Census TopDown: Differentially Private Data, Incremental Schemas, and Consistency with Public Knowledge

John Abowd[*]
Daniel Kifer[†]
Brett Moran
U.S. Census Bureau

Robert Ashmead
Philip Leclerc
William Sexton
U.S. Census Bureau

Simson Garfinkel
Ashwin Machanavajjhala[‡]
U.S. Census Bureau

ABSTRACT

The Census TopDown algorithm is a differentially private mechanism for creating microdata that captures person-level demographics of a population. It has two guiding principles: incremental schema extension and consistency with public knowledge.

Incremental schema extension is the process of taking privacy-preserving microdata having schema S^0 and adding additional fields to each record in order to obtain privacy-preserving microdata having schema S (with $S^0 \subset S$). This requirement can arise from data collection operations, where some fields are not yet available but differentially private data must be published anyway. The differentially private data are then updated once more fields become available. This requirement can also arise due to the need for scalability, as such an approach can break one large optimization problem (a key component of many differentially private algorithms) into a set of smaller optimization problems.

Consistency with public knowledge is a real-world requirement that has been virtually unexplored in the differential privacy literature. Some information, such as exact state population totals, may be deemed so important that they must be released without any perturbation. Other information, such as the (unperturbed) number of occupied group quarters facilities may be contained in public datasets. In such cases, the differentially private microdata must be consistent with this information either for legal/contractual reasons, to maintain the public's trust in the data, or to improve accuracy by incorporating public knowledge.

[*]The views expressed in this technical paper are those of the authors and not those of the U.S. Census Bureau. There are no sensitive data in this paper. This draft has not undergone complete internal Census Bureau peer review. Comments welcome. Please send comments to daniel.kifer@census.gov. This draft: July 2019.

[†]Penn State University and U.S. Census Bureau

[‡]Duke University

1. INTRODUCTION

Differential Privacy [9] (henceforth DP) is considered a gold standard in privacy protection—it allows organizations to collect and publish statistics about a group of people while protecting their individual responses. Initially adopted by the U.S. Census Bureau in 2008 for the OnTheMap product [27, 4], it has since seen development by Google [10, 3], Apple [34], Uber [21], Microsoft [7], and is now being adopted for the 2020 Census of Population and Housing [1].

The 2020 Census implementation will be the first large-scale deployment of differential privacy in the centralized model and, arguably, will have the highest stakes since the decennial census data are used for apportionment, redistricting, allocation of funds, public policy, and research.

TopDown is the name given to the current prototype of the collection of DP algorithms that will be used to generate confidentiality-preserving microdata with demographic information from the resident United States population. TopDown is based on two design principles: incremental schema extension and consistency with public knowledge. The combination of these principles leads to NP-complete problems even for public knowledge expressible as simple counting queries.

In this paper, we formalize these two principles, study their computational complexity and then describe the TopDown algorithm. We note that these problems and their solutions may also be of interest in other DP applications to official statistics and computational advertising.

1.1 Incremental Schema Extension

Let S^0 and S be two schemas (sets of attributes) with $S^0 \subset S$. Given a differentially private table $\widetilde{T}^0$ with schema S^0 we want to create a DP table $\widetilde{T}$ by adding fields from $S \setminus S^0$ to every record in $\widetilde{T}^0$ (i.e., by adding columns to $\widetilde{T}^0$). We say that $\widetilde{T}$ is an extension of $\widetilde{T}^0$. Note these tables are thus mutually consistent: any query performed on $\widetilde{T}^0$ that only uses attributes in S^0 will have the same answer when run on $\widetilde{T}$.

The need for extensions arises due to the complexity of data management in large surveys. For example, in the 2010 Census, the first data release was PL94-171 [36]—redistricting data that contain basic histograms on population totals in each geography broken down by race and ethnicity. The next two major waves, Summary File 1 (SF1) [37] and the Urban/Rural update [38], contained additional demographic information about people, households, and group quarters.

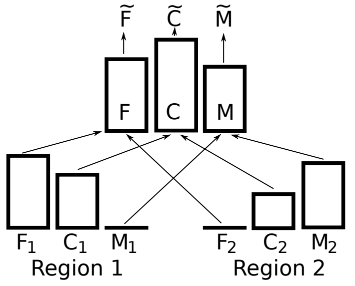


Figure 1: Publishing differentially private estimates $\widetilde{M}, \widetilde{F}, \widetilde{C}$ with implied constraints. The true counts are $M = M_1 + M_2$, $F = F_1 + F_2$ and $C = C_1 + C_2$.

Another use case for extensions is to make algorithms scale. A typical DP algorithm obtains noisy query answers from the true table T and then solves an optimization problem to obtain a privatized table $\widetilde{T}$. The number of variables in this problem is often equal to the number of possible record values (e.g., the "universe" size, or sample space in statistics) [16, 24]. For large domains, such an optimization problem cannot be performed in main memory or in reasonable time because of super-linear computational complexity. One solution is to first project T onto a table T^0, which has fewer attributes, then to create a DP version $\widetilde{T}^0$ that requires solving a much smaller optimization problem. Then $\widetilde{T}^0$ is extended to the full schema, taking advantage of parallelization. For example, to create a DP table with schema S =(Age, Sex, Race, Ethnicity, State, County), we could first create DP microdata with schema S^0 =(Age, Sex, Race, Ethnicity, State), partition the records by state and then for each state extend the records with county information (resulting in 50 smaller optimization problems being performed in parallel).

1.2 Consistency with Public Knowledge

Some information is considered so vital that a policy decision may be made to release it exactly. One such candidate is the total population in each state, which is used for apportionment and allocation of federal funds to states. In other cases, there can exist auxiliary datasets with overlapping information. For example, the LUCA dataset [5] contains information about group quarters housing facilities in each census block in the United States, which can provide lower bounds on sub-population totals. In other cases, common knowledge and data editing rules enforced by a data collector restrict the set of feasible tables. For example, "the number of spouses of householders cannot exceed the number of householders" is a data-editing rule that could be enforced during data collection or pre-tabulation editing. Thus, for legal and contractual reasons, as well as to maintain public trust, a data publisher may be required to provide DP tables $\widetilde{T}^0$ (and later $\widetilde{T}$) that are consistent with this public knowledge.

Consistency with published information raises an interesting question: under what conditions on $\widetilde{T}^0$ does an extension $\widetilde{T}$ that is consistent with public knowledge exist? This is a subtle and often computationally intractable question. To see the subtlety, consider the following example.

EXAMPLE 1. *Figure 1 represents a small college town that is divided into two regions R_1 and R_2, all living in dorms. For each student we collect the geographic region where they live and the type of dorm they live in (male-only, female-only, co-ed). T_0 contains only information about dorm type, while T also contains the region information. Information in T_0 can be summarized by 3 numbers: M, F, C representing the total number of students living in Male, Female, Co-ed dorms, respectively. T can be summarized by 6 numbers $M_1, F_1, C_1, M_2, F_2, C_2$, where, for example, F_2 is the number of students in Female dorms in Region 2.*

The public knowledge is that (1) both regions have 98 students each; (2) Region R_1 has one female dorm, one co-ed dorm, and no male dorms; (3) Region 2 has no female dorms; one co-ed dorm, and one male dorm. It can be represented by the following linear constraints on those variables (the specific values of the variables are not public knowledge).

$$F_1 \geq 0 \qquad C_1 \geq 0 \qquad M_1 = 0 \tag{1}$$

$$F_2 = 0 \qquad C_2 \geq 0 \qquad M_2 \geq 0 \tag{2}$$

$$F_1 + C_1 + M_1 = 98 \qquad F_2 + C_2 + M_2 = 98 \tag{3}$$

Suppose DP table $\widetilde{T}^0$ has already been produced (which is equivalent to differentially private counts $\widetilde{M}, \widetilde{F}, \widetilde{C}$). What restrictions (i.e., constraints involving only $\widetilde{M}, \widetilde{F}, \widetilde{C}$) are needed to ensure that we can extend $\widetilde{T}^0$ into a DP table $\widetilde{T}$ whose associated DP counts $\widetilde{M}_1, \widetilde{F}_1, \widetilde{C}_1, \widetilde{M}_2, \widetilde{F}_2, \widetilde{C}_2$ satisfy the above equations?

A seemingly "obvious" set of conditions can be derived as follows. First, we must have $\widetilde{C} \geq 0$, $\widetilde{F} \geq 0$ and $\widetilde{M} \geq 0$. The total population is 196, so $\widetilde{F} + \widetilde{M} + \widetilde{C} = 196$. These are the obvious constraints we obtain by adding up matching equalities/inequalities in regions R_1 and R_2.

Are these constraints enough? Surprisingly, it turns out they are not. Suppose a DP algorithm produced the counts $\widetilde{F} = 48, \widetilde{C} = 49$, $\widetilde{M} = 99$. These counts satisfy the constraints that we just listed but they are inconsistent with public knowledge—the entire population of male-only dorms in the town must be contained in Region R_2 but there are 98 students in R_2, so $\widetilde{M} = 99$ is not a valid value for the population in male-only dorms. It turns out that a full set of necessary and sufficient constraints on $\widetilde{T}^0$ (and hence $\widetilde{M}, \widetilde{F}, \widetilde{C}$) implied by public knowledge is:

$$\widetilde{F} \geq 0 \qquad \widetilde{C} \geq 0 \qquad \widetilde{M} \geq 0 \tag{4}$$

$$\widetilde{F} + \widetilde{C} \geq 98 \qquad \widetilde{M} + \widetilde{C} \geq 98 \qquad \widetilde{C} + \widetilde{F} + \widetilde{M} = 196 \tag{5}$$

The constraints on $\widetilde{T}^0$ can be thought of as (1) constraints $\widetilde{T}^0$ must satisfy to be consistent with public knowledge; (2) constraints $\widetilde{T}^0$ (with schema S^0) must satisfy in order to be extendable to a DP dataset $\widetilde{T}$ (with schema $S \supset S^0$).

1.3 Contributions of the Paper

The DP conceptual and implementation issues are not specific to the U.S. Census Bureau—they are relevant to official statistics (government-produced statistics) in other agencies and countries. We also note that consistency of DP data with public knowledge has other applications as well.

In transactional systems, like in online advertising and shopping, information relevant to billing must be exactly revealed. For instance, advertisers on Google must know exactly the number of individuals who clicked an ad, or sellers on Amazon must know exactly the number of individuals who bought their products, so that they can be billed correctly. Additionally, platforms like Amazon and Google might want to release more fine-grained demographic information about the people who click on the ads or buy their products for the purposes of forecasting and product/market research. There is no justification for releasing these fine-grained demographic characteristics exactly, and in fact, it makes sense to release them under strong privacy guarantees of differential privacy. Nevertheless, these differentially private counts should be consistent with the exact statistics revealed for billing purposes.

In this paper we introduce and formalize the problem of consistency with public data and study its interaction with incremental schema extension. We show that the associated decision problem is NP-complete even for public knowledge consisting of simple types of counting queries. We then present the 2020 Census TopDown algorithms that uses incremental schema extension. In the case of linearly expressible public knowledge, we explain how to obtain appropriate constraints on the differentially private table $\widetilde{T}^0$. We then apply this technique derive constraints based on the expected public knowledge for the 2020 Decennial Census. In cases where deriving constraints on $\widetilde{T}^0$ is computationally expensive, we also present several workarounds.

This paper is organized as follows. In Section 2, we introduce notation and formalize the problem statement. We review related work in Section 3. We present a complexity analysis of the problem in Section 4. We then describe the TopDown algorithm in Section 5. In Section 6, we discuss the mathematical tools that we use to work with implied constraints (specifically, Fourier-Motzkin Elimination and Network Flows). In Section 7 we use these tools to derive implied constraints for a variety of mathematical classes of constraints that can represent pieces external knowledge. In Section 8, we list the expected constraints for the 2020 Decennial Census and how they can be accommodated in our framework. For situations where deriving implied constraints becomes infeasible, we discuss a workaround called the *failsafe* in Section 9. We then discuss conclusions and future work in Section 10.

2. PROBLEM STATEMENT

2.1 Notation

Let $\lfloor x \rfloor$ be the operation that rounds a number (or vector) down to the nearest integer.

Let $S^0 = \{R_1, R_2, \dots, R_{d^0}\}$ be a set of d^0 discrete attributes having finite domains $\Omega_1, \dots, \Omega_{d^0}$ (numeric attributes can be discretized; for example age can be viewed as a discrete attribute with domain $\{0, 1, \dots, 115\}$). Let $S = \{R_1, \dots, R_{d^0}, R_{d^0+1}, \dots, R_d\}$ be a superset of S^0, containing $d > d^0$ discrete attributes. We say that S is a schema extension of S^0.

We will use the notation T^0 (resp. T) to represent a table of n records with schema S^0 (resp. S). We say T^0 and T are *consistent* if T^0 is the projection of T onto the attributes in S^0; in this case we say T is an extension of T^0. The table T^0 can be converted into a histogram H^0 with $|\Omega_1| \times \dots \times |\Omega_{d^0}|$ cells, one for each possible distinct record. The value of a cell, $H^0[i_1, \dots, i_{d^0}]$, is the number of times the corresponding record appeared in T^0. In particular, this is a nonnegative integer. Similarly, H is the corresponding histogram of T.

We let size(H^0) represent the number of cells in H^0, which is equal to the domain size of records from T^0, which is equal to $|\Omega_1| \times \dots \times |\Omega_{d^0}|$. Similarly size$(H)$ is the number of cells in H.

In the case of Example 1, the table T^0 has a record for every person and has one column which specifies the dorm type (female, male, co-ed) while T has two columns: dorm type and region. H^0 is thus a 3×1 histogram while H is a 3×2 histogram where, for example, $H[2, 0]$ is the number of people in the first region who live in co-ed dorms.

We say that a query Q^0 is linear over table T^0 if Q^0 is a linear function of H^0 (similarly for queries Q over T). Counting queries are an example of a linear queries.

Let $\mathcal{C}$ be a set of *linear* constraints on H (and hence T), where each constraint i has the form $Q_i(H) \circledast c_i$, where Q_i is a linear query, $\circledast$ is one of the following comparison operators $\leq, =, \geq$, and c_i is a scalar. We will use $\mathcal{C}^0$ to denote constraints on T^0.

Privacy-preserving versions of T and T^0 are denoted as $\widetilde{T}$ and $\widetilde{T}^0$ and their corresponding histograms are $\widetilde{H}$ and $\widetilde{H}^0$.

2.2 Privacy

We use ϵ-differential privacy as the formal privacy definition.

Definition 1 (ϵ-differential privacy [9]). *Given a privacy-loss budget $\epsilon > 0$, an algorithm M satisfies ϵ-differential privacy if for any subset V of the range of M and for any pair of tables T_1, T_2 that differ in the value of one record, $P(M(T_1) \in V) \leq e^\epsilon P(M(T_2) \in V)$*

The parameter ϵ is known as the *privacy-loss budget* and setting its value is a job for policy-makers.

Differential privacy has several important properties. The first is *transparency*—an organization can release the source code (but not random bits) of M without compromising the privacy guarantees [9]. The second property is *post-processing* [29]: if we run an algorithm $\mathcal{A}$ with no direct access to T on the output of the ϵ-differentially private algorithm $M(T)$, then this composed algorithm $\mathcal{A}(M(T))$ also satisfies ϵ-differential privacy. Finally, differential privacy has an *adaptive composition* property: if we run an ϵ_1-differentially private algorithm $M_1(T)$ to produce an output ω and an ϵ_2-differentially private algorithm $M_2(\omega, T)$, then the combined release of the outputs of both algorithms satisfies $(\epsilon_1 + \epsilon_2)$-differential privacy [29].

These properties are vital for our setting. First, we create a privatized table $\widetilde{T}^0 = M_1(T^0)$ using an algorithm M_1 that satisfies ϵ_1-differential privacy. Note that since T^0 is obtainable from T (by removing columns), then we can think of M_1 as also an ϵ_1-differentially private algorithm on T. Then for some $\epsilon_2 > \epsilon_1$, we run an $(\epsilon_2 - \epsilon_1)$-differentially private algorithm M_2 to create $\widetilde{T} = M_2(\widetilde{T}^0, T)$ for a total privacy cost of $\epsilon_1 + (\epsilon_2 - \epsilon_1) = \epsilon_2$.

In this situation we say that $\widetilde{T}^0$ was created using ϵ_1 privacy-loss budget and it was extended to $\widetilde{T}$ using an additional $\epsilon_2 - \epsilon_1$ privacy-loss budget, for a total privacy-loss budget of ϵ_2.

2.3 Problem Statement

We are given a table T^0 with schema S^0 and a table T, consistent with T^0, having schema $S \supset S^0$. We must first release a privacy-preserving version $\widetilde{T}^0$ of T^0 using ϵ_1 privacy-loss budget and then extend it into $\widetilde{T}$, a privacy-preserving version of T, using an additional $(\epsilon_2 - \epsilon_1)$ privacy-loss budget.

Recall that there are public knowledge constraints $\mathcal{C}$ on T and any differentially private version $\widetilde{T}$ that we release must satisfy them as well. Thus, when we generate $\widetilde{T}^0$ we must make sure that it is possible to extend it to a table $\widetilde{T}$ that satisfies $\mathcal{C}$. Therefore, $\widetilde{T}^0$ must satisfy some constraints $\mathcal{C}^0$ that are implied by $\mathcal{C}$. In Example 1, we saw that all the constraints on $\widetilde{T}$ referenced the location attribute,[1] which is not present in $\widetilde{T}^0$. This means that $\mathcal{C}^0$ is generally not a subset of $\mathcal{C}$ and is not always trivial to derive.

In order to formalize the requirements on $\mathcal{C}^0$, we need the following conditions.

Definition 2 (Necessary Implied Constraints). *Let S and S^0 be two schemas with $S^0 \subset S$. Let $\mathcal{C}$ (resp. $\mathcal{C}^0$) be a set of constraints over tables with schema S (resp. S^0). We say that $\mathcal{C}^0$ is a necessary set of implied constraints of $\mathcal{C}$ if, for all tables T with schema S that satisfy $\mathcal{C}$, their projections T^0 satisfy $\mathcal{C}^0$.*

Definition 3 (Sufficient Implied Constraints). *We say that $\mathcal{C}^0$ is a sufficient set of implied constraints of $\mathcal{C}$ if, for each table T^0 with schema S^0 that satisfies $\mathcal{C}^0$, there exists an extension T with schema S that satisfies $\mathcal{C}$.*

We say that $\mathcal{C}^0$ is a *complete* set of implied constraints of $\mathcal{C}$ if it is both necessary and sufficient. Intuitively, the necessary condition means that we do not add incorrect constraints and the sufficient condition means that we do not accidentally omit any constraints.

Instead of working directly with $\widetilde{T}^0$ and $\widetilde{T}$, our algorithms will be working with the corresponding histograms $\widetilde{H}^0$ and $\widetilde{H}$. The reason is that the histograms can be manipulated numerically and the constraints $\mathcal{C}$ and $\mathcal{C}^0$ most often encountered in practice are linear functions over the histograms. Histograms are only equivalent to tables of records when the histogram entries are integers and non-negative. Thus our algorithms must produce nonnegative integer histograms using differential privacy (despite its importance, such a requirement is often ignored in the literature [16, 24, 39]).

We say that $\widetilde{H}$ is an extension of $\widetilde{H}^0$ whenever $\widetilde{T}$ is an extension of $\widetilde{T}^0$. Note that $\widetilde{H}$ is an extension of $\widetilde{H}^0$ if and only if $\widetilde{H}^0$ is a marginal of $\widetilde{H}$. That is, for all $i_1, \dots, i_{d^0}$:

$$\widetilde{H}^0[i_1, \dots, i_{d^0}] = \sum_{i_{d^0+1}} \cdots \sum_{i_d} \widetilde{H}[i_1, \dots, i_{d^0}, i_{d^0+1}, \dots, i_d]$$

Now we can formally state the problem:

Problem 1 (External Consistency). *Let H^0 be a histogram on d^0 attributes and let H be a histogram on d attributes that is an extension of H^0. Given positive numbers ϵ_1, ϵ_2 with $\epsilon_1 < \epsilon_2$ and a set of linear constraints $\mathcal{C}$ on H,*

1. *Identify a complete set of implied constraints $\mathcal{C}^0$ on H^0.*
2. *Using ϵ_1 privacy-loss budget, generate $\widetilde{H}^0$, a differentially private version of H^0 that only contains nonnegative integer counts and satisfies $\mathcal{C}^0$.*
3. *Using an additional $\epsilon_2 - \epsilon_1$ of the privacy-loss budget, generate $\widetilde{H}$, a differentially private version of H that only contains nonnegative integer counts, that satisfies $\mathcal{C}$, and is an extension of $\widetilde{H}^0$.*

2.4 Applications to 2020 Census Data

The generation of differentially private microdata for the 2020 Census of Population and Housing will be performed by the Disclosure Avoidance System. It uses an algorithm called Census TopDown, which is described in this paper. The first data release will be the PL94-171 redistricting data [36], which is used for redrawing every legislative district in the country. Subsequent data releases provide more detailed tabulations of demographic characteristics for persons and households.

TopDown will be used to generate a differentially private version of the PL94-171 microdata—a table with attributes **R**ace (63 values), **E**thnicity (Hispanic or not), **VA** (whether age is 18+, or age is ≤ 17), **H**ousing **T**ype (in a household or in one of 8 types of group quarters), and location attributes **S**tate, **C**ounty, **T**ract, **B**lock **G**roup and **B**lock.[2] For brevity, we refer to these attributes as **R**, **E**, **VA**, **HT**, $\mathbf{L_S}$, $\mathbf{L_C}$, $\mathbf{L_T}$, $\mathbf{L_{BG}}$, $\mathbf{L_B}$,.

The PL94-171 dataset is too large to process in memory, thus the TopDown algorithm must repeatedly generate extensions of intermediate tables. It generates a differentially private table with the schema **R**, **E**, **VA**, **HT** (i.e., national level demographics), then extends it to the schema **R**, **VA**, **HT**, $\mathbf{L_S}$ (i.e state level demographics), and then extends four more times to add attributes for county, then tract, then block group, and then block.

Each intermediate differentially private table must ultimately be extendable to the schema for the more detailed person and households tabulations that have previously been called Summary File 1 [37] and are now called the Demographic and Housing Characteristics. This larger dataset also has a predefined constraint set.

3. RELATED WORK

The requirement that the output of a differentially private algorithm satisfy predefined constraints is known as *consistency*. It was introduced by Barak et al. [2] in the publication of overlapping differentially private histograms. For example, a differentially private histogram on Age by Race and a differentially private histogram on Age by Ethnicity can both be used to answer queries that are purely about age. Consistency means that the answers to those queries would be the same no matter which table they were computed from.

Consistency can often be obtained by adding noise to each query and setting up a least squares optimization problem that finds a single table such that queries computed over the table best match the noisy answers. Hay et al. [17]

[1]In Example 1, Table T had constraints on the population in dorms *in each region*, while the resulting constraints on T^0 were over combined populations each dorm type.

[2]Location is hierarchical: states are subdivided into counties (or county-equivalent regions), which are subdivided into tracts, block groups, and then blocks. In 2010 there were over 6.2 million inhabited blocks.

introduced a specialized algorithm for hierarchical queries, which was later extended by Cormode et al. [6] to the case where the noisy queries had different variances. Ding et al. [8] provided a different extension for lattice-based queries. While the least squares formulation is most common, other approaches are possible. Lee et al. [23] and Barak et al. [2] used an L_1 optimization problem; Lin and Kifer used Bayesian decision theory [25]; and Proserpio et al. [33] used an MCMC approach; Hardt et al. [16] used a multiplicative update rule that could be viewed as a mirror descent optimization of a least squares problem while Gaboardi et al. [12] worked directly with records rather than histograms.

In the differential privacy literature, several works considered situations where certain exact query answers over the data, like one-dimensional marginals, were publicly known [22, 18, 35]. They studied how privacy guarantees were affected [22, 18] and how they simplified differentially private algorithms [35]. However, earlier work did not encounter implied constraints, which is a novel contribution of our work. Implied constraints however do show up in other fields unrelated to privacy, like data editing [11] and the study of linear inequalities [20, 30].

4. HARDNESS RESULTS

In this section, we prove NP-completeness even for simple and naturally arising versions of this problem. The corresponding decision problem is the following.

PROBLEM 2 (IMPLIED CONSTRAINTS DECISION PROBLEM). *Given two sets S^0 and S of attributes with $S^0 \subset S$, a set $\mathcal{C}$ of linear constraints on tables with schema S and a table $\widetilde{T}^0$ with schema S^0, is it possible to extend $\widetilde{T}^0$ to a table $\widetilde{T}$ that has schema S and satisfies $\mathcal{C}$?*

Note that if implied constraints could be constructed and evaluated in polynomial time, then this decision problem would be polynomially-time solvable. So, proving that the decision problem is NP-complete means that constructing the set $\mathcal{C}^0$ of implied constraints is intractable in the general case. Our first result is even that if we start with a table on one attribute ($|S^0| = 1$) and wish to extend it to two attributes ($|S| = 2$) then the decision problem can encode any 3-SAT [13] formula and so is NP-complete.

THEOREM 1. *The implied constraints decision problem is NP-complete in the size of $\mathcal{C}$ even when $|S^0| = 1$, $|S| = 2$.*

For proof see Appendix B.

In practice, one would not expect the set $\mathcal{C}$ to be as arbitrarily complex as a 3-SAT problem. The types of problems that arise in practice have a much simpler structure. For example, suppose we have a table $\widetilde{T}^0$ of demographic characteristics for the entire population—a schema $S^0 = \{R_1, R_2\}$—and we want to extend it to a county-level demographics table $\widetilde{T}$—a schema $S = \{R_1, R_2, \text{County}\}$. Further suppose that a one-dimensional histogram on R_1 is publicly known in each county and a one-dimensional histogram on R_2 is also known at each county.[3] Let us call this problem the *location extension with 2 one-way marginal equality constraints at each locale.* It turns out that even when the smallest attribute domain (say, the domain of R_1) only has 3 possible values, the decision problem is still NP-complete in the size of the domains of all the variables.

[3]Such a situation occurred in the 2010 Census when the exact number of voting-age and non-voting age persons were known in each block (R_1 is voting age status), along with the number of householders and non-householders in each block (attribute R_2).

THEOREM 2. *The decision problem for location extension with 2 one-way marginal equality constraints at each locale is NP-complete in the number of possible locations and the domain size of the attributes when each attribute (including the location attribute) has domain size at least 3.*

For proof, see Appendix C.

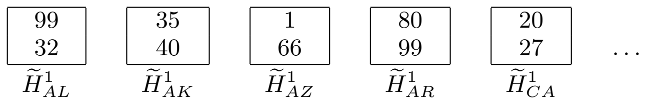
Thus, even seemingly simple variations of this problem can be intractable and a general polynomial time solution is out of the question (unless P=NP). Therefore, in the remainder of the paper, we first explain how incremental schema extension is used inside the Census TopDown algorithm and then we discuss special cases that are of interest to Census Bureau applications. We also discuss some theoretical tools that are useful in solving them, which include Fourier-Motzkin elimination [20, 30] and network flows [32].

5. THE CENSUS TopDown ALGORITHM

We now present the TopDown framework our system uses for constructing large scale differentially private histograms with external consistency. Abstracting the setup from Section 2.4, we have demographic attributes $R_1, \dots, R_{d-1}$ along with a hierarchical location attribute L (e.g., nation, state, county, tract, block group, and block). Our goal is to generate a sequence of histograms $\widetilde{H}^0, \widetilde{H}^1, \dots, \widetilde{H}^k$ where $\widetilde{H}^0$ is a histogram on $R_1, \dots, R_{d-1}$, then $\widetilde{H}^1$ extends it with State information (so it is a histogram on the attributes $R_1, \dots, R_{d-1}$ and state), $\widetilde{H}^2$ extends it with county, etc., and $\widetilde{H}^k = \widetilde{H}$ is our desired full histogram.

In order to make the algorithm run in parallel, we split the histograms along the geography dimension. For example $\widetilde{H}^1$ is a histogram on the attributes $R_1, \dots, R_{d-1}$ and state. Alternatively, we can view $\widetilde{H}^1$ as 50 histograms $\widetilde{H}^1_{AK}, \widetilde{H}^1_{AL}, \dots, \widetilde{H}^1_{WY}$ where, for example, $\widetilde{H}^1_{AK}$ is a histogram on $R_1, \dots, R_{d-1}$ in Alaska and $\widetilde{H}^1_{WY}$ is a histogram on $R_1, \dots, R_{d-1}$ in Wyoming. In general, if $\gamma_1, \dots, \gamma_\ell$ are the locations covered by $\widetilde{H}^j$, we use the notation $\widetilde{H}^j_{\gamma_i}$ to refer to the part of the histogram that deals with location γ_i. Note that each $\widetilde{H}^j_{\gamma_i}$ is a histogram on attributes $R_1, \dots, R_{d-1}$. For example, Big Horn County is a county in Wyoming (and "county" is level 2 of the location hierarchy), so $\widetilde{H}^2_{BHC}$ is a histogram on attributes $R_1, \dots, R_{d-1}$ of people in that county. A visual example of this notation is shown below.

		AL	**AK**	**AZ**	**AR**	**CA**	...
$\widetilde{H}^1$:	$\mathbf{R_1 = 0}$	99	35	1	80	20	...
	$\mathbf{R_1 = 1}$	32	40	66	99	27	...



For each geographic region γ in level i of the hierarchy, there is a workload W^i_γ of linear queries—these are the queries about the histogram at that location that end-users process. If γ is a leaf (e.g., represents a block at level k of the hierarchy), there is also a set of external linear constraints

$\mathcal{C}^k_\gamma$ that should be satisfied by $\widetilde{H}^k_\gamma$ (the part of the full table that refers to γ). Thus our set $\mathcal{C}$ of external constraints is equivalent to the set $\{\mathcal{C}^k_\gamma \mid \gamma \text{ is a leaf}\}$.

For $i = 0, 1, \ldots, k-1$, let $\mathcal{C}^i$ be the set of implied constraints on $\widetilde{H}^i$. Since $\widetilde{H}^i$ can be represented as set of histograms $\{\widetilde{H}^i_{\gamma_j} \mid \gamma_j \text{ is a node at level } i\}$, we can also write $\mathcal{C}^i_{\gamma_j}$ as the implied constraints on $\widetilde{H}^i_{\gamma_j}$. Thus, we have $\mathcal{C}^i = \{\mathcal{C}^i_{\gamma_j} \mid \gamma_j \text{ is a node at level } i\}$.

With this notation, the Census TopDown algorithm is shown in Algorithm 1. We next describe its pieces.

5.1 Initialization.

The first step (line 2) is to obtain privacy-preserving measurements of the workload queries. For each level i and each node γ at level i, we use ϵ/k of the privacy-loss budget[4] to answer the workload queries W^i_γ. These workload queries can be answered using the Laplace mechanism [9], Geometric mechanism [14], or more advanced techniques such as the high dimensional matrix mechanism [28]. As long as the mechanism provides ϵ/k-differential privacy for the workloads it is given, this entire phase satisfies ϵ-differential privacy. A noisy answer to a query is referred to as a *measurement.*

The next step (line 3) is given the external knowledge constraints $\mathcal{C}^k_\gamma$ for each leaf node γ and determines the implied constraints for the nodes at levels $i = 0, 1, \ldots, k-1$. The generation of implied constraints is discussed in the rest of the paper, starting with Section 7. The rest of the algorithm is post-processing—creating nonnegative integer histograms and incrementally extending them (i.e., adding more geographic detail).

5.2 Construction of $\widetilde{H}^0$.

The first goal of the postprocessing step is to create the initial differentially private histogram $\widetilde{H}^0$ that will later be extended. In our case, this is the histogram nation-level demographic characteristics. This histogram is constructed in two phases. First, we create a nonnegative fractional histogram H^* (Line 8) by solving a least squares optimization problem. Then, using linear programming (or integer programming), we round it to get the nonnegative integer histogram $\widetilde{H}^0$ (Line 9). These optimization problems are solved using commercial state-of-the-art optimization software like Gurobi [15] or CPlex [19].

The nonnegative fractional histogram H^* is obtained by solving the following problem (which we explain next):

$$\begin{aligned}
&\arg\min_{H^*} \sum_{Q_i \in W^0_{\gamma_0}} ||Q_i(H^*) - m_{\gamma_0,i}||^2_2 \qquad (6)\\
&\quad \text{s.t. } H^* \succeq 0 \quad \text{(nonnegativity)}\\
&\qquad \sum_x H^*[x] = \text{population total}\\
&\qquad Q'_j(H^*)\ \text{op}_j\ c_j \text{ is true for } (Q'_j, \text{op}_j, c_j) \in \mathcal{C}^0_{\gamma_0}
\end{aligned}$$

Here γ_0 represents the root node (of the geographic hierarchy) and $W^0_{\gamma_0}$ is the query workload—the set of queries over the national histogram for which we obtained noisy measurements in the initialization step. For each query $Q_i \in W^0_{\gamma_0}$,

[4]Since nodes in the same level cover disjoint regions, we exploit parallel composition within a level and sequential composition between levels.

```
1  function Driver($H, \epsilon, Workload$):
2      $A \leftarrow$ PrivacyPreservingAnswers($H, \epsilon$, Workload)
3      $B \leftarrow$ ImpliedConstraints($\{\mathcal{C}^k_\gamma\ :\ \gamma \in leaves\}$)
4      $\widetilde{H} \leftarrow$ TopDownPostprocess($A, B$, Workload)
5      return $\widetilde{H}$
6  function TopDownPostprocess($A, B$):
7      node-queue $\leftarrow \emptyset$      // List of processed nodes
       /* Generate root node histogram $\widetilde{H}^0$        */
8      $H^* \leftarrow$ solution to Equation 6
9      $\widetilde{H}^0 \leftarrow$ solution to Equation 7
       /* Recurse down the hierarchy                  */
10     node-queue.append(root node $\gamma_0$)
11     while node-queue is not empty do
12         $\gamma \leftarrow$node-queue.pop()
13         $i \leftarrow$level($\gamma$) $m \leftarrow |child(\gamma)|$
14         $\gamma_1, \ldots, \gamma_m \leftarrow$ children($\gamma$)
           /* Generate child histograms $\widetilde{H}^{i+1}_\ell$         */
15         $H^*_1, \ldots, H^*_m \leftarrow$ solution to Equation 9
16         $\widetilde{H}^{i+1}_{\gamma_1}, \ldots, \widetilde{H}^{i+1}_{\gamma_m} \leftarrow$ solution to Equation 11
17         for each $\gamma_j$ do
18             if $\gamma_j$ has children then
19                 node-queue.append($\gamma_j$)
20             end
21         end
22     end
23     Concatenate the leaf histograms ($\widetilde{H}^k_\gamma$ for $\gamma \in$ leaves)
        into the single histogram $\widetilde{H}$
24     return $\widetilde{H}$
```

Algorithm 1: Census TopDown Algorithm

the noisy answer is denoted by $m_{\gamma_0,i}$. Hence the objective of the optimization problem is to find a histogram H^* that minimizes the squared error between the values of the queries evaluated on H^* (i.e. $Q_i(H^*)$) and the corresponding noisy measurements $m_{\gamma_0,i}$. This minimization must respect several constraints: the values in the cells are nonnegative, and the sum of the cells is the total population. The last line of the optimization enforces implied constraints $\mathcal{C}^0_{\gamma_0}$ on the national histogram. Specifically, each constraint j has a linear query Q'_j, a comparison op_j that is either $\leq$, $=$, or $\geq$, and a constant c_j that $Q'_j(H^*)$ is compared to. We discuss implied constraints in detail in Section 7.

The solution to this optimization problem is a histogram H^* that satisfies the implied constraints and is nonnegative, but almost always fractional. The next step is to convert it into a nonnegative integer histogram $\widetilde{H}^0$ that satisfies the implied constraints (so that it can eventually be extended to the full histogram on all attributes including location). This can be viewed as a rounding step that is performed by solving the following minimization problem.

$$\begin{aligned}
\widetilde{H}^0 = &\arg\min_{H^\dagger} -(H^\dagger - \lfloor H^* \rfloor) \cdot (H^* - \lfloor H^* \rfloor) \qquad (7)\\
&\quad \text{s.t. } H^\dagger \succeq 0 \text{ (nonnegativity)}\\
&\qquad \sum_x H^\dagger[x] = \sum_x H^*[x] \text{ (total sum constraint)}\\
&\qquad Q'_j(H^\dagger)\ \text{op}_j\ c_j \text{ is true for } (Q'_j, \text{op}_j, c_j) \in \mathcal{C}^0_{\gamma_0}
\end{aligned}$$

$$|H^\dagger[x] - H^*[x]| \leq 1 \text{ for all cells } x$$
$$\forall x: \quad H^\dagger[x] \text{ is an integer} \tag{8}$$

The goal of this problem is to find a histogram $H^\dagger$ that is close to H^* subject to the constraints. It turns out that in the presence of these constraints, the objective function is equal to $||H^\dagger - H^*||_1$.

First, we examine the constraints. The constraints are that the solution is nonnegative, has the same total sum as H^*, and satisfies all of the implied constraints. Furthermore, we require $|H^\dagger[x] - H^*[x]| \leq 1$ so that no cell in $H^\dagger$ is very different from its value in H^* (this equation can be interpreted as saying we obtain $H^\dagger$ from H^* by rounding each cell either up or down). Finally, the last constraint (Equation 8) is that $H^\dagger$ only has integer entries.

To show that the objective function is equivalent to minimizing L_1 norm in the presence of the constraints, note that $||H^\dagger - H^*||_1$ is equal to $||(H^\dagger - \lfloor H^* \rfloor) - (H^* - \lfloor H^* \rfloor)||_1$. Noting that H^* is a fixed constant in this problem, the constraints force $(H^\dagger - \lfloor H^* \rfloor)$ to be a vector of zeroes and ones. Therefore, the L_1 norm is then equal to

$$(H^\dagger - \lfloor H^* \rfloor)\cdot(\vec{1} - (H^* - \lfloor H^* \rfloor)) + (\vec{1} - (H^\dagger - \lfloor H^* \rfloor))\cdot(H^* - \lfloor H^* \rfloor)$$

where the left term accounts for the entries that are rounded up to the nearest integer (e.g., $H^\dagger[i] - \lfloor H^*[i] \rfloor = 1$) and the right term accounts for the entries that are rounded down (e.g., $H^\dagger[i] - \lfloor H^*[i] \rfloor = 0$). Since the constraints on total population force $(H^\dagger - \lfloor H^* \rfloor)\cdot \vec{1}$ to be a constant—that is, equal to $(H^* - \lfloor H^* \rfloor)\cdot\vec{1}$—minimizing the above equation is equivalent to minimizing $-(H^\dagger - \lfloor H^* \rfloor)\cdot(H^* - \lfloor H^* \rfloor)$.

The integrality condition can sometimes be dropped. If we drop it, the result is a linear program. If the constraints in this linear program have a special property called *total unimodularity* [31] (discussed in Section 6.1), then the solution to this program automatically returns integers as long as we use the simplex algorithm or barrier+crossover algorithm to solve the linear program [31].

If the constraints are not totally unimodular, then the integer constraints in Equation 8 are necessary and require an optimizer to solve an integer program, which can be very slow. Thus, the efficiency of this phase of the algorithm depends on two factors:

- How quickly we can compute the implied constraints.
- Whether the optimization problem in Equation 7 (which depends on the implied constraints) is totally unimodular.

These are the main questions we consider when presenting the derivation of implied constraints.

5.3 Recursive Schema Extension.

Once the national histogram $\widetilde{H}^0$ has been created, the next step is to extend it to $\widetilde{H}^1$ (adding state-level information), $\widetilde{H}^2$ (adding county information), ..., $\widetilde{H}^k$ (adding block information). This process happens recursively—first, we fix (i.e., hold constant) the root node and generate its children (e.g., histograms for each state) with the constraint that the child histograms add up to the parent histogram while satisfying their own implied constraints. Then, for each state histogram, we fix the histogram and generate its county-level children such that they add up to the state, and so forth down to the block.

Generally, after a histogram $\widetilde{H}^i_\gamma$ has been generated for a node γ in level i of the hierarchy (initially γ is the root node), then we generate its children by solving a least squares optimization problem followed by a linear (or integer) program. The least squares optimization problem generates nonnegative fractional histograms and the subsequent optimization problem rounds them to nonnegative integer histograms.

The least squares optimization problem is the following:

$$\arg\min_{H_1^*,\dots,H_m^*} \sum_{j=1}^{m} \sum_{Q_\ell \in W^{i+1}_{\gamma_j}} ||Q_\ell(H_j^*) - m_{\gamma_j,\ell}||_2^2 \tag{9}$$
$$\text{s.t. } H_j^* \succeq 0 \quad \text{for all } j$$
$$\sum_{j=1}^{m} H_j^*[x] = \widetilde{H}^i_\gamma[x] \text{ for all cells } x$$
$$Q'_\ell(H_j^*) \text{ op}_\ell\ c_\ell \text{ is true for all } j$$
$$\text{and for all}(Q'_\ell, \text{op}_\ell, c_\ell) \in \mathcal{C}^{i+1}_{\gamma_j}$$

Here we have query workloads $W^{i+1}_{\gamma_j}$ for each child γ_j. For each query Q_ℓ in a workload $W^{i+1}_{\gamma_j}$ we have its noisy answer $m_{\gamma_j,\ell}$. The objective is to find histograms $H_1^*, H_2^*, \dots, H_m^*$ such that the answers to queries evaluated over those histograms are as close as possible to the noisy answers. These histograms must satisfy several constraints. First, they must be nonnegative. Second, the sum of the child histograms must add up to the parent histogram. This is the standard parent-child constraint – e.g., the number of female, hispanic, 33 year-olds at the national level $\widetilde{H}^0$ should equal the total number of female, hispanic, 33-year-olds in the state level histograms $\widetilde{H}^1_{AL}, \widetilde{H}^1_{AK}, \dots, \widetilde{H}^1_{WY}$. The last set of constraints is that each child histogram must satisfy its implied constraints.

Then we round these fractional histograms $H_1^*, \dots, H_m^*$ to obtain the nonnegative integer child histograms $\widetilde{H}^{i+1}_{\gamma_1}, \dots, \widetilde{H}^{i+1}_{\gamma_m}$ using the following optimization problem.

$$\widetilde{H}^{i+1}_{\gamma_1}, \dots, \widetilde{H}^{i+1}_{\gamma_m} \tag{10}$$
$$= \arg\min_{H_1^\dagger,\dots,H_m^\dagger} \sum_{j=1}^{m} -(H_j^\dagger - \lfloor H_j^* \rfloor)\cdot(H_j^* - \lfloor H_j^* \rfloor) \tag{11}$$
$$\text{s.t. } H_j^\dagger \succeq 0 \text{ for all } j$$
$$Q'_\ell(H_j^\dagger) \text{ op}_\ell\ c_\ell \text{ is true for all } j$$
$$\text{and for all}(Q_\ell, \text{op}_\ell, c_\ell) \in \mathcal{C}^{i+1}_{\gamma_j}$$
$$|H_j^\dagger[x] - H_j^*[x]| \leq 1 \text{ for all } j \text{ and cells } x$$
$$\sum_j H_j^\dagger[x] = \widetilde{H}^i_\gamma[x] \text{ for all cells } x$$
$$\forall x, j: \ H_j^\dagger[x] \text{ is an integer} \tag{12}$$

This linear/integer program is similar to Equation 7. We want to find histograms $H_1^\dagger, \dots, H_m^\dagger$ that are as close as possible to the fractional histograms $H_1^*, \dots, H_m^*$ (a similar argument to that of Equation 7, but applied to all terms of this objective function simultaneously shows that this is equivalent to minimizing the sum of L_1 norms). This histograms must be nonnegative and satisfy the implied constraints for $\widetilde{H}^{i+1}_{\gamma_1}, \dots, \widetilde{H}^{i+1}_{\gamma_m}$. For each j we also have the constraint $|H_j^\dagger[x] - H_j^*[x]|$, which can be interpreted as saying that $H_j^\dagger$ is obtained from H_j^* by rounding each cell either up or down. We must also include the parent-child summation constraint, which says that $H_1^\dagger, \dots, H_m^\dagger$ (which will become the child histograms) must add up to the parent. Finally, Equation 12 is used to ensure the histograms are integers. Again, if the problem is totally unimodular, the last constraint is not needed because a linear program solver (e.g., the simplex or barrier + crossover algorithms) will automatically return integer solutions.

Once the leaf histograms $\widetilde{H}_\gamma^k$ have been generated, the algorithm concatenates them to form the final table, a demographic characteristics histogram $\widetilde{H}$ that includes all levels of geography.

In the subsequent sections, we consider situations in which implied constraints can be computed efficiently and the rounding optimization problems have totally unimodular constraints, as these conditions ensure that the TopDown algorithm runs in polynomial time.

6. MATHEMATICAL TOOLS

In this section we discuss the mathematical tools used to derive implied constraints.

6.1 Total Unimodularity

In feasible linear programs of the form:

$$\arg\min_{\vec{x}} \vec{c}^{\,T}\vec{x}$$
$$\text{s.t. } A\vec{x} \preceq \vec{b}$$

The set of optimal solutions forms a polytope. If the vector $\vec{b}$ only contains integers, if the matrix A has the total unimodularity property, and if the polytope of optimal solutions is bounded, then and all vertices of this polytope are integers. This means that some of the optimal solutions for $\vec{x}$ are vectors of integers. Furthermore, algorithms like Simplex will return one of these integer vectors [31].

A matrix is *totally unimodular* (TUM) if the determinant of every square sub-matrix is either $-1, 0$, or 1 [31]. Thus, the efficiency of the L_1 solves in the TopDown algorithm depends on the L_1 solve being equivalent to a linear program with a TUM constraint matrix.

6.2 Fourier-Motzkin Elimination (FME)

Fourier-Motzkin elimination (FME) [20, 30] is a technique for eliminating variables in a system of linear inequality constraints. Starting with a system of constraints A on variables $x_1, x_2, \dots, x_d'$, one can obtain a system of constraints B on variables $x_1, x_2, \dots, x_k'$ (with $k' < d'$) such that A has a feasible real-valued solution if and only if B has a feasible (real-valued) solution.

The process is fairly simple. To eliminate a variable x_d', we find all inequalities involving it (equalities of the form $\sum_i a_i x_i = c_i$ can be treated as a pair of inequalities $\sum_i a_i x_i \geq c_i$, $\sum_i a_i x_i \leq c_i$) and isolate x_j on one side to get:

$$\begin{aligned}
x_{d'} &\leq a_{1,1}x_1 + a_{1,2}x_2 + \dots + a_{1,d'-1}x_{d'-1}\\
x_{d'} &\leq a_{2,1}x_1 + b_{2,2}x_2 + \dots + b_{2,d'-1}x_{d'-1}\\
&\vdots \ \vdots\\
x_{d'} &\geq b_{1,1}x_1 + b_{1,2}x_2 + \dots + b_{1,d'-1}x_{d'-1}\\
x_{d'} &\geq b_{2,1}x_1 + b_{2,2}x_2 + \dots + b_{2,d'-1}x_{d'-1}\\
&\vdots \ \vdots
\end{aligned}$$

Then, for each pair of $\geq$ and $\leq$ constraints $x_{d'} \leq a_{i,1}x_1 + a_{i,2}x_2 + \dots + a_{i,d'-1}x_{d'-1}$ and $x_{d'} \geq b_{j,1}x_1 + b_{j,2}x_2 + \dots + b_{j,d'-1}x_{d'-1}$ one introduces a new constraint $a_{i,1}x_1 + a_{i,2}x_2 + \dots + a_{i,d'-1}x_{d'-1} \geq b_{j,1}x_1 + b_{j,2}x_2 + \dots + b_{j,d'-1}x_{d'-1}$. After all new constraints are generated, the old ones are removed and hence $x_{d'}$ is eliminated since it does not appear in the new system of inequalities.

Given a solution to the new system of inequalities, it is easy to see that it can be extended to the original system of inequalities by choosing any value for $x_{d'}$ that satisfies:

$$\max_j b_{j,1}x_1 + b_{j,2}x_2 + \dots + b_{j,d'-1}x_{d'-1} \leq x_{d'}$$
$$\min_i a_{i,1}x_1 + a_{i,2}x_2 + \dots + a_{i,d'-1}x_{d'-1} \geq x_{d'}$$

since a solution to the new system of inequalities implies that for all i, j, $a_{i,1}x_1 + a_{i,2}x_2 + \dots + a_{i,d'-1}x_{d'-1} \geq b_{j,1}x_1 + b_{j,2}x_2 + \dots + b_{j,d'-1}x_{d'-1}$.

Note that this extension is for real-valued variables. After constraints are derived, we typically often need to prove that integer-valued solutions to the final system of inequalities can be extended to integer-valued solutions of the original problem.

One can repeatedly use FME to sequentially eliminate $x_{d'}, x_{d'-1}, x_{d'-2}$, etc. The Fourier-Motzkin algorithm runs in double exponential time in the number of eliminated variables (in the worst case) but acceleration techniques [20, 30] can sometimes make it practical.

6.3 Network Flows

Consider a directed acyclic graph with exactly one source s (a node with no incoming edges), one sink t (a node with no outgoing edges), and nonnegative weights on each edge (called edge-capacities). A flow is an assignment of nonnegative numbers to each edge such that at each node (except for the source and sink) the sum on the incoming edges equals the sum on the outgoing edges and can be thought of as the amount of liquid flowing on each edge as it goes from the source to the sink. A flow is feasible if the flow on each edge is at most the weight on the edge. The amount of flow is the sum of the flow on each edge coming out of the source (or, equivalently, going into the sink). A cut is a partition of the vertices of the graph into two sets S and T, where $s \in S$ and $t \in T$. The value of a cut is the sum of the capacities on all edges of the form (a, b) where $a \in S$ and $b \in T$.

The Max-Flow/Min-Cut theorem [32] states that the maximum possible amount of flow is equal to the minimum value of any cut. Furthermore, if all edges have integer or infinite edge capacities (except for edges incident to the source and sink), then maximum flow amount is achieved by an integral flow (the flow on each edge is an integer).

7. IMPLIED CONSTRAINTS

In this section we provide examples of complete implied constraints for several situations of interest to the Census TopDown algorithm.

7.1 Generating Implied Constraints via FME

In this section we explain how to use FME to generate implied constraints. To minimize the amount of notation required, we explain it with a simple example related to one set of external constraints that was under consideration.

Example 2. *The hierarchy γ on location L contains the root and two children A and B. The full data schema contains the attributes R_H (whether someone is a householder or not), R_V (whether someone is voting age or not) and L. We want to release a differentially private histogram $\widetilde{H}^0$ over attributes R_H, R_V and then extend it with location. Suppose the public knowledge is the total population in regions A and B along with the number of voting age persons in each region and the total number of householders in each region. This is one scenario under considerations at the Census Bureau. This public knowledge is summarized as follows:*[5]

	Region A			***Region B***		
	$\mathbf{R_V = 0}$	$\mathbf{R_V = 1}$		$\mathbf{R_V = 0}$	$\mathbf{R_V = 1}$	
$\mathbf{R_H = 0}$	*?*	*?*	*6*	*?*	*?*	*15*
$\mathbf{R_H = 1}$	*?*	*?*	*16*	*?*	*?*	*5*
	17	*5*		*5*	*15*	

What should be the constraints on the 2×2 histogram at the root level?

Let $h_A[i,j]$ (resp, $h_B[i,j]$) denote the unknown cell counts in the histogram for Region A (resp, B). The public knowledge can then be expressed as

$$h_A[0,0] + h_A[0,1] = 6 \qquad h_B[0,0] + h_B[0,1] = 15 \qquad (13)$$
$$h_A[1,0] + h_A[1,1] = 16 \qquad h_B[1,0] + h_B[1,1] = 5 \qquad (14)$$
$$h_A[0,0] + h_A[1,0] = 17 \qquad h_B[0,0] + h_B[1,0] = 5 \qquad (15)$$
$$h_A[0,1] + h_A[1,1] = 5 \qquad h_B[0,1] + h_B[1,1] = 15 \qquad (16)$$

$$h_A[i,j] \geq 0 \quad h_B[i,j] \geq 0 \quad \text{for } i = 0,1 \ \ j = 0,1 \qquad (17)$$

Now, $\widetilde{H}^0$ is intended to be a differentially private histogram on R_H, R_V at the root (national) level. We let $h[i,j]$ refer to its cell counts. The goal is to determine the allowable values of the $h[i,j]$ so that it is consistent with the public knowledge formalized in Equations 13–17. The relationship between $h[i,j]$ to $h_A[i,j]$ and $h_B[i,j]$ is:

$$h[i,j] = h_A[i,j] + h_B[i,j] \qquad \text{for } i = 0,1; \quad j = 0,1 \qquad (18)$$

Equations 13–18 are therefore the initial system of (in)equalities. Our goal is to apply FME to eliminate the variables $h_A[i,j]$ and $h_B[i,j]$ (for $i = 0,1$ and $j = 0,1$). This would leave only the desired constraints on $h[i,j]$.

The resulting constraints (full derivation appears in Appendix D) are:

$$\begin{aligned} h[0,0] &= 1 + h[1,1] \\ h[0,1] &= 20 - h[1,1] \\ h[1,0] &= 21 - h[1,1] \\ 10 \geq h[1,1] &\geq 0 \end{aligned}$$

So any differentially private histogram at the root level must satisfy these constraints.

[5]Note that knowing the number of voting age people and total population in a region means we also know the number of people who are not voting age.

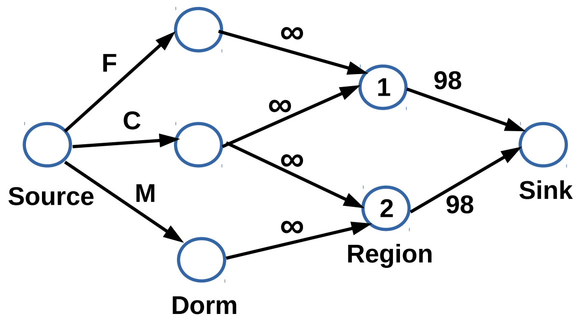


Figure 2: Encoding Example 1 as a Network Flow

Integer feasibility. The types of implied constraints considered in Example 2 are formalized in the lemma below. It turns out that this is an easy case: two one-way marginal constraints are imposed on every leaf and each marginal only has two values. As shown previously in Theorem 2, the complexity increases dramatically when the size of the marginals increases.

Lemma 1. *Let h be a 2×2 matrix of nonnegative integers and let $h_1, \dots, h_m$ be nonnegative matrices such that $\sum_i h_i = h$. Suppose that the following constraints are imposed.*

- *For each i, j and ℓ, we have $h_\ell[i,j] \geq 0$*
- *For each i and ℓ, the row sums are fixed: $h_\ell[i,0] + h_\ell[i,1] = r_\ell[i]$ for a pre-specified vector r_ℓ*
- *For each j and ℓ, the columns sums are fixed: $h_\ell[0,j] + h_\ell[i,j] = c_\ell[j]$ for a pre-specified vector c_ℓ*

then a complete set of implied constraints on h are:

$$\begin{aligned} \forall i, \ \ h[i,0] + h[i,1] &= \sum_\ell r_\ell[i] \\ \forall j, \ \ h[0,j] + h[1,j] &= \sum_\ell c_\ell[j] \\ h[1,1] &\leq \sum_\ell \min(r_\ell[1], c_\ell[1]) \\ h[1,1] &\geq \sum_\ell c_\ell[1] - \min(c_\ell[1], r_\ell[0]) \end{aligned}$$

Furthermore, if h is an integer histogram that satisfies those constraints, then there exist integer histograms $h_1, \dots, h_\ell$ that add up to h and satisfy the external knowledge row and column sum constraints for each ℓ

See Appendix E for the proof.

7.2 Generating Complete Implied Constraints via Network Flows

Network flow is a more specialized tool than FME – it is not as automatic as FME and applies in fewer cases, but it is simpler to use as a proof technique. In this section we show how the complete implied constraints of Example 1 can be derived with network flows.

Example 1 can be represented as a network flow as shown in Figure 7.2. There are 3 nodes corresponding to dormitories - a Female dorm node with an incoming edge labeled with edge capcity F, a Co-ed dorm node with an incoming edge labeled with edge capacity C, and a Male dorm node with an incoming edge labeled with edge capacity M. There are also two region nodes. In this graph, people flow from the source to the different dorm types. The capacities on these edges (F, C, M) will be determined later. People in female dorms flow into Region 1 (because only that region has female dorms). Similarly, people in male dorms flow into Region 2, while people in co-ed dorms can flow into both regions. People in both regions then flow into the sink. Since there is a capacity of 98 on the edge from Region 1 to the sink, any feasible flow must have at most 98 people flowing into Region 1 and similarly for Region 2.

Now, if we have the base histogram $\widetilde{H}^0$ which, in this case, consists of the differentially private estimates $\widetilde{F}$, $\widetilde{C}$, $\widetilde{M}$ then this partially specifies a flow (outgoing flow from the source, with flow $\widetilde{F}$ on the edge with capacity F, etc.). The question is whether we can complete the flow (i.e., assign flows on the rest of the edges) that satisfy the capacity constraints. If we can, then we can try to construct the extended table $\widetilde{H}$ from the flow as follows: the number of people flowing from co-ed dorms to Region 1 would be the variable $\widetilde{C}_1$ in Example 1, the number of people flowing from co-ed dorms to Region 2 would be the variable $\widetilde{C}_2$, etc. This table satisfies our desired constraints if and only if the total flow is equal to $98 + 98$ (which means a total of 98 people flow into Region 1, some from female and co-ed dorms but none from male dorms; and a total of 98 people flow into Region 2, some from male and co-ed dorms, but none from female dorms). Notice that $98 + 98$ must be an upper bound on the maximum flow in the network (because that is the most that is allowed to flow into the sink) and so $\widetilde{H}^0$ is extendable if and only if the value of the max flow is equal to 98+98. If there is such a max flow, then it will also exist if the edge capacities F, C, M are set to the actual amount of flow we put on these edges: $\widetilde{F}$, $\widetilde{C}, \widetilde{M}$, respectively. Thus implied constraints on $\widetilde{F}$, $\widetilde{C}, \widetilde{M}$ are equivalent to constraints on the capacities F, C, M if we add the condition that $F + C + M = 98 + 98$.

Thus, the question is under what conditions on F, C, M does the maximum flow in the network have value 196. To answer this question, we must examine all of the cuts and ensure that every cut has value at least 98+98 (note that the cut in which the sink is by itself has value equal to $98 + 98$). The cuts that have finite value are easy to determine because we must avoid edges with infinite capacity going from the sets S to T of the cut. Thus the cuts with finite value are:

- $S = \{\text{Source}, N_f, N_m, N_c, N_1, N_2\}$ and $T = \{\text{Sink}\}$, where N_f, N_m, N_c are the nodes for female, male, and co-ed dorms, and N_1, N_2 are the nodes for Regions 1 and 2. This cut has a value of 196.
- $S = \{\text{Source}, N_m, N_2\}$ and $T = \{\text{Sink}, N_1, N_f, N_c\}$. This cut has value $F + C + 98$.
- $S = \{\text{Source}, N_2\}$ and $T = \{\text{Sink}, N_1, N_f, N_c, N_m\}$. This cut has value $F + C + M + 98$.
- $S = \{\text{Source}, N_f, N_1\}$ and $T = \{\text{Sink}, N_2, N_m, N_c\}$. This cut has value $M + C + 98$.
- $S = \{\text{Source}, N_1\}$ and $T = \{\text{Sink}, N_2, N_f, N_c, N_m\}$. This cut has value $F + M + C + 98$.
- $S = \{\text{Source}\}$ and $T = \{\text{Sink}, N_1, N_2, N_f, N_c, N_m\}$. This cut has value $F + C + M$.

Thus we want $F + C + M = 98 + 98$ and each cut to have a value at least 196. This results in the following inequalities:

$$\begin{aligned} F + C + M &= 196 & F &\geq 0\\ F + C &\geq 98 & M &\geq 0\\ M + C &\geq 98 & C &\geq 0 \end{aligned}$$

By the Max-Flow/Min-Cut theorem, there is a maximum flow (with integer coefficients) equal to 196 if and only if these constraints are satisfied, hence these are the complete implied constraints on $\widetilde{H}^0$.

Generalization. We generalize this discussion as follows. Suppose the cells of a histogram (e.g., the national histogram) are grouped into k buckets $x_1, \ldots, x_k$ (e.g., this a partition of the space of demographic attributes). When we add a geographic attribute, then for each bucket (i.e., combination of demographic attributes) x_j and for each leaf γ_i, let $x_{j,i}$ be the number of people from region γ_i that would fall in bucket x_j. Thus, at each leaf γ_i we have a sum constraint: $\sum_j x_{j,i} = n_i$ (where n_i is a constant). For each leaf γ_i and bucket x_j we also have either the constraint $x_{j,i} = c_{j,i}$ or the constraint $x_{j,i} \geq c_{j,i}$ (where $c_{j,i}$ is a constant that is at least 0).

THEOREM 3. *Consider an attribute R_1 that can take k values and R_2 that can take m values. Let the base schema $\mathcal{S}^0 = \{R_1\}$ and the extended schema be $\mathcal{S} = \{R_1, R_2\}$. Let $\widetilde{H}^0 = (x_1, \ldots, x_k)$ be a histogram constructed from a table with schema $\mathcal{S}^0$. For histograms built from tables over schema $\mathcal{S}$, we will use the notation $x_{j,i}$ to refer to the count of the number of people with $R_1 = j$ and $R_2 = i$. Suppose the external constraints have the following form:*

$$\sum_j x_{j,i} = n_i \quad \text{for } i = 1, \ldots, m$$

$$x_{j,i}\ op_{j,i}\ c_{j,i} \quad \text{where } op_{j,i} \text{ is either} \geq \text{or} =, \text{ for } {}^{i=1,\ldots,m}_{j=1,\ldots,k}$$

where the $c_{j,i}$ are all nonnegative and $n_i \geq \sum_j c_{j,i}$ for all i. For any set $A \subseteq \{1, \ldots, k\}$ let neigh(A) denote the set $\{i \mid x_{j,i} \geq c_{j,i}$ is a constraint for some $j \in A\}$. Then a complete set of implied constraints is:

$$\sum_j x_j = \sum_i n_i$$

$$\forall j\ x_j \geq \sum_i c_{j,i}$$

$$\forall A \subset \{1, \ldots, k\} : \sum_{j \in A} (x_j - \sum_i c_{j,i}) \leq \sum_{i \in neigh(A)} (n_i - \sum_j c_{j,i})$$

For proof see Appendix F. The way to interpret the last set of constraints (in terms of the group quarters setup of Example 1) is the following: for any combination of group quarters types, the total number of people living in them must be at most the total population in all regions that have group quarters of those types. Note that the constraints given by this theorem are not the same as the constraints derived at the beginning of this section, but they are equivalent.

In the next section we show that we can replace these constraints with a more compact representation that, when added to the optimization problems used in the TopDown algorithm, results in that algorithm always producing histograms that can be extended to the leaves of the geographic hierarchy.

7.3 Network flows during postprocessing.

The implied linear constraints on $\widetilde{H}^0$ that were generated by Theorem 3 have a drawback—there are exponentially many in k (e.g., exponentially many in the number of group quarters types). Since these constraints are implicitly encoded in a much smaller network flow problem, it turns out that we can make the constrained optimization problem smaller by encoding network flows directly into the optimization problems in Equations 6, 7, 9, and 11.

7.3.1 An Example

We will illustrate this approach first with a concrete example (to avoid getting bogged down in notation) and then present the general case. Consider reduced 3-level location hierarchy:

- Leaves: Suffolk, Nassau, Queens, Washington, Kent, Providence, Newport.
- States: NY (consisting of Suffolk, Nassau, Queens counties) and RI (consisting of Washington, Kent, Providence, Newport).
- National: here it consists of two states: NY and RI (or more precisely, the parts of those states covered by the chosen counties).

Suppose there are 3 GQ types: A (dorm), B (nursing home), C (households) and the following public knowledge:

- Suffolk – Population: N_0, Possible GQ: $\{A, C\}$ with min populations a_0, c_0, respectively
- Nassau – Population: N_1, Possible GQ: $\{A, C\}$ with min populations a_1, c_1, respectively
- Queens – Population: N_2, Possible GQ: $\{C\}$ with min population c_2
- Washington – Population N_3, Possible GQ: $\{A, B\}$ with min populations a_3, b_3, respectively
- Kent – Population N_4, Possible GQ: $\{A, B\}$ with min populations a_4, b_4, respectively
- Providence – Population N_5, Possible GQ: $\{A, C\}$ with min populations a_5, c_5, respectively
- Newport – Population N_6, Possible GQ: $\{A, B\}$ with min populations a_6, b_6, respectively

Note that there are only 3 distinct GQ combinations: $\{A, C\}$, $\{A, B\}$, $\{C\}$. Suppose at each level of geography γ, the corresponding histogram H_γ is a 2×3 histogram on ethnicity $\times$ GQ. So, for example, $H_\gamma[0, 2]$ refers to the number of non-Hispanic people in GQ type C.

Nation-level Solve. At the national level, we introduce a variable g_i for each group quarters type (g_0 will count how many people are in GQ of type A, etc) and a variable r_j for each GQ combination that appears in the leaves: r_0 for $\{A, C\}$, r_1 for $\{A, B\}$, r_2 for $\{C\}$. We also define flow variables $f_{i,j}$ for how many people from g_i flow into a region with gq combination r_j. Eliminating impossible flows, we get the variables $f_{0,0}$ (people flowing from GQ A to region with combination $\{A, C\}$), $f_{0,1}$, $f_{1,1}$, $f_{2,0}$, $f_{2,2}$. The point of these flow variables is that they will show that it is possible to extend our national level table in a way that satisfies the external knowledge. That is the only role of the flow variables. Once we get a solution, we discard them.

Given noisy measurements to the workload queries at the national level (using the same notation as in Section 5.2), we first solve the following least squares optimization:

$$\arg\min_{H^*, g_i, r_j} \sum_{Q_i \in W^0_{\gamma_0}} ||Q_i(H^*) - m_{\gamma_0,i}||_2^2 \tag{19}$$

$$\begin{aligned}
&\text{s.t. } H^*[i,j] \geq 0 \quad (\text{for all } i, j)\\
&\sum_{i=0}^{1}\sum_{j=0}^{2} H^*[i,j] = \sum_{\ell=0}^{6} N_\ell\\
&(\text{define the } g_i)\\
&g_0 = H^*[0,0] + H^*[1,0]\\
&g_1 = H^*[0,1] + H^*[1,1]\\
&g_2 = H^*[0,2] + H^*[1,2]\\
&g_0 \geq a_0 + a_1 + a_3 + a_4 + a_5 + a_6\\
&g_1 \geq b_3 + b_4 + b_6\\
&g_2 \geq c_0 + c_1 + c_2 + c_5\\
&(\text{Lower bounds on flow})\\
&f_{0,0} \geq a_0 + a_1 + a_5\\
&f_{0,1} \geq a_3 + a_4 + a_6\\
&f_{1,1} \geq b_3 + b_4 + b_6;\\
&f_{2,0} \geq c_0 + c_1 + c_5\\
&f_{2,2} \geq c_2\\
&(\text{flow out of } g)\\
&g_0 = f_{0,0} + f_{0,1}\\
&g_1 = f_{1,1}\\
&g_2 = f_{2,0} + f_{2,2}\\
&(\text{flow into } r)\\
&r_0 = f_{0,0} + f_{2,0}\\
&r_1 = f_{0,1} + f_{1,1}\\
&r_2 = f_{2,2}\\
&(\text{Populations in r})\\
&r_0 = N_0 + N_1 + N_5\\
&r_1 = N_3 + N_4 + N_6\\
&r_2 = N_2
\end{aligned}$$

This problem finds an H^* such that workload queries on H^* match the noisy answers as close as possible with the following constraints. First, the entries of H^* are nonnegative. Second, the population total matches that of the real data. Third, the set of equations define g_0, g_1, g_2 (i.e., g_0 consists of Hispanic and non-Hispanic people in GQ A) and provide lower bounds on the amount of people in each GQ type (obtained by adding the lower bounds of that GQ in each region).

The next set of equations provide a lower bound on the flow. The idea is that a flow variable $f_{i,j}$ is the number of people in GQ type i that we want to move into a region with GQ combination j. For example $f_{0,0}$ is the number of people from GQ A that are assigned to regions having GQ combination $\{A, C\}$. These regions are 0,1,5 (Suffolk, Nassau, Providence) and require at least a_0, a_1, a_5 people in GQ A in each of them. Hence $f_{0,0} \geq a_0 + a_1 + a_5$.

The next set of equations define the flow out of the group quarters types. For example, g_0 will be the number of people in GQ A. These people will only flow into regions with GQ combinations $\{A, C\}$ and $\{A, B\}$ hence the total number of people flowing out of GQ A is $f_{0,0} + f_{0,1}$.

The next set of equations define the corresponding inflow. For example, regions with GQ combinations $\{A, C\}$ will receive flow from GQ A and GQ C, so $r_0 = f_{0,0} + f_{2,0}$.

The final set of equations list the population totals in regions with the same GQ combinations. For example, regions with GQ combination $\{A, C\}$ have $N_0 + N_1 + N_5$ people.

All but the first 3 lines of Equation 19 define the implied constraints. These are the same implied constraints that we will use for the L1 rounding part of the national solve (i.e. we put those implied constraints into Equation 7).

Recursive Solve. We use the recursive solve to create the child histograms (e.g., state-level histograms) from the national histogram. Then, for each state we create its child histograms (e.g., county-level histograms). The construction of implied constraints is similar—we create a set of implied constraints for each child.

For example, when creating the state-level histograms from the national histogram, we define $g_{NY,0}$, $g_{NY,1}$, $g_{NY,2}$ to be the populations in NY in GQ types A,B,C. We create $r_{NY,0}$ for the GQ combination $\{A, C\}$ and $r_{NY,2}$ for the GQ combination $\{C\}$ (note there is no $r_{NY,1}$ because there is no combination $\{A, B\}$ in NY). We set the lower bounds for $g_{NY,0}$, $g_{NY,1}$, $g_{NY,2}$ based on the leaves (counties) in NY, we set the equality constraints on $r_{NY,0}$ and $r_{NY,2}$ from the leaves (counties) in NY, etc. Then, we define flow variables for NY that flow from GQ types into GQ combinations, etc. For the child RI, we create a similar network flow.

7.3.2 *The general case*

For simplicity, we will discuss these extensions in the context of group quarters facilities. That is there are n_g types of group quarters ("in a household" is considered one of those types, and implies "not in group quarters"). For $i = 1, \ldots, n_g$, we let X_i be the set of cells in the histogram $\widetilde{H}^0$ that correspond to sub-populations that belong to group quarters type i (note that X_i is then also the set of cells in the histogram $\widetilde{H}^1_{NY}$ that corresponds to the sub-populations in NY in group quarters type i).

We will consider the following background knowledge on the leaves: the types of GQ that are present in each leaf is known and a lower bound on the population in each GQ type in each leaf is known. If a GQ type is not present in a leaf, there are no people in that GQ type in the leaf. The public knowledge also includes the total population in each leaf.

Nation-level solves. Now we discuss what constraints need to be added to the national solves. We will introduce the following variables/constants:

- First, we introduce a variable GQ_i for each gq type. Let there be n_g types. Recall X_i is the set of histogram cells for people in GQ type i.
- For each GQ type, there is the corresponding constant gq_i equal to the sum (over all blocks) of the minimum guaranteed population in GQ of type i.
- Let $Y_1, Y_2, \ldots$ be the distinct GQ combinations that appear in some block (in our previous example, these would be $Y_1 = \{A, C\}$, $Y_2 = \{A, B\}$ and $Y_3 = \{C\}$). Let there be n_c such combinations.
- For each Y_i we introduce a variable CB_i.
- For each Y_i, let N_i be the total population in blocks that have GQ combination Y_i.
- Let $H^*[1], ..., H^*[m]$ be the variables corresponding to the national histogram cells that we want to solve for.
- We define flow variables F_{ij} if a GQ type i appears in combination j (i.e. if $i \in Y_j$).
- For each legal flow F_{ij} variable (i.e. $i \in Y_j$), let f_{ij} be a lower bound on it. f_{ij} looks at all blocks that have GQ combination Y_j and adds up the minimum populations for GQ of type i in those blocks. In particular, this means $\sum_j f_{ij} = gq_i$.

At the national level, optimization problem with the added network flow constraints are:

$$\arg \min_{H^*, GQ_j, CB_\ell} \sum_{Q_i \in W^0_{\gamma_0}} ||Q_i(H^*) - m_{\gamma_0,i}||_2^2 \tag{20}$$

$$\begin{aligned}
&\text{s.t. } H^*[i] \geq 0 \quad (\text{for } i = 1, \ldots, m)\\
&\sum_{i=0}^{m} H^*[i] = \text{Total Population}\\
&(\text{define the } GQ_i) \qquad (21)\\
&GQ_j = \sum_{i \in X_j} H^*[i] \quad \text{for } j = 1, \ldots, n_g\\
&GQ_j \geq gq_j \quad \text{for } j = 1, \ldots, n_g\\
&(\text{Lower bounds on Flow})\\
&F_{ij} \geq f_{ij} \quad \text{for all } i, j \text{ with } i \in Y_j\\
&(\text{Flow out of GQ})\\
&GQ_i = \sum_{j\,:\,i \in Y_j} F_{ij} \quad \text{for } i = 1, \ldots, n_g\\
&(\text{Flow into GQ combination regions})\\
&CB_j = \sum_{i\,:\,i \in Y_j} F_{ij} \quad \text{for } j = 1, \ldots, n_c\\
&(\text{Populations in GQ regions})\\
&CB_j = N_j \quad \text{for } j = 1, \ldots, n_c
\end{aligned}$$

These constraints (following Eq 21) are the same added constraints that we use in place of the implied constraints in the L_1 solve (Equation 7):

$$\widetilde{H}^0 = \arg \min_{H^\dagger} -(H^\dagger - \lfloor H^* \rfloor) \cdot (H^* - \lfloor H^* \rfloor) \tag{22}$$

$$\begin{aligned}
&\text{s.t. } H^\dagger \succeq 0 \text{ (nonnegativity)}\\
&\sum_x H^\dagger[x] = \sum_x H^*[x] \text{ (total sum constraint)}
\end{aligned}$$

$$
\begin{aligned}
&H^\dagger[x] - H^*[x] \leq 1 \text{ for all cells } x\\
&-H^\dagger[x] + H^*[x] \leq 1 \text{ for all cells } x\\
&\text{(define the } GQ_i\text{)}\\
&GQ_j = \sum_{i\in X_j} H^\dagger[i] \quad \text{for } j=1,\dots,n_g\\
&GQ_j \geq gq_i \quad \text{for } j=1,\dots,n_g\\
&\text{(Lower bounds on Flow)}\\
&F_{ij} \geq f_{ij} \quad \text{for all } i,j \text{ with } i\in Y_j\\
&\text{(Flow out of GQ)}\\
&GQ_i = \sum_{j\,:\,i\in Y_j} F_{ij} \quad \text{for } i=1,\dots,n_g\\
&\text{(Flow into GQ combination regions)}\\
&CB_j = \sum_{i\,:\,i\in Y_j} F_{ij} \quad \text{for } j=1,\dots,n_c\\
&\text{(Populations in GQ regions)}\\
&CB_j = N_j \quad \text{for } j=1,\dots,n_c
\end{aligned}
$$

THEOREM 4. *Suppose the public knowledge is the following: (1) the types of GQ that are present in each leaf, (2) the lower bound on the population in each GQ type in each leaf, (3) the total population in each leaf, (4) if a GQ type is not present in a leaf, there are no people in that GQ type in the leaf. The network flow constraints (following Eq 21), when added to the L_2 solve (Eq 6) without any other implied constraints (resulting in Eqn 20) produce a fractional histogram at the national level that can be extended to fractional histograms at the leaves that satisfy the public knowledge about GQ populations.*

For proof, see Appendix G.

THEOREM 5. *Suppose the public knowledge is the following: (1) the types of GQ that are present in each leaf, (2) he lower bound on the population in each GQ type in each leaf, (3) the total population in each leaf, (4) if a GQ type is not present in a leaf, there are no people in that GQ type in the leaf. The network flow constraints (following Eq 21), when added to the L_1 solve (Eq 7) without any other implied constraints (resulting in Equation 22), and solved using the simplex algorithm, produce an integer histogram at the national level that can be extended to integer histograms at the leaves that satisfy the public knowledge about GQ populations.*

For proof, see Appendix H.

Recursive solves. The recursive solves involve adding a set of network flow constraints for each child. This is straightforward, but for completeness, we provide the optimization problems here. For this purpose, let Y_j^τ refer to sets of GQ combinations in leaves under child τ with N_j^τ being the total population of leaves under child τ with GQ combination Y_j^τ. Similarly, gq_i^τ is the sum of the minimum populations in GQ i for leaves under child τ and f_{ij}^τ looks at all blocks under child τ that have GQ combination Y_j^τ and add up the minimum populations for GQ of type i in those blocks. Again, $\sum_j f_{ij}^\tau = gq_i^\tau$. n_g is the number of group quarters types (does not depend on τ) and n_c^τ is the number of group quarter combinations appearing in leaves of child τ. We introduce variables GQ^τ, F_{ij}^τ, CB^τ.

The L_2 solve is:

$$
\arg\min_{H_1^*,\dots,H_m^*} \sum_{\tau=1}^{m} \sum_{Q_\ell\in W_{\gamma_\tau}^{i+1}} ||Q_\ell(H_\tau^*) - m_{\gamma_\tau,\ell}||_2^2 \tag{23}
$$

$$
\begin{aligned}
\text{s.t. } & H_\tau^* \succeq 0 \quad \text{for all } \tau\\
&\sum_{\tau=1}^{m} H_\tau^*[x] = \widetilde{H}_\gamma^i[x] \text{ for all cells } x\\
&Q'_\ell(H_\tau^*) \text{ op}_\ell\ c_\ell \text{ is true for all } \tau\\
&\text{(define the } GQ_i^\tau\text{)}\\
&GQ_j^\tau = \sum_{i\in X_j} H_\tau^*[i] \quad \text{for } j=1,\dots,n_g \text{ and all } \tau\\
&GQ_j^\tau \geq gq_j^\tau \quad \text{for } j=1,\dots,n_g\\
&\text{(Lower bounds on Flow)}\\
&F_{ij}^\tau \geq f_{ij}^\tau \quad \text{for all } \tau \text{ and } i,j \text{ with } i\in Y_j^\tau\\
&\text{(Flow out of GQ)}\\
&GQ_i^\tau = \sum_{j\,:\,i\in Y_j^\tau} F_{ij}^\tau \quad \text{for all } \tau \text{ and } i=1,\dots,n_g\\
&\text{(Flow into GQ combination regions)}\\
&CB_j^\tau = \sum_{i\,:\,i\in Y_j^\tau} F_{ij}^\tau \quad \text{for all } \tau \text{ and } j=1,\dots,n_c^\tau\\
&\text{(Populations in GQ regions)}\\
&CB_j^\tau = N_j^\tau \quad \text{for all } \tau \text{ and } j=1,\dots,n_c^\tau
\end{aligned}
$$

and the L_1 solve is:

$$
\widetilde{H}_{\gamma_1}^{i+1},\dots,\widetilde{H}_{\gamma_m}^{i+1} \tag{24}
$$

$$
= \arg\min_{H_1^\dagger,\dots,H_m^\dagger} \sum_{\tau=1}^{m} -(H_\tau^\dagger - \lfloor H_\tau^* \rfloor)\cdot(H_\tau^* - \lfloor H_\tau^* \rfloor) \tag{25}
$$

$$
\begin{aligned}
&\text{s.t. } H_\tau^\dagger \succeq 0 \text{ for all } \tau\\
&H_\tau^\dagger[x] - H_\tau^*[x] \leq 1 \text{ for all } \tau \text{ and cells } x\\
&-H_\tau^\dagger[x] + H_\tau^*[x] \leq 1 \text{ for all } \tau \text{ and cells } x\\
&\sum_\tau H_\tau^\dagger[x] = \widetilde{H}_\gamma^i[x] \text{ for all cells } x\\
&\text{(define the } GQ_i^\tau\text{)}\\
&GQ_j^\tau = \sum_{i\in X_j} H_\tau^*[i] \quad \text{for } j=1,\dots,n_g \text{ and all } \tau\\
&GQ_j^\tau \geq gq_j^\tau \quad \text{for } j=1,\dots,n_g\\
&\text{(Lower bounds on Flow)}\\
&F_{ij}^\tau \geq f_{ij}^\tau \quad \text{for all } \tau \text{ and } i,j \text{ with } i\in Y_j^\tau\\
&\text{(Flow out of GQ)}\\
&GQ_i^\tau = \sum_{j\,:\,i\in Y_j^\tau} F_{ij}^\tau \quad \text{for all } \tau \text{ and } i=1,\dots,n_g\\
&\text{(Flow into GQ combination regions)}\\
&CB_j^\tau = \sum_{i\,:\,i\in Y_j^\tau} F_{ij}^\tau \quad \text{for all } \tau \text{ and } j=1,\dots,n_c^\tau\\
&\text{(Populations in GQ regions)}\\
&CB_j^\tau = N_j^\tau \quad \text{for all } \tau \text{ and } j=1,\dots,n_c^\tau
\end{aligned}
$$

THEOREM 6. *Suppose the public knowledge is the following: (1) the types of GQ that are present in each leaf, (2) the lower bound on the population in each GQ type in each leaf, (3) the total population in each leaf, (4) if a GQ type is not present in a leaf, there are no people in that GQ type in the leaf. The network flow constraints, when added to the L_2 solve without any other implied constraints (resulting in Eqn 23) produce a fractional child histogram that can be extended to fractional histograms at the leaves that satisfy the public knowledge about GQ populations.*

For proof, see Appendix I.

THEOREM 7. *Suppose the public knowledge is the following: (1) the types of GQ that are present in each leaf, (2) the lower bound on the population in each GQ type in each leaf, (3) the total population in each leaf, (4) if a GQ type is not present in a leaf, there are no people in that GQ type in the leaf. The network flow constraints, when added to the L_1 solve without any other implied constraints (resulting in Equation 25), and solved using the simplex algorithm, produce an integer child histogram that can be extended to integer histograms at the leaves that satisfy the public knowledge about GQ populations.*

For proof, see Appendix J.

7.4 Composing Implied Constraints

In general, constraints do not compose well—for a constraint set A it may be easy to find a complete set of implied constraints, and for a constraint set B it may be easy to find a complete set of implied constraints, but for the constraint set $A \cup B$ it might be infeasible to find a complete set of implied constraints. A simple example is if A is the set of equality constraints on a one-way marginal on an attribute that can have at least 3 values and B is the set of equality constraints on a one-way marginal on a different attribute that can have at least 3 values. Each set is easy by itself, but combining A and B results in an NP-complete problem (Theorem 2). In this section we consider special cases where constraints compose. In particular, we identify situations in which adding (or removing) a constraint on $\widetilde{H}$ does not affect the implied constraints on $\widetilde{H}^0$.

7.4.1 Structural Zero Cells

Let Z be a set such that for every leaf node τ, the demographic characteristics histogram $\widetilde{H}_\tau$ at that leaf must have zeros in the cells specified by Z (i.e. $\widetilde{H}_\tau[i] = 0$ for all $i \in Z$). The next result shows that these structural zero cells do not interfere with implied constraints.

THEOREM 8. *Let $\widetilde{H}_p$ be a demographic characteristics histogram at node p that we must extend to histograms $\widetilde{H}_{\tau_1}, \dots, \widetilde{H}_{\tau_k}$, where $\tau_1, \dots, \tau_k$ are the leaf nodes under p. For each τ_i, let $\mathcal{C}_{\tau_i}$ be a set of constraints on $\widetilde{H}_{\tau_i}$ and let $\mathcal{C}_p$ be a complete set of implied constraints for $\widetilde{H}_p$. Let Z be a set of indices. Define $\mathcal{C}^\dagger_{\tau_i} = \mathcal{C}_{\tau_i} \cup \left\{ \widetilde{H}_{\tau_i}[j] = 0 \mid j \in Z \right\}$ and $\mathcal{C}^\dagger_p = \mathcal{C}_p \cup \left\{ \widetilde{H}_p[j] = 0 \mid j \in Z \right\}$. Then $\mathcal{C}^\dagger_p$ is a complete set of implied constraints for $\mathcal{C}^\dagger_{\tau_1}, \dots, \mathcal{C}^\dagger_{\tau_k}$.*

For proof see Appendix K.

8. APPLICATIONS TO INVARIANTS UNDER CONSIDERATION FOR THE 2020 CENSUS

We first describe the 2020 Census data that are the inputs to the TopDown algorithm. Second we list the potential invariants. Finally we show how the algorithms developed in this paper will be applied.

8.1 Detailed Description of Census Data

The microdata used to generate PL94-171 data can be viewed as a table with the following attributes:

- R: Race. It has 63 possible values. Each value corresponds to a non-empty subset of the following 6 OMB categories: *American Indian or Alaskan Native*, *Asian*, *Black or African American*, *Native Hawaiian or Other Pacific Islander*, *White*, *Other*.
- E: Ethnicity. It has 2 values: *Hispanic or Latino*, and *not Hispanic or Latino*.
- V: Voting Age Status. It has 2 values: whether a person has age ≥ 18, or age < 17.
- H: Housing/Group Quarters status. There are 8 possible values that describe the type of housing an individual lives in: *household*, *Correctional Facilities for Adults*, *Juvenile Facilities*, *Nursing Facilities/Skilled-Nursing Facilities*, *Other Institutional Facilities*, *College/University Student Housing*, *Military Quarters*, *Other Noninstitutional Facilities*.
- L: Location. It represents the 2020 Census tabulation block that an individual lives in. This is a hierarchical attribute that is coded as a 15 digit number. The first 2 digits represent the state; the next 3 represent the county within the state; the next 6 refer to tract within county; the last 4 refer to block within tract. The first digit of the block code is called the block group. There were over 6.2 million inhabited blocks in 2010.

For Demographic and Housing Characteristics (DHC)-Persons, the microdata table used to generate person-level tabulations formerly denoted Summary File 1 [37], is an extension of the table used to create PL94-171. DHC-Persons adds the following information:

- S. Sex: two values (male, female).
- A. Detailed age (0-115). It extends the V attribute (voting age status).
- RH. Relation to householder. It has 43 values and extends the H attribute (Housing/Group Quarters status). For people living in households, the 15 possible values in 2010 were *householder*, *husband or wife*, *biological son or daughter*, *adopted son or daughter*, *stepson or stepdaugther*, *brother or sister*, *father or mother*, *grandchild*, *parent-in-law*, *son-in-law or daughter-in-law*, *other relative*, *roomer or boarder*, *housemate or roommate*, *unmarried partner*, *other nonrelative*. For people living in group quarters, there are 28 possible values that provide more detail than the H attribute.

8.2 Invariants and Structural Zeros

We have the following invariants for PL94-171, as implemented for the 2018 End-to-End Census Test: [6]

1. Number of housing units in each block is invariant. Equivalent to upper bounds on number of householders.
2. Number of occupied group quarters by major type (7 levels) in each block is invariant. Equivalent to lower bounds on number of people with living in that type of GQ in each block.
3. Population of each state is invariant (under consideration is also an alternative where the block-level population is invariant).

For the DHC-Persons table, in addition to the PL94-171 invariants, we have the following invariant:

1. Number of occupied group quarters by detailed type (29 levels) by single-sex status (3 levels) is invariant. Equivalent to lower bounds on number of people with certain sex and GQ attribute combinations in persons table.

DHC-Persons also has the following structural zeros:

1. Some GQ are single sex or co-ed
2. Some GQ have upper and lower bounds on age.
3. Householder age must be at least 15
4. Number of householders $\geq$ number of spouses and unmarried partners
5. Number of householders $\leq$ 2 times number of parents of householder
6. Number of householders $\leq$ 4 times number of grandparents of householder
7. Each block that contains a person living in a household must also have a householder
8. Children of householder must have certain age gap with householder
9. Parents of householder must have a certain age gap with householder
10. Grandparents of householder must have a certain age gap with householder

8.2.1 *Enforced Invariants and Structural Zeros*

When the initial table includes attributes on detailed GQ type (or whether a person lives in a household and is or is not a householder), age, and sex, it is possible to enforce all of the PL94 invariants and the DHC invariants related to group quarters using the network flows of Section 7.3. The structural zeros from Items 1, 2, 3 can also be incorporated by Theorem 8.

[6]There was only one county, Providence, RI, in the End-to-End Test, so the state-level population invariant was actually a county population invariant. In addition, occupied housing units were invariant in the End-to-End test, but that invariant has now been removed.

8.2.2 *Unenforced Structural Zeros*

When constructing the histogram corresponding to the DHC-Persons table, we note that some of the structural zeros are *not* linear constraints on the histogram—Items 7, 8, 9, 10. Except in rare situations (e.g., a block has no GQ and at most 1 housing unit), we can only directly verify that these constraints hold if we have a household identifier attribute, and even then, verification of those constraints involves multiple self-joins between the resulting table. It is an open problem how to derive implied constraints in such a situation.

Thus, when creating the DHC-Persons table, these constraints will not be enforced except on a best-effort basis. For example, if there is a block with no GQ but at most 1 housing unit, then all the generated people records are from the same household. We may edit the age or relation attribute to ensure the structural zeros hold at that block.

8.2.3 *Partially Enforced Structural Zeros*

Items 4, 5, 6 are linear constraints on a one-way marginal at each block. Note that adding one-way marginal constraints to other constraints can make finding implied constraints NP-hard (for example adding a one-way equality constraint). Thus, if the full relation to householder variable is used in the initial table, we can add (to our existing implied constraints) linear equations corresponding to Items 4, 5, 6 at each level of geography. If infeasibility occurs in any of the solves performed by TopDown, the solve can be redone without the constraints corresponding to Items 4, 5, 6. Since this solve only uses the original implied constraints based on network flows, it will be feasible (by definition of implied constraints). After TopDown finishes, it may be possible to add an edit phase, similar to the Fellegi-Holt model [11] to resolve them. One idea is that if the number of blocks for which Items 4, 5, 6 fail to hold is relatively small, we can set up an integer linear program to correct those variables while minimizing the change in an objective function (such as the error in answering the queries for which we have noisy measurements).

9. THE FAILSAFE

The TopDown algorithm solves a series of constrained optimization problems and uses careful analyses of the constraints to ensure that they are feasible. However, in a complex system, occasional infeasibility should be expected. This can happen either as a bug in setting up the quadratic/linear/integer program or when some implied constraints are omitted for performance reasons.

When a histogram $\widetilde{H}^i$ at level i of the hierarchy is being extended (for example, when a demographic characteristics histogram at the county level is being extended to histograms at the tract level) either the L_2 or the L_1 solve may fail. When this happens, the *failsafe* (a backup piece of code), is invoked. It consists of the following steps.

1. First, a *distance to feasibility* is computed. The constraint that the child histograms add up to the parent histogram is removed. The objective function is also removed. Instead, the optimization problem uses the rest of the constraints (which are constraints on the children) and tries to find a set of child histograms that satisfies those constraints, while minimizing the L_1 distance between the sum of the child histograms

and $\widetilde{H}^i$ (in the L_1 metric). This distance, denoted d^*, is the distance to feasibility.

2. Now we take the infeasible optimization problem, remove the constraint that the child histograms add up to the parent, and add the constraint that the L_1 distance between the sum of the children and $\widetilde{H}^i$ is at most $d^* + 1$ (the +1 is a fudge factor). For attributes that are not involved in the implied constraints, we also force the marginal of $\widetilde{H}$ on those attributes to equal the sum of the child marginals. In particular, this forces the sum of the populations in the child nodes to equal the population in $\widetilde{H}^i$. Assuming that there exist child nodes that satisfy the implied constraints (or the subset of those constraints that was implemented) with population totals that add up to the population total in $\widetilde{H}^i$, then the marginal constraints (involving attributes that are not part of the implied constraints) do not cause infeasibility. Intuitively, this is because we can arbitrarily redistribute those demographic characteristics among the children without affecting any of the implied constraints.

10. CONCLUSIONS AND FUTURE WORK

This paper presents a description of the TopDown algorithm that is being developed to produce differentially private microdata for the 2020 Census of Population and Housing. The algorithm creates an initial set of microdata and then extends it (for example, by adding geographic attributes to the records). The key technical challenge of this algorithm is to preserve invariants—certain queries where the differentially private data must exactly match the true data. Invariants lead to implied constraints which ensure that a set of microdata can be extended with additional attributes while satisfying the invariants. Generally, finding implied constraints is intractable. In some cases where the implied constraints are not intractable, network flows can be used to find them.

11. REFERENCES

[1] John Abowd. The U.S. Census Bureau adopts differential privacy. KDD Invited Talk, August 2018.

[2] Boaz Barak, Kamalika Chaudhuri, Cynthia Dwork, Satyen Kale, Frank McSherry, and Kunal Talwar. Privacy, accuracy, and consistency too: A holistic solution to contingency table release. In *Proceedings of the Twenty-sixth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, 2007.

[3] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, 2017.

[4] U. S. Census Bureau. On the map: Longitudinal employer-household dynamics. `https://onthemap.ces.census.gov/`.

[5] U.S. Census Bureau. Local update of census addresses operation (luca). `https://www.census.gov/programs-surveys/decennial-census/about/luca.html`, 2018.

[6] Graham Cormode, Cecilia Procopiuc, Divesh Srivastava, Entong Shen, and Ting Yu. Differentially private spatial decompositions. In *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering*, 2012.

[7] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[8] Bolin Ding, Marianne Winslett, Jiawei Han, and Zhenhui Li. Differentially private data cubes: Optimizing noise sources and consistency. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, 2011.

[9] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.

[10] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, 2014.

[11] I. P. Fellegi and D. Holt. A systematic approach to automatic edit and imputation. *Journal of the American Statistical Association*, 71(353):17–35, 1976.

[12] Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Zhiwei Steven Wu. Dual query: Practical private query release for high dimensional data. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, 2014.

[13] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[14] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. Universally utility-maximizing privacy mechanisms. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, 2009.

[15] LLC Gurobi Optimization. Gurobi optimizer reference manual, 2018.

[16] Moritz Hardt, Katrina Ligett, and Frank McSherry. A simple and practical algorithm for differentially private data release. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*, 2012.

[17] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. Boosting the accuracy of differentially private histograms through consistency. *Proc. VLDB Endow.*, 2010.

[18] Xi He, Ashwin Machanavajjhala, and Bolin Ding. Blowfish privacy: Tuning privacy-utility trade-offs using policies. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, 2014.

[19] IBM. Ilog cplex, 2019.

[20] Jean-Louis Imbert. About redundant inequalities generated by fourier's algorithm. In *Proceedings of the Fourth International Conference on Artificial Intelligence: Methodology, Systems, Applications*, 1990.

[21] Noah Johnson, Joseph P. Near, and Dawn Song.

Towards practical differential privacy for sql queries. *Proc. VLDB Endow.*, 11(5), 2018.

[22] Daniel Kifer and Ashwin Machanavajjhala. A rigorous and customizable framework for privacy. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, 2012.

[23] Jaewoo Lee, Yue Wang, and Daniel Kifer. Maximum likelihood postprocessing for differential privacy under consistency constraints. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015.

[24] Chao Li, Gerome Miklau, Michael Hay, Andrew Mcgregor, and Vibhor Rastogi. The matrix mechanism: Optimizing linear counting queries under differential privacy. *The VLDB Journal*, 24(6):757–781, 2015.

[25] Bing-Rong Lin and Daniel Kifer. Information measures in statistical privacy and data processing applications. *ACM Trans. Knowl. Discov. Data*, 9(4), 2015.

[26] Jesús A. De Loera and Shmuel Onn. The complexity of three-way statistical tables. *SIAM J. Comput.*, 33(4):819–836, 2004.

[27] Ashwin Machanavajjhala, Daniel Kifer, John Abowd, Johannes Gehrke, and Lars Vilhuber. Privacy: From theory to practice on the map. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, pages 277–286, 2008.

[28] Ryan McKenna, Gerome Miklau, Michael Hay, and Ashwin Machanavajjhala. Optimizing error of high-dimensional statistical queries under differential privacy. *Proc. VLDB Endow.*, 2018.

[29] Frank D. McSherry. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, 2009.

[30] David Monniaux. Quantifier elimination by lazy model enumeration. In *CAV*, 2010.

[31] George L. Nemhauser and Laurence A. Wolsey. *Integer and Combinatorial Optimization.* Wiley-Interscience, New York, NY, USA, 1988.

[32] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1982.

[33] Davide Proserpio, Sharon Goldberg, and Frank McSherry. Calibrating data to sensitivity in private data analysis: A platform for differentially-private analysis of weighted datasets. *Proc. VLDB Endow.*, 2014.

[34] Apple Differential Privacy Team. Learning with privacy at scale. *Apple Machine Learning Journal*, 1(8), 2017.

[35] Caroline Uhler, Aleksandra Slavkovic, and Stephen E. Fienberg. Privacy-preserving data sharing for genome-wide association studies. *Journal of Privacy and Confidentiality*, 5(1):137–166, 2013.

[36] 2010 Census Redistricting Data (Public Law (P.L.) 94-171) Summary File – United States machine-readable data files/prepared by the U.S. Census Bureau, 2011.

[37] 2010 Census Summary File 1 United States/prepared by the U.S. Census Bureau, 2011.

[38] 2010 Census Summary File 1 Urban/Rural Update United States/prepared by the U.S. Census Bureau, 2011.

[39] Jia Xu, Zhenjie Zhang, Xiaokui Xiao, Yin Yang, Ge Yu, and Marianne Winslett. Differentially private histogram publication. *The VLDB Journal*, 22(6), 2013.

APPENDIX

A. IMPLIED CONSTRAINTS FROM EQUALITY-CONSTRAINT COUNT QUERIES

EXAMPLE 3. *Suppose the only variables we are interested in are* geography R_G *(which state a person is in),* voting age status R_V *(whether a person is voting age or not, and* householder status R_H *(whether a person is a householder or not). Only the national-level histogram will be published. However there is public knowledge at the state level state: the one-dimensional marginal on R_V and the one-dimensional marginal on R_H are known exactly. For our simple example, suppose we only have 2 states A and B.*

Region A Invariants

	$\mathbf{R_V = 0}$	$\mathbf{R_V = 1}$	
$\mathbf{R_H = 0}$	*?*	*?*	*5*
$\mathbf{R_H = 1}$	*?*	*?*	*15*
	15	*5*	

Region B Invariants

	$\mathbf{R_V = 0}$	$\mathbf{R_V = 1}$	
	?	*?*	*15*
	?	*?*	*5*
	5	*15*	

What conditions should the national level histogram satisfy so that it can be extended to state level histograms having those row and column sums? A naive, but intuitive, suggestion is that the margins of the national histogram should equal the sum of the margins of the state level histograms as follows:

Aggregate Invariants?

	$\mathbf{R_V = 0}$	$\mathbf{R_V = 1}$	***row sum***
$\mathbf{R_H = 0}$	z_1	z_2	*20*
$\mathbf{R_H = 1}$	z_3	z_4	*20*
col sum	*20*	*20*	

It turns out that these conditions are not sufficient. The following table satisfies the proposed restrictions at the national level yet it cannot be extended to the state level.

Aggregate Invariants Counterexample

	$\mathbf{R_V = 0}$	$\mathbf{R_V = 1}$	***row sum***
$\mathbf{R_H = 0}$	*11*	*9*	*20*
$\mathbf{R_H = 1}$	*9*	*11*	*20*
col sum	*20*	*20*	

This counterexample contains 11 records with attributes $R_H = 0, R_V = 0$. We know there can be at most 5 such records in State A (because there are at most 5 records with $R_H = 0$ in State A) and at most 5 such records in State B (because there are at most 5 records with $R_V = 0$ in State B). Hence, the national level data had too many records with $R_H = 0, R_V = 0$. It turns out that an additional constraint is needed at the national level: $z_1 \in [0, 10]$.

B. PROOF OF THEOREM 1

PROOF. Clearly the problem is in NP. We do a reduction from 3-SAT. Let ϕ be a 3-SAT formula with w variables and c clauses. We define two attributes R_1 with domain $\Omega_1 = \{1, 2, \dots, w\}$ and R_2 with domain $\Omega_2 = \{0, 1\}$. We set our schemas to be $S^0 = \{R_1\}$ and $S = \{R_1, R_2\}$. We set the corresponding histogram $\widetilde{H}^0$ to be w-dimensional vector $[1, 1, 1, \dots, 1]$. Then we define the constraint set $\mathcal{C}$ (over histograms of dimension $w \times 2$) so that it encodes the 3-SAT problem ϕ. The intuition is that we would like $\widetilde{H}[j, 0]$ to encode the truth value of variable v_j and $\widetilde{H}[j, 1]$ to encode the truth value of $\neg v_j$.

So, for every clause involving, say, variables v_i, v_j, v_ℓ we add the linear constraint $\widetilde{H}[i, a] + \widetilde{H}[j, b] + \widetilde{H}[\ell, c] \geq 1$, where $a = 1$ if v_i is negated and 0 if v_i is not negated. Similarly, b and c, respectively, encode whether v_j and v_ℓ are negated or not. Thus there are a total of c inequality constraints.

The fact that $\widetilde{H}$ is supposed to be an extension of our chosen $\widetilde{H}^0$ means that $\widetilde{H}[i, 0] + \widetilde{H}[i, 1] = 1$ for all i. Now, the entries of $\widetilde{H}$ must be nonnegative integers, so this means that for all i, either $\widetilde{H}[i, 0] = 1$ or $\widetilde{H}[i, 1] = 1$ and so $\widetilde{H}$ can be interpreted as a truth value assignment for each variable.

Clearly, if $\widetilde{H}^0$ can be extended into a histogram $\widetilde{H}$ that satisfies $\mathcal{C}$ then this gives us a satisfying assignment for 3-SAT and any satisfying assignment for 3-SAT produces a histogram $\widetilde{H}$ that satisfies $\mathcal{C}$ and extends $\widetilde{H}^0$. Thus the decision problem is NP-complete. □

C. PROOF OF THEOREM 2

PROOF. This is equivalent to the 3-table/2-marginal existence problem [26]. In this problem, there are three attributes A, B, C and one specifies a histogram H_1 on attributes A, B, H_2 on attributes B, C and H_3 on attributes A, C. The problem is to determine if there is a histogram H on A, B, C that is consistent with H_1, H_2, H_3 (that is, H_1, H_2, H_3 should be the two-way marginals of H – if we add up H over the third dimension, we should obtain H_1, adding up over the first dimension should yield H_2 and adding up over the second dimension should yield H_3). In this problem, if all attributes have domain size at least 3, then the problem is NP-complete in the domain size of A and B (i.e., even if we fix $|C| = 3$).

In our case, the full histogram we want to construct is $\widetilde{H}$, a histogram on 3 attributes $\{R_1, R_2, \text{Location}\}$. We start with $\widetilde{H}^0$, a histogram on $\{R_1, R_2\}$. Thus $\widetilde{H}^0$ would be a two-dimensional marginal of $\widetilde{H}$. Meanwhile, the one-dimensional histogram on R_1 at each location can be represented as a two-dimensional histogram over $\{R_1, \text{Location}\}$ (again, it is supposed to be a 2-d marginal of $\widetilde{H}$) and similarly, the one-dimensional histogram on R_2 at each location can be represented as a two-dimensional histogram over $\{R_2, \text{Location}\}$ (also, it is supposed to be a 2-d marginal of $\widetilde{H}$). Thus we have 3 two-dimensional marginals and are asking if there exists a table $\widetilde{H}$ that is consistent with all of them. This is exactly the same as the 3-table/2-marginal existence problem [26]. □

D. FME IN ACTION IN EXAMPLE 2

We write out the full constraints:

$$h_A[0,0] + h_A[0,1] = 6 \qquad h_B[0,0] + h_B[0,1] = 15 \tag{26}$$
$$h_A[1,0] + h_A[1,1] = 16 \qquad h_B[1,0] + h_B[1,1] = 5 \tag{27}$$
$$h_A[0,0] + h_A[1,0] = 17 \qquad h_B[0,0] + h_B[1,0] = 5 \tag{28}$$
$$h_A[0,1] + h_A[1,1] = 5 \qquad h_B[0,1] + h_B[1,1] = 15 \tag{29}$$

$$h_A[0,0] \geq 0 \quad h_A[0,1] \geq 0 \quad h_A[1,0] \geq 0 \quad h_A[1,1] \geq 0 \tag{30}$$
$$h_B[0,0] \geq 0 \quad h_B[0,1] \geq 0 \quad h_B[1,0] \geq 0 \quad h_B[1,1] \geq 0 \tag{31}$$

$$h[0,0] = h_A[0,0] + h_B[0,0] \qquad h[0,1] = h_A[0,1] + h_B[0,1] \tag{32}$$
$$h[1,0] = h_A[1,0] + h_B[1,0] \qquad h[1,1] = h_A[1,1] + h_B[1,1] \tag{33}$$

First we use Gaussian Eliminatin to replace $h_A[0,0]$ in Equations 28, 30,32 with $6 - h_A[0,1]$ (obtained from Equation 26. This results in

$$\begin{aligned} h_A[0,0] + h_A[0,1] &= 6 & h_B[0,0] + h_B[0,1] &= 15 \\ h_A[1,0] + h_A[1,1] &= 16 & h_B[1,0] + h_B[1,1] &= 5 \\ 6 - h_A[0,1] + h_A[1,0] &= 17 & h_B[0,0] + h_B[1,0] &= 5 \\ h_A[0,1] + h_A[1,1] &= 5 & h_B[0,1] + h_B[1,1] &= 15 \end{aligned}$$

$$6 - h_A[0,1] \geq 0 \quad h_A[0,1] \geq 0 \quad h_A[1,0] \geq 0 \quad h_A[1,1] \geq 0$$
$$h_B[0,0] \geq 0 \quad h_B[0,1] \geq 0 \quad h_B[1,0] \geq 0 \quad h_B[1,1] \geq 0$$

$$h[0,0] = 6 - h_A[0,1] + h_B[0,0] \qquad h[0,1] = h_A[0,1] + h_B[0,1]$$
$$h[1,0] = h_A[1,0] + h_B[1,0] \qquad h[1,1] = h_A[1,1] + h_B[1,1]$$

Then we use FME to eliminate $h_A[0,0]$ altogether. Note that it only appears in one place: $h_A[0,0] + h_A[0,1] = 6$, which can be rewritten as $h_A[0,0] \geq 6 - h_A[0,1]$ and $h_A[0,0] \leq 6 - h_A[0,1]$. Applying the FME technique to these two inequalities yields the vacuous inequality $6 - h_A[0,1] \geq 6 - h_A[0,1]$ that can be dropped. Thus we get:

$$\begin{aligned} & & h_B[0,0] + h_B[0,1] &= 15 \\ h_A[1,0] + h_A[1,1] &= 16 & h_B[1,0] + h_B[1,1] &= 5 \\ 6 - h_A[0,1] + h_A[1,0] &= 17 & h_B[0,0] + h_B[1,0] &= 5 \\ h_A[0,1] + h_A[1,1] &= 5 & h_B[0,1] + h_B[1,1] &= 15 \end{aligned}$$

$$6 - h_A[0,1] \geq 0 \quad h_A[0,1] \geq 0 \quad h_A[1,0] \geq 0 \quad h_A[1,1] \geq 0$$
$$h_B[0,0] \geq 0 \quad h_B[0,1] \geq 0 \quad h_B[1,0] \geq 0 \quad h_B[1,1] \geq 0$$

$$h[0,0] = 6 - h_A[0,1] + h_B[0,0] \qquad h[0,1] = h_A[0,1] + h_B[0,1]$$
$$h[1,0] = h_A[1,0] + h_B[1,0] \qquad h[1,1] = h_A[1,1] + h_B[1,1]$$

We can apply the same process to $h_A[1,0]$ to obtain:

$$\begin{aligned} & & h_B[0,0] + h_B[0,1] &= 15 \\ & & h_B[1,0] + h_B[1,1] &= 5 \\ 6 - h_A[0,1] + 16 - h_A[1,1] &= 17 & h_B[0,0] + h_B[1,0] &= 5 \\ h_A[0,1] + h_A[1,1] &= 5 & h_B[0,1] + h_B[1,1] &= 15 \end{aligned}$$

$$6 - h_A[0,1] \geq 0 \quad h_A[0,1] \geq 0 \quad 16 - h_A[1,1] \geq 0 \quad h_A[1,1] \geq 0$$
$$h_B[0,0] \geq 0 \quad h_B[0,1] \geq 0 \quad h_B[1,0] \geq 0 \quad h_B[1,1] \geq 0$$

$$h[0,0] = 6 - h_A[0,1] + h_B[0,0] \qquad h[0,1] = h_A[0,1] + h_B[0,1]$$
$$h[1,0] = 16 - h_A[1,1] + h_B[1,0] \qquad h[1,1] = h_A[1,1] + h_B[1,1]$$

We can repeat the same procedures for $h_B[0,0]$ and then $h_B[1,1]$ to obtain:

$$6 - h_A[0,1] + 16 - h_A[1,1] = 17 \qquad 15 - h_B[0,1] + 5 - h_B[1,1] = 5$$
$$h_A[0,1] + h_A[1,1] = 5 \qquad h_B[0,1] + h_B[1,1] = 15$$

$$6 \geq h_A[0,1] \geq 0 \quad 16 \geq h_A[1,1] \geq 0$$
$$15 \geq h_B[0,1] \geq 0 \quad 5 \geq h_B[1,1] \geq 0$$

$$h[0,0] = 6 - h_A[0,1] + 15 - h_B[0,1] \qquad h[0,1] = h_A[0,1] + h_B[0,1]$$
$$h[1,0] = 16 - h_A[1,1] + 5 - h_B[1,1] \qquad h[1,1] = h_A[1,1] + h_B[1,1]$$

In these new set of equations, the first line is equivalent to the second, so we can drop it to get

$$h_A[0,1] + h_A[1,1] = 5 \qquad h_B[0,1] + h_B[1,1] = 15$$

$$6 \geq h_A[0,1] \geq 0 \quad 16 \geq h_A[1,1] \geq 0$$
$$15 \geq h_B[0,1] \geq 0 \quad 5 \geq h_B[1,1] \geq 0$$

$$h[0,0] = 6 - h_A[0,1] + 15 - h_B[0,1] \qquad h[0,1] = h_A[0,1] + h_B[0,1]$$
$$h[1,0] = 16 - h_A[1,1] + 5 - h_B[1,1] \qquad h[1,1] = h_A[1,1] + h_B[1,1]$$

We now eliminate $h_A[0,1]$ using the same procedures as before (a Gaussian elimination to remove it from all but one equation, followed by FME) and then do the same for $h_B[0,1]$.

$$6 \geq 5 - h_A[1,1] \geq 0 \quad 16 \geq h_A[1,1] \geq 0$$
$$15 \geq 15 - h_B[1,1] \geq 0 \quad 5 \geq h_B[1,1] \geq 0$$

$$h[0,0] = 6 - (5 - h_A[1,1]) + 15 - (15 - h_B[1,1])$$
$$h[0,1] = 5 - h_A[1,1] + 15 - h_B[1,1]$$
$$h[1,0] = 16 - h_A[1,1] + 5 - h_B[1,1]$$
$$h[1,1] = h_A[1,1] + h_B[1,1]$$

Removing redundant inequalities (and simplifying the equalities), we get

$$5 \geq h_A[1,1] \geq 0$$
$$5 \geq h_B[1,1] \geq 0$$
$$h[0,0] = 1 + h_A[1,1] + h_B[1,1])$$
$$h[0,1] = 20 - (h_A[1,1] + h_B[1,1])$$
$$h[1,0] = 21 - (h_A[1,1] + h_B[1,1])$$
$$h[1,1] = h_A[1,1] + h_B[1,1]$$

We use the last equation with Gaussian elimination to get

$$5 \geq h_A[1,1] \geq 0$$
$$5 \geq h_B[1,1] \geq 0$$
$$h[0,0] = 1 + h[1,1])$$
$$h[0,1] = 20 - h[1,1]$$
$$h[1,0] = 21 - h[1,1]$$
$$h[1,1] = h_A[1,1] + h_B[1,1]$$

Rewriting the last equation:

$$5 \geq h_A[1,1] \geq 0$$
$$5 \geq h_B[1,1] \geq 0$$
$$h[0,0] = 1 + h[1,1])$$
$$h[0,1] = 20 - h[1,1]$$
$$h[1,0] = 21 - h[1,1]$$
$$h[1,1] \geq h_A[1,1] + h_B[1,1]$$
$$h[1,1] \leq h_A[1,1] + h_B[1,1]$$

Then applying FME to $h_A[1,1]$ and then $h_B[1,1]$, we get

$$h[0,0] = 1 + h[1,1])$$
$$h[0,1] = 20 - h[1,1]$$
$$h[1,0] = 21 - h[1,1]$$
$$10 \geq h[1,1] \geq 0$$

E. PROOF OF LEMMA 1

Proof. Note that since external knowledge is assumed to be self-consistent, we must have $r_\ell[0] + r_\ell[1] = c_\ell[0] + c_\ell[1]$ as both sides of the equation represent the total population in region γ_ℓ.

The proof using FME is long and tedious, so instead we opt for a simpler proof that shows why these are the correct constraints. Note that these constraints are redundant (any one of the equality constraints can be dropped).

First note $h_\ell[1,1] \leq \min(r_\ell[1], c_\ell[1])$ (because the count in that cell is at most the count in that row or that column). For the same reason, $h_\ell[0,1] \leq \min(r_\ell[0], c_\ell[1])$. Since $h_\ell[1,1] + h_\ell[0,1] = c_\ell[1]$, we must have $h_\ell[1,1] \geq c_\ell[1] - \min(r_\ell[0], c_\ell[1])$ (in particular, this means that $c_\ell[1] - \min(r_\ell[0], c_\ell[1]) \leq h_\ell[1,1] \leq \min(r_\ell[1], c_\ell[1])$). Since $h[1,1] = \sum_\ell h_\ell[1,1]$, an upper (resp lower) bound on $h[1,1]$ can be obtained by summing up the upper (resp lower) bounds on the $h_\ell[1,1]$. Hence the proposed constraints are necessary.

To show that the proposed constraints are sufficient, we construct integer histograms h_ℓ (consistent with external knowledge) from a nonnegative integer histogram h that satisfies the claimed constraints. First we set values for $h_\ell[1,1]$ for all ℓ. This can be done with the following iterative procedure: First, note that

```
1  function Allocate():
       // Set values for $h_\ell[1,1]$
2      leftover ← $h[1,1]$ for $\ell = 1, \ldots, m$ do
3          amount ← $c_\ell[1] - \min(r_\ell[0], c_\ell[1])$
4          $h_\ell[1,1]$ ← amount
5          leftover ← leftover - amount
6      end
7      $\ell \leftarrow 1$
8      while leftover > 0 do
9          increment ← $\min(r_\ell[1], c_\ell[1]) - h_\ell[1,1]$
             $h_\ell[1,1] \leftarrow h_\ell[1,1] + \min(\text{increment},\text{leftover})$
10         leftover ← leftover - min(increment,leftover)
11     end
       // Set values for all other entries of $h_\ell$
12     for $\ell = 1, \ldots, m$ do
13         $h_\ell[0,1] = c_\ell[1] - h_\ell[1,1]$
14         $h_\ell[1,0] = r_\ell[1] - h_\ell[1,1]$
15         $h_\ell[0,0] = c_\ell[0] - h_\ell[1,0]$
16     end
```

this procedure constructs the leaf histograms h_ℓ that have integer entries. We need to prove the entries are nonnegative, satisfy the appropriate row and column constraints, and that they add up to h.

It is easy to see that right before line 12, $\sum_\ell h_\ell[1,1] = h[1,1]$. Furthermore, after line 7, it maintains the invariant that $0 \leq c_\ell[1] - \min(r_\ell[0], c_\ell[1]) \leq h_\ell[1,1] \leq \min(r_\ell[1], c_\ell[1])$ and after the algorithm finishes, it is easy to see that $\sum_\ell h_\ell[i,j] = h[i,j]$ for $i = 0,1$ and $j = 0,1$ and they satisfy the row and column sum equality constraints from external knowledge. Thus we just need to show that the $h_\ell[i,j]$ are nonnegative.

Since the algorithm, after line 7, maintains the invariant that $h_\ell[1,1] \leq \min(r_\ell[1], c_\ell[1])$, then $h_\ell[0,1]$ and $h_\ell[1,0]$ are nonnegative by construction. Finally, we use the lines 14 and 15 to show:

$$\begin{aligned}
h_\ell[0,0] &= c_\ell[0] - h_\ell[1,0] \\
&= c_\ell[0] - (r_\ell[1] - h_\ell[1,1]) \\
&= h_\ell[1,1] + c_\ell[0] - r_\ell[1] \\
&\geq c_\ell[1] - \min(r_\ell[0], c_\ell[1]) + c_\ell[0] - r_\ell[1] \\
&= r_\ell[0] + r_\ell[1] - \min(r_\ell[0], c_\ell[1]) - r_\ell[1] \\
&\text{(since } r_\ell[0] + r_\ell[1] = c_\ell[0] + c_\ell[1] \text{ by self-consistency} \\
&\text{of external knowledge)} \\
&= r_\ell[0] - \min(r_\ell[0], c_\ell[1]) \\
&\geq 0
\end{aligned}$$

□

F. PROOF OF THEOREM 3

PROOF. We first consider the case where the $c_{j,i}$ are all 0 and then consider the general case. Construct a network with a source node s and sink node t. For $j = 1, \dots, k$ create nodes labeled X_j and place a directed edge from s to X_j with capacity x_j. For $i = 1, \dots, m$ create a node labeled Y_i and place a directed edge from Y_i to t with edge weight n_i. For each pair X_j and Y_i, add a directed edge from X_j to Y_i if the constraint involving is $x_{i,j} \geq 0$ (i.e., it is not equal to 0). This is the same construction as used in Figure 7.2.

The flow incoming to X_j will be interpreted as the total number of people in demographic bucket j. The capacity constraint limits it to be at most x_j and we want the maximum flow to have those edges equal to their capacities.

The flow coming out from Y_i to the sink T is interpreted as the total number of people in region γ_i. The capacity constraint limits it to be at most n_i and we want the maximum flow to have those edges equal to their capacities.

The flow between X_j and Y_i is interpreted as the number of people from bucket j who belong to region γ_i. The conservation of flow means that the total flow on edge coming into Y_i must be at most n_i (since that is a bound on the outgoing flow from Y_i). Hence, if we find a maximum integer flow in which the edges coming out of the sink are saturated (equal to their capacity) and the edges coming into the source are saturated, and the flow values are all integers, then setting $x_{j,i}$ equal to the flow from X_j to Y_i is the desired extension of $\widetilde{H}^0$.

Since we want the maximum flow to saturate the edges coming out of S and going into T, we need the constraint

$$\sum_j x_j = \sum_i n_i \tag{34}$$

and we need the maximum flow to equal $\sum_i n_i$. By the max-flow/min-cut theorem, we want the minimum cut S, T to be at least $\sum_i n_i$ (the cut where $S = \{s\}$ and T contains everything else already equals $\sum_j x_j = \sum_i n_i$). We can avoid considering cuts where an edge with infinite capacity leaves S. This gives us the rules:

1. If S contains a node X_j then it must also contain all Y_i for which $x_{j,i} \geq 0$.
2. If S contains all the Y_i that an X_j that points to, then we can reduce the cost of the cut by putting that X_j in S (so that the capacity on the edge $s \to X_j$ does not add to the cost).

Thus the cuts we need to consider are of the form $\{s\} \cup A \cup B$ where A is a subset of the X_j and B contains all the Y_i what are pointed to by nodes in A. The cost of this cut is the sum of capacities on edges from s to the X_j that are not in A plus the sum of capacities on edges from B to the sink t and we must require it to be at least $\sum_i n_i$ (the minimum cut cost we want). Item 2 also suggests that if two regions Y_{i_1} and Y_{i_2} have exactly two same incoming neighbors, then we can merge those two nodes (and the capacity of the outgoing edge from the merged node to t is the $n_{i_1} + n_{i_2}$ (thus we only need to deal with equivalence classes of regions that have the same set of incoming neighbors, and this makes the network smaller).

Thus, a complete set of implied constraints is:

$$\sum_j x_j = \sum_i n_i$$

$$\text{for all } A \subset \{1, \dots, k\} : \sum_i n_i - \sum_{j \in A} x_j + \sum_{i \in \text{neigh}(A)} n_i \geq \sum_i n_i$$

where neigh(A) is the set of i such that there is an edge from some $X_j \in A$ to Y_i which is the set of regions i for which $x_{j,i}$ is not forced to be 0. After re-arranging, we get

$$\sum_j x_j = \sum_i n_i$$

$$\text{for all } A \subset \{1, \dots, k\} : \sum_{j \in A} x_j \leq \sum_{i \in \text{neigh}(A)} n_i$$

Now, for the general case where the $c_{j,i}$ are not necessarily 0, we note that we can reduce this to the above problem by removing $c_{j,i}$ people from each demographic bucket x_j in location γ_i so that we can work with variables $x'_j = x_j - \sum_i c_{j,i}$ and constants $n'_i = \sum_j c_{j,i}$ and so we also need the conditions that $x_j \geq \sum_i c_{j,i}$ and $n_i \geq \sum_j c_{j,i}$. □

G. PROOF OF THEOREM 4

PROOF. **Initial placement:** Note that the original data provides a feasible solution to the constraints, so the optimization problem will produce a fractional histogram. To create an extension to the leaves, for each leaf, we put in the minimum number of people in each GQ that it has. The leaf histograms under construction now satisfy all the GQ constraints and we need to allocate the rest of the people to satisfy the known population constraints. Let ϕ_ℓ be the number of people assigned to leaf ℓ in this phase and let n_ℓ be its known population (note $n_\ell \geq \phi_\ell$).

Excess flow: After accounting for this initial placement of people, each flow F_{ij} from a GQ of type i to the set of leaves with GQ combination Y_j is reduced by f_{ij}. The network flow constraints ensure that $F_{ij} - f_{ij} \geq 0$ (call $F_{ij} - f_{ij}$ the excess flow from i to j). For each i, this leaves $G_i - qq_i$ fractional people in group quarters type i to redistribute among the leaves (since $gq_i = \sum_j f_{ij}$). Arbitrarily partition the fractional people in cells belonging to X_i (cells of H^* that correspond to people in GQ of type i) so that $F_{ij} - f_{ij}$ of them are set aside for the set of leaves with GQ combination Y_j.

Now fractional people in the excess flow are assigned to each set of leaves having the same GQ combination. For each GQ combination Y_j, the total number of people assigned to leaves with that combination (based on the excess flow and the initial placement) matches the total population of those leaves, by the network flow constraints. Thus there are M_j people from the excess flow assigned to leaf set corresponding to Y_j and each leaf ℓ already has ϕ_ℓ people assigned to it. To each leaf we can assign arbitrary people as long as they come from a GQ belonging to combination Y_j and the population limit of leaf ℓ is not exceeded. Noting that $M_j = \sum n_\ell - \phi_\ell$ (where the summation is over the leaves associated with GQ combinations Y_j), we can randomly assign $\sum n_\ell - \phi_\ell$ of the excess flow to leaf ℓ, thus letting it reach its population total. □

H. PROOF OF THEOREM 5

PROOF. Note that the fractional histogram H^* produced by the L_2 solve provides a feasible point that satisfies the constraints in the L_1 problem. Thus we need to show that the problem is totally unimodular (so that the histogram $H^\dagger$ and its flow variables are all integer valued in the solution). After that, the same allocation arguments as in Theorem 4 apply to show that this integer histogram can be extended to integer leaf histograms. Since an optimal solution to a constrained L_1 minimization lies at one of the corner points of the simplex of feasible values, it is enough to make sure that all of the corner points are integers. We do this by showing that any square sub-matrix of the constraint matrix has a determinant of $1, 0$ or -1.

We will make heavy use of the following fact. Suppose a matrix A' has a row or a column that has only one nonzero entry, and that entry is ± 1 and occurs at coordinates (i, j). Then A' has determinant in $\{-1, 0, 1\}$ if and only if the matrix B, formed by removing row i and column j has determinant in $\{-1, 0, 1\}$.

Applying this fact means that we only need to consider square sub-matrices formed from some of the following constraints:

$$\begin{aligned}
&\sum_x H^\dagger[x] = \sum_x H^*[x] \text{ (total sum constraint)}\\
&GQ_j = \sum_{i \in X_j} H^\dagger[i] \quad \text{for } j = 1, \dots, n_g\\
&GQ_i = \sum_{j\,:\,i \in Y_j} F_{ij} \quad \text{for } i = 1, \dots, n_g\\
&CB_j = \sum_{i\,:\,i \in Y_j} F_{ij} \quad \text{for } j = 1, \dots, n_c
\end{aligned}$$

Applying this fact again to the CB_j, we can drop constraints of the last type above so that we only need to consider square submatrices formed from:

$$\begin{aligned}
&\sum_x H^\dagger[x] = \sum_x H^*[x] \text{ (total sum constraint)}\\
&GQ_j = \sum_{i \in X_j} H^\dagger[i] \quad \text{for } j = 1, \dots, n_g\\
&GQ_i = \sum_{j\,:\,i \in Y_j} F_{ij} \quad \text{for } i = 1, \dots, n_g
\end{aligned}$$

Applying this fact recursively again, we can drop constraints of the third type above (because of the F_{ij} variables, each such variable appears in just one equation) and then the second type above (because now the fact would apply to the GQ_j variables).Now we are down to one constraint with coeffeicients being $-1, 0,$ or 1 then any square submatrix has determinant $-1, 0, 1$ and so we are done. □

I. PROOF OF THEOREM 6

PROOF. This proof is virtually the same as the proof of Theorem 4, applied to each child. □

J. PROOF OF THEOREM 7

PROOF. This proof is virtually the same as the proof of Theorem 5. □

K. PROOF OF THEOREM 8

PROOF. Clearly $C_p^\dagger$ is a necessary set of implied constraints. To show that it is sufficient, suppose $\widetilde{H}_p$ satisfies those constraints. Since $C_p \subseteq C_p^\dagger$ then $\widetilde{H}_p$ is extendable to histograms $\widetilde{H}_{\tau_1}, \dots, \widetilde{H}_{\tau_k}$ that satisfy $C_{\tau_1}, \dots, C_{\tau_k}$, respectively. Since $\widetilde{H}_{\tau_1}, \dots, \widetilde{H}_{\tau_k}$ is essentially equivalent to extending $\widetilde{H}_p$ with a location attribute, it is clear that for cells in $\widetilde{H}_p$ that are 0, the corresponding cells in each of the $\widetilde{H}_{\tau_i}$ is also 0, which means that $C_p^\dagger$ is then sufficient as well. □

Note that if the implied constraints C_p cause the TopDown algorithm to solve TUM problems, then these new implied constraints $C_p^\dagger$ will also cause the TopDown algorithm to solve TUM problems as these constraints add equations with only 1 variable in each row.

L. INVARIANTS AND STRUCTURAL ZEROS

The Pl94-171 dataset is too large to create in memory

Invariants for PL94-171:

1. Number of housing units in each block is invariant. Equivalent to upper bounds on number of householders.
2. Number of occupied group quarters by major type (7 levels) is invariant. Equivalent to lower bounds on number of people with GQ attributes in persons table
3. Population of each state is invariant

Invariants for DHC:

1. Number of housing units in each block is invariant. Equivalent to upper bounds on number of householders.
2. Number of occupied group quarters by detailed type (29 levels) by single-sex status (3 levels) is invariant. Equivalent to lower bounds on number of people with certain sex and GQ attribute combinations in persons table
3. Population of each state is invariant

structural zeros

1. Number of householders $\geq$ number of spouses and unmarried partners
2. Number of householders $\leq$ 2 times number of parents of householder
3. Number of householders $\leq$ 4 times number of grandparents of householder
4. children of householder must have certain age gap with householder
5. Householder age must be at least 15
6. Parents of householder must have a certain age gap with householder
7. grandparents of householder must have a certain age gap with householder
8. Some GQ are single sex or co-ed
9. Some GQ have upper and lower bounds on age.

Each invariant specifies a collection of cells (the sum of counts in those cells is given the upper/lower/equality bound in the invariant).

Structural zeros also specify a collection of cells (each cell is 0, or equivalently the sum of counts in those cells is 0).

The main mathematical distinction between structural zeros and invariants is that (a) most importantly, structural zeros are independent of geography while the bounds in the invariants depend on geography and (b) less importantly structural zeros are equality constraints with 0 (so there is no leeway in setting values in cells specified by structural zeros). This is the source of difficulty.

Given that we have a table T_1 that we want to extend to table T_2 by adding columns, what are the implied constraints on T_1.

Some constraints on T_2 are irrelevant when creating T_1.