Sample Debiasing in the Themis Open World Database System

Laurel Orr, Magdalena Balazinska, Dan Suciu
ljorr1,magda,suciu@cs.washington.edu
University of Washington
Seattle, Washington

ABSTRACT

Open world database management systems assume tuples *not in* the database still exist and are becoming an increasingly important area of research. We present Themis, the first open world database that automatically rebalances arbitrarily biased samples to approximately answer queries as if they were issued over the entire population. We leverage apriori population aggregate information to develop and combine two different approaches for automatic debiasing: sample reweighting and Bayesian network probabilistic modeling. We build a prototype of Themis and demonstrate that Themis achieves higher query accuracy than the default AQP approach, an alternative sample reweighting technique, and a variety of Bayesian network models while maintaining interactive query response times. We also show that Themis is robust to differences in the support between the sample and population, a key use case when using social media samples.

ACM Reference Format:
Laurel Orr, Magdalena Balazinska, Dan Suciu. 2020. Sample Debiasing in the Themis Open World Database System. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (SIGMOD'20), June 14–19, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3318464.3380606

1 INTRODUCTION

Data samples are increasingly easy to access and analyze with the help of websites such as Facebook and Twitter and data repositories such as data.gov, data.world, and kaggle.com. Additionally, data analytic toolkits, like Python, are becoming more mainstream. These two factors have led to data science becoming tightly coupled with sample analysis.

Modern data scientists, however, face the added challenge that the data samples they seek to analyze are not always an accurate representation of the population they are sampled from. For example, social scientists today study migration patterns from Twitter samples [66], but Twitter users are a non-uniform subset of all people. This phenomenon is known as sample selection bias [19] and is problematic because it can lead to inaccurate analyses.

Correcting this bias, however, is difficult because the sampling mechanism in today's data sources, i.e. the probability of some population tuple being included in the sample, is typically not known. This means common techniques like the Horvitz-Thompson estimator [11] are not applicable.

There is, however, another increasingly available data source scientists leverage for debiasing: population aggregates. Along with the increase in the number of publicly available data samples, there is a recent push for more data transparency and reporting by corporations and governments, e.g. the United State's OPEN Government Data Act passed in 2018 [1] and the InFuse UK aggregate population statistics tool [2]. These reports are often in the form of population aggregate queries. For example, in the FBI's 2017 Internet Crime Report [3], they present a table showing a `GROUP BY, COUNT(*)` aggregate query over crime type, counting the number of victims in each crime type group.

These aggregates can facilitate data debiasing, but the process remains tedious and error prone. There is no general, automatic technique or system for debiasing using aggregates. With the ultimate goal of answering queries approximately over the population, data scientists are forced to manually implement one-off, specialized solutions [66] tailored towards specific datasets, such as census reports [51].

In this paper, we present Themis, which is, to our knowledge, the first open world database management system (OW-DMBS) that automates and encapsulates the debiasing process. The data scientist simply inserts a sample and aggregates and then asks queries, getting approximate results as if the queries were issued on the population. This novel query processing paradigm, which we call open world query processing (OWQP), is fundamentally different from other paradigms where queries are processed over populations (standard query processing) or *representative* samples (approximate query processing (AQP) or cardinality estimation).

Supporting OWQP and data debiasing remains unaddressed because DBMSs are traditionally built for closed world data. They only support samples for AQP where DMBSs leverage samples to achieve interactive speeds. However, AQP techniques [16, 23, 30, 50, 55, 57, 58] make one (or both) of the following assumptions. (1) They have access to the entire population, or (2) they have knowledge of the error from querying the sample. Neither of these assumptions hold in OWQP, making standard AQP techniques not applicable. The default AQP solution is therefore uniform reweighting, which is inaccurate.

As DBMSs become increasingly used by data scientists [4], however, they need to support OWQP to meet to the needs of these new users. Themis takes the first step in that direction.

To achieve our goal, at the heart of our system, we develop and combine two different debiasing techniques: reweighting the sample and learning the probability distribution of the population. The former allows us to more accurately answer heavy hitter queries while the later ensures we can answer queries about tuples that may not exist in the sample.

For sample reweighting, we investigate two different approaches: modifying linear regression and applying an existing aggregate fitting procedure. For learning the probability distribution, we utilize Bayesian networks to build an approximate population probability distribution. The uniqueness of our system is in not only building two separate debiasing techniques but also combining them into one unified system for query answering.

We build a prototype database system called Themis, named after the Greek titan for balance and order. Themis treats relations as samples and automatically corrects for sample bias using population-level aggregates. We evaluate Themis on three datasets to show that Themis is more accurate at answering point queries than the default AQP technique, linear regression reweighting, and a variety of Bayesian network probabilistic approaches. We further demonstrate that Themis can handle more advanced aggregate queries, depending on the apriori knowledge. In summary, the contributions of this paper are as follows:

- A new query processing paradigm (OWQP).
- The first OW-DMBS that automatically debiases data from a sample and population aggregates for OWQP (Sec. 3).
- The development and application of debiasing techniques and a novel hybrid approach integrating them (Sec. 4).
- Two optimization techniques for faster preprocessing: aggregate pruning and model simplification (Sec. 5).
- Detailed experiments on two datasets showing that Themis achieves a 70 percent improvement in the median error when compared to the default AQP approach when asking about heavy hitter tuples (Table 4, Sec. 6). We further show Themis is robust to differences in the support of the sample and the population.

2 MOTIVATING EXAMPLE

A data scientist is trying to estimate the number of flights under 30 min in different states of the United States in a year. She has a sample of all flights in the United States biased towards four major states, but she does not know how badly it is biased. Further, she has access to how many flights in total leave from each state. She decides to analyze this data and focus only on short flights on either the East or West Coast of the country.

Being a database user, she ingests the data into a SQL database. As this dataset is a sample, she has three choices for how to prepare her data for analysis: do nothing, uniformly rebalance (default AQP), or use state information to reweight flights based on the number of flights leaving each state. For the second option, she knows there are 7 million flights in the United States per year but only 700,000 in her sample. Therefore, she adds a `weight` attribute to the dataset and gives each tuple a weight of 10, indicating the each tuple in her sample represents 10 tuples in the real world. For the third option, if she knows there are N flights leaving from some state per year but only n leaving that state in her sample, she sets the weight of each flight from that state to be N/n.

After preprocessing, she starts issuing point queries of the form

```
SELECT SUM(weight) AS num_flights
FROM flights WHERE flight_time <= 30 min
AND origin_state = `<state>';
```

The results of a few queries are shown in Table 1 where Raw represents option one, AQP represents option two, US State represents option three, and Themis represents our system's answer. Themis and US State use the single aggregate to produce more accurate answers than Raw and AQP because they are correcting for the fact that some flights leaving the four major states are overrepresented in the sample. More importantly, Themis does the re-balancing automatically, which will become time consuming to do manually for more complex aggregates. Themis also answers queries about tuples not in the sample, like ME.

3 THEMIS MODEL

At a high level, Themis uses a sample and population aggregate data to build a model to perform OWQP. We use the term *model* because it encapsulates that we use both a reweighted sample and a probabilistic model to answer queries. Both techniques treat the aggregates as constraints to be satisfied.

Query	True	Raw	AQP	US State	Themis
CA	7855	2846	28460	7843	7843
FL	2	1	10	3	3
OH	119	1	10	70	70
ME	2	0	0	0	3

Table 1: Query results of the data scientists using the raw sample, a uniformly scaled sample, a state-scaled sample, and Themis.

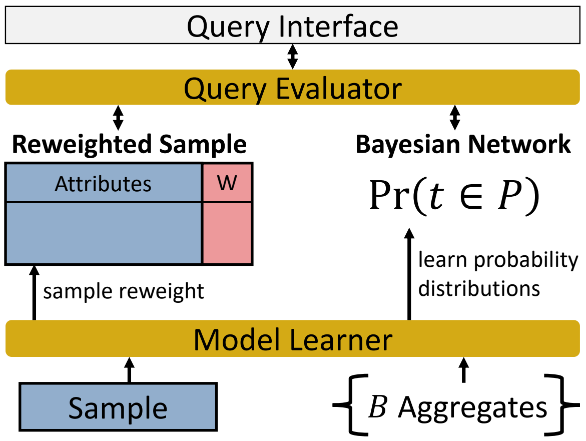

Figure 1: Architecture

Note that the population aggregates do not need to be exact. They may contain errors, be computed at different times, or be purposely perturbed. For example, the 2020 US census will add random noise to their reports to be differentially private [24]. Themis will still treat these aggregates as marginal constraints to be satisfied.

We assume there is a well defined, but unavailable population P of (approximate) size n with m attributes $\mathcal{A} = \{A_1, \ldots, A_m\}$. P is unavailable because either it does not exist (e.g. a dataset of all graduate students in the US) or is not released to the public (e.g. a hospital's private medical data). The active domain of each attribute A_i is assumed to be discrete and ordered[1].

We assume there is a sample S drawn independently but not uniformly from P of size n_S such that for each tuple $t \in P$, t has probability $\Pr_S(t) \geq 0$ of being included in S. This probability, however, is not known apriori.

Lastly, we have Γ, a set of results of B aggregate `COUNT(*)` queries of various dimensions computed over the population denoted

$$\Gamma = \{G_{\boldsymbol{\gamma}_i,\text{COUNT}(*)}(P) : i = 1, B\}$$

where $G_{\boldsymbol{\gamma}_i,\text{COUNT}(*)}(P)$ is an aggregate query of dimension d_i; i.e., $\boldsymbol{\gamma}_i \subseteq \mathcal{A}$ (see Example 3.1). Each aggregate query Γ_i returns a set of M_i attribute value-count pairs denoted

$$\Gamma_i = \{(\boldsymbol{a}_{i,k}, c_{i,k}) : k = 1, \ldots, M_i\}$$

where $\boldsymbol{a}_{i,k}$ is the vector of d_i attribute values associated with group k of aggregate i, and $c_{i,k}$ is the group's count. Further, we let $\bigcup_{i=1,B} \boldsymbol{\gamma}_i \subseteq \mathcal{A}$, meaning the aggregates may not cover all domain attributes.

Example 3.1. *Following the example from Sec. 2, assume the population P and sample S are the following sets of domestic flights in the United States.* `date` *is the month and* `o_st` *and* `d_st` *are origin and destination states, respectively.*

$P =$

date	o_st	d_st
01	FL	FL
01	FL	FL
02	FL	NY
01	NC	FL
02	NC	NY
02	NC	NY
02	NC	NY
01	NY	FL
01	NY	NC
02	NY	NY

$S =$

date	o_st	d_st
01	FL	FL
01	FL	FL
02	NC	NY
01	NY	NC

Let $\Gamma = \{\Gamma_1, \Gamma_2\}$ with $d_1 = 1$ and $d_2 = 2$ be the following two aggregate queries.

$$\Gamma_1 = G_{\texttt{date},\texttt{COUNT(*)}}(P) = \{([01], 5), ([02], 5)\}$$
$$\Gamma_2 = G_{\texttt{o\_st},\texttt{d\_st},\texttt{COUNT(*)}}(P) =$$
$$\{([FL, FL], 2), ([FL, NY], 1), ([NC, FL], 1),$$
$$([NC, NY], 3), ([NY, FL], 1), ([NY, NC], 1), ([NY, NY], 1)\}.$$

In this case, $n = 10$ and $B = 2$.

We are given a user query Q over the population. As we do not have P, we need to use S and Γ to perform OWQP and estimate $Q(P)$. While Q can be any SQL query, we focus on point[2] and `GROUP BY` queries to study the improvement in accuracy. To answer $Q(P)$, we build a model $\mathcal{M}(\Gamma, S)$ such that $Q(\mathcal{M}(\Gamma, S))$ is an approximate answer to $Q(P)$.

4 DATA DEBIASING

Themis's model $\mathcal{M}(\Gamma, S)$ has two components: a reweighted sample and a probabilistic model. We present each technique and then describe how Themis merges them into a unique hybrid approach for OWQP.

4.1 Sample Reweighting

In sample reweighting, each tuple $t \in S$ gets assigned a weight $w(t)$ indicating the number of tuples it represents in P. Queries get transformed to run on weighted tuples by, for

[1]We support continuous data types by bucketizing their active domains.

[2]We define a d-dimensional point query as `SELECT COUNT(*) FROM R WHERE A1 = v1 AND ... AND Ad = vd`.

example, translating `COUNT(*)` to be `SUM(weight)`. If the sampling mechanism, $\Pr_S(t)$, is known, we can use the Horvitz-Thompson estimator which reweights each tuple by $1/\Pr_S(t)$ [19, 50].

The challenge is that we do not have the sampling mechanism. The default approach used by standard AQP systems is to perform uniform reweighting by setting $w(t)$ to be $|P|/|S|$. When the sample is biased, this achieves low accuracy (see Sec. 6). To correct for the bias, we present two solutions: linear regression adaptation and Iterative Proportional Fitting (IPF) [37, 45].

4.1.1 Linear Regression Reweighting. Following propensity score research [10, 49, 59], we assume a tuple's weight depends on the attributes of t. In particular, we let $w(t)$ be a linear combination of its attributes; i.e., if $\boldsymbol{t}^{0/1}$ represents the one-hot encoded tuple t, then $w(t) = \boldsymbol{\beta} \cdot \boldsymbol{t}^{0/1}$ where $\boldsymbol{\beta}$ is a vector of weights.

To solve for $\boldsymbol{\beta}$, we set up a system of linear equations, one for each $c_{i,k}$. These equations enforce that, for aggregate i, group k of $G_{\boldsymbol{\gamma}_i,\text{COUNT(*)}}(S)$ equals $c_{i,k}$. Note that we enforce that $w(t) \geq 0$ so each tuple in S gets some representation. Details can be found in [56].

4.1.2 Iterative Proportional Fitting. An alternative approach to finding $w(t)$ is to assume every $w(t)$ is independent and can be solved for directly. Inspired by the technique of population synthesis in demography, we apply a technique called Iterative Proportional Fitting (IPF) [12, 21, 26, 45, 51, 61] to solve for $w(t)$. While IPF is not new, our approach of using IPF for arbitrary data debiasing is novel.

To briefly review, IPF is a iterative procedure for calibrating sample weights to match given population aggregates. It examines each individual aggregate iteratively and rescales the weights of the participating tuples if that aggregate is not satisfied. The process repeats until convergence[3]. For pseudocode and details, see [56].

4.2 Probabilistic Model Learning

It is important to understand when either of our two sample reweighting techniques will fail. For one, the Horvitz-Thompson estimator, which we are approximating by $w(t)$, assumes the support of the sample is the same as the population; i.e. $Pr_S(t) > 0 \; \forall t$. When this does not hold, e.g. when the sampling design is flawed, sample reweighting is inaccurate. Secondly, even if the support is the same, sample reweighting will fail when tuples exist in P but not in S because the sample will always say those tuples do not exist. This occurs with rare groups and small sample sizes. While we could impute missing rows to S, this risks losing important structural information (S gives us partial information about the manifold P lives on) and slowing down queries.

[3]If no convergent scaling exists, the algorithm may not converge and can only give an approximate reweighting.

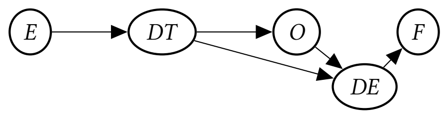

Figure 2: Example Bayesian network of flights in the United States (see Table 2 for abbreviations).

Our solution is to build a probabilistic model of P using S and Γ and answer queries using this model [22, 55][4]. When building a probabilistic model, the first consideration is what class of distributions to use. As we have no prior knowledge on the population, our main concern is choosing a distribution that can be learned from aggregate data. Similar to [61], we use a Bayesian network (BN) to model the population distribution as a Bayesian network is parameterized by aggregate queries and can scale to many attributes and large data [29]. Unlike [61], which builds the BN from the sample only, the novelty of our BN framework is that is merges S and Γ into BN learning.

4.2.1 Why Standard Bayesian Network Algorithms Do Not Apply. A Bayesian network is a probabilistic graphical model representing a set of random variables and their conditional dependencies through a directed, acyclic graph. Each edge represents a conditional dependency factor of the form $\Pr(X_i|Pa(X_i))$ where $Pa(X_i)$ are the parents of node X_i. The example in Fig. 2 represents the joint distribution is $\Pr(E, DT, O, DE, F) = \Pr(DT|E)\Pr(O|DT)\Pr(DE|O, DT)\Pr(F|DE)\Pr(E)$.

Standard BN learning algorithms assume access to the population P, meaning we fundamentally cannot use standard, black-box BN learning algorithms as we do not have access to P. We could learn the BN just using S, but that ignores the population aggregate data Γ. We cannot just use Γ as Γ may not have information on all the attributes. We need to combine S and Γ for the highest accuracy (we evaluate this hypothesis in [56]). However, doing so is non-trivial, especially for parameter learning.

To see this, take the flights example from Fig. 2. Assume we have some sample S and one 2D population aggregate over `DE` (destination) and `DT` (distance). The first step is to learn the structure. Structure learning algorithms are either greedy (altering edges one at a time) or constraint-based (using independence tests to find a satisfying structure). For constraint-based algorithms, it is unclear how to test for independence over the attributes `DE`, `DT`, and `E` as only two of them are covered in the aggregate; therefore, we must use

[4]In order to reason about the population probability distribution, we use the possible world semantics.

a greedy algorithm. We must modify the greedy algorithm, however, to incorporate information from both the sample and the aggregates. We show our solution to this in Sec. 4.2.2.

Parameter learning raises an even greater challenge than structure learning. Suppose we learn the structure shown in Fig. 2. Further, suppose the aggregate gives us that the probability of a flight distance of 500 miles is 0.20. Take the edge from E to DT. How do we learn the parameters for this edge's factor $\Pr(DT|E)$? We cannot learn the parameters from S as S may not have any flights traveling 500 miles. Further, how can we ensure that $\Pr(\texttt{DT} = 500\text{mi}) = \sum_{\texttt{E}} \Pr(E)\Pr(\texttt{DT} = 500\text{mi}|E)$ is equal to 0.20? This problem becomes more complex as we add aggregates. We solve this in Sec. 4.2.3.

4.2.2 Learning Network Structure. As mentioned above, we adapt the greedy hill-climbing algorithm [31, 47]. The traditional hill-climbing algorithm's goal is to find the structure that maximizes some score. At each step of the algorithm, it makes the "move" (adding, removing, or reversing a directed edge) that improves the score the most. If the score cannot be further improved, the algorithm terminates.

We modify the algorithm as follows. To focus on learning from the population before the sample, our algorithm runs in two phases: building from Γ and building from S. As Γ represents ground truth information, we want to build as many edges from Γ before adding edges from S. In the first phase, we make "moves" using Γ until all attributes from Γ are added to the network. Then, if there are any remaining attributes in S not in Γ, we use S to continue building. For details and pseudocode, see [56].

4.2.3 Learning Network Parameters. As mentioned above, it is non-trivial how to learn the parameters using S and Γ. Recall that BN parameters are learned by maximizing the likelihood of the data subject to the BN structure. Inspired by [20, 54] adding parameter sharing constraints to the BN parameter learning optimization, we instead add constraints enforcing each aggregate is satisfied. To our knowledge, we are the first to add aggregate constraints to BN parameter learning.

To see how the added constraints function, following Fig. 2, suppose an aggregate gives us that 0.2 percent of flights have O = KA, DE = NM, and ET = 60. Letting $\theta_{i,j,k}$ represent $\Pr(X_i = j|Pa(X_i) = k)$, the added constraint from that aggregate is

$$\sum_{dt \in dom(DT)} \sum_{f \in dom(F)} \theta_{DT,dt,\{60\}} * \theta_{O,\texttt{KA},\{dt\}} * \theta_{DE,\texttt{NM},\{\texttt{KA},dt\}} * \theta_{F,f,\{\texttt{NM}\}} * \theta_{E,60,\emptyset} = 0.2$$

Intuitively, by summing the joint probability over all possible values of the attributes that do *not* participate in the aggregate (e.g. $\sum_{dt \in dom(DT)} \sum_{f \in dom(F)}$), we are calculating the probability of the values that do participate (e.g. $\Pr(O = \texttt{KA}, DE = \texttt{NM}, ET = 60)$). We then enforce that this sum equals 0.2. Note that this formulation is non-linear.

For a mathematical formulation of our approach, see [56].

4.2.4 Query Answering. Once we have learned the probability distribution of our population, we can answer selection (point) queries probabilistically by calculating $n * \Pr(X_1 = x_1, \ldots, X_m = x_m)$. To answer GROUP BY queries, we use the BN to generate K samples of data that are representative of the population via forward/logic sampling [35, 40, 61]. Once the samples S'_k are generated, tuples are uniformly scaled up (i.e. the weight of each tuple is $|P|/|S'_k|$), and the query is answered as it is for reweighted samples. After receiving K answers, we return the groups appearing in all K answers, averaging the aggregate value. Using K samples reduces the variance and the number of incorrect "phantom groups" (groups that are returned but do not exist).

4.3 Hybrid Query Evaluator

To perform OWQP, Themis's hybrid approach integrates the previous two methods into a unified technique. For point queries, when a point query gets issued, if the tuple being queried is in the sample, we use the reweighted sample. Otherwise, we do direct BN inference. For GROUP BY queries, we return all values from our reweighted sample unioned with any groups that appear in the BN query but not the reweighted sample query.

The motivation for these techniques is due to the inherent problems with sample reweighting. If the tuple does not exist in the sample, the sample achieves poor accuracy. We are simply capturing this failure in our query evaluator by only using the BN answer to handle missing tuples or groups. The technique is critical when handling samples do not have the same support as the population, as shown in Sec. 6.

5 OPTIMIZATION

There are two main challenges to implementing our techniques efficiently: the potentially large number of aggregates[5] and the nonlinearity of our BN constraints (Sec. 4.2.3). In regards to the former, each new aggregates adds a new constraint in our BN and linear regression solver and is one more iteration of weight rescaling in IPF. With the latter problem, it is well known that nonlinear constraints add complexity and makes solving constrained optimizations computationally expensive [60]. As our constraints are sums over tuples that do not participate in an aggregate, without simplifying, solving is intractable (see Sec. 6).

To solve these problems, we present two optimization techniques: pruning the least informative aggregates and simplifying the constrained optimization. Simplifying our

[5]InFuse [2] has more that 900 aggregate queries.

constrained optimization is critical to the success of integrating S and Γ into parameter learning as it makes solving tractable and allows for optimizing each BN factor independently.

5.1 Aggregate Selection

Our goal is to reduce the number of aggregates, i.e. $|\Gamma|$, *before* using them in reweighting and Bayesian network learning. Given a budget B, the natural choice is to choose the B most informative aggregates; i.e., the B aggregates that minimize the distance between the true population distribution and some approximate distribution parametrized by the aggregates. We choose to minimize the Kullback-Leibler (KL) divergence because if we assume, like BNs, that our approximate distribution is a product distribution, we can use the fact that Chow-Liu trees [18] and their higher order counterparts minimize the KL divergence. Note that although Γ can contain aggregates with $d_i > 2$, in Sec. 6, we limit ourselves to $d_i \leq 2$; therefore, we only present our optimization in terms of Chow-Liu trees (see [56] for extension to $d_i > 2$ via k-order t-cherry junction trees [14, 62]).

Similar to the problem with black-box BN techniques, we cannot use standard Chow-Liu algorithms as they assume access to the entire population. Therefore, we modify the Chow-Liu algorithm is two ways. First, we only add edges that have support in Γ, meaning we can calculate the mutual information from Γ alone. Second, as our aggregate budget B may be larger than the number of attributes, we allow for multiple iterations of our algorithm. To avoid creating duplicate tree edges, we disallow previous edges. Once our algorithm generates B edges, we filter Γ so that each $\boldsymbol{\gamma}_i$ must be equal to the attributes associated with one of the edges. See [56] for pseudocode.

5.2 Bayesian Network Simplification

Our most critical optimization is to simplify our constrained optimization. To make solving tractable, we want each BN factor $\Pr(X_i|Pa(X_i))$ to be optimized independently with *linear* constraints. Recall that the BN structure is already known. To do this, we enforce a topological solving order (every parent node is optimized before its children nodes) and limit the aggregates added to our model.

To enforce linear constraints and independent solving, we restrict our model to only add aggregate constraints that act on single factors, i.e. aggregate constraints over a child node X_i and its parents. This means for child node X_i, the constraints will only contain the product of the child parameter $\theta_{i,j,k}$ with its ancestors because the other factors have been marginalized out. By itself, this has only reduced the number of product factors. The key is topological solving order. By insuring that the parents are solved for before the children, at the time of solving for $\theta_{i,j,k}$ for a particular X_i, the ancestor terms are already known and become a constant in the constraint, meaning the $\theta_{i,j,k}$ for X_i are the only parameters. In summary, by removing aggregate constraints that act on multiple different BN factors, we can turn our nonlinear constraints into linear ones. Further, as we only include constraints on single factors and only those factor's parameters are unknown, we can solve factors independently. See [56] for an example and mathematical formulation.

Flights	**Abrv**
fl_date	F
origin_state	O
dest_state	DE
elapsed_time	E
distance	DT

IMDB	**Abrv**
movie_year	MY
movie_country	MC
name	N
gender	G
actor_birth	B
rating	RG
top_250_rank	TR
runtime	RT

Table 2: `Flights` and `IMDB` attributes.

6 EVALUATION

In this section, we evaluate the accuracy and execution time of Themis for OWQP. We compare Themis's hybrid approach to the standard AQP solution (uniform reweighting), sample reweighting, and Bayesian network generation. We briefly discuss Themis in comparison an AQP approach [30] that can be modified to leverage aggregates, but as their technique makes different assumptions and we were unable to get code from the authors, we only evaluate one of their techniques. We then investigate the performance of the two different sample reweighting techniques. For experiments on our Bayesian network learning techniques and the benefits of using our pruning technique, see [56]. We do not show timing results for our optimization from Sec. 5.2 as experiments did not finish in under 10 hours without using the optimization, indicating the necessity of the optimizations in making learning tractable. For implementation details see the source code at [5] and [56].

6.1 Datasets

We use a flights dataset [6] (all United States flights in 2005) and an IMDB dataset [42] (actor-movie pairs released in the United States, Great Britain, and Canada). We preprocess the datasets to remove null values and bucketize the real-valued attributes into equi-width buckets. For the two real-world datasets, the attributes and their abbreviates are in Table 2.

We take three samples from `Flights`: uniform (Unif), flight month of June (June), and flights leaving from a four corner state of CA, NY, FL, WA (SCorners)[6]. Each are 10

[6]The S stands for the supported Corners sample.

B	1	2	3	4
`Flights`	E & DT	DE & DT	O & DT	F & DE
`IMDB`	MY & RT	RG & RT	MY & MC	MY & G

Table 3: The 4 2D `Flights` and 4 2D `IMDB` data aggregate attributes chosen by the pruning technique.

percent samples with a 90 percent bias, meaning 90 percent of the rows are from the selection criteria. We also take a corner states 10 percent sample with 100 percent bias (Corners).

It is important to understand the motivation for Corners compared to SCorners. Corners represents generating a sample by performing a selection on the population, a common use case. For example, US social media data is a 100 percent biased sample of the population of the US. Only users who have a social media account are in these samples. Because it is common for datasets on the web to be 100-percent biased (e.g. [7–9]) yet still serve as foundations for analysis, Themis needs to handle queries on these types of samples.

We likewise take three samples from `IMDB`: uniform (Unif), movie country of Great Britain (GB), and movies with ratings 1, 5, or 9 (SR159). We similarly give these 10 percent samples a 90 percent bias and take a 10 percent sample with 100 percent bias of the ratings sample (R159).

6.2 Experimental Setup

As real population reports typically have aggregates of one or two dimensions (e.g. Excel tables), we use $d = 1$ or 2 (see [56] for extended results on $d = 3$). Note that as the dimensionality of all aggregates is the same, we use d rather than d_i. We prune all possible aggregates to produce from $B = 1$ to 4 aggregates. Table 3 shows the aggregates chosen.

To measure accuracy, we run point queries[7] where the query selection values are selected from the population's light hitters (smallest values), heavy hitters (largest values), and random values (any existing value). We run 100 point queries for each of the three selections per attribute set. For `Flights`, we issue point queries over all possible attribute sets of size two to five (total of 26). For `IMDB`, as there are too many attributes to run all possible point queries, we randomly choose 20 three dimensional attribute sets[8]. Lastly, we use the error metric of percent difference, $2 * |true\_value - est\_value| / |true\_value + est\_value|$ rather than percent error to avoid over emphasizing errors where the true value is small and to ensure missed (not in the result but should exist) and phantom (in the result but should not exist) groups get the maximum error of 200 percent.

We also investigate how Themis performs for more advanced SQL aggregate queries. We run the six SQL queries shown in Table 5 and measure the average percent difference across the returned groups.

Finally, when measuring runtime (Sec. 6.5), as all reweighted samples are stored and accessed the same, we only look at the runtime for one reweighted sample.

[7]We define a d-dimensional point query as `SELECT COUNT(*) FROM R WHERE A1 = v1 AND ... AND Ad = vd`.

[8]We use all attributes, not just those covered by aggregates.

Hitters	Percentile	Unif	June	SCorners	Corners
Heavy	25	4.2	13.6	168.3	6.1
	50	1.8	69.7	61.9	2.7
	75	1.4	29.6	34.4	2.2
Light	25	∞	∞	∞	45
	50	1.7	1.7	1.7	1.4
	75	1.0	1.0	1.0	1.0

Table 4: Percent improvement of percentiles for hybrid compared to AQP for the queries from Fig. 3. The infinite value represents that hybrid has zero error.

6.3 Overall Accuracy

Using $B = 4$ and $d = 2$, we compare Themis's hybrid approach (pink) to the standard AQP approach (uniform reweighting), best linear reweighting technique of IPF (orange), and the best Bayesian network technique of BB (blue) (BB means it uses both Γ and S to learn the BN).

Fig. 3 and Fig. 4 show boxplots of the percent difference of 100 heavy and 100 light hitter point queries across the samples. The median value is the black line and the average is the black X. For reference, Table 4 shows the percent improvement of the 25th, 50th, and 75th percentiles of Themis's hybrid approach to uniform reweighting for `Flights`.

We see that for the samples that have the same support as the population (first three), Themis's hybrid technique achieves the lowest error for heavy and light hitters. For the `Flights` sample without support (Corners), the BN technique (BB) performs best, but hybrid performs better than IPF, indicating that hybrid mitigates the problem of mismatching support, a key requirement for Themis's applicability for real world use cases. For light hitters, BB performs better than IPF and AQP, and it is because of this that Themis's hybrid approach achieves the lowest error for light hitters. Themis uses IPF in the rare case that the tuple is in the sample, which is why hybrid achieves lower error than BB.

BB does not perform best for R159 because of queries over the very dense attribute `N` (48,000 distinct values). BB learns that `N` is uniformly distributed and underestimates queries over `N` because all values are equally likely.

To examine how the amount of sample bias impacts accuracy, we measure the average percent difference for 100 random point queries using 4 2D aggregates on the `Flights` sample of Corners as we decrease the percent bias from 100 percent (Corners sample) to 90 percent (SCorners sample), shown in Fig. 5.

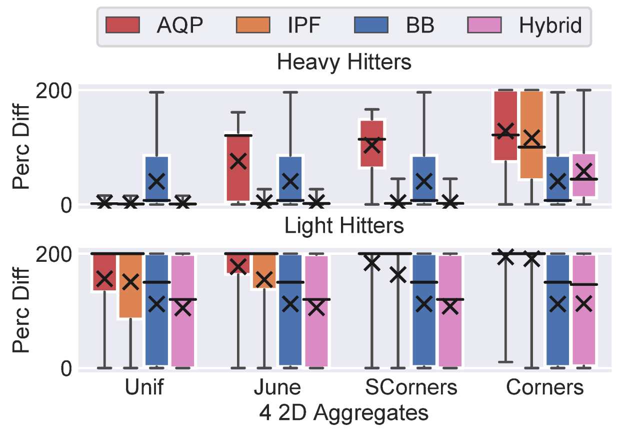

Figure 3: 100 heavy and light hitter point query percent difference for `Flights` biased samples ($B = 4$).

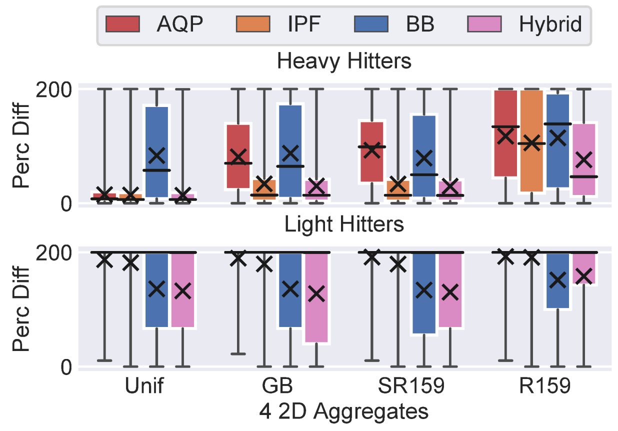

Figure 4: 100 heavy and light hitter point query percent difference for `IMDB` biased samples ($B = 4$).

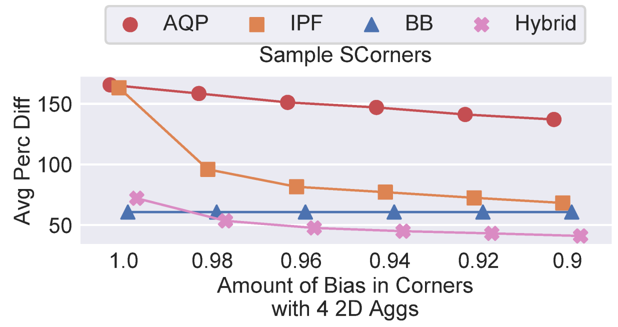

Figure 5: Average percent difference of 100 random point queries for the SCorners using 4 2D aggregates as we decrease the percent bias.

As soon as the support is the same (bias < 100), sample reweighting techniques start performing significantly better. Themis's hybrid approach is able to mitigate this difference and performs better than IPF for 100 percent bias.

Id	Query
1	`SELECT O, AVG(E) FROM F`
2	`SELECT O, AVG(E) FROM F WHERE DE = 'CA'`
3	`SELECT DE, AVG(E) FROM F WHERE O = 'CA'`
4	`SELECT O, COUNT(*) FROM F WHERE E < 120`
5	`SELECT DE, COUNT(*) FROM F WHERE E < 120`
6	`SELECT t.O, s.DE, COUNT(*) FROM F t, F s WHERE f.DE=fs.O AND (f1.DE IN ['CO', 'WY'])`

Table 5: The six SQL queries run in Fig. 6. We leave out the `GROUP BY` clause and replace `Flights` with `F` for space.

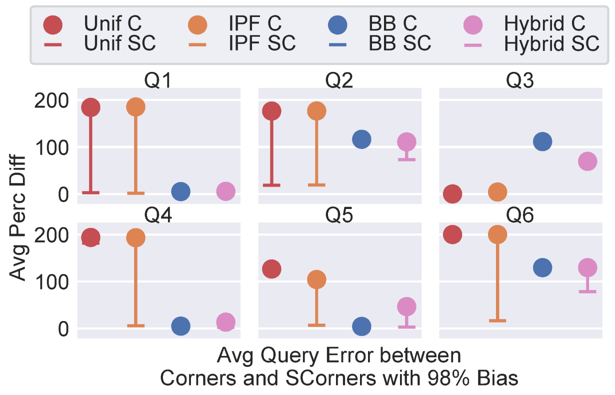

Figure 6: Average query error for six queries between Corners with 100 percent bias (C) and 98 percent bias (SC).

We further examine the impact of bias on accuracy on more complex SQL queries. Using the benchmark queries in IDEBench [25], we chose six SQL queries (Table 5) to show the strengths and weaknesses of Themis. We adapt them to be general `GROUP BY` queries which are supersets of the chained filter queries presented in [25], but we maintain the core properties of IDEBench queries that they have an aggregate, zero or more filter predicates, and zero or more joins. With the same setup as Fig. 5, we run the queries on Corners with 100 percent bias and 98 percent bias (SCorners) and measure the change in the average percent difference (Fig. 6). The queries are run on the post-bucketized data. The circle and horizontal line represent the results on Corners and SCorners, respectively. Note that the horizontal line is sometimes obfuscated by the circle, indicating little to no change in error.

First, all queries except Q3 demonstrate that hybrid and BB outperform the alternatives for 100 percent bias because they miss fewer groups. This does not hold for Q3 because the selection is the same as that for the bias; i.e., CA tuples are already present in the sample. This is also why there is no change in the error between Corners and SCorners.

Bias	100	98	96	94	92	90
O-DE	1.3	0.58	0.98	0.94	0.97	0.97
DT-DE	1.0	3.8	4.8	5.3	5.6	5.7

Table 6: Ratio of Themis relative to [30] for group by queries over O-DE and DT-DE.

Second, hybrid and BB perform suboptimal in Q2, Q3, and Q6 because they produce more phantom groups than the number of missed groups in IPF and AQP. As BB is a BN, unless BB has a factor (edge) containing all the attributes in a query, BB cannot know exactly which values exist or not in the population (a known problem with using probability distributions to answer queries [55]). Although BB mitigates this problem by using multiple generated samples to answer queries (Sec. 4.2.4), it still produces phantom groups. Q1, Q4, and Q5 are all over attribute pairs that are contained in an edge in BB, meaning BB will not generate phantom groups and have excellent performance.

Lastly, Q6 is a join query looking at flights with a layover in CO or WY. We see that due to phantom tuples, BB and hybrid are not optimal for SCorners, but IPF achieves the lowest error, far surpassing AQP. IPF more accurately rebalances the underrepresented flights leaving CO and WY in the sample.

Finally, we compare Themis to the technique presented in [30]. This, as far as we know, is the only AQP technique that does not require access to the population and can be modified to leverage the aggregates (see Sec. 7). However, note that [30] assumes a normally distributed error in query answers and access to a light hitter index, which does not hold in our setting. As we were unable to get the code, we examine their technique of reformulating the joint probability with conditional probabilities for two attributes as we increase the bias. Their motivation for this query rewrite was to reuse prior query answers, but we can modify it to use aggregate query answers.

With the same setup as for Fig. 5, we measure the ratio of hybrid's error over [30]'s error (i.e. $err_{\text{THEMIS}}/err_{[30]}$) using one 1D aggregate over O. We issue GROUP BY, COUNT(*) queries over the attribute pairs O-DE and DT-DE. We use hybrid to answer the query directly and, following the technique from [30], use the known distribution of O with the conditional probability from the sample to answer the query. The results are shown in Table 6.

For the query over O-DE, Themis achieves approximately the same error as [30]. It is, on average, 0.96x the error of [30]. The outlier is for the 98 percent bias where hybrid is 0.58x the error of [30]. This is because hybrid has more phantom groups than the sample has missing groups. For the query over DT-DE, Themis achieves the lowest error. Themis is able to debias the sample using the aggregate information, while [30] cannot use the information and is equivalent to the standard AQP approach of uniform reweighting. As the number of aggregates increase, Themis is able to learn from all of the aggregate information, while [30] must choose which information to use per query.

For experiments showing how changing the number and dimensionality of the aggregates impacts accuracy, see [56].

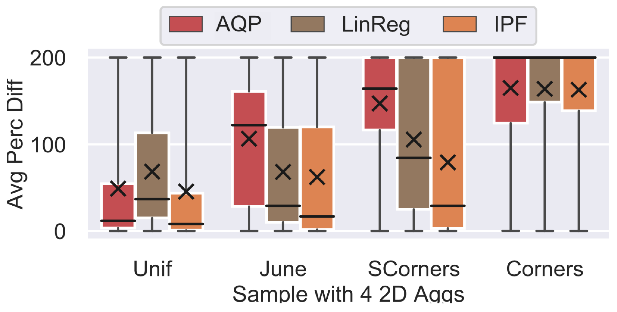

Figure 7: Percent difference of 100 random point queries over four different Flights samples with 4 2D aggregates.

6.4 Sample Reweighting Performance

We now compare the two different sample reweighting techniques of linear regression (LinReg) and IPF. We focus on comparing how they perform on the different Flights samples by measuring error on 100 random point queries using 4 2D aggregates.

Fig. 7 shows the percent difference of LinReg, IPF, and AQP. Note that AQP does not achieve near zero error on Unif because some of the random point queries are over light hitters which are not in the sample. We see clearly that IPF outperforms LinReg on all cases. While LinReg does outperform AQP in all biased samples, it does not outperform IPF due to correlations in the data. For example, to satisfy aggregates on the DT attribute, LinReg will add weight to the highly correlated attribute values of E. This will help satisfy aggregates on DT but will overall hurt performance because any other tuple with the correlated attribute values will have an incorrect weight.

From Fig. 7, it may seem that IPF is always superior to LinReg, but when the data is uncorrelated, LinReg should achieve approximately the same error as IPF. Further, if more rows are added to the sample, LinReg does not have to be retrained while IPF does.

6.5 Execution Time

Lastly, we examine the query execution time and solver time of the different approaches. We run all timing experiments on the IMDB SR159 sample as IMDB has the larger active domain. For the query execution time, we run 100 random point queries. As the query execution time did not noticeably

	1D	1	2	3	4	5	5			
	2D	0					1	2	3	4
S	BB	0.3	0.3	0.4	0.7	0.7	2.3	5.3	8.4	13.2
P	Reg	0.4	0.5	0.6	1.0	3.1	4.6	6.1	6.6	7.0
	IPF	0.2	0.6	0.8	1.7	10.8	16.1	22.5	30.0	38.5
	BB	75.1	103	103	161	295	148	68.7	63.0	58.8

Table 7: Structure (S) and parameter (P) learning execution times in seconds for LinReg (denoted Reg for space), IPF, and BB using SR159 sample as 1D and 2D aggregates are added.

change as we added more 2D aggregates, we only report results for 4 2D aggregates.

We find that running a query over a reweighted sample (i.e. AQP, LinReg, IPF, or a BN reweighted sample) takes, on average, 25.3 ms. A query issued directly on a BN takes, on average, 2.07 ms. See [56] for execution time results on other BN variations.

As the main bottle neck to using these methods is the solver time, we report the time it takes to learn the structure for BB and the time it takes to learn the parameters of LinReg, IPF, and BB in Table 7 as we increase the number of aggregates.

We see that the structure learning time is negligible compared to the solver time for BB. LinReg is the fastest, followed by IPF, and then by BB. The solver time increases with all methods as we increase the number of 1D aggregates. Surprisingly, as we add more 2D aggregates, the solver time of BB decreases. This is because as we add more 2D constraints to our model, the constraint solver has more direct equality constraints which are instantaneous to solve for.

For more execution time results, see [56].

7 RELATED WORK

Our technique is related to population synthesis where the goal is to directly generate a population dataset from a sample and either historical population data [46] or aggregate data [12, 26, 41, 45, 51, 61]. Themis, however, combines two different techniques, does not assume the sample is representative, and more accurately answers queries over tuples not in the sample.

Themis is related to bootstrapping, which is a resampling technique for understanding the uncertainty in queries during AQP [16, 44, 50, 68], but in Themis, the sample is not representative of the population and the sample probabilities are not known, a requirement for accurate bootstrap.

As discussed in Sec. 1, standard AQP techniques cannot be applied to OWQP. Most related to Themis is the work in AQP that reuses past or known results [30, 57, 58]. [58] uses pre-computed data cube aggregates to improve sample query accuracy, but it is required that the dimensions of the data cube match that of the aggregate queries, which is not true in Themis. [30] assumes sample query errors to be normally distribution and requires a light hitter index, and [57] assumes accurate knowledge of the query error. Neither of these conditions hold or are available in Themis.

From a data integration standpoint, Themis is closely related to answering queries over views (samples) [33, 43] but we model the data missing from the data sources whereas data integration deals with knowing when answers are certain or not. In regards to data cleaning [15, 38, 64], while Themis is trying to infer missing values from the sample, we are missing entire rows, not just attribute values.

Lastly, Themis is related to the machine learning fields of one class classification [32, 39], learning from aggregate labels [13, 17, 53], removing bias from machine learning algorithms [27, 67], handling selection bias for downstream models [36, 65], and calculating propensity scores [49, 59]. The fundamental difference is that Themis only has the sample that sampled with some unknown distribution and does not have any data outside of the sample. A possible solution is to generate the data outside of the sample [34]. We leave this possible technique as future work.

Using aggregates constrain query answers is the work of [63]. The work in [48] discusses using Bayesian networks to approximate a data cube. Our work is similar to both of these except our goal is not to merely answer queries but to also debias a sample.

The work of [31] uses BNs over relations to do selectivity estimation for queries. [52] performs conjunctive query selectivity estimation using both samples and synopses. Themis also uses samples and synopses (synopses are population histograms). Our overall goal, however, is to debias the data.

8 CONCLUSION

We introduce a novel query processing paradigm, OWQP. We then present Themis, the first prototype OW-DBMS that uses a biased sample and population-level aggregate information to perform OWQP. More importantly, our data debiasing is automatic. Themis's hybrid approach merges sample reweighting with population probabilistic modeling to achieve a 70 percent improvement in the median error when compared to uniform reweighting for heavy hitter queries. Further, as shown in Fig. 5, Themis is robust to differences in the support between the sample and population.

Future work is to extend Themis to support continuous data by extending our BN to allow for continuous distributions, as done in [28]. We further want to integrate multiple samples into the debiasing process and investigate alternative techniques to integrate the sample and population into our probabilistic model.

Acknowledgements. This work is supported by NSF AITF 1535565, NSF IIS 1907997, and a gift from Intel.

REFERENCES

[1] https://www.congress.gov/115/bills/s760/BILLS-115s760is.pdf.

[2] http://infusecp.mimas.ac.uk.

[3] https://pdf.ic3.gov/2017_IC3Report.pdf.

[4] https://dawn.cs.stanford.edu/2018/04/11/db-community/.

[5] https://gitlab.cs.washington.edu/uwdb/project_themis_debias.

[6] https://www.transtats.bts.gov/.

[7] https://cran.r-project.org/web/packages/nycflights13/index.html.

[8] https://data.bloomington.in.gov/dataset/traffic-data.

[9] https://data.baltimorecity.gov/Neighborhoods/Education-and-Youth-2010-2013-Shape/ifsa-6r6x/about.

[10] P. C. Austin. An introduction to propensity score methods for reducing the effects of confounding in observational studies. *Multivariate behavioral research*, 46(3):399–424, 2011.

[11] J.-F. Beaumont. A new approach to weighting and inference in sample surveys. *Biometrika*, 95(3):539–553, 2008.

[12] R. J. Beckman, K. A. Baggerly, and M. D. McKay. Creating synthetic baseline populations. *Transportation Research Part A: Policy and Practice*, 30(6):415–429, 1996.

[13] A. Bhowmik, J. Ghosh, and O. Koyejo. Generalized linear models for aggregated data. In *Artificial Intelligence and Statistics*, pages 93–101, 2015.

[14] J. Bukszár and A. Prekopa. Probability bounds with cherry trees. *Mathematics of Operations Research*, 26(1):174–192, 2001.

[15] S. Chaudhuri, A. Das Sarma, V. Ganti, and R. Kaushik. Leveraging aggregate constraints for deduplication. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 437–448. ACM, 2007.

[16] S. Chaudhuri, B. Ding, and S. Kandula. Approximate query processing: No silver bullet. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 511–519. ACM, 2017.

[17] B.-C. Chen, L. Chen, R. Ramakrishnan, and D. R. Musicant. Learning from aggregate views. In *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*, pages 3–3. IEEE, 2006.

[18] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory*, 14(3):462–467, 1968.

[19] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh. Sample selection bias correction theory. In *International Conference on Algorithmic Learning Theory*, pages 38–53. Springer, 2008.

[20] C. P. De Campos, Y. Tong, and Q. Ji. Constrained maximum likelihood learning of bayesian networks for facial action recognition. In *European Conference on Computer Vision*, pages 168–181. Springer, 2008.

[21] W. E. Deming and F. F. Stephan. On a least squares adjustment of a sampled frequency table when the expected marginal totals are known. *The Annals of Mathematical Statistics*, 11(4):427–444, 1940.

[22] A. Deshpande and S. Madden. Mauvedb: supporting model-based user views in database systems. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 73–84. ACM, 2006.

[23] B. Ding, S. Huang, S. Chaudhuri, K. Chakrabarti, and C. Wang. Sample+seek: Approximating aggregates with distribution precision guarantee. In *Proceedings of the 2016 International Conference on Management of Data*, pages 679–694. ACM, 2016.

[24] C. Dwork. Differential privacy and the us census. In *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 1–1. ACM, 2019.

[25] P. Eichmann, C. Binnig, T. Kraska, and E. Zgraggen. Idebench: A benchmark for interactive data exploration. *arXiv preprint arXiv:1804.02593*, 2018.

[26] B. Farooq, M. Bierlaire, R. Hurtubia, and G. Flötteröd. Simulation based population synthesis. *Transportation Research Part B: Methodological*, 58:243–263, 2013.

[27] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268. ACM, 2015.

[28] N. Friedman, M. Goldszmidt, and T. J. Lee. Bayesian network classification with continuous attributes: Getting the best of both discretization and parametric fitting. In *ICML*, volume 98, pages 179–187, 1998.

[29] N. Friedman, I. Nachman, and D. Peér. Learning bayesian network structure from massive datasets: the "sparse candidate" algorithm. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 206–215. Morgan Kaufmann Publishers Inc., 1999.

[30] A. Galakatos, A. Crotty, E. Zgraggen, C. Binnig, and T. Kraska. Revisiting reuse for approximate query processing. *Proceedings of the VLDB Endowment*, 10(10):1142–1153, 2017.

[31] L. Getoor, B. Taskar, and D. Koller. Selectivity estimation using probabilistic models. In *ACM SIGMOD Record*, volume 30, pages 461–472. ACM, 2001.

[32] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[33] A. Y. Halevy. Answering queries using views: A survey. *The VLDB Journal*, 10(4):270–294, 2001.

[34] H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge & Data Engineering*, (9):1263–1284, 2008.

[35] M. Henrion. Propagating uncertainty in bayesian networks by probabilistic logic sampling. In *Machine Intelligence and Pattern Recognition*, volume 5, pages 149–163. Elsevier, 1988.

[36] J. Huang, A. Gretton, K. Borgwardt, B. Schölkopf, and A. J. Smola. Correcting sample selection bias by unlabeled data. In *Advances in neural information processing systems*, pages 601–608, 2007.

[37] M. Idel. A review of matrix scaling and sinkhorn's normal form for matrices and positive maps. *arXiv preprint arXiv:1609.06349*, 2016.

[38] I. F. Ilyas, X. Chu, et al. Trends in cleaning relational data: Consistency and deduplication. *Foundations and Trends® in Databases*, 5(4):281–393, 2015.

[39] S. S. Khan and M. G. Madden. A survey of recent trends in one class classification. In *Irish conference on artificial intelligence and cognitive science*, pages 188–197. Springer, 2009.

[40] K. B. Korb and A. E. Nicholson. *Bayesian artificial intelligence*. CRC press, 2010.

[41] D.-H. Lee and Y. Fu. Cross-entropy optimization model for population synthesis in activity-based microsimulation models. *Transportation Research Record*, 2255(1):20–27, 2011.

[42] V. Leis, A. Gubichev, A. Mirchev, P. Boncz, A. Kemper, and T. Neumann. How good are query optimizers, really? *Proceedings of the VLDB Endowment*, 9(3):204–215, 2015.

[43] M. Lenzerini. Data integration: A theoretical perspective. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246. ACM, 2002.

[44] K. Li and G. Li. Approximate query processing: what is new and where to go? *Data Science and Engineering*, 3(4):379–397, 2018.

[45] R. Lovelace, M. Birkin, D. Ballas, and E. van Leeuwen. Evaluating the performance of iterative proportional fitting for spatial microsimulation: new tests for an established technique. *Journal of Artificial Societies and Social Simulation*, 18(2), 2015.

[46] M. C. Lovell. Seasonal adjustment of economic time series and multiple regression analysis. *Journal of the American Statistical Association*, 58(304):993–1010, 1963.

[47] D. Margaritis. Learning bayesian network model structure from data. Technical report, Carnegie-Mellon Univ Pittsburgh Pa School of Computer Science, 2003.

[48] D. Margaritis, C. Faloutsos, and S. Thrun. Netcube: A scalable tool for fast data mining and compression. In *VLDB*, pages 311–320, 2001.

[49] D. F. McCaffrey, B. A. Griffin, D. Almirall, M. E. Slaughter, R. Ramchand, and L. F. Burgette. A tutorial on propensity score estimation for multiple treatments using generalized boosted models. *Statistics in medicine*, 32(19):3388–3414, 2013.

[50] B. Mozafari and N. Niu. A handbook for building an approximate query engine. *IEEE Data Eng. Bull.*, 38(3):3–29, 2015.

[51] K. Müller. *A generalized approach to population synthesis*. PhD thesis, ETH Zurich, 2017.

[52] M. Müller, G. Moerkotte, and O. Kolb. Improved selectivity estimation by combining knowledge from sampling and synopses. *Proceedings of the VLDB Endowment*, 11(9):1016–1028, 2018.

[53] D. R. Musicant, J. M. Christensen, and J. F. Olson. Supervised learning by training on aggregate outputs. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 252–261. IEEE, 2007.

[54] R. S. Niculescu. Exploiting parameter domain knowledge for learning in bayesian networks. *Technical Report CMU-TR-05-147*, 2005.

[55] L. Orr, M. Balazinska, and D. Suciu. Probabilistic database summarization for interactive data exploration. *Proceedings of the VLDB Endowment*, 10(10):1154–1165, 2017.

[56] L. Orr, M. Balazinska, and D. Suciu. Sample debiasing in the themis open world database system (extended version). *arXiv preprint arXiv:2002.09799*, 2020.

[57] Y. Park, A. S. Tajik, M. Cafarella, and B. Mozafari. Database learning: Toward a database that becomes smarter every time. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 587–602. ACM, 2017.

[58] J. Peng, D. Zhang, J. Wang, and J. Pei. Aqp++: connecting approximate query processing with aggregate precomputation for interactive analytics. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1477–1492. ACM, 2018.

[59] P. R. Rosenbaum and D. B. Rubin. Reducing bias in observational studies using subclassification on the propensity score. *Journal of the American statistical Association*, 79(387):516–524, 1984.

[60] D. Solow. Linear and nonlinear programming. *Wiley Encyclopedia of Computer Science and Engineering*, 2007.

[61] L. Sun and A. Erath. A bayesian network approach for population synthesis. *Transportation Research Part C: Emerging Technologies*, 61:49–62, 2015.

[62] T. Szántai and E. Kovács. Discovering a junction tree behind a markov network by a greedy algorithm. *Optimization and Engineering*, 14(4):503–518, 2013.

[63] M. Vartak, V. Raghavan, E. A. Rundensteiner, and S. Madden. Refinement driven processing of aggregation constrained queries. In *EDBT*, pages 101–112, 2016.

[64] J. Wang, S. Krishnan, M. J. Franklin, K. Goldberg, T. Kraska, and T. Milo. A sample-and-clean framework for fast and accurate query processing on dirty data. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 469–480. ACM, 2014.

[65] B. Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning*, page 114. ACM, 2004.

[66] E. Zagheni, V. R. K. Garimella, I. Weber, et al. Inferring international and internal migration patterns from twitter data. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 439–444. ACM, 2014.

[67] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. Learning fair representations. In *International Conference on Machine Learning*, pages 325–333, 2013.

[68] K. Zeng, S. Gao, B. Mozafari, and C. Zaniolo. The analytical bootstrap: a new method for fast error estimation in approximate query processing. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 277–288. ACM, 2014.