mathbb{E}[W_E]$$
$$+ 2\,\mathbb{E}[W_A]\mathbb{E}[W_Y]\mathbb{E}[W_B]\mathbb{E}[W_Z]\mathbb{E}[W_C] + 2\,\mathbb{E}[W_B]\mathbb{E}[W_Y]\mathbb{E}[W_E]\mathbb{E}[W_Z]\mathbb{E}[W_C].$$

This property leads us to consider a structure related to the lineage polynomial.

▶ **Definition 1.6.** *For any polynomial $\Phi(\mathbf{X})$ corresponding to a c-TIDB (henceforth, c-TIDB-lineage polynomial), define the reduced polynomial $\widetilde{\Phi}(\mathbf{X})$ to be the polynomial obtained by setting all exponents $e > 1$ in the standard monomial basis (SMB) [1] form of $\Phi(\mathbf{X})$ to 1.*

This is incorrect. You replace each X_t by $\sum_{j=1}^{c} j \cdot X_{t,j}$

With $\Phi^2(A, B, C, E, X, Y, Z)$ as an example, we have:

$$\widetilde{\Phi^2}(A, B, C, E, X, Y, Z) = AXB + BYE + BZC + 2AXBYE + 2AXBZC + 2BYEZC.$$

Note that we have argued that for our specific example the expectation that we want is $\widetilde{\Phi^2}(Pr(A = 1), Pr(B = 1), Pr(C = 1)), Pr(E = 1), Pr(X = 1), Pr(Y = 1), Pr(Z = 1))$. Lemma 1.7 generalizes the equivalence to *all* $\mathcal{RA}^+$ queries on c- TIDBs (proof in Appendix B.5).

[1] This is the representation, typically used in set-PDBs, where the polynomial is reresented as sum of 'pure' products. See Definition 2.2 for a formal definition.

This lemma should be for c-TIDBs:

▶ **Lemma 1.7.** *Let $\mathcal{D}$ be a* 1-BIDB *such that the probability distribution $\mathcal{P}$ over $\mathbf{W} \in \{0,1\}^{|D|}$ (the set of all worlds) is induced by the disjoint condition and the probability vector $\mathbf{p} = \left(p_1, \ldots, p_{|D|}\right)$ where $p_i = Pr\left(W_i = 1\right)$. For any* 1-BIDB*-lineage polynomial $\Phi\left(\mathbf{X}\right) = \Phi[Q, \overline{D}, t](\mathbf{X})$, it holds that $\mathbb{E}_{\mathbf{W}\sim\mathcal{P}}\left[\Phi\left(\mathbf{W}\right)\right] = \widetilde{\Phi}\left(\mathbf{p}\right)$.*

Sec 1.2 starts here

mention e.b for 1-TIDB

To prove our hardness result we show that for the same Q from the example above, for an arbitrary 'product width' k, the query Q^k is able to encode various hard graph-counting problems (assuming $O\left(n\right)$ tuples rather than the $O(1)$ tuples in Fig. 2). We do so by considering an arbitrary graph G (analogous to the *Route* relation of Q) and analyzing how the coefficients in the (univariate) polynomial $\widetilde{\Phi}\left(p, \ldots, p\right)$ relate to counts of subgraphs in G that are isomorphic to various graphs with k edges. E.g., we exploit the fact that the leading coefficient in Φ corresponding to Q^k is proportional to the number of k-matchings in G, a known hard problem in parameterized/fine-grained complexity literature.

This should be under upper bound techniques

For an upper bound on approximating the expected count, it is easy to check that if all the probabilties are constant then $\Phi\left(p_1, \ldots, p_n\right)$ (i.e. evaluating the original lineage polynomial over the probability values) is a constant factor approximation. For example, using Q^2 from above, using p_A to denote $Pr\left[A = 1\right]$ (and similarly for the other variables), we can see that

$$\Phi^2\left(\mathbf{p}\right) = p_A^2p_X^2p_B^2 + p_B^2p_Y^2p_E^2 + p_B^2p_Z^2p_C^2 + 2p_Ap_Xp_B^2p_Yp_E + 2p_Ap_Xp_B^2p_Zp_C + 2p_B^2p_Yp_Ep_Zp_C$$
$$\leq p_Ap_Xp_B + p_Bp_Yp_E + p_Bp_Zp_C + 2p_Ap_Xp_Bp_Yp_E + 2p_Ap_Xp_Bp_Zp_C + 2p_Bp_Yp_Ep_Zp_C = \widetilde{\Phi}\left(\mathbf{p}\right)$$

If we assume that all seven probability values are at least $p_0 > 0$, we get that $\Phi^2\left(\mathbf{p}\right)$ is in the range $\left[\left(p_0\right)^3 \cdot \widetilde{\Phi}\left(\mathbf{p}\right), \widetilde{\Phi}\left(\mathbf{p}\right)\right]$. To get an $(1 \pm \epsilon)$-multiplicative approximation we uniformly sample monomials from the SMB representation of Φ and 'adjust' their contribution to $\widetilde{\Phi}\left(\cdot\right)$.

Upper Bound Techniques. Our negative results (Table 1) indicate that c-TIDBs can not achieve comparable performance to deterministic databases for exact results (under complexity assumptions). In fact, under plausible hardness conjectures, one cannot (drastically) improve upon the trivial algorithm to exactly compute the expected multiplicities for c-TIDBs. A natural followup is whether we can do better if we are willing to settle for an approximation to the expected multiplities. In the remainder of this section ~~work~~, we demonstrate that a $(1 \pm \epsilon)$ (multiplicative) approximation with competitive performance is achievable.

(even for c=1)

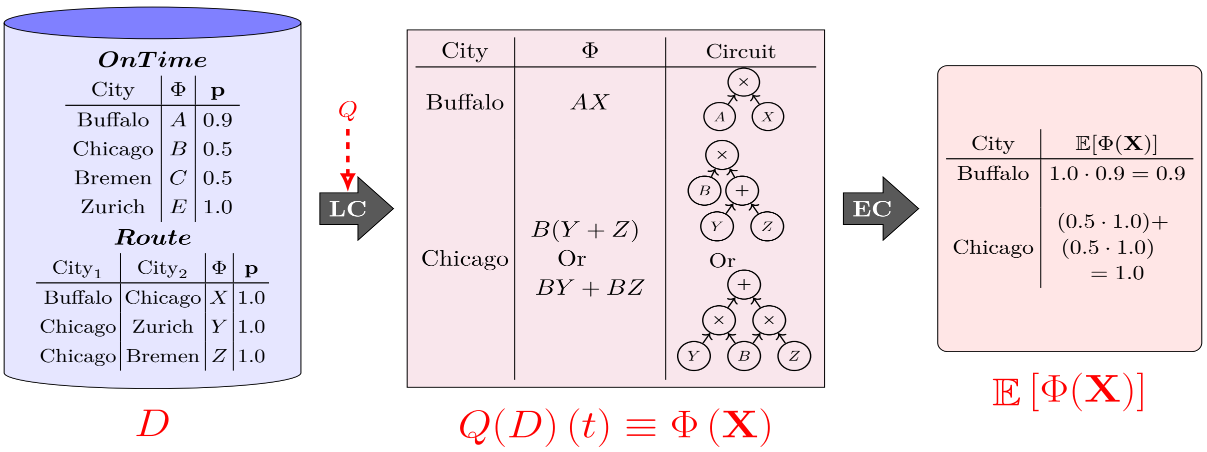

Figure 2 Intensional Query Evaluation Model ($Q = \pi_{\text{City}}\left(Route \bowtie_{\text{City}_1 = City} OnTime\right)$).

D

We adopt the two-step intensional model of query evaluation used in set-PDBs, as illustrated in Fig. 2: (i) Lineage Computation (LC): Given input D and Q, output every tuple t that possibly satisfies Q, annotated with its lineage polynomial ($\Phi(\mathbf{X}) = \Phi[Q, \overline{D}, t]\left(\mathbf{X}\right)$); (ii) Expectation Computation (EC): Given $\Phi(\mathbf{X})$ for each tuple, compute $\mathbb{E}\left[\Phi(\mathbf{W})\right]$. Let $T_{LC}(Q, D_{\overline{\Omega}}, \mathtt{C})$ denote the runtime of LC when it outputs $\mathtt{C}$ (which is a representation of Φ

D

include ϵ in def?

as an arithmetic circuit — more on this representation shortly). Denote by $T_{EC}(\mathtt{C})$ (recall $\mathtt{C}$ is the output of LC) the runtime of EC, allowing us to formally define our objective:

▶ **Problem 1.8** (Bag-c-TIDB linear time approximation)**.** *Given c-TIDB $\mathcal{D}$, $\mathcal{RA}^+$ query Q, is there a $(1 \pm \epsilon)$-approximation of $\mathbb{E}_{\mathbf{D}\sim\mathcal{P}_{\overline{\Omega}}}\left[Q\left(\mathbf{D}\right)(t)\right]$ for all result tuples t where $\exists \mathtt{C} : T_{LC}(Q, D, \mathtt{C}) + T_{EC}(\mathtt{C}) \leq O_{\epsilon}(T^*_{det}(Q, D))$?*

Some more work is needed to connect this to Problem 1.3

We show in Appendix E.2.1 an $O(T^*_{det}(Q, D))$ algorithm for constructing the lineage polynomial for all result tuples of an $\mathcal{RA}^+$ query Q (or more more precisely, a single circuit $\mathtt{C}$ with one sink per tuple representing the tuple's lineage). A key insight of this paper is that the representation of $\mathtt{C}$ matters. For example, if we insist that $\mathtt{C}$ represent the lineage polynomial in SMB, the answer to the above question in general is no, since then we will need $|\mathtt{C}| \geq \Omega\left((T^*_{det}(Q, D))^k\right)$, and hence, just $T_{LC}(Q, D, \mathtt{C})$ will be too large.

However, systems can directly emit compact, factorized representations of $\Phi(\mathbf{X})$ (e.g., as a consequence of the standard projection push-down optimization [23]). For example, in Fig. 2, $B(Y + Z)$ is a factorized representation of the SMB-form $BY + BZ$. Accordingly, this work uses (arithmetic) circuits[2] as the representation system of $\Phi(\mathbf{X})$.

Need some discussion on how we solve P1.9

Given that there exists a representation $\mathtt{C}^*$ such that $T_{LC}(Q, D, \mathtt{C}^*) \leq O(T^*_{det}(Q, D))$, we can now focus on the complexity of EC. We can represent the factorized lineage polynomial by its correspoding arithmetic circuit $\mathtt{C}$ (whose size we denote by $|\mathtt{C}|$). As we also show in Appendix E.2.2, this size is also bounded by $T^*_{det}(Q, D)$ (i.e., $|\mathtt{C}^*| \leq O(T^*_{det}(Q, D))$). Thus, the question of approximation can be reframed as:

reframed is not correct. P1.9 is not equivalent to P1.8 but it is sufficient

▶ **Problem 1.9** (Problem 1.8 reframed)**.** *Given one circuit $\mathtt{C}$ that encodes $\Phi[Q, \overline{D}, t]$ for all result tuples t (one sink per t) for* bag-PDB *$\mathcal{D}$ and $\mathcal{RA}^+$ query Q, does there exist an algorithm that computes a $(1 \pm \epsilon)$-approximation of $\mathbb{E}_{\mathbf{D}\sim\mathcal{P}_{\overline{\Omega}}}\left[Q\left(\mathbf{D}\right)(t)\right]$ (for all result tuples t) in $O\left(|\mathtt{C}|\right)$ time?*

Imp. to clearly state this.

Old Stuff

A probabilistic database (PDB) $\mathcal{D}$ is a pair $\left(\overline{\Omega}, \mathcal{P}_{\overline{\Omega}}\right)$, where $\overline{\Omega}$ is a set of deterministic database instances called possible worlds and $\mathcal{P}_{\overline{\Omega}}$ is a probability distribution over $\overline{\Omega}$. A tuple independent database (TIDB) (to which we will refer to later) is a PDB such that each tuple is an independent random event. A commonly studied problem in probabilistic databases is, given a query Q, PDB $\mathcal{D}$, and possible query result tuple t, to compute the tuple's *marginal probability* of being in the query's result, i.e., computing the expectation of a Boolean random variable over $\mathcal{P}_{\overline{\Omega}}$ that is 1 for every $D \in \overline{\Omega}$ for which $t \in Q(D)$ and 0 otherwise. In this work, we are interested in bag semantics, where each tuple is associated with a multiplicity. Following [25], we model bag databases (resp., relations) as functions from each t to the tuple's multiplicity $D(t) \in \mathbb{N}$ in a possible world D. ~~We refer to such a probabilistic database as a bag-probabilistic database or bag-PDB for short.~~

The natural generalization of the (set) problem of computing marginal probabilities of query result tuples to bag semantics is to compute the expectation of a random variable over $\mathcal{P}_{\overline{\Omega}}$ that is assigned value $Q(D)(t) \in \mathbb{N}$ in world $D \in \overline{\Omega}$, formally $\mathbb{E}_{\mathbf{D}\sim\mathcal{P}_{\overline{\Omega}}}\left[Q\left(\mathbf{D}\right)(t)\right]$.

[2] An arithmetic circuit is a DAG with variable and/or numeric source nodes and internal, each nodes representing either an addition or multiplication operator.