2017

Data Privacy Beyond Differential Privacy

Zhiwei Steven Wu
University of Pennsylvania, wuzhiwei@cis.upenn.edu

Data Privacy Beyond Differential Privacy

Abstract
Computing technologies today have made it much easier to gather personal data, ranging from GPS locations to medical records, from online behavior to social exchanges. As algorithms are constantly analyzing such detailed personal information for a wide range of computations, data privacy emerges as a paramount concern. As a strong, meaningful and rigorous notion of privacy, Differential

Privacy has provided a powerful framework for designing data analysis algorithms with provable privacy guarantees. Over the past decade, there has been tremendous progress in the theory and algorithms for differential privacy, most of which consider the setting of centralized computation where a single, static database is subject to many data analyses. However, this standard framework does not capture many complex issues in modern computation. For example, the data might be distributed across self-interested agents, who may have incentive to misreport their data; and different individuals in the computation may have different expectations to privacy.

The goal of this dissertation is to bring the rich theory of differential privacy to several computational problems in practice. We start by studying the problem of private counting query release for high-dimensional data, for which there are well-known computational hardness results. Despite the worst-case intractability barrier, we provide a solution with practical empirical performances by leveraging powerful optimization heuristics. Then we tackle problems within different social and economic settings, where the standard notion of differential privacy is not applicable. To that end, we use the perspective of differential privacy to design algorithms with meaningful privacy guarantees.

(1) We provide privacy-preserving algorithms for solving a family of economic optimization problems under a strong relaxation of the standard definition of differential privacy---joint differential privacy.

(2) We also show that (joint) differential privacy can serve as a novel tool for mechanism design when solving these optimization problems: Under our private mechanisms, the agents are incentivized to behave truthfully.

(3) Finally, we consider the problem of using social network metadata to guide a search for some class of targeted individuals (for whom we cannot provide any meaningful privacy guarantees). We give a new variant of differential privacy---protected differential privacy---that guarantees differential privacy only for a subgroup of protected individuals. Under this privacy notion, we provide a family of algorithms for searching targeted individuals in the network while ensuring the privacy for the protected (un-targeted) ones.

Degree Type
Dissertation

Degree Name
Doctor of Philosophy (PhD)

Graduate Group
Computer and Information Science

DATA PRIVACY BEYOND DIFFERENTIAL PRIVACY

Zhiwei Steven Wu

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

2017

Co-Supervisor of Dissertation

Michael Kearns, Professor and National Center Chair, University of Pennsylvania

Co-Supervisor of Dissertation

Aaron Roth, Associate Professor, University of Pennsylvania

Graduate Group Chairperson

Lyle Ungar, Professor, University of Pennsylvania

Dissertation Committee

Sampath Kannan, Henry Salvatori Professor, University of Pennsylvania

Alexander Rakhlin, Associate Professor, University of Pennsylvania

Tim Roughgarden, Professor, Stanford University

Rakesh Vohra, George A. Weiss and Lydia Bravo Weiss University Professor, University of Pennsylvania

DATA PRIVACY BEYOND DIFFERENTIAL PRIVACY

Dedicated to my parents.

ACKNOWLEDGEMENT

First and foremost, I must begin by thanking my advisors Michael Kearns and Aaron Roth. Without their guidance and support, the work contained in this dissertation wouldn't have been possible and my journey through graduate school wouldn't have been such an enjoyable ride. Michael and Aaron are the *dream team* of advisors that I could ever wish for. Their knowledge, genius, and eloquence have been my perpetual source of inspiration. Their keen intuition for important problems and elegant solutions have helped me shape my own research style. Their constant challenge and encouragement have given me the confidence to be an independent researcher. I will always be proud to be one of the first students co-advised by Michael and Aaron.

I am also deeply grateful to have many other mentors during graduate school. Thanks to Nicole Immorlica, Bobby Kleinberg, Brendan Lucier, Yishay Mansour, Alex Slivkins, and Vasilis Syrgkanis, I had the opportunity to explore other topics outside of my dissertation research during my internships at Microsoft Research. I especially want to thank Alex for his timely advices on different aspects of my career. I also want to thank Jonathan Ullman for teaching me the importance of clarity in thinking, writing, and speaking.

Other than my mentors, I am also indebted to my numerous terrific collaborators, some of whom have also become my closest friends: Rachel Cummings, Marco Gaboradi, Emilio Jesús Gallego Arias, Paul Goldberg, Justin Hsu, Zhiyi Huang, Shahin Jabarri, Sampath Kannan, Katrina Ligett, Francisco Javier Marmolejo Cossío, Jamie Morgenstern, Seth Neel, Kobbi Nissim, Mallesh Pai, Jaikumar Radhakrishnan, Ryan Rogers, Tim Roughgarden, Sam Taggart, Rakesh Vohra, Bo Waggoner, Grigory Yaroslavtsev, and Juba Ziani.

Many thanks to my parents– Wang Wu and Xiaozhen Qin – for all of their unconditional love and support. Finally, to my dearest girlfriend Veronica for her enormous support throughout graduate school. She is clearly my most important finding in this wonderful journey.

ABSTRACT

DATA PRIVACY BEYOND DIFFERENTIAL PRIVACY

Zhiwei Steven Wu

Michael Kearns and Aaron Roth

Computing technologies today have made it much easier to gather personal data, ranging from GPS locations to medical records, from online behavior to social exchanges. As algorithms are constantly analyzing such detailed personal information for a wide range of computations, data privacy emerges as a paramount concern. As a strong, meaningful and rigourous notion of privacy, *Differential Privacy* has provided a powerful framework for designing data analysis algorithms with provable privacy guarantees. Over the past decade, there has been tremendous progress in the theory and algorithms for differential privacy, most of which consider the setting of centralized computation where a single, static database is subject to many data analyses. However, this standard framework does not capture many complex issues in modern computation. For example, the data might be distributed across self-interested agents, who may have incentive to misreport their data; and different individuals in the computation may have different expectations to privacy.

The goal of this dissertation is to bring the rich theory of differential privacy to several computational problems in practice. We start by studying the problem of private counting query release for high-dimensional data, for which there are well-known computational hardness results. Despite the worst-case intractability barrier, we provide a solution with practical empirical performances by leveraging powerful optimization heuristics. Then we tackle problems within different social and economic settings, where the standard notion of differential privacy is not applicable. To that end, we use the perspective of differential privacy to design algorithms with meaningful privacy guarantees.

- We provide privacy-preserving algorithms for solving a family of economic opti-

mization problems under a strong relaxation of the standard definition of differential privacy—*joint differential privacy*.

- We also show that (joint) differential privacy can serve as a novel tool for mechanism design when solving these optimization problems: Under our private mechanisms, the agents are incentivized to behave truthfully.

- Finally, we consider the problem of using social network metadata to guide a search for some class of targeted individuals (for whom we cannot provide any meaningful privacy guarantees). We give a new variant of differential privacy—*protected differential privacy*—that guarantees differential privacy only for a subgroup of protected individuals. Under this privacy notion, we provide a family of algorithms for searching targeted individuals in the network while ensuring the privacy for the protected (un-targeted) ones.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

List of Algorithms

CHAPTER 1

Introduction

People are producing more information than ever. Algorithms today have access to an superabundance of fine-grained personal data, including everything from medical records to GPS locations, from online behaviors to social exchanges. By analyzing such a wide range of information, powerful algorithms have been making tremendous impact on many domains as diverse as academic research, policy making and electronic commerce. However, in reality many of the most informative data sets happen to contain the most sensitive personal information. As a result, the tension between the *privacy* of the individuals and the *usefulness* of the analysis on their data is inevitable: on the one hand, revealing the individuals' private information (directly or indirectly) raises moral and legal concerns; but on the other hand, analyzing data in aggregate can provide insights that largely benefit the society.

This thesis studies on the problem of *privacy-preserving data analysis* that focuses on the following question: *How can we perform useful analysis on sensitive data while preserving the privacy of the individuals?* A key step in tackling this problem is to first understand what it means to protect the individuals' privacy in a computation. One tempting privacy measure is the so-called "data anoymization" or "de-identification," where the basic idea is to "de-identify" the dataset by removing personally identifiable information (for example, name, gender, and age), and then publish the "anonymous" dataset. Ironically, these de-identification mechanisms are not resilient to the so-called "re-identification attacks." One notable example is the attack on Netflix challenge dataset. During the Netflix challenge machine learning contest, the company released its users' movie rating dataset with all of the user names removed, so researchers could study the datasets in order to design new learning algorithms for movie recommendations. While the machine learning contest

itself was a huge success, the allegedly anonymous data sets was far from private. By cross-linking the Internet Movie Database (IMDb), two researchers were able to to de-anonymize most of the users in the dataset [Narayanan and Shmatikov, 2008]. As a result, Netflix was faced with a costly lawsuit and had to cancel all subsequent contests. Other examples of re-identification attacks abound, including the one on the AOL search logs [Barbaro and Zeller, 2006] (see Ohm [2009] for a survey on the failure of anonymization). As Cynthia Dwork put it, "de-identified data isn't," and de-identification is simply not a solution for privacy-preserving data analysis.

Privacy risk arises even when we never publish the data. For example, it may seem innocuous to release aggregate statistics (that are not specific to any individual), but all of the statistics in combination can reveal a great deal about a single individual in the dataset. A striking example is the application of a statistical method, originally developed to determine whether an individual is contributing trace amounts of genomic DNA to a complex forensic mixture, to data from Genomic Wide Association Study (GWA) genetic studies. In particular, the genetic studies contain the allele frequencies of certain case groups for large numbers of single nucleotide polymorphisms (SNPs), which are essentially a large collection of summary statistics that are not about any single individual. Researchers demonstrate that given the genomic data of any single individual they can accurately determine whether the individual participated in the study or not, and can therefore further infer whether the individual has disease associated with the study [Homer et al., 2008]. As a result, the GWA genomic dataset was removed from the public domain. In general, such privacy risk persists whenever too many summary statistics that are overly accurate are released (see Dwork et al. [2017] for a survey of privacy attacks on aggregate data).

Privacy attacks have taught us a valuable lesson: Protecting privacy is a delicate task, and ad-hoc privacy measures have proven to be problematic. Privacy-preserving data analysis calls for a more principled approach that is based on a strong, meaningful, and mathematically rigorous notion of privacy. From the standpoint of algorithm design, the desiderata

for a meaningful privacy notion are three-fold.

- First, it is crucial to work with a stringent and worst-case privacy guarantee. The privacy notion should be a property of the *algorithm*, with no assumption on what the data "looks like" or what knowledge the adversary has. As a result, privacy preservation will be a provable mathematical guarantee.

- Second, it is important that such notion enables effective use of data. One potential disadvantage of working with a stringent privacy notion is that it may rule out all algorithms with non-trivial *utility* guarantee. Thus, a meaningful privacy notion should allow one to effectively acquire useful aggregate information while revealing nothing about any single individual.

- Lastly, a meaningful notion should also facilitate private algorithm design. In particular, it should also come with a collection basic principles and tools for reasoning and designing private algorithms, so that we can build algorithmic solutions for a broad class of data analysis tasks.

As we will see, *Differential Privacy* is such a definition.

1.1. Differential Privacy

The notion of differential privacy was first proposed by the seminal work of Dwork, McSherry, Nissim, and Smith [Dwork et al., 2006]. Informally, differential privacy is a *stability* constraint on algorithms—it requires that no single individual's data in the input data set has a significant influence on the output information. The promise of such stability guarantee is that no data subject will be affected for contributing her private data to any study or data analysis subject to differential privacy. The exact definition of differential privacy formalizes such stability notion in a mathematical and quantifiable way, and provides a privacy measure "epsilon", which allows algorithm designer to reason about privacy loss in the computation.

To understand the strength of this privacy notion, imagine an adversary who is trying to learn about the private information of "Bobby", whose data may or may not be part of the database in the computation. The adversary gets to observe the algorithm's output (as a function of the input database), and may also has access to any arbitrary side information. For example, he might even know the private data of every other individual except Bobby, and quite a bit about Bobby. Differential privacy guarantees that no matter what such side information might be, the adversary will learn *almost nothing new* about Bobby whether or not his data is in the database.

Despite being a stringent privacy notion, differential privacy does not prevent useful data analysis. To illustrate the compatibility between learning and differential privacy, consider a medical study releases that smokers have much higher chances of getting lung cancer. Suppose that our protagonist Bobby happens to be a smoker. After learning his health risk, his insurance company decides to raise his insurance premium. In this case, is Bobby's private information "leaked" from this study? Differential privacy takes the stance that this is not a privacy violation: if the medical study is carried out subject to differential privacy, then the correlation between smoking and lung cancer will still be released whether or not Bobby's data is in the study.

Differential privacy as an algorithmic stability notion is not only a desideratum, but also a useful tool for data analysis. There is a intimate connection between differential privacy and generalization in adaptive analysis (formally established in [Dwork et al., 2015]). One way to interpret the guarantee of differential privacy is the following: if we replace any single data record in the database by a random person from the population, the output (in distributional sense) will be approximately the same. As a result, any statistical claim derived from a differentially private analysis also generalizes to the underlying distribution from which the data is drawn. Therefore, techniques in differential privacy can serve as a tool to prevent over-fitting for adaptive data analysis even when privacy is not a concern! The stability guarantee of differential privacy is also useful when there are

incentives involved in the computation. In a strategic environment, we can interpret the stability guarantee of differential privacy as follows: if any self-interested agent manipulate his or her private data input to the mechanism, the agent will not be able to affect the outcome (e.g. the set of prices imposed on the goods) by much. Consequently, every agent participating in the computation has almost no incentive to misreport the data. In Chapter 6, we will leverage this connection between differential privacy and truthfulness to develop incentive-compatible mechanisms that truthfully elicit private data from self-interested agents for computation.

Differential privacy also admits a powerful algorithmic framework. There are two important features that allow a rich class of differentially private algorithms. The first one is *robustness to post-processing*: any data-independent post-processing procedure on a differentially private output continues to satisfy differential privacy. In other words, if we promise differential privacy in the algorithm's output, no data analyst can incur more privacy loss by running further analysis on the output. The second one is its *compositional property*: the composition of any two private algorithms remains private with the privacy loss parameter increasing gradually. This compositional reasoning greatly facilitates algorithm design—we can divide the computation into different building blocks and design private solutions for these parts in isolation. We will provide more details in Chapter 2.

1.2. Beyond Differential Privacy

Most work in differential privacy literature focuses on the setting of centralized computation: given access to a private static database, the algorithm performs its data analysis and outputs aggregate information (e.g. summary statistics) in privacy-preserving manner. While this is a powerful framework that allows us to develop meaningful private algorithms for optimization, statistical analysis, and machine learning, it has limitations when we wish to solve problems beyond such a centralized setting.

1.2.1. Privacy in Economic Environments

Let us consider the following simple allocation problem that the centralized framework does not capture. There is a collection of goods and a group agents with their private values over the goods. Our goal is to compute a feasible allocation to maximize the social welfare—the sum of the values over the assigned items across all buyers. The private values might be related to treatment of some embarrassing disease or indicative of some business strategy, so we wish to compute the allocation while protecting the agents' private values. However, it is impossible to compute an allocation with high welfare under the standard definition of differential privacy. The intuition is quite simple: if we view the entire assignment as the output by the algorithm, differential privacy requires that the assignment to each agent to be insensitive to the change of his own private values. However, if we want to achieve high-welfare, we must give each agent what he prefers! As a result, there is a direct contradiction between differential privacy and high welfare.

The incompatibility between differential privacy and useful algorithms is common in other domains including driving route suggestion and residency matching assignment. Can we use perspective of differential privacy to provide meaningful private algorithms in these domains despite the obvious incompatibility? To answer this question, let us take the special structure of this type of problems—both the input and the output to the algorithm is naturally partitioned amongst the participants in the problem. For example, in the allocation problem we can view the agents individually submitting their private values to the algorithm, and then the algorithm separately tell each agent what item they get. Taking this distributed view, we can reformulate the goal of privacy protection differently. Since the assignment is no longer publicly visible, we can allow the each agent's assignment to be sensitive to his private information. However, we still need to protect any single agent's private information from the output assignment to all the other agents: fix any agent i, we would like the assignment to all the other agents except i to reveal little information about i's private value.

We can formulate such privacy desideratum as a relaxation of the standard notion of differential privacy—*joint differential privacy* first proposed by Kearns et al. [2014]. Informally, it requires that the output to all the other agents to be insensitive to the change of any single agent's private data. Despite being a relaxation, joint differential privacy is still extremely strong: it implies that for any agent i in the computation, even if all the other agents except i collude and share their information, they would not be able to learn about agent i's private information!

Now that we have a meaningful privacy notion for this large class of economic optimization problems, how do we design algorithms subject to this new privacy constraint? Do we need to "re-invent" all the powerful tools we have for the standard notion of differential privacy, including all the basic algorithms and the composition theorem? Fortunately, we give a remarkable structural result that allows us to "stand on the shoulders of the giants" of (standard) differential privacy. We present a algorithmic framework, termed as the *billboard model*, under which the algorithm publishes signals that satisfies *standard* differential privacy to all the agents, and each agent is able to compute his or her part of the output solution based on the private signal and the agent's private data. The billboard model is in fact very compatible with many distributed computing protocols, including the deferred acceptance algorithm [Kelso and Crawford, 1982], and the dual decomposition optimization algorithm [Boyd et al., 2011]. To design algorithms under the billboard model, we can simply focus on computing a private version of the signal (e.g. prices on the goods), and demonstrate that the resulting solution has good quality despite the noise we introduce in the computation. Based on this powerful framework, we will provide algorithms for a family of economic optimization problems under this strong notion of joint differential privacy in Chapters 4 and 5,

1.2.2. Heterogeneous Privacy Guarantees Across Populations

The standard framework of differential privacy aims to provide privacy guarantee for everyone in the population. However, such universal privacy guarantee may not be appro-

priate if not all members of the population have an equal right to, or demand for, privacy. Consider the problem of using social network metadata to guide a search for some class of *targeted individuals* such as terrorists and organized criminals. Since the goal of the network search is to identify members of the targeted class in the network, we cannot possibly provide any meaningful privacy guarantees to the targeted individuals (and few would argue that actual criminals have the same right to privacy). However, since the network search will eventually encounter members not in the targeted class, we do wish to provide privacy for the private network data of such *protected individuals*.

A natural question arises: *how can we balance the privacy guarantee for the protected individuals while allowing effective search for the targeted?* To address this question, we propose another relaxation of differential privacy—*protected differential privacy* proposed by Kearns et al. [2016]. Under this relaxation, we aim to provide differential privacy guarantee *only* to the protected individuals. However, the reformulation of the privacy guarantee is not the end of the story. The critical technical challenge in this graph search problem is that we don't know who the targeted individuals are a-priori.[1]

More generally, the notion of protected differential privacy is an important step towards a more general computational framework that manages the different privacy guarantees we wish to provide to different sub-groups in the population. This additional degree of freedom is very desirable and opens up more possibility of designing meaningful privacy-preserving algorithms when the underlying population may have different expectations to privacy.

1.3. Outline of Results

In the remainder of this thesis, we will structure the contents as follows.

In Chapter 2, we will provide the basic definition of differential privacy along with the

[1] This is exactly what we are trying to identify!

basic algorithmic tools for designing private algorithms.

In Chapter 3, we will focus on a fundamental problem in privacy-preserving data analysis—private counting query release. For this problem in (standard) differential privacy, our goal is to overcome the *curse of dimensionality* in practice and provide an algorithm with practical empirical run-time performance. The result is based on a joint work with Gaboardi, Gallego Arias, Hsu, and Roth [Gaboardi et al., 2014].

In Chapter 4, we will introduce the constriant of joint differential privacy, a meaningful privacy guarantee for a wide range of problems in economic environments, including distributed optimization and equilibrium computation. In particular, we will focus on solving the allocation problem under the billboard model. This result is based on a joint work with Hsu, Huang, Roth, and Roughgarden [Hsu et al., 2014a].

In Chapter 5, we will provide a general algorithm for solving a family of convex programs under the constraint of joint differential privacy. In particular, the algorithm can be used to solve allocation problems with more a more general class of valuation functions, which gives an improvement over the result in Chapter 4. We will continue to study the same class of optimization problems in Chapter 6, but from the angle of mechanism design. In particular, we will demonstrate how (joint) differential privacy can be used as a tool to obtain incentive-compatible mechanisms. This is based on a joint work with Hsu, Huang, and Roth [Hsu et al., 2016].

Finally, in Chapter 7, we will focus on a graph search problem for which we only seek to provide differential privacy guarantee for a subgroup in the population. In particular, we will provide a family of graph search algorithms under the constraint of *protected differential privacy*. This is based on a joint work with Kearns, Roth, and Yaroslavtsev [Kearns et al., 2016].

CHAPTER 2

Background

I am in the database, but nobody knows.

Cynthia Dwork

2.1. The Definition of Differential Privacy

Differential Privacy is an algorithmic property, specifically for randomized algorithms.[1] Roughly speaking, it requires that the any single individual in the data set has little influence on the information output by the algorithm. More concretely, it is a stability notion on (randomized) algorithms such that the change to any single data record in the dataset does not change the probability of any event (based on the output) by much.

In this chapter, we will mostly focus on the standard setting of centralized computation and formalize the guarantee above. We assume the existence a trusted and trustworthy curator who has access to a sensitive database D that contains the private data records of n individuals. We also assume a data universe $\mathcal{X}$, which is the set of all possible data types (e.g., all possible genotypes). It is useful to think of each data record corresponding to a "row" in the database, and we will write D as an element in the set $\mathcal{X}^n$.[2] A query is a function of the database. We assume a data analyst that asks the data curator a set of queries (possibly adaptively), and the data analyst provide the answers by running some *privacy mechanism* (or simply *private algorithm*) $\mathcal{M}$ with output range R.

To formally introduce differential privacy, it is important to define the *neighboring rela-*

[1] It is a folklore that any non-trivial differentially private algorithm is necessarily randomized.

[2] Sometimes a database is defined as a *multi-set*, that is $D \subseteq \mathcal{X}$, which is useful when the size of the database is not publicly known.

tionship among databases. In particular, two databases $D, D' \subset \mathcal{X}$ are *neighboring* if they differ by at most a single data element, that is their hamming distance $\|D \triangle D'\| \leq 1$. More generally, one can define the neighboring relationship with respect to other metric over the databases, and the privacy guarantee will be adapted accordingly. Differential privacy requires that the algorithm has "close" output distributions on any pair of neighboring databases. Formally:

Definition 2.1.1 ([Dwork et al., 2006])**.** *A mechanism $\mathcal{M}\colon \mathcal{X}^n \to R$ satisfies (ε, δ)-differential privacy if for every $S \subseteq R$ and for all neighboring databases $D, D' \in \mathcal{X}^n$, the following holds:*

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(D') \in S] + \delta$$

If $\delta = 0$ we say M satisfies ε-differential privacy.

The most important parameter in the definition is the privacy loss parameter ε. The smaller the value of ε is, the more the stable the algorithm is, and hence the better privacy guarantee we can provide. To understand the privacy guarantee, let us consider the following quantity: for any pair of databases D and D', any outcome o in the range R, let

$$\mathcal{L}(D, D', o) = \ln\left(\frac{\Pr[\mathcal{M}(D) = o]}{\Pr[\mathcal{M}(D') = o]}\right),$$

which is commonly referred to as the *privacy loss* incurred by observing the outcome o. In particular, ε-differential privacy (with $\delta = 0$) implies that for any pairs of neighboring databases D, D' and outcome o, the absolute value of $\mathcal{L}(D, D', o)$ is bounded by ε. When ε goes to 0, this means D and D' are almost equally likely to produce the outcome o. As a result of such indistinguishability guarantee, the privacy of any single individual is protected even when the adversary is equipped with really strong side information. Suppose that the adversary knows the entire database except for the i-th person's data (that is, he has full knowledge about D_{-i}), and he knows that the i-th record is either a or b, which correspond to a pair of neighboring databases D and D'. Even in this case, the adversary can learn almost nothing more about the individual i by observing the output

by the algorithm.

When δ is non-zero, we are typically interested in values of δ that are less than the inverse of any polynomial in the size of the database n. For example, even if a mechanism randomly outputs a few data records in the dataset, it will still satisfy $(0, \delta)$-differential privacy with δ in the order of $O(1/n)$.

We can also express the guarantee of differential privacy through the notion of *max divergence*. Fix any pair of random variables Y and Z, we will write

$$D_\infty(Y\|Z) = \max_{S \subseteq \text{Supp}(Y)} \left[\ln \frac{\Pr[Y \in S]}{\Pr[Z \in S]} \right]$$

to denote the max divergence between two random variables Y and Z. Also, the δ-approximate max divergence between Y and Z is defined to be

$$D_\infty^\delta(Y\|Z) = \max_{S \subseteq \text{Supp}(Y): \Pr[Y \in S] \geq \delta} \left[\frac{\Pr[Y \in S] - \delta}{\Pr[Y \in S]} \right].$$

Claim 2.1.2. *A mechanism $\mathcal{M}$ is*

1. *ε-differentially private if and only if on every pair of neighboring datasets D and D', $D_\infty(\mathcal{M}(D)\|\mathcal{M}(D')) \leq \varepsilon$ and $D_\infty(\mathcal{M}(D')\|\mathcal{M}(D)) \leq \varepsilon$; and is*

2. *(ε, δ)-differentially private if and only if on every pair of neighboring datasets D and D': $D_\infty^\delta(\mathcal{M}(D)\|\mathcal{M}(D')) \leq \varepsilon$ and $D_\infty^\delta(\mathcal{M}(D')\|\mathcal{M}(D)) \leq \varepsilon$.*

2.2. Properties of Differential Privacy

We will highlight some of the most important properties of differential privacy. The first one is its *resilience to post-processing*. To put it simply, a data analyst cannot incur more privacy loss by "thinking" very hard about the output from a differentially private algorithm. The claim is formalized below, and see [Dwork and Roth, 2014] (Proposition 2.1) for a formal proof.

Lemma 2.2.1 (Post-Processing). *Let $\mathcal{M}\colon \mathcal{X}^* \to R$ be a (ε, δ)-differentially private mechanism. Let $f : R \to R'$ be an arbitrary randomized mapping. Then $f \circ \mathcal{M}\colon \mathcal{X}^* \to R'$ is (ε, δ)-differentially private.*

Next, we will discuss the *compositional property* of differential privacy. Suppose we have several differentially private subroutines at hand, and we wish to build a more sophisticated algorithm by composing these subroutines. How do we perform the privacy analysis for the entire algorithm? The composition theorem of differential privacy says that the "privacy losses add up." We will first provide the following basic composition theorem (see [Dwork and Roth, 2014] (Theorem B.1) for a formal proof).

Theorem 2.2.2 (Basic Composition). *Let $\mathcal{M}_i\colon \mathcal{X}^* \to R_i$ be an $(\varepsilon_i, \delta_i)$-differentially private algorithm for $i \in [k]$. Then if $\mathcal{M}_{[k]}\colon \mathcal{X}^* \to \prod_{i=1}^k R_i$ is defined to be $\mathcal{M}_{[k]}(D) = (\mathcal{M}_1(D), \ldots, \mathcal{M}_k(D))$, then $\mathcal{M}_{[k]}$ is $(\sum_{i=1}^k \varepsilon_i, \sum_{i=1}^k \delta_i)$-differentially private.*

Now we will give a more sophisticated version of the composition theorem. To formally introduce the result, we consider a model in which an adversary can affect the input databases to the future mechanisms, along with the queries to these mechanisms. Let $\mathcal{F}$ be class of differentially private mechanisms. (For instance, the class can just be the set of all ε-differentially private mechanisms given a fixed level of ε.) Fix any adversary A, we consider two experiments, Experiment 0 and Experiment 1, defined as follows.

Experiment b for family $\mathcal{F}$ and adversary A: For each round $i = 1, \ldots, k$:

1. A outputs two neighboring datasets D_i^0 and D_i^1, a mechanism $\mathcal{M}_i \in \mathcal{F}$, and parameters w_i.

2. A receives $y_i \in_R \mathcal{M}_i(w_i, D_i^b)$.

In this experiment, the adversary is allowed to be adaptive, and thus is may choose the pairs of databases D_i^0, D_i^1, mechanisms, and the parameters based on the outputs of previous mechanisms. The *view* V^b of the adversary A in the experiment b includes A's internal

randomness and all of the mechanism outputs $(y_1, \ldots, y_k)$.[3]

How does this experiment relate to the composition of differential privacy? Consider an adversary who always choose D_i^0 to hold Bob's data and D_i^1 to differ only in that Bob's data is replaced by Alice's data. Then experiment 0 can be thought of as the "real world," where Bob's data is repeatedly used in the computation, and the experiment 1 can be thought of as the "parallel world," where Bob's data is never used. The fact that the mechanisms in class $\mathcal{F}$ satisfies differential privacy guarantees that the two experiments need to be "close" to each other. We will now define the notion of *k-fold adaptive composition*, which will be useful for describing the advanced composition theorem.

Definition 2.2.3. *We say that the family $\mathcal{F}$ of mechanisms satisfies ε-differential privacy under k-fold adaptive composition if for every adversary A, we have $D_\infty(V^0 \| V^1) \leq \varepsilon$ where V^b denotes the view of A in the experiment b defined above.*

Similarly, $\mathcal{F}$ satisfies (ε, δ)-differential privacy under k-fold adaptive composition if for every adversary A, we have $D_\infty^\delta(V^0 \| V^1) \leq \varepsilon$

Theorem 2.2.4 (Advanced Composition[Dwork et al., 2010b])**.** *Let $\varepsilon, \delta, \delta' > 0$, the class of (ε, δ)-differentially private mechanisms satisfies $(\varepsilon', k\delta + \delta')$-differential privacy under k-fold adaptive composition for:*

$$\varepsilon' = \sqrt{2k \ln(1/\delta')} + k\varepsilon(e^\varepsilon - 1).$$

We will often use the following corollary, which demonstrate that the privacy loss in an adaptive composition of k ε-differentially private mechanisms grows roughly as $\sqrt{k}\varepsilon$.

Corollary 2.2.5. *Given target privacy parameters $\varepsilon' \in (0, 1)$ and $\delta' > 0$, to ensure $(\varepsilon', k\delta + \delta')$-differential privacy over k mechanisms, it suffices that each mechanism is (ε, δ)-differentially private, where*

$$\varepsilon = \frac{\varepsilon'}{2\sqrt{2k \ln(1/\delta')}}.$$

[3] Note that the datasets D_i^b's, $\mathcal{M}_i$'s, and parameters w_i's can all be determined from $(y_1, \ldots, y_k)$.

2.3. Basic Tools

The most basic differentially private mechanism is the Laplace mechanism, which allows us to release noisy answer for any real-valued query based on its ℓ_1 sensitivity. The Laplace Distribution centered at 0 with scale b is the distribution with probability density function

$$\mathrm{Lap}(z|b) = \frac{1}{2b}\exp\left(-\frac{|z|}{b}\right).$$

We say $X \sim \mathrm{Lap}(b)$ when X has Laplace distribution with scale b. Let $f\colon \mathcal{X}^* \to \mathbb{R}^d$ be an arbitrary d-dimensional function. The ℓ_1 *sensitivity* of f is defined to be

$$\Delta_1(f) = \max_{D\sim D'} \|f(D) - f(D')\|_1.$$

The *Laplace mechanism* with parameter ε simply adds noise drawn independently from $\mathrm{Lap}\left(\frac{\Delta_1(f)}{\varepsilon}\right)$ to each coordinate of $f(x)$.

Theorem 2.3.1 (Laplace Mechanism [Dwork et al., 2006])**.** *Let $\varepsilon > 0$. The Laplace mechanism is ε-differentially private.*

A similar basic tool is the *Gaussian Mechanism*, which releases private perturbations of vector valued functions $f : \mathcal{X}^n \to \mathbb{R}^k$ by adding Gaussian noise with scale proportional to the ℓ_2 sensitivity of the function f:

$$\Delta_2(f) = \max_{\text{neighboring } D,D'\in\mathcal{X}^n} \|f(D) - f(D')\|_2.$$

Theorem 2.3.2 (Gaussian Mechanism (see e.g. Dwork and Roth [2014] for a proof))**.** *Let $\varepsilon, \delta \in (0,1)$. Let c be a number such that $c^2 \geq 2\log(1.25/\delta)$, then the Gaussian Mechanism (which outputs $f(D)+Z$ where $Z \sim \mathcal{N}(0,\sigma^2)^k$ with $\sigma \geq c\Delta_2(f)/\varepsilon$) is (ε,δ)-differentially private.*

Another basic tool is called the *exponential mechanism*, which allows us to privately solve

the following selection problem: given a set of alternatives, select one that is the "best" fit for the sensitive dataset. Here how good an alternative is for the dataset is measured by a quality score function. More formally:

Theorem 2.3.3 (Exponential Mechanism[McSherry and Talwar, 2007]). *Given some arbitrary output range R, the* exponential mechanism *with score function S selects and outputs an element $r \in R$ with probability proportional to*

$$\exp\left(\frac{\varepsilon S(D,r)}{2\Delta}\right),$$

where Δ is the sensitivity *of S, defined as*

$$\Delta = \max_{D,D':|D\triangle D'|=1, r\in R} |S(D,r) - S(D',r)|.$$

The exponential mechanism is ε-differentially private.

The sparse vector mechanism from differential privacy takes a numeric threshold and a sequence of (possibly adaptively chosen) queries. Sparse vector outputs $\perp$ while the current query has answer substantially less than the threshold, and outputs $\top$ and halts when the query has answer near or exceeding the threshold. The code is in Algorithm 1.

Algorithm 1 Sparse vector mechanism $\mathsf{Sparse}(\varepsilon, T)$

Input: Privacy parameter $\varepsilon > 0$, threshold T, and stream of Δ-sensitive queries $q_1, q_2, \ldots$.

Initialize: $\hat{T} := T + \mathrm{Lap}\left(\frac{2\Delta}{\epsilon}\right)$.
for each query q_i:
  Let $y := q_i + \mathrm{Lap}\left(\frac{4\Delta}{\epsilon}\right)$.
  if $y \geq \hat{T}$:
    Output $a_i = \top$, **Halt**.
  else
    Output $a_i := \perp$.

We will use a standard accuracy result about the sparse vector mechanism.

Lemma 2.3.4 (see e.g., Dwork and Roth [2014] for a proof). *Let $\alpha > 0, \beta \in (0,1)$. Say sparse vector on a sequence $q_1, \ldots, q_k$ of Δ sensitive queries and threshold T is* (α, β)-accurate *if with*

probability at least $1-\beta$*, it outputs* $\perp$ *while* q_i *has value at most* $T-\alpha$*, and halts on the first query with value greater than* $T+\alpha$*. Then, sparse vector with privacy parameter* ε *is* ε*-differentially private and* (α, β)*-accurate for*

$$\alpha = \frac{8\Delta(\log k + \log(2/\beta))}{\varepsilon}.$$

CHAPTER 3

Private Counting Query Release

3.1. Introduction

Differentially private query release is one of the most fundamental tasks for privacy-preserving data analysis. The goal of *query release* is to release accurate answers to a set of statistical queries. A statistical query is asking "what is the fraction of individuals in the database satisfies some specified property?" As observed early on by Blum et al. [2005], performing private query release is sufficient to simulate any learning algorithm in the *statistical query model* by Kearns [1998].

Since then, the query release problem has been extensively studied in the differential privacy literature. While simple perturbation can be used to privately answer a small number of queries [Dwork et al., 2006], more sophisticated approaches can accurately answer nearly exponentially many queries in the size of the private database [Blum et al., 2013, Dwork et al., 2009, 2010b, Roth and Roughgarden, 2010, Hardt and Rothblum, 2010, Gupta et al., 2012, Hardt et al., 2012]. A natural approach, employed by many of these algorithms, is to answer queries by generating *synthetic data*: a safe version of the dataset that approximates the real dataset on every statistical query of interest.

Unfortunately, even the most efficient approaches for query release have a per-query running time linear in the size of the *data universe*, which is exponential in the dimension of the data [Hardt and Rothblum, 2010]. Moreover, this running time is necessary in the worst case [Ullman, 2013], especially if the algorithm produces synthetic data [Ullman and Vadhan, 2011].

This exponential runtime has hampered practical evaluation of query release algorithms. One notable exception is due to Hardt et al. [2012], who perform a thorough experimental evaluation of one such algorithm, which they called MWEM. They find that MWEM has quite good accuracy in practice and scales to higher dimensional data than suggested by a theoretical (worst-case) analysis. Nevertheless, running time remains a problem, and the approach does not seem to scale to high dimensional data (with more than 30 or so attributes for general queries, and more when the queries satisfy special structure[1]). The critical bottleneck is the size of the state maintained by the algorithm: MWEM, like many query release algorithms, needs to manipulate an object that has size linear in the size of the data universe (i.e., exponential in the dimension). This quickly becomes impractical as the record space grows more complex.

In this chapter, we present DualQuery, an alternative algorithm which is *dual* to MWEM in a sense that we will make precise. Rather than manipulating an object of exponential size, DualQuery solves a concisely represented (but NP-hard) optimization problem. Critically, the optimization step does not require a solution that is private or exact, so it can be handled by existing, highly optimized solvers. Except for this step, all parts of our algorithm are extremely efficient. As a result, DualQuery requires (worst-case) space and (in practice) time only linear in the number of *queries* of interest, which is often significantly smaller than the number of possible records. Like existing algorithms for query release, DualQuery has a provable accuracy guarantee and satisfies the strong differential privacy guarantee.

We evaluate DualQuery on a variety of datasets by releasing *3-way marginals* (also known as *conjunctions* or *contingency tables*), demonstrating that it solves the query release problem accurately and efficiently even when the data includes hundreds of thousands of features. We know of no other algorithm to perform accurate, private query release for rich

[1] Hardt et al. [2012] are able to scale up to 1000 features on synthetic data when the features are partitioned into a number of small buckets, and the queries are chosen to never depend on features in more than one bucket.

classes of queries on real data with more than even 100 features.

3.2. Related Work

There has been a significant amount of work on privately releasing synthetic data based on a true dataset while preserving the answers to large numbers of statistical queries [Blum et al., 2013, Dwork et al., 2009, Roth and Roughgarden, 2010, Dwork et al., 2010b, Hardt and Rothblum, 2010, Gupta et al., 2012]. These results are extremely strong in an information theoretic sense: they ensure the consistency of the synthetic data with respect to an exponentially large family of statistics. However, all of these algorithms (including the notable multiplicative weights algorithm of Hardt and Rothblum [2010], which achieves the theoretically optimal accuracy and runtime) have running time exponential in the dimension of the data. With standard cryptographic assumptions, this is necessary in the worst case for mechanisms that answer arbitrary statistical queries [Ullman, 2013].

Nevertheless, there have been some experimental evaluations of these approaches on real datasets. Most related to our work is the evaluation of the MWEM mechanism by Hardt et al. [2012], which is based on the private multiplicative weights mechanism [Hardt and Rothblum, 2010]. This algorithm is inefficient (it manipulates a probability distribution over a set exponentially large in the dimension of the data space) but with some heuristic optimizations, Hardt et al. [2012] were able to implement the multiplicative weights algorithm on several real datasets with up to 77 attributes (and even more when the queries are restricted to take positive values only on a small number of disjoint groups of features). However, it seems difficult to scale this approach to higher dimensional data.

Another family of query release algorithms are based on the Matrix Mechanism [Li et al., 2015, Li and Miklau, 2015]. The runtime guarantees of the matrix mechanism are similar to the approaches based on multiplicative weights—the algorithm manipulates a "matrix" of queries with dimension exponential in the number of features. Yaroslavtsev et al. [2013] evaluate an approach based on this family of algorithms on low dimensional datasets, but

scaling to high dimensional data also seems challenging. A recent work by Zhang et al. [2014] proposes a low-dimensional approximation for high-dimensional data distribution by privately constructing Bayesian networks, and shows that such a representation gives good accuracy on some real datasets.

Our algorithm is inspired by the view of the synthetic data generation problem as a zero-sum game, first proposed by Hsu et al. [2013]. In this interpretation, Hardt et al. [2012] solves the game by having a *data player* use a no-regret learning algorithm, while the *query player* repeatedly best responds by optimizing over queries. In contrast, our algorithm swaps the roles of the two players: the query player now uses the no-regret learning algorithm, whereas the data player now finds best responses by solving an optimization problem. This is reminiscent of "Boosting for queries," proposed by Dwork et al. [2010b]; the main difference is that our optimization problem is over single records rather than sets of records. As a result, our optimization can be handled non-privately.

3.3. The Query Release Game

First, we will give the formal definition of a counting query.

Definition 3.3.1. *For any predicate* $\varphi\colon \mathcal{X} \to \{0,1\}$, *the* counting query (or statistical query) $Q_\varphi\colon \mathcal{X}^n \to [0,1]$ *is defined by*

$$Q_\varphi(D) = \frac{\sum_{x\in D} \varphi(x)}{|D|},$$

where $|D|$ *denotes the size of the database* D.

The analysis of our algorithm relies on the interpretation of query release as a two player, zero-sum game [Hsu et al., 2013]. In the present section, we review this idea and related tools.

Game Definition Suppose we want to answer a set of queries $\mathcal{Q}$. For each query $q \in \mathcal{Q}$, we can form the *negated query* $\bar{q}$, which takes values $\bar{q}(D) = 1 - q(D)$ for every database D. Equivalently, for a linear query defined by a predicate φ, the negated query is defined

by the negation $\neg\varphi$ of the predicate. For the remainder, we will assume that $\mathcal{Q}$ is closed under negation; if not, we may add negated copies of each query to $\mathcal{Q}$.

Let there be two players, whom we call the *data* player and *query* player. The data player has action set equal to the data universe $\mathcal{X}$, while the query player has action set equal to the query class $\mathcal{Q}$. Given a play $x \in \mathcal{X}$ and $q \in \mathcal{Q}$, we let the payoff be

$$A(x, q) := q(D) - q(x), \tag{3.1}$$

where D is the true database. As a zero sum game, the data player will try to minimize the payoff, while the query player will try to maximize the payoff.

Equilibrium of the Game Let $\Delta(\mathcal{X})$ and $\Delta(\mathcal{Q})$ be the set of probability distributions over $\mathcal{X}$ and $\mathcal{Q}$. We consider how well each player can do if they randomize over their actions, i.e., if they play from a probability distribution over their actions. By von Neumann's minimax theorem,

$$\min_{u \in \Delta(\mathcal{X})} \max_{w \in \Delta(\mathcal{Q})} A(u, w) = \max_{w \in \Delta(\mathcal{Q})} \min_{u \in \Delta(\mathcal{X})} A(u, w),$$

for any two player zero-sum game, where

$$A(u, w) := \mathbb{E}_{x \sim u, q \sim w} A(x, q)$$

is the expected payoff. The common value is called the *value of the game*, which we denote by v_A. Intuitively, von Neumann's theorem states that there is no advantage in a player going first: the minimizing player can always force payoff at most v_A, while the maximizing player can always force payoff at least v_A.

This suggests that each player can play an optimal strategy, assuming best play from the opponent—this is the notion of equilibrium strategies, which we now define. We will soon interpret these strategies as solutions to the query release problem.

Definition 3.3.2. *Let $\alpha > 0$. Let A be the payoffs for a two player, zero-sum game with action*

sets $\mathcal{X}, \mathcal{Q}$. Then, a pair of strategies $u^* \in \Delta(\mathcal{X})$ and $w^* \in \Delta(\mathcal{Q})$ form an α-approximate mixed Nash equilibrium *if*

$$A(u^*, w) \leq v_A + \alpha \qquad \text{and} \qquad A(u, w^*) \geq v_A - \alpha$$

for every strategy $u \in \Delta(\mathcal{X}), w \in \Delta(\mathcal{Q})$.

If the true database D is normalized to be a distribution $\hat{D}$ in $\Delta(\mathcal{X})$, then $\hat{D}$ always has zero payoff:

$$A(\hat{D}, w) = \mathbb{E}_{x \sim \hat{D}, q \sim w}[q(x) - q(D)] = 0.$$

Hence, the value of the game v_A is at most 0. Also, for any data strategy u, the payoff of query q is the negated payoff of the negated query $\bar{q}$:

$$A(u, \bar{q}) = \mathbb{E}_{x \sim u}[\bar{q}(x) - \bar{q}(D)] = \mathbb{E}_{x \sim u}[q(D) - q(x)],$$

which is $A(u, \bar{q})$. Thus, any query strategy that places equal weight on q and $\bar{q}$ has expected payoff zero, so v_A is at least 0. Hence, $v_A = 0$.

Now, let (u^*, w^*) be an α-approximate equilibrium. Suppose that the data player plays u^*, while the query player always plays query q. By the equilibrium guarantee, we then have $A(u^*, q) \leq \alpha$, but the expected payoff on the left is simply $q(D) - q(u^*)$. Likewise, if the query player plays the negated query $\bar{q}$, then

$$-q(D) + q(u^*) = A(u^*, \bar{q}) \leq \alpha,$$

so $q(D) - q(u^*) \geq -\alpha$. Hence for every query $q \in \mathcal{Q}$, we know $|q(u^*) - q(D)| \leq \alpha$. This is precisely what we need for query release: we just need to privately calculate an approximate equilibrium.

Solving the Game To construct the approximate equilibrium, we will use the multiplicative weights update algorithm (MW).[2]

This algorithm maintains a distribution over actions (initially uniform) over a series of steps. At each step, the MW algorithm receives a (possibly adversarial) loss for each action. Then, MW reweights the distribution to favor actions with less loss.

The algorithm is presented in Algorithm 2.

Algorithm 2 The Multiplicative Weights Algorithm

Let $\eta > 0$ be given, let $\mathcal{A}$ be the action space
Initialize $\tilde{A}^1$ uniform distribution on $\mathcal{A}$
For $t = 1, 2, \ldots, T$:
  Receive loss vector ℓ^t
  For each $a \in \mathcal{A}$**:**
    Update $A_a^{t+1} = e^{-\eta \ell_a^t} \tilde{A}_a^t$ for every $a \in \mathcal{A}$
  Normalize $\tilde{A}^{t+1} = \frac{A^{t+1}}{\sum_i A_i^{t+1}}$

For our purposes, the most important application of MW is to solving zero-sum games. Freund and Schapire [1996] showed that if one player maintains a distribution over actions using MW, while the other player selects a *best-response* action versus the current MW distribution (i.e., an action that maximizes his expected payoff), the average MW distribution and empirical best-response distributions will converge to an approximate equilibrium rapidly.

Theorem 3.3.3 ([Freund and Schapire, 1996])**.** *Let $\alpha > 0$, and let $A(i,j) \in [-1,1]^{m \times n}$ be the payoff matrix for a zero-sum game. Suppose the first player uses multiplicative weights over their actions to play distributions $p^1, \ldots, p^T$, while the second player plays $(\alpha/2)$-approximate best responses $x^1, \ldots, x^T$, i.e.,*

$$A(p^t, x^t) \geq \max_x A(p^t, x) - \alpha/2.$$

[2] The MW algorithm has wide applications; it has been rediscovered in various guises several times. More details can be found in the comprehensive survey by Arora et al. Arora et al. [2012].

Setting $T = 16 \log n/\alpha^2$ *and* $\eta = \alpha/4$ *in the MW algorithm, the empirical distributions*

$$\frac{1}{T}\sum_{i=1}^{T} p^i \quad \textit{and} \quad \frac{1}{T}\sum_{i=1}^{T} x^i$$

form an α*-approximate mixed Nash equilibrium.*

3.4. Dual Query Release

By the game interpretation, the algorithm of Hardt and Rothblum [2010] (and the MWEM algorithm of Hardt et al. [2012]) uses MW for the data player, while the query player plays best responses. For privacy, their algorithm selects the query best-responses privately via the exponential mechanism of McSherry and Talwar [2007]. Our algorithm simply reverses the roles.

While MWEM uses a no-regret algorithm to maintain the data player's distribution, we will instead use a no-regret algorithm for the query player's distribution. Likewise, instead of finding a maximum payoff query at each round, our algorithm selects a minimum payoff record at each turn. The full algorithm can be found in Algorithm 3.

Our privacy argument differs slightly from the analysis for MWEM. There, the data distribution is public, and finding a query with high error requires access to the private data. Our algorithm instead maintains a distribution Q over queries which depends directly on the private data, so we cannot use Q directly. Instead, we argue that *queries sampled from* Q are privacy preserving. Then, we can use a non-private optimization method to find a minimal error record versus queries sampled from Q. We then trade off privacy (which degrades as we take more samples) with accuracy (which improves as we take more samples, since the distribution of sampled queries converges to Q).

Given known hardness results for the query release problem [Ullman, 2013], our algorithm must have worst-case runtime polynomial in the universe size $|\mathcal{X}|$, so is not theoretically more efficient than prior approaches. In fact, even compared to prior work on query

release (e.g., [Hardt and Rothblum, 2010]), our algorithm has a weaker accuracy guarantee. However, our approach has an important practical benefit: the computationally hard step can be handled with standard, non-private solvers.

The iterative structure of our algorithm, combined with our use of constraint solvers, also allows for several heuristics improvements. For instance, we may run for fewer iterations than predicted by theory. Or, if the optimization problem turns out to be hard (even in practice), we can stop the solver early at a suboptimal (but often still good) solution. These heuristic tweaks can improve accuracy beyond what is guaranteed by our accuracy theorem, while always maintaining a strong *provable* privacy guarantee.

Algorithm 3 DualQuery

Input: Database $D \in \mathbb{R}^{|\mathcal{X}|}$ (normalized) and linear queries $q_1, \ldots, q_k \in \{0,1\}^{|\mathcal{X}|}$.
Initialize: Let $\mathcal{Q} = \bigcup_{j=1}^{k} q_j \cup \bar{q}_j$, Q^1 uniform distribution on $\mathcal{Q}$,

$$T = \frac{16 \log |\mathcal{Q}|}{\alpha^2}, \qquad \eta = \frac{\alpha}{4}, \qquad s = \frac{48 \log\left(\frac{2|\mathcal{X}|T}{\beta}\right)}{\alpha^2}.$$

For $t = 1, \ldots, T$:
  Sample s queries $\{q_i\}$ from $\mathcal{Q}$ according to Q^t.
  Let $\tilde{q} := \frac{1}{s} \sum_i q_i$.
  Find x^t with $A(\tilde{q}, x^t) \geq \max_x A(\tilde{q}, x) - \alpha/4$.
  Update: For each $q \in \mathcal{Q}$:
    $Q_q^{t+1} := \exp(-\eta A(q, x^t)\rangle) \cdot Q_q^t$.
  Normalize Q^{t+1}.
Output synthetic database $\hat{D} := \bigcup_{t=1}^{T} x^t$.

Privacy The privacy proofs are largely routine, based on the composition theorems. Rather than fixing ε and solving for the other parameters, we present the privacy cost ε as function of parameters T, s, η. Later, we will tune these parameters for our experimental evaluation.

We first prove pure ε-differential privacy.

Theorem 3.4.1. DualQuery *is ε-differentially private for*

$$\varepsilon = \frac{\eta T (T-1) s}{n}.$$

Proof. We will argue that sampling from Q^t is equivalent to running the exponential mechanism with some quality score. At round t, let $\{x^i\}$ for $i \in [t-1]$ be the best responses for the previous rounds. Let $r(q, d)$ be defined by

$$r(q, D) = \sum_{i=1}^{t-1} (q(x^i) - q(D)),$$

where $q \in \mathcal{Q}$ is a query and D is the true database. This function is evidently $((t-1)/n)$-sensitive in D: changing D changes each $q(D)$ by at most $1/n$. Now, note that sampling from Q^t is simply sampling from the exponential mechanism, with quality score $r(q, D)$. Thus, the privacy cost of each sample in round t is $\varepsilon'_t = 2\eta(t-1)/n$ (Theorem 2.3.3).

By the standard composition theorem (Theorem 2.2.2), the total privacy cost is

$$\varepsilon = \sum_{t=1}^{T} s\varepsilon'_t = \frac{2\eta s}{n} \cdot \sum_{t=1}^{T} (t-1) = \frac{\eta T(T-1)s}{n}.$$

□

We next show that DualQuery is (ε, δ)-differentially private, for a much smaller ε.

Theorem 3.4.2. *Let* $0 < \delta < 1$. DualQuery *is* (ε, δ)*-differentially private for*

$$\varepsilon = \frac{2\eta(T-1)}{n} \cdot \left[\sqrt{2s(T-1)\log(1/\delta)} + s(T-1)\left(\exp\left(\frac{2\eta(T-1)}{n}\right) - 1\right)\right].$$

Proof. Let ε be defined by the above equation. By the advanced composition theorem (Theorem 2.2.4), running a composition of k ε'-private mechanisms is (ε, δ)-private, for

$$\varepsilon = \sqrt{2k\log(1/\delta)}\varepsilon' + k\varepsilon'(\exp(\varepsilon') - 1).$$

Again, note that sampling from Q^t is simply sampling from the exponential mechanism, with a $(T-1)/n$-sensitive quality score. Thus, the privacy cost of each sample is $\varepsilon' = 2\eta(T-1)/n$ (Theorem 2.3.3). We plug in $k = s(T-1)$ samples, as in the first round our

samplings are 0-differentially private. □

Accuracy The accuracy proof proceeds in two steps. First, we show that "average query" formed from the samples is close to the true weighted distribution Q^t. We will need a standard Chernoff bound.

Lemma 3.4.3 (Chernoff bound)**.** *Let $X_1, \ldots, X_N$ be IID random variables with mean μ, taking values in $[0,1]$. Let $\bar{X} = \frac{1}{N}\sum_i X_i$ be the empirical mean. Then,*

$$\Pr[|\bar{X} - \mu| > T] < 2\exp(-NT^2/3)$$

for any T.

Lemma 3.4.4. *Let $\beta \in (0,1)$, and let p be a distribution over queries. Suppose we draw*

$$s = \frac{48\log\left(\frac{2|\mathcal{X}|}{\beta}\right)}{\alpha^2}$$

samples $\{\hat{q}_i\}$ from p, and let $\bar{q}$ be the aggregate query

$$\bar{q} = \frac{1}{s}\sum_{i=1}^{s} \hat{q}_i.$$

Define the true weighted answer $Q(x)$ to be

$$Q(x) = \sum_{i=1}^{|\mathcal{Q}|} p_i q_i(x).$$

Then with probability at least $1-\beta$, we have $|\bar{q}(x) - Q(x)| < \alpha/4$ for every $x \in \mathcal{X}$.

Proof. For any fixed x, note that $\bar{q}(x)$ is the average of random variables $\hat{q_1}(x), \ldots, \hat{q_s}(x)$. Also, note that $\mathbb{E}[\bar{q}(x)] = Q(x)$. Thus, by the Chernoff bound (Lemma 3.4.3) and our choice of s,

$$\Pr[|\bar{q}(x) - Q(x)| > \alpha/4] < 2\exp(-s\alpha^2/48) = \beta/|\mathcal{X}|.$$

By a union bound over $x \in \mathcal{X}$, this equation holds for all $x \in \mathcal{X}$ with probability at least $1 - \beta$. □

Next, we show that we compute an approximate equilibrium of the query release game. In particular, the record best responses form a synthetic database that answer all queries in $\mathcal{Q}$ accurately. Note that our algorithm doesn't require an exact best response for the data player; an approximate best response will do.

Theorem 3.4.5. *With probability at least* $1 - \beta$, DualQuery *finds a synthetic database that answers all queries in* $\mathcal{Q}$ *within additive error* α.

Proof. As discussed in Section 3.3, it suffices to show that the distribution of best responses $x^1, \ldots, x^T$ forms is an α-approximate equilibrium strategy in the query release game. First, we set the number of samples s according to in Lemma 3.4.4 with failure probability β/T. By a union bound over T rounds, sampling is successful for every round with probability at least $1 - \beta$; condition on this event.

Since we are finding an $\alpha/4$ approximate best response to the sampled aggregate query $\bar{q}$, which differs from the true distribution by at most $\alpha/4$ (by Lemma 3.4.4), each x^i is an $\alpha/4 + \alpha/4 = \alpha/2$ approximate best response to the true distribution Q^t. Since q takes values in $[0, 1]$, the payoffs are all in $[-1, 1]$. Hence, Theorem 3.3.3 applies; setting T and η accordingly gives the result. □

Remark 3.4.6. *The guarantee in Theorem 3.4.5 may seem a little unusual, since the convention in the literature is to treat* ε, δ *as inputs to the algorithm. We can do the same: from Theorem 3.4.2 and plugging in for* T, η, s, *we have*

$$\begin{aligned}\varepsilon &= \frac{4\eta T\sqrt{2sT\log(1/\delta)}}{n} \\ &= \frac{256\log^{3/2}|\mathcal{Q}|\sqrt{6\log(1/\delta)\log(2|\mathcal{X}|T/\beta)}}{\alpha^3 n}.\end{aligned}$$

Solving for α, we find

$$\alpha = O\left(\frac{\log^{1/2}|\mathcal{Q}|\log^{1/6}(1/\delta)\log^{1/6}(2|\mathcal{X}|/\gamma)}{n^{1/3}\varepsilon^{1/3}}\right),$$

for $\gamma < \beta/T$.

3.5. Case study: 3-way marginals

In our algorithm, the computationally difficult step is finding the data player's approximate best response against the query player's distribution. As mentioned above, the form of this problem depends on the particular query class $\mathcal{Q}$. In this section, we first discuss the optimization problem in general, and then specifically for the well-studied class of *marginal* queries [Thaler et al., 2012, Gupta et al., 2013, Dwork et al., 2014]. For instance, in a database of medical information in binary attributes, a particular marginal query may be: What fraction of the patients are over 50, smoke, and exercise?

The Best-Response Problem Recall that the query release game has payoff $A(x,q)$ defined by Equation (3.1); the data player tries to minimize the payoff, while the query player tries to maximize it. If the query player has distribution Q^t over queries, the data player's best response minimizes the expected loss:

$$\operatorname*{argmin}_{x\in\mathcal{X}} \mathop{\mathbb{E}}_{q\leftarrow Q^t}[q(D)-q(x)].$$

To ensure privacy, the data player actually plays against the distribution of samples $\hat{q_1},\ldots,\hat{q_s}$. Since the database D is fixed and $\hat{q_i}$ are linear queries, the best-response problem is

$$\operatorname*{argmin}_{x\in\mathcal{X}} \frac{1}{s}\sum_{i=1}^{s}\hat{q_i}(D)-\hat{q_i}(x) = \operatorname*{argmax}_{x\in\mathcal{X}} \sum_{i=1}^{s}\hat{q_i}(x).$$

By Theorem 3.4.5 it even suffices to find an approximate maximizer, in order to guarantee

accuracy.

3-Way Marginal Queries To look at the precise form of the best-response problem, we consider *3-way marginal* queries. We think of records as having d binary attributes, so that the data universe $|\mathcal{X}|$ is all bitstrings of length d. We write x_i for $x \in \mathcal{X}$ to mean the ith bit of record x.

Definition 3.5.1. *Let $\mathcal{X} = \{0,1\}^d$. A* 3-way marginal query *is a linear query specified by 3 integers $a \neq b \neq c \in [d]$, taking values*

$$q_{abc}(x) = \begin{cases} 1 & : x_a = x_b = x_c = 1 \\ 0 & : \textit{otherwise.} \end{cases}$$

Though everything we do will apply to general k-way marginals, for concreteness we work with 3-way marginals.

Given sampled conjunctions $\{\hat{u_i}\}$ and negated conjunctions $\{\hat{v_i}\}$, the best-response problem is

$$\operatorname*{argmax}_{x \in \mathcal{X}} \sum_i \hat{u_i}(x) + \sum_j \hat{v_j}(x).$$

In other words, this is a MAXCSP problem—we can associate a clause to each conjunction:

$$q_{abc} \Rightarrow (x_a \wedge x_b \wedge x_c) \quad \text{and} \quad q_{\bar{a}bc} \Rightarrow (\bar{x_a} \vee \bar{x_b} \vee \bar{x_c}),$$

and we want to find $x \in \{0,1\}^d$ satisfying as many clauses as possible.[3]

Since most solvers do not directly handle MAXCSP problems, we convert this optimization problem into a more standard, integer program form. We introduce a variable x_i for each literal x_i, a variable c_i for each sampled conjunction $\hat{u_i}$, a variable d_i for each sampled

[3]Note that this is almost a MAX3SAT problem, except there are also "negative" clauses.

negated conjunction $\hat{v_i}$, and we form the following integer program.

$$\max \sum_i c_i + \sum_j d_j$$

$$\text{with } \forall \hat{u_i} = q_{abc} : \; x_a + x_b + x_c \geq 3c_i$$

$$\forall \hat{v_j} = \bar{q_{abc}} : \; (1 - x_a) + (1 - x_b) + (1 - x_c) \geq d_j$$

$$x_i, c_i, d_i \in \{0, 1\}$$

Note that $x_i, 1 - x_i$ corresponds to the literals $x_i, \bar{x_i}$, and $c_i = 1$, $d_i = 1$ exactly when their respective clauses are satisfied. Thus, the objective is the number of satisfied clauses. The resulting integer program can be solved using any standard solver; we use CPLEX.

3.6. Case study: Parity queries

We can also apply DualQuery to another well-studied class of queries: *parities*. Each specified by a subset S of features, these queries measure the number of records with an even number of bits on in S compared to the number of records with an odd number of bits on in S.

Definition 3.6.1. *Let $\mathcal{X} = \{-1, +1\}^d$. A* k-wise parity query *is a linear query specified by a subset of features $S \subseteq [d]$ with $|S| = k$, taking values*

$$q_S(x) = \begin{cases} +1 & : \textit{even number of } x_i = +1 \textit{ for } i \in S \\ -1 & : \textit{otherwise.} \end{cases}$$

Like before, we can define a negated k-wise parity query*:*

$$\bar{q_S}(x) = \begin{cases} +1 & : \textit{odd number of } x_i = +1 \textit{ for } i \in S \\ -1 & : \textit{otherwise.} \end{cases}$$

Barak et al. [2007] observed that answering k-way marginal queries can be reduced to answering k-wise parity queries to the same accuracy, in the following sense.

Theorem 3.6.2 ([Barak et al., 2007])**.** *Let q_S be a k-way marginal query specified by the set of features S, and let D be a database of records. Then,*

$$q_S(D) = \frac{1}{2^k} \sum_{T \subseteq S} p_T(D),$$

where p_T is the parity query for features T.

Note that the coefficients sum to 1: hence, answering parity queries with additive error α is enough to answer marginal queries with additive error α. We now consider how to handle these queries with our algorithm. For the remainder, we specialize to $k = 3$. Like marginal queries, it suffices to give the best-response optimization problem; unlike marginal queries, we need to handle k-wise parities for every $k \leq 3$ in order to apply Theorem 3.6.2.

Given sampled parity queries $\{\hat{u_i}\}$ and negated parity queries $\{\hat{v_i}\}$, the best response problem is to find the record $x \in \mathcal{X}$ that takes value 1 on as many of these queries as possible. We can construct an integer program for this task: introduce d variables x_i, and two variables c_q, d_q for each sampled query. The following integer program encodes the best-response problem.

$$\begin{aligned} \max & \sum_i c_i \\ \text{such that } \forall \hat{u_i} = q_S. & \sum_j x_{S_j} = 2d_i + c_i - 1 \\ \forall \hat{v_i} = \bar{q_S}. & \sum_j x_{S_j} = 2d_i + c_i \\ & x_i, c_i, d_i \in \{0, 1\} \end{aligned}$$

Consider the (non-negated) parity queries first. The idea is that each variable c_i can be set

to 1 exactly when the corresponding parity query takes value 1, i.e., when x has an even number of bits in S set to +1. Since $|S| \leq 3$, this even number will either be 0 or 2, hence is equal to $2d_i$ for $d_i \in \{0,1\}$. A similar argument holds for the negated parity queries.

3.7. Experimental Evaluation

Dataset	Size	Attributes	Binary attributes
Adult	30162	14	235
KDD99	494021	41	396
Netflix	480189	17,770	17,770

Table 1: Test Datasets for DualQuery

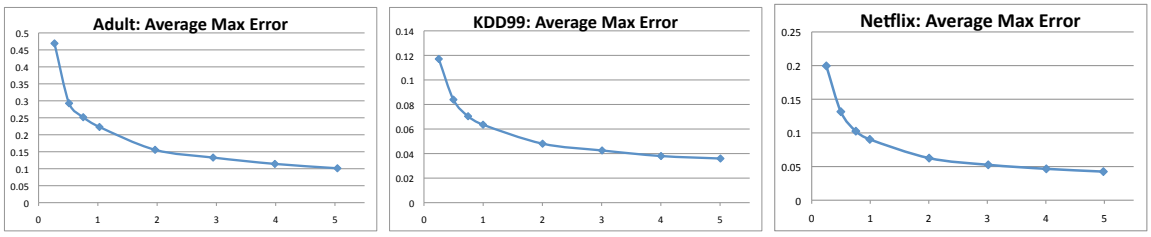

Figure 1: Average max error of $(\varepsilon, 0.001)$-private DualQuery on 500,000 3-way marginals versus ε.

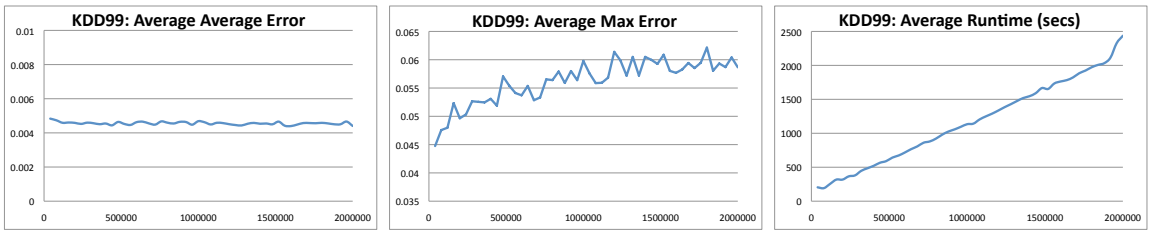

Figure 2: Error and runtime of $(1, 0.001)$-private DualQuery on KDD99 versus number of queries.

We evaluate DualQuery on a large collection of 3-way marginal queries on several real datasets (Table 1) and high dimensional synthetic data. Adult (census data) and KDD99 (network packet data) are from the UCI repository [Bache and Lichman, 2013], and have a mixture of discrete (but non-binary) and continuous attributes, which we discretize into binary attributes. We also use the (in)famous Netflix movie ratings dataset, with more than

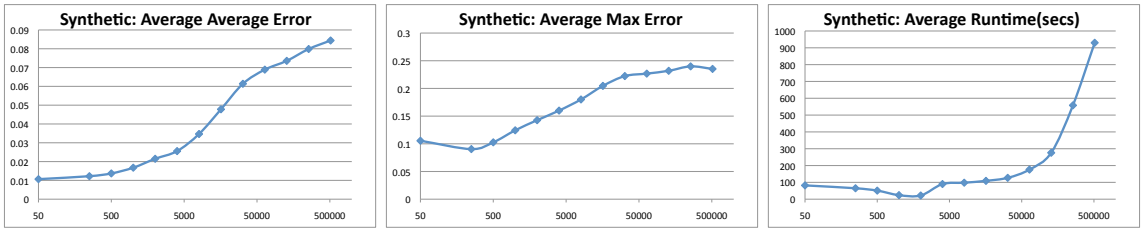

Figure 3: Error and runtime of $(1, 0.001)$-private DualQuery on 100,000 3-way marginal queries versus number of attributes.

17,000 binary attributes. More precisely, we can consider each attribute (corresponding to a movie) to be 1 if a user has watched that movie, and 0 otherwise.

Rather than set the parameters as in Algorithm 3, we experiment with a range of parameters. For instance, we frequently run for fewer rounds (lower T) and take fewer samples (lower s). As such, the accuracy guarantee (Theorem 3.4.5) need not hold for our parameters. However, we find that our algorithm gives good error, often much better than predicted. In all cases, our parameters satisfy the privacy guarantee Theorem 3.4.2.

Accuracy We evaluate the accuracy of the algorithm on 500,000 3-way marginals on Adult, KDD99 and Netflix. We report maximum error in Figure 1, averaged over 5 runs. (Marginal queries have range $[0, 1]$, so error 1 is trivial.)

The runs are $(\varepsilon, 0.001)$-differentially private, with ε ranging from 0.25 to 5.[4]

For the Adult and KDD99 datasets, we set step size $\eta = 2.0$, sample size $s = 1000$ while varying the number of steps T according to the privacy budget ε, using the formula from Theorem 3.4.2. For the Netflix dataset, we adopt the same heuristic except we set s to be 5000.

The accuracy improves noticeably when ε increases from 0.25 to 1 across 3 datasets, and

[4] Since our privacy analysis follows from the composition theorem of differential privacy, our algorithm actually satisfies (ε, δ)-privacy for smaller values of δ. For example, our algorithm is also $(\sqrt{2}\varepsilon, \delta')$-private for $\delta' = 10^{-6}$. Similarly, we could choose any arbitrarily small value of δ, and the composition theorem would tell us that our algorithm was (ε', δ)-differentially private for an appropriate value ε', which depends only sub-logarithmically on $1/\delta$.

the improvement diminishes gradually with larger ε. With larger sizes, both KDD99 and Netflix datasets allow DualQuery to run with more steps and get significantly better error.

Scaling to More Queries Next, we evaluate accuracy and runtime when varying the number of queries. We use a set of 40,000 to 2 million randomly generated marginals $\mathcal{Q}$ on the KDD99 dataset and run DualQuery with $(1, 0.001)$-privacy.

For all experiments, we use the same set of parameters: $\eta = 1.2, T = 170$ and $s = 1750$. By Theorem 3.4.2, each run of the experiment satisfies $(1, 0.001)$-differential privacy. These parameters give stable performance as the query class $\mathcal{Q}$ grows.

As shown in Figure 2, both average and max error remain mostly stable, demonstrating improved error compared to simpler perturbation approaches. For example, the Laplace mechanism's error growth rate is $O(\sqrt{|\mathcal{Q}|})$ under (ε, δ)-differential privacy.

The runtime grows almost linearly in the number of queries, since we maintain a distribution over all the queries.

Scaling to Higher Dimensional Data Finally, we evaluate accuracy and runtime behavior for data dimension ranging from 50 to 512,000. We evaluate DualQuery under $(1, 0.001)$-privacy on 100,000 3-way marginals on synthetically genearted datasets. We report runtime, max, and average error over 3 runs in Figure 3; note the logarithmic scale for attributes axis. We do not include query evaluation in our time measurements—this overhead is common to all approaches that answer a set of queries.

When generating the synthetic data, one possibility is to set each attribute to be 0 or 1 uniformly at random. However, this generates very uniform synthetic data: a record satisfies any 3-way marginal with probability 1/8, so most marginals will have value near 1/8. To generate more challenging and realistic data, we pick a separate bias $p_i \in [0, 1]$ uniformly at random for each attribute i. For each data point, we then set attribute i to be 1 independently with probability equal to p_i. As a result, different 3-way marginals have different

answers on our synthetic data.

For parameters, we fix step size η to be 0.4, and increase the sample size s with the dimension of the data (from 200 to 50,000) at the expense of running fewer steps. For these parameters, our algorithm is $(1, 0.001)$-differentially private by Theorem 3.4.2. With this set of parameters, we are able to obtain 8% average error in an average of 10 minutes of runtime, excluding query evaluation.

Methodology In this section, we discuss our experimental setup in more detail.

Implementation details. The implementation is written in OCaml, using the CPLEX constraint solver. We ran the experiments on a mid-range desktop machine with a 4-core Intel Xeon processor and 12 Gb of RAM. Heuristically, we set a timeout for each CPLEX call to 20 seconds, accepting the best current solution if we hit the timeout. For the experiments shown, the timeout was rarely reached.

Data discretization. We discretize KDD99 and Adult datasets into binary attributes by mapping each possible value of a discrete attribute to a new binary feature. We bucket continuous attributes, mapping each bucket to a new binary feature. We also ensure that our randomly generated 3-way marginal queries are sensible (i.e., they don't require an original attribute to take two different values).

Setting free attributes. Since the collection of sampled queries may not involve all of the attributes, CPLEX often finds solutions that leave some attributes unspecified. We set these *free* attributes heuristically: for real data, we set the attributes to 0 as these datasets are fairly sparse;[5] for synthetic data, we set attributes to 0 or 1 uniformly at random.[6]

[5]The adult and KDD99 datasets are sparse due to the way we discretize the data; for the Netflix dataset, most users have only viewed a tiny fraction of the 17,000 movies.

[6]For a more principled way to set these free attributes, the sparsity of the dataset could be estimated at a small additional cost to privacy.

Parameter tuning. DualQuery has three parameters that can be set in a wide variety of configurations without altering the privacy guarantee (Theorem 3.4.2): number of iterations (T), number of samples (s), and learning rate (η), which controls how aggressively to update the distribution. For a fixed level of ε and δ, there are many feasible private parameter settings.

Performance depends strongly on the choice of parameters: T has an obvious impact, increasing s increases the number of constraints in the integer program for CPLEX.

For sparse datasets like the ones in Table 1, with larger η (in the range of 1.5 to 2.0) and smaller s (in the same order as the number of attributes), we obtain good accuracy when we set the free attributes to be 0. But for denser data like our synthetic data, we get better accuracy when we have smaller η (around 0.4) and set the free attributes randomly. As for runtime, we observe that CPLEX could finish running quickly (in seconds) when the sample size is in about the same range as the number of attributes (factor of 2 or 3 different).

Parameter setting should be done under differential privacy for a truly realistic evaluation. Overall, we do not know of a principled approach to handle this issue; private parameter tuning is an area of active research (see e.g., [Chaudhuri and Vinterbo, 2013]).

3.8. Discussion and Conclusion

We have given a new private query release mechanism that can handle datasets with dimensionality multiple orders of magnitude larger than what was previously possible.

Indeed, it seems we have not reached the limits of our approach—even on synthetic data with more than 500,000 attributes, DualQuery continues to generate useful answers with about 30 minutes of overhead on top of query evaluation (which by itself is on the scale of hours). We believe that DualQuery makes private analysis of high dimensional data practical for the first time.

However, this remarkable improvement in running time is not free: our theoretical accuracy bounds are worse than those of previous approaches [Hardt and Rothblum, 2010, Hardt et al., 2012]. For low dimensional datasets for which it is possible to maintain a distribution over records, the MWEM algorithm of Hardt et al. [2012] likely remains the state of the art. Our work complements MWEM by allowing private data analysis on higher-dimensional data sets.

CHAPTER 4

Jointly Private Matchings and Allocations

4.1. Introduction

In the classic maximum-weight matching problem in bipartite graphs, there are k goods $j \in \{1,\dots,k\}$ and n buyers $i \in \{1,\dots,n\}$. Each buyer i has a value $v_{ij} \in [0,1]$ for each good j, and the goal is to find a matching μ between goods and buyers which maximizes the social welfare $\mathrm{SW} = \sum_{i=1}^{n} v_{i,\mu(i)}$. When the buyers' values are sensitive information,[1] it is natural to ask for a matching that hides the reported values of each of the players.

It is not hard to see that this goal is impossible under the standard notion of differential privacy, which requires that the allocation must be insensitive to the reported valuations of each player. We formalize this observation later in this chapter, but the intuition is simple. Consider the case with two types of goods with n identical copies each, and suppose that each buyer has a private preference for one of the two types: value 1 for the good that he likes, and value 0 for the other good. There is no contention since the supply of each good is larger than the total number of buyers, so any allocation achieving social welfare $\mathrm{OPT} - \alpha n$ can be used to reconstruct a $(1-\alpha)$ fraction of the preferences; this is plainly impossible for non-trivial values of α under differential privacy.

In light of this obstacle, is there any hope for privately solving maximum-weight matching problems? In this chapter, we show that the answer is *yes*: it is possible to solve matching problems (and more general allocation problems) to high accuracy assuming a small number of identical copies of each good, while still satisfying an extremely strong variant of differential privacy. We observe that the matching problem has the following two features:

[1] For instance, the goods might be related to the treatment of disease, or might be indicative of a particular business strategy, or might be embarrassing in nature.

1. Both the input and solution are naturally partitioned amongst the same n people: each buyer i receives the item $\mu(i)$ they are matched to in the solution.

2. The problem is not solvable privately because the item given to each buyer must reflect their own private data.

By utilizing these two features, we show that the matching problem can be accurately solved under the constraint of *joint differential privacy* [Kearns et al., 2014]. Informally speaking, this requires that for every buyer i, the joint distribution on items $\mu(j)$ for $j \neq i$ must be differentially private in the reported valuation of buyer i. As a consequence, buyer i's privacy is protected even if *all* other buyers collude, potentially sharing the identities of the items they receive. As long as buyer i does not reveal their own item, i's privacy is protected.

We then show that our techniques generalize beyond the max-matching problem to the more general *allocation* problem. Here, each buyer i has a valuation function defined over subsets of goods $v_i : 2^{[k]} \to [0,1]$ from some class of valuations, and the goal is to find a partition of the goods $S_1, \ldots, S_n$ maximizing social welfare; note that the maximum-weight matching problem is the special case when agents are *unit demand*, i.e., only want bundles of size 1. More specifically, we consider buyers with *gross substitutes* valuations. This is an economically meaningful class of valuation functions that is a strict subclass of submodular functions and are the most general class of valuations for which our techniques apply.

4.1.1. Our Techniques and Results

Our approach makes a novel connection between *market clearing prices* and differential privacy. Prices have long been considered as a low-information way to coordinate markets; our work formalizes this intuition in the context of differentially private allocation. Specifically, we will use *Walrasian equilibrium prices*: prices under which each buyer is simultaneously able to buy a most preferred bundle of goods, and no good is over-demanded.

Although the allocation itself cannot be computed under standard differential privacy, we show how to differentially privately compute the Walrasian equilibrium prices while coordinating a high welfare allocation under joint differential privacy.

We start from the classic analysis of Kelso and Crawford [1982], who show how to use *ascending price* auctions to compute Walrasian equilibrium prices. In the classical ascending price auction, each good begins with a price of 0 and each agent is initially unmatched to any good. Unmatched agents i take turns bidding on the good j^* that maximizes their utility at the current prices: i.e., $j^* \in \arg\max(v_{ij} - p_j)$. When a bidder bids on a good j^*, they become the new high bidder and the price of j^* is incremented. Bidders are tentatively matched to a good as long as they are the high bidder. The auction continues until there are no unmatched bidders who prefer to be matched at the current prices. The algorithm converges because each bid increases the the prices, which are bounded by some finite value.[2] Moreover, every bidder ends up matched to their most preferred good given the prices. Finally, by the *first welfare theorem* of Walrasian equilibria, any matching that corresponds to equilibrium prices maximizes social welfare. We emphasize that this final implication is key: "prices" play no role in our problem description, nor do we ever actually charge "prices" to the agents—the prices are purely a device to coordinate the matching.

We give an approximate, private version of Kelso and Crawford's algorithm based on several observations. First, in order to implement this algorithm, it is sufficient to maintain the sequence of prices of the goods privately: given a record of the price trajectory, each agent can figure out what good they are matched to. Second, in order to privately maintain the prices, it suffices to maintain a private count of the number of bids each good has received over the course of the auction; we can accomplish this task using private counters due to Dwork et al. [2010a], Chan et al. [2011]. Finally, it is possible to halt the algorithm early without significantly harming the quality of the final matching. By doing so, we re-

[2] Bidders do not bid on goods for which they have negative utility; in our case, $v_{ij} \in [0, 1]$.

duce the number of bids from each bidder, enabling us to bound the sensitivity of the bid counters, reducing the amount of noise needed for privacy.

The result is an algorithm that converges to a matching together with prices that form an approximate Walrasian equilibrium. We complete our analysis by proving an approximate version of the first welfare theorem, which shows that the matching has high weight.

The algorithm of Kelso and Crawford [1982] extends to the general allocation problem when players have gross substitute preferences, and our private algorithm does as well. We note that this class of preferences is the natural limit of our approach, which makes crucial use of equilibrium prices as a coordinating device: in general, when agents have valuations over bundles of goods that do not satisfy the gross substitutes condition, Walrasian equilibrium prices may not exist.

We first state our main result informally in the special case of max-matchings, which we prove in Section 4.3. We prove our more general theorem for allocation problems with gross substitutes preferences in Section 4.4. Here, privacy is protected with respect to a single agent i changing their valuations v_{ij} for possibly *all* goods j.

Theorem 4.1.1 (Informal)**.** *Suppose there are n agents and k types of goods, with each with s identical copies. There is a computationally efficient ϵ-joint differentially private algorithm which computes a matching of weight* $\mathrm{OPT} - \alpha n$ *as long as*

$$s \geq O\left(\frac{1}{\alpha^3 \epsilon} \cdot \mathrm{polylog}\left(n, k, \frac{1}{\alpha}\right)\right).$$

For certain parameter ranges, the welfare guarantee can be improved to $(1-\alpha)\mathrm{OPT}$.

Our algorithms actually work in a privacy model that is stronger than joint differential privacy, called the *billboard model*. We can view the algorithm as a mechanism that posts the prices publicly on a *billboard* as a differentially private signal such that every player can deduce what object they should be matched to just from their own private information and the contents of the billboard. As we show, algorithms in the billboard model automatically

satisfy joint differential privacy.

Furthermore, we view implementations in the billboard model as preferable to arbitrary jointly differentially private implementations. Algorithms in the billboard model only need the ability to publish sanitized messages to all players, and do not need a secure channel to communicate the mechanisms' output to each player (though of course, there still needs to be a secure channel from the player to the mechanism). The previous work by McSherry and Mironov [2009] and some of the results by Gupta et al. [2010] can be viewed as existing examples of algorithms in the billboard model.

In Section 4.5, we complement our positive results with lower bounds showing that our results are qualitatively tight. Not only is the problem impossible to solve under the standard differential privacy, assuming multiple copies of each good is also necessary to get any non-trivial solution even under *joint* differential privacy.

Theorem 4.1.2 (Informal)**.** *No joint differentially private algorithm can compute matchings of weight greater than* $\mathrm{OPT} - \alpha n$ *on instances in which there are* n *agents and* s *copies of each good, when*

$$s \leq O\left(\frac{1}{\sqrt{\alpha}}\right).$$

In particular, no algorithm can compute matchings of weight $\mathrm{OPT} - o(n)$ on instances for which the supply $s = O(1)$. In addition, we show that when goods have supply only $s = O(1)$, it is not even possible to compute the equilibrium prices privately under standard differential privacy. Our lower bounds are all reductions to database reconstruction attacks. Our technique for proving this lower bound may be of general interest, as the construction may be useful for other lower bounds for joint differential privacy.

4.1.2. Related Work

The privacy of our algorithms relies on work by Dwork et al. [2010a] and Chan et al. [2011], who show how to release a running count of a stream of bits under *continual ob-*

servation—i.e., report the count as the stream is revealed, provide high accuracy at every point in time, and keep the transcript differentially private.

Much work in differential privacy has focused on answering numeric valued queries on a private dataset. In contrast, work on private combinatorial optimization problems has been sporadic (e.g., Nissim et al. [2007], Gupta et al. [2010]). Part of the challenge is that many combinatorial optimization problems, including the allocation problems we consider in this chapter, are impossible to solve under differential privacy. To sidestep this problem, we employ the solution concept of *joint differential privacy*. First formalized by Kearns et al. [2014], similar ideas are present in the vertex and set-cover algorithms of Gupta et al. [2010], the private recommendation system of McSherry and Mironov [2009], and the analyst private data analysis algorithms of Dwork et al. [2012], Hsu et al. [2013].

Our algorithm is inspired by Kelso and Crawford [1982], who study the problem of matching *firms* to *workers* when the firms have preferences that satisfy the *gross substitutes* condition. They give an algorithm based on simulating simultaneous ascending auctions that converge to *Walrasian equilibrium prices* and a corresponding matching. In some respect, this approach does not generalize to more general valuations: Gul and Stacchetti [1999] show that gross substitutes preferences are precisely the set of preferences for which Walrasian equilibrium prices are guaranteed to exist.

Our work is closely related to recent work on computing various kinds of equilibrium or efficient outcomes under the constraint of joint differential privacy (e.g., correlated equilibrium [Kearns et al., 2014], Nash equilibrium [Rogers and Roth, 2014], and minmax equilibrium [Hsu et al., 2013]). In particular, our work establishes the billboard model as a basic paradigm for designing algorithms under joint differential privacy, which serves as a building blocks for later works, including [Cummings et al., 2015, Rogers et al., 2015, Kannan et al., 2015, Cummings et al., 2016].

4.2. Preliminaries

4.2.1. The Allocation Problem

We consider allocation problems defined by a set of goods G, and a set of n agents $[n]$. Each agent $i \in [n]$ has a *valuation function* $v_i : 2^G \to [0,1]$ mapping bundles of goods to values. A *feasible allocation* is a collection of sets $S_1, \ldots, S_n \subseteq G$ such that $S_i \cap S_j = \emptyset$ for each $i \neq j$: i.e., a partition of goods among the agents. The *social welfare* of an allocation $S_1, \ldots, S_n$ is $\sum_{i=1}^{n} v_i(S_i)$, the sum of the agent's valuations for the allocation; we are interested in finding allocations which maximize this quantity. Given an instance of an allocation problem, we write $\text{OPT} = \max_{S_1,\ldots,S_n} \sum_{i=1}^{n} v_i(S_i)$ to denote the social welfare of the optimal feasible allocation.

A particularly simple valuation function is a *unit demand valuation*, where bidders demand at most one item. Such valuation functions take the form $v_i(S) = \max_{j \in S} v_i(\{j\})$ and can be specified by numbers $v_{i,j} = v_i(\{j\}) \in [0,1]$, which represent the value that bidder i places on good j. When bidders have unit demand valuations, the allocation problem corresponds to computing a maximum weight matching in a bipartite graph.

Our results will also hold for *gross substitute valuations*, which include unit demand valuations as a special case. Informally, for gross substitute valuations, any set of goods S' that are in a most-demanded bundle at some set of prices p remain in a most-demanded bundle if the prices of *other* goods are raised, keeping the prices of goods in S' fixed. Gross substitute valuations are a standard class of valuation functions: they are a strict subclass of submodular functions, and they are precisely the valuation functions with Walrasian equilibria in markets with indivisible goods [Gul and Stacchetti, 1999]. Two other simple examples of gross substitute valuations are (1) *additive functions*, which takes the form $v(S) = \sum_{j \in S} v(\{j\})$ and (2) *symmetric submodular functions*, such that $v(S) = f(|S|)$ for some monotone concave function $f : \mathbb{R}_+ \to \mathbb{R}_+$.

To give the formal definition, we will need some notation. Given a vector of prices $\{p_g\}_{g \in G}$, the (quasi-linear) *utility* that player i has for a bundle of goods S_i is defined to be $u_i(S_i, p) = v_i(S_i) - \sum_{j \in S_i} p_j$.[3]

Given a vector of prices p, for each agent i we can define the set of *most demanded bundles*: $\omega(p) = \arg\max_{S \subseteq G} u_i(S, p)$. Given two price vectors p, p', we write $p \preceq p'$ if $p_g \leq p'_g$ for all g.

Definition 4.2.1. *A valuation function $v_i : 2^G \to [0,1]$ satisfies the* gross substitutes condition *if for every two price vectors $p \preceq p'$ and for every bundle $S \in \omega(p)$, if $S' \subseteq S$ satisfies $p'_g = p_g$ for every $g \in S'$, then there is a bundle $S^* \in \omega(p')$ with $S' \subseteq S^*$.*

Finally, we will typically consider markets with multiple copies of each type of good. Two goods $g_1, g_2 \in G$ are *identical* if for every bidder i and for every bundle $S \subseteq G$, $v_i(S \cup \{g_1\}) = v_i(S \cup \{g_2\})$: i.e., the two goods are indistinguishable according to every valuation function. Formally, we say that a set of goods G consists of k *types* of goods with s *supply* if there are k representative goods $g_1, \ldots, g_k \in G$ such that every good $g' \in G$ is identical to one of $g_1, \ldots, g_k$, and for each representative good g_i, there are s goods identical to g_i in G. For simplicity of presentation we will assume that the supply of each good is the same, but this is not necessary; all of our results continue to hold when the supply s denotes the *minimum* supply of any type of good.

4.2.2. Joint Differential Privacy

Suppose agents have valuation functions v_i from a class of functions C. A database $D \in C^n$ is a vector of valuation functions, one for each of the n bidders. Two databases D, D' are *i-neighbors* if they differ in only their i'th index: that is, if $D_j = D'_j$ for all $j \neq i$. If two databases D, D' are i-neighbors for some i, we say that they are *neighboring databases*.

When the range of a mechanism is also a vector with n components (e.g., $\mathcal{R} = (2^G)^n$), we

[3] This is a natural definition of utility if agents must pay for the bundles they buy at the given prices. In this chapter, we are concerned with the purely algorithmic allocation problem, so our algorithm will not actually charge prices. However, prices will be a convenient abstraction throughout our work.

can define *joint differential privacy*: this requires that simultaneously for all i, the *joint* distribution on outputs given to players $j \neq i$ is differentially private in the input of agent i. Given a vector $x = (x_1, \ldots, x_n)$, we write $x_{-i} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ to denote the vector of length $n-1$ which contains all coordinates of x except the i'th coordinate.

Definition 4.2.2 (Kearns et al. [2014]). *An algorithm $\mathcal{M} : C^n \to (2^G)^n$ is (ϵ, δ)-*joint differentially private *if for every i, for every pair of i-neighbors $D, D' \in C^n$, and for every subset of outputs $S \subseteq (2^G)^{n-1}$,*

$$\Pr[\mathcal{M}(D)_{-i} \in S] \leq e^{\epsilon} \Pr[\mathcal{M}(D')_{-i} \in S] + \delta.$$

*If $\delta = 0$, we say that $\mathcal{M}$ is ϵ-*joint differentially private.

Note that this is still an extremely strong definition that protects i from arbitrary coalitions of adversaries—it weakens the constraint of differential privacy only in that the output given specifically to agent i may be sensitive in the input of agent i.

4.2.3. Differentially Private Counters

The central tool in our algorithm is the private streaming counter proposed by Chan et al. [2011] and Dwork et al. [2010a]. Given a bit stream $\sigma = (\sigma_1, \ldots, \sigma_T) \in \{0,1\}^T$, a streaming counter $\mathcal{M}(\sigma)$ releases an approximation to $c_\sigma(t) = \sum_{i=1}^{t} \sigma_i$ at every time step t. The counters release accurate approximations to the running count at every time step.

Definition 4.2.3. *A streaming counter $\mathcal{M}$ is (α, β)-*useful *if with probability at least $1-\beta$, for each time $t \in [T]$,*

$$|\mathcal{M}(\sigma)(t) - c_\sigma(t)| \leq \alpha.$$

For the rest of this chapter, let $\mathsf{Counter}(\epsilon, T)$ denote the Binary mechanism of Chan et al. [2011], instantiated with parameters ϵ and T. The mechanism produces a monotonically increasing count, and satisfies the following accuracy guarantee. Further details may be found in Section 4.6.

Theorem 4.2.4 (Chan et al. [2011]). *For $\beta > 0$, $\mathsf{Counter}(\epsilon, T)$ is ϵ-differentially private with*

respect to a single bit change in the stream, and (α, β)*-useful for*

$$\alpha = \frac{2\sqrt{2}}{\epsilon} \ln(2/\beta) \log(T)^{5/2}.$$

4.3. Private Max-Weight Matching

In this section, we study the special case of unit demand valuations. Though our later algorithm for gross substitutes valuations generalizes this case, we first present our algorithm in this simpler setting to highlight the key features of our approach.

Consider a matching market with n bidders and k different types of goods, where each good has supply s and bidder i has valuation $v_{ij} \in [0,1]$ for good j. Some agents may not end up being matched to a good: to simplify notation, we will say that unmatched agents are matched to a special dummy good $\perp$.

To reach a maximum weight matching, we first aim to privately compute prices $p \in [0,1]^k$ and an allocation of the goods $\mu\colon [n] \to [k] \cup \{\perp\}$ such that *most* bidders are matched with their *approximately* favorite goods *given the prices* and each over-demanded good almost clears, where a good is *over-demanded* if its price is strictly positive.[4] We will show that if we can achieve this intermediate goal, then in fact we have computed an approximate maximum weight matching.

Definition 4.3.1. *A price vector* $p \in [0,1]^k$ *and an assignment* $\mu\colon [n] \to [k] \cup \{\perp\}$ *of bidders to goods is an* (α, β, ρ)-approximate matching equilibrium *if:*

1. *all but a* ρ *fraction of bidders* i *are matched to an* α*-approximate favorite good: i.e.,*$v_{i\mu(i)} - p_{\mu(i)} \geq v_{ij} - p_j - \alpha$ *for every good* j*, for at least* $(1-\rho)n$ *bidders* i *(we call these bidders* satisfied*);*

2. *the number of bidders assigned to any type of good is below its supply; and*

[4] This is the notion of approximate Walrasian equilibrium we will use.

3. *each over-demanded good clears except for at most β supply.*

4.3.1. Overview of the Algorithm

Our algorithm takes the valuations as input, and outputs a trajectory of prices that can be used by the agents to figure out what they are matched to. For the presentation, we will sometimes speak as if the bidders are performing some action, but this actually means that our algorithm simulates the actions of the bidders internally—the actual agents do not interact with our algorithm.

Algorithm 4 (PMatch) is a variant of a *deferred acceptance* algorithm first proposed and analyzed by Kelso and Crawford [1982], which runs k simultaneous ascending price auctions: one for each type of good. At any given moment each type of good has a *proposal price* p_j. In a sequence of rounds where the algorithm passes through each bidder once in some fixed, publicly known order, unsatisfied bidders bid on a good that maximizes their utility at the current prices: that is, a good j that maximizes $v_{ij} - p_j$. (This is the **Propose** function.)

The s most recent bidders for a type of good are tentatively matched to that type of good; these are the current *high bidders*. A bidder tentatively matched to a good with supply s becomes unmatched once the good receives s subsequent bids; we say this bidder has has been *outbid*. Every s bids on a good increases its price by a fixed increment α. Bidders keep track of which good they are matched to, if any, and determine whether they are currently matched or unmatched by looking at a count of the number of bids received by the last good they bid on.

To implement this algorithm privately, we count the number of bids each good has received using private counters. Unsatisfied bidders can infer the prices of all goods based on the number of bids each has received, and from this information, they determine their favorite good at the given prices. Their bid is recorded by sending the bit 1 to the appropriate counter. (This is the **Bid** function.) Matched bidders store the reading of the bid counter

on the good they are matched to at the time that they last bid (in the variable d_i); when the counter ticks s bids past this initial count, bidders conclude that they have been outbid and become unmatched. The final matching is communicated implicitly: the real agents observe the full published price trajectory and simulate what good they would have been matched to had they bid according to the published prices.

Since the private counters are noisy, more than s bidders may be matched to a good. To maintain feasibility, the algorithm reserves some supply m: i.e., it treats the supply of each good as $s - m$, rather than s. The *reserved supply* m is used to satisfy the demand of excess bidders who believe themselves to be matched to a good; the number of such bidders is at most s, with high probability.

Our algorithm stops as soon as fewer than ρn bidders place bids in a round. We show that this early stopping condition does not significantly harm the welfare guarantee of the matching, while it substantially reduces the *sensitivity* of the counters: no bidder ever bids more than $O(1/(\alpha\rho))$ times in total. Crucially, this bound is independent of both the number of types of goods k and the number of bidders n. By stopping early, we greatly improve the accuracy of the prices since the amount we must perturb the bid counts to protect privacy increases with the sensitivity of the counters.

To privately implement the stopping condition, the algorithm maintains a separate counter (counter_0) which counts the number of unsatisfied bidders throughout the run of the algorithm. At the end of each round, bidders who are unsatisfied will send the bit 1 to this counter, while bidders who are matched will send the bit 0. If this counter increases by less than roughly ρn in any round, the algorithm halts. (This is the **CountUnsatisfied** function.)

4.3.2. Privacy Analysis

In this section, we show that the allocation output by our algorithm satisfies joint differential privacy with respect to any single bidder changing *all* of their valuations. We will use

Algorithm 4 PMatch$(\alpha, \rho, \varepsilon)$

Input: Bidders' valuations $(\{v_{1j}\}_{j=1}^{k}, \ldots, \{v_{nj}\}_{j=1}^{k})$
Initialize: for bidder i **and good** j**,**

$$T = \frac{8}{\alpha\rho}, \qquad \epsilon' = \frac{\epsilon}{2T}, \qquad E = \frac{2\sqrt{2}}{\epsilon'}(\log nT)^{5/2}\log\left(\frac{4k}{\gamma}\right), \qquad m = 2E + 1$$

$$\text{counter}_j = \textbf{Counter}(\epsilon', nT) \qquad p_j = c_j = 0,$$

$$\mu(i) = \emptyset, \qquad d_i = 0, \qquad \text{counter}_0 = \textbf{Counter}(\epsilon', nT)$$

Propose T times; **Output:** prices p and allocation μ.

Propose:
for all bidders i **do**
  if $\mu(i) = \emptyset$ **then**
    Let $\mu(i) \in \text{argmax}_j\, v_{ij} - p_j$, breaking ties arbitrarily
    if $v_{i\mu(i)} - p_{\mu(i)} \leq 0$ **then**
      Let $\mu(i) := \perp$ and **Bid**$(\mathbf{0})$.
    else Save $d_i := c_{\mu(i)}$ and **Bid**$(\mathbf{e}_{\mu(i)})$.
  else Bid$(\mathbf{0})$
CountUnsatisfied

Bid: On input bid vector $\mathbf{b}$
for all goods j **do**
  Feed $\mathbf{b}_j$ to counter_j.
  Update count $c_j := \text{counter}_j$.
  if $c_j \geq (p_j/\alpha + 1)(s - m)$ **then**
    Update $p_j := p_j + \alpha$.

CountUnsatisfied:
for all bidders i **do**
  if $\mu(i) \neq \perp$ and $c_{\mu(i)} - d_i \geq s - m$ **then**
    Feed 1 to counter_0.
    Let $\mu(i) := \perp$.
  else Feed 0 to counter_0.
if counter_0 increases by less than $\rho n - 2E$ **then**
  Halt and output μ.

a basic but useful lemma: to show joint differential privacy, it is sufficient to show that the output sent to each agent i is an arbitrary function of (i) some global signal that is computed under the standard constraint of differential privacy, and (ii) agent i's private data.

We call this model the *billboard model*: agents can compute their output by combining a common signal—as if posted on a public billboard—with their own private data. In our case, the price history over the course of the auction is the differentially private message posted on the billboard. Combined with their personal private valuation, each agent can compute their personal allocation.

Lemma 4.3.2 (Billboard Lemma)**.** *Suppose $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ is (ϵ, δ)-differentially private. Consider any set of functions $f_i : \mathcal{D}_i \times \mathcal{R} \to \mathcal{R}'$, where $\mathcal{D}_i$ is the portion of the database containing i's data. The composition $\{f_i(\Pi_i D, \mathcal{M}(D))\}$ is (ϵ, δ)-joint differentially private, where $\Pi_i : \mathcal{D} \to \mathcal{D}_i$ is the projection to i's data.*

Proof. We need to show that for any agent i, the view of the other agents is (ϵ, δ)-differentially private when i's private data is changed. Suppose databases D, D' are i-neighbors, so $\Pi_j D = \Pi_j D'$ for $j \neq i$. Let $\mathcal{R}_{-i}$ be a set of possible outputs to the bidders besides i. Let $\mathcal{R}^* = \{r \in \mathcal{R} \mid \{f_j(\Pi_j D, r)\}_{-i} \in \mathcal{R}_{-i}\}$. Then, we need

$$\begin{aligned} \Pr[\{f_j(\Pi_j D, \mathcal{M}(D))\}_{-i} \in \mathcal{R}_{-i}] &\leq e^{\epsilon} \Pr[\{f_j(\Pi_j D', \mathcal{M}(D'))\}_{-i} \in \mathcal{R}_{-i}] + \delta \\ &= e^{\epsilon} \Pr[\{f_j(\Pi_j D, \mathcal{M}(D'))\}_{-i} \in \mathcal{R}_{-i}] + \delta \\ \text{so } \Pr[\mathcal{M}(D) \in \mathcal{R}^*] &\leq e^{\epsilon} \Pr[\mathcal{M}(D') \in \mathcal{R}^*] + \delta, \end{aligned}$$

but this is true since $\mathcal{M}$ is (ϵ, δ)-differentially private. □

Theorem 4.3.3. *The sequence of prices and counts of unsatisfied bidders released by* $\mathsf{PMatch}(\alpha, \rho, \varepsilon)$ *satisfies ϵ-differential privacy.*

Sketch. We give a rough intuition here, and defer the full proof to Section 4.6. Note that the prices can be computed from the noisy counts, so it suffices to show that the counts are private. Since no bidder bids more than $T \approx 1/(\alpha\rho)$ times in total, the *total* sensitivity of the k price streams to a single bidder's valuations is only $O(1/(\alpha\rho))$ (independent of k) even though a single bidder could in principle bid $\Omega(1/\alpha)$ times on each of the k streams. Hence the analysis of these k simultaneously running counters is akin to the analysis of answering

histogram queries, multiple queries whose joint sensitivity is substantially smaller than the sum of their individual sensitivities.

By setting the counter for each good with privacy parameter $\epsilon' = \epsilon/2T$, the prices are $\epsilon/2$ differentially private. By the same reasoning, setting the unsatisfied bidders counter with privacy parameter $\epsilon' = \epsilon/2T$ also makes the unsatisfied bidders count $\epsilon/2$ private. Thus, these outputs together satisfy ϵ-differential privacy.

While this intuition is roughly correct, there are some technical details. Namely, Chan et al. [2011] show privacy for a single counter with sensitivity 1 on a non-adaptively chosen stream. Since intermediate outputs (i.e., prices) from our counters will affect the future streams (i.e., future bids) for other counters, this is not sufficient. In fact, it is possible to prove privacy for multiple counters running on adaptively chosen streams, where the privacy parameter depends only on the joint sensitivity of the streams and not on the number of streams. We show this result using largely routine arguments; details can be found in Section 4.6. □

Theorem 4.3.4. PMatch$(\alpha, \rho, \varepsilon)$ *is* ϵ*-joint differentially private.*

Proof Sketch. Note that given the sequence of prices, counts of unsatisfied bidders, and the private valuation of any bidder i, the final allocation to that bidder can be computed by simulating the sequence of bids made by bidder i, since the bids are determined by the price when bidder i is slotted to bid and by whether the auction has halted or not. Bidder i's final allocation is simply the final item that i bids on. The prices and halting condition are computed as a deterministic function of the noisy counts, which are ϵ-differentially private by Theorem 4.2.4. So, Lemma 4.3.2 shows that PMatch is ϵ-joint differentially private. □

4.3.3. Utility Analysis

In this section, we compare the weight of the matching produced by PMatch with OPT. As an intermediate step, we first show that the resulting matching paired with the prices computed by the algorithm forms an approximate matching equilibrium. We next show that any such matching must be an approximately max-weight matching.

The so-called *first welfare theorem* from general equilibrium theory guarantees that an exact (i.e., a $(0,0,0)$-) matching equilibrium gives an exact maximum weight matching. Compared to this ideal, PMatch loses welfare in three ways. First, a ρ fraction of bidders may end up unsatisfied. Second, the matched bidders are not necessarily matched to goods that maximize their utility given the prices, but only to goods that do so approximately (up to additive α). Finally, the auction sets aside part of the supply to handle over-allocation from the noisy counters. This reserved supply may end up unused, say, if the counters are accurate or actually under-allocate. In other words, we compute an equilibrium of a market with reduced supply, so our welfare guarantee holds if the supply s is significantly larger than the necessary reserved supply m.

The key performance metric is *how much* supply is needed to achieve a given welfare approximation in the final matching. On the one hand, we will show later that the problem is impossible to solve privately if $s = O(1)$ (Section 4.5). On the other hand, the problem is trivial if $s \geq n$: agents can be simultaneously matched to their favorite good with no coordination; this allocation is trivially both optimal and private. Our algorithm will achieve positive results in the intermediate supply range, when $s \geq \text{polylog}(n)$.

Theorem 4.3.5. *Let $\alpha > 0$, and μ be the matching computed by* $\mathsf{PMatch}(\alpha/3, \alpha/3, \varepsilon)$. *Let* OPT *denote the weight of the optimal matching. Then, if the supply satisfies*

$$s \geq \frac{16E' + 4}{\alpha} = O\left(\frac{1}{\alpha^3 \epsilon} \cdot \text{polylog}\left(n, k, \frac{1}{\alpha}, \frac{1}{\gamma}\right)\right),$$

and $n > s$, the matching μ has social welfare at least $\mathrm{OPT} - \alpha n$ *with probability* $\geq 1 - \gamma$*, where*

$$E' = \frac{288\sqrt{2}}{\alpha^2 \epsilon}\left(\log\left(\frac{72n}{\alpha^2}\right)\right)^{5/2} \log\left(\frac{4k}{\gamma}\right).$$

Remark 4.3.6. *Our approximation guarantee here is* additive*. Later in this section, we show that if we are in the* unweighted *case—$v_{ij} \in \{0, 1\}$—we can find a matching μ with welfare at least* $(1-\alpha)\mathrm{OPT}$*. This* multiplicative *guarantee is unusual for a differentially private algorithm.*

The proof follows from the following lemmas.

Lemma 4.3.7. *We call a bidder who wants to continue bidding* unsatisfied*; otherwise bidder i is* satisfied*. At termination of* $\mathsf{PMatch}(\alpha, \rho, \varepsilon)$*, all satisfied bidders i are matched to a good $\mu(i)$ such that*

$$v_{i,\mu(i)} - p_{\mu(i)} \geq \max_j (v_{i,j} - p_j) - \alpha.$$

Proof. Fix any satisfied bidder i matched to $j^* = \mu(i)$. At the time that bidder i last bid on j^*, by construction, $v_{ij^*} - p_{j^*} \geq \max_j (v_{ij} - p_j)$. Since i remained matched to j^*, its price could only have increased by at most α, and the prices of other goods $j \neq j^*$ could only have increased. Hence, at completion of the algorithm,

$$v_{i,\mu(i)} - p_{\mu(i)} \geq \max_j (v_{ij} - p_j) - \alpha$$

for all matched bidders i. □

Lemma 4.3.8. *Assume all counters have error at most E throughout the run of* $\mathsf{PMatch}(\alpha, \rho, \varepsilon)$*. Then the number of bidders assigned to any good is at most s and each over-demanded good clears except for at most β supply, where*

$$\beta = 4E + 1 = O\left(\frac{1}{\alpha\rho\epsilon} \cdot \mathrm{polylog}\left(\frac{1}{\alpha}, \frac{1}{\rho}, \frac{1}{\gamma}, k, n\right)\right).$$

Proof. Since the counter for each under-demanded good never exceeds $s - m$, we know that each under-demanded good is matched to no more than $s - m + E < s$ bidders. Consider any

counter c for an over-demanded good. Let t be a time step such that

$$c(nT) - c(t+1) \leq s - m < c(nT) - c(t),$$

where $c(t)$ denotes the output of the counter at time t. Note that the bidders who bid after time t are the only bidders matched to this good at time nT. Let σ be the true bid stream for this good and let the sum of bids in σ up to time t be $h(\sigma, t)$. Then, the total number of bidders allocated to this good at time nT is

$$\begin{aligned} h(\sigma, nT) - h(\sigma, t) &\leq h(\sigma, nT) - h(\sigma, t+1) + 1 \\ &\leq (c(nT) + E) - (c(t+1) - E) + 1 \\ &\leq s - m + 2E + 1 = s. \end{aligned}$$

Similarly, we can lower bound the number of bidders allocated to this good:

$$\begin{aligned} h(\sigma, nT) - h(\sigma, t) &= (h(\sigma, nT) - c(nT)) + (c(nT) - c(t)) + (c(t) - h(\sigma, t)) \\ &> s - m - 2E > s - 4E - 1. \end{aligned}$$

Therefore, every over-demanded good clears except for at most $\beta = 4E + 1$ supply, which gives

$$\begin{aligned} \beta &= \frac{16\sqrt{2}}{\alpha\rho\epsilon}\left(\log\left(\frac{6n}{\alpha\rho}\right)\right)^{5/2}\log\left(\frac{4k}{\gamma}\right) + 1 \\ &= O\left(\frac{1}{\alpha\rho\epsilon}\cdot \text{polylog}\left(\frac{1}{\alpha}, \frac{1}{\rho}, \frac{1}{\gamma}, k, n\right)\right). \end{aligned}$$

□

Lemma 4.3.9. *Assume all counters have error at most E throughout the run of* $\mathsf{PMatch}(\alpha, \rho, \varepsilon)$. *Then at termination all but a ρ fraction of bidders are satisfied, so long as $s \geq 8E + 1$ and $n \geq 8E/\rho$.*

Proof. First, we show that the total number of bids made over the course of the algorithm is bounded by $3n/\alpha$. We account separately for the under-demanded goods (those with price 0 at the end of the auction) and the over-demanded goods (those with positive price). For the under-demanded goods, since their prices remain 0 throughout the algorithm, their corresponding noisy counters never exceeded $(s - m)$. Since no bidder is ever unmatched after having been matched to an under-demanded good, the set of under-demanded goods can receive at most one bid from each agent; together the under-demanded goods can receive at most n bids.

Next, we account for the over-demanded goods. Note that the bidders matched to these goods are precisely the bidders who bid within $s - m$ ticks of the final counter reading. Since the counter has error bounded by E at each time step, this means at least $s - m - 2E$ bidders end up matched to each over-demanded good. Since no agent can be matched to more than one good there can be at most $n/(s - m - 2E)$ over-demanded goods in total.

Likewise, we can account for the number of price increases per over-demanded good. Prices never rise above 1 (because any bidder would prefer to be unmatched than to be matched to a good with price higher than 1). Therefore, since prices are raised in increments of α, the price of every over-demanded good increases at most $1/\alpha$ times. Since there can be at most $(s - m + 2E)$ bids between each price update (again, corresponding to $s - m$ ticks of the counter), the total number of bids received by all of the over-demanded goods in total is at most

$$\frac{n}{s - m - 2E} \cdot \frac{1}{\alpha} \cdot (s - m + 2E).$$

Since each bid is either on an under or over-demanded good, we can upper bound the *total* number of bids B by

$$B \leq n + \frac{n}{\alpha}\left(\frac{s - m + 2E}{s - m - 2E}\right) = \frac{n}{\alpha}\left(\alpha + \frac{s - m + 2E}{s - m - 2E}\right).$$

The algorithm sets the reserved supply to be $m = 2E + 1$ and by assumption, we have

$s \geq 8E + 1$. Since we are only interested in cases where $\alpha < 1$, we conclude

$$B \leq n + \frac{n}{\alpha}\left(\frac{s - m + \alpha}{s - m - \alpha}\right) \leq \frac{3n}{\alpha}. \tag{4.1}$$

Now, consider the halting condition. Either the algorithm halts early, or it does not. We claim that at termination, at most ρn bidders are unsatisfied. The algorithm halts early if at any round of **CountUnsatisfied**, counter_0 (which counts the number of unsatisfied bidders) increases by less than $\rho n - 2E$, when there are at most $\rho n - 2E + 2E = \rho n$ unsatisfied bidders.

Otherwise, suppose the algorithm does not halt early. At the start of each round there must be at least $\rho n - 4E$ unsatisfied bidders. Not all of these bidders must bid during the **Propose** round since price increases while they are waiting to bid might cause them to no longer demand any item, but this only happens if bidders prefer to be unmatched at the new prices. Since prices only increase, these bidders remain satisfied for the rest of the algorithm. If the algorithm runs for R rounds and there are B true bids,

$$B \geq R(\rho n - 4E) - n.$$

Combined with our upper bound on the number of bids (Equation (4.1)) and our assumption $\rho n \geq 8E$, we can upper bound the number of rounds R:

$$R \leq \left(\frac{3n}{\alpha} + n\right) \cdot \left(\frac{1}{\rho n - 2E}\right) \leq \left(\frac{4n}{\alpha}\right)\left(\frac{2}{\rho n}\right) = \frac{8}{\alpha\rho} := T.$$

Thus, running the algorithm for T rounds leads to all but ρn bidders satisfied. □

Lemma 4.3.10. *With probability at least* $1 - \gamma$, $\mathsf{PMatch}(\alpha, \rho, \varepsilon)$ *computes an* (α, β, ρ)*-matching equilibrium, where*

$$\beta = 4E + 1 = O\left(\frac{1}{\alpha\rho\epsilon} \cdot \text{polylog}\left(\frac{1}{\alpha}, \frac{1}{\rho}, \frac{1}{\gamma}, k, n\right)\right)$$

so long as $s \geq 8E + 1$ *and* $n \geq 8E/\rho$.

Proof. By Theorem 4.2.4, counter_0 is $(\lambda_1, \gamma/2)$-useful, and each of the k good counters is $(\lambda_2, \gamma/2)$-useful, where

$$\lambda_1 = \frac{2\sqrt{2}}{\epsilon'}(\log nT)^{5/2}\log\left(\frac{4}{\gamma}\right) \quad \text{and} \quad \lambda_2 = \frac{2\sqrt{2}}{\epsilon'}(\log nT)^{5/2}\log\left(\frac{4k}{\gamma}\right).$$

Since we set $E = \lambda_2 > \lambda_1$, all counters are $(E, \gamma/2)$-useful, and thus with probability at least $1 - \gamma$, all counters have error at most E. The theorem then follows by Lemmas 4.3.7 to 4.3.9. □

With these lemmas in place, it is straightforward to prove the welfare theorem (Theorem 4.3.5).

Theorem 4.3.5. By Lemma 4.3.10, $\mathsf{PMatch}(\alpha/3, \alpha/3, \epsilon)$ calculates a matching μ that is an $(\alpha/3, \beta, \alpha/3)$-approximate matching equilibrium with probability at least $1 - \gamma$, where $\beta = 4E' + 1$. Let p be the prices at the end of the algorithm, and S be the set of satisfied bidders. Let μ^* be the optimal matching achieving welfare $\sum_{i=1}^{n} v_{i,\mu^*(i)} = \text{OPT}$. We know that $|S| \geq (1 - \alpha/3)n$ and

$$\sum_{i \in S}(v_{i\mu(i)} - p_{\mu(i)}) \geq \sum_{i \in S}(v_{i\mu^*(i)} - p_{\mu^*(i)}) - \alpha|S|/3.$$

Let N_j^* and N_j be the number of goods of type j matched in μ^* and μ respectively, and let G be the set of over-demanded goods at prices p.

Since each over-demanded good clears except for at most β supply, and since each of the n agents can be matched to at most one good, we know that $|G| \leq n/(s - \beta)$. Since the true supply in OPT is at most s, we also know that $N_j^* - N_j \leq \beta$ for each over-demanded good j.

Finally, by definition, under-demanded goods j have price $p_j = 0$. So,

$$\begin{aligned}\sum_{i\in S} v_{i\mu^*(i)} - \sum_{i\in S} v_{i\mu(i)} &\le \sum_{i\in S} p_{\mu^*(i)} - \sum_{i\in S} p_{\mu(i)} + \alpha|S|/3 \\ &= \sum_{j\in G} p_j(N_j^* - N_j) + \alpha|S|/3 \\ &\le \sum_{j\in G} \beta + \alpha|S|/3 \le \frac{n\beta}{s-\beta} + \alpha|S|/3.\end{aligned}$$

If $s \ge 4\beta/\alpha$, the first term is at most $\alpha n/3$. Finally, since all but $\alpha n/3$ of the bidders are matched with goods in S, and their valuations are upper bounded by 1, so

$$\sum_i v_{i\mu(i)} - \sum_i v_{i\mu^*(i)} \le \alpha n/3 + \alpha|S|/3 + \alpha n/3 \le \alpha n.$$

Unpacking β from Lemma 4.3.10, we get the stated bound on supply. □

4.3.4. Multiplicative Approximation to Welfare

In certain situations, a slight variant of PMatch (Algorithm 4) can give a multiplicative welfare guarantee. In this section, we will assume that the value of the maximum weight matching OPT is known; it is often possible to privately estimate this quantity to high accuracy. Our algorithm is PMatch with a different halting condition: rather than count the number of unmatched bidders each round, count the number of bids per round. Once this count drops below a certain threshold, halt the algorithm.

More precisely, we use a function **CountBids** (Algorithm 5) in place of **CountUnsatisfied** in Algorithm 4.

Theorem 4.3.11. *Suppose bidders have valuations $\{v_{ij}\}$ over goods such that*

$$\min_{v_{ij}>0} v_{ij} \ge \lambda.$$

Algorithm 5 Modified Halting Condition CountBids

CountBids:
for all bidders i **do**
  if $\mu(i) \neq \perp$ and $c_{\mu(i)} - d_i \geq s - m$ **then**
    Let $\mu(i) := \emptyset$
  if i bid this round **then**
    Feed 1 to counter$_0$.
  else Feed 0 to counter$_0$.
if counter$_0$ increases by less than $\frac{\alpha OPT}{2\lambda} - 2E$ **then**
  Halt; For each i with $\mu(i) = \emptyset$, let $\mu(i) = \perp$

Then Algorithm 4, with

$$T = \frac{24}{\alpha^2}$$

rounds, using stopping condition **CountBids** *(Algorithm 5) in place of* **CountUnsatisfied** *and stopped once the total bid counter increases by less than*

$$\frac{\alpha \,\mathrm{OPT}}{2\lambda} - 2E$$

bids in a round, satisfies ϵ-joint differential privacy and outputs a matching that has welfare at least $O((1-\alpha/\lambda)\mathrm{OPT})$, so long as

$$s = \Omega\left(\frac{1}{\alpha^3\epsilon} \cdot \mathrm{polylog}\left(n, k, \frac{1}{\alpha}, \frac{1}{\gamma}\right)\right) \text{and} \qquad \mathrm{OPT} = \Omega\left(\frac{\lambda}{\alpha^3\epsilon} \cdot \mathrm{polylog}\left(n, k, \frac{1}{\alpha}, \frac{1}{\gamma}\right)\right).$$

Proof. Privacy follows exactly like Theorem 4.3.4. We first show that at termination, all but $\alpha\,\mathrm{OPT}/\lambda$ bidders are matched to an α-approximate favorite item. The analysis is very similar to Lemma 4.3.10. Note that every matched bidder is matched to an α-approximate favorite good, since it was an exactly favorite good at the time of matching, and the price increases by at most α. Thus, it remains to bound the number of unsatisfied bidders at termination.

Condition on all counters having error bounded by E at all time steps; by Theorem 4.2.4 and a union bound over counters, this happens with probability at least $1-\gamma$. Like above, we write $s' = s - m$ for the effective supply of each good. Let us first consider the case where

the algorithm stops early. If the total bid counter changes by less than $\frac{\alpha \mathrm{OPT}}{2\lambda} - 2E$, the true number of bids that round is at most

$$Q = \frac{\alpha \mathrm{OPT}}{2\lambda}.$$

We will upper bound the number of unsatisfied bidders at the end of the round. Note that the number of unsatisfied bidders at the end of the round is the number of bidders who have been rejected in the current round. Suppose there are N goods that reject bidders during this round. The total count on these goods must be at least

$$(s' - 2E) \cdot N - Q$$

at the start of the round, since each counter will increase by at most $2E$ due to error, and there were at most Q bids this round. By our conditioning, there were at least

$$(s' - 2E) \cdot N - Q - 2EN$$

bidders matched at the beginning of the round. Since bidders are only matched when their valuation is at least λ, and the optimal weight matching is OPT, at most $\frac{OPT}{\lambda}$ bidders can be matched at any time. Hence,

$$N \leq \left(\frac{\mathrm{OPT}}{\lambda} + Q\right) \cdot \frac{1}{s' - 4E}.$$

Then, the total number of bidders rejected this round is at most $2EN + Q$. Simplifying,

$$\begin{aligned} 2EN + Q &\leq \frac{2E}{s' - 4E} \cdot \left(\frac{\mathrm{OPT}}{\lambda} + Q\right) + Q \\ &\leq \left(\frac{6E}{s' - 4E}\right)\left(\frac{\mathrm{OPT}}{\lambda}\right) + \frac{\alpha \mathrm{OPT}}{2\lambda}. \end{aligned}$$

To make the first term at most $\frac{\alpha \mathrm{OPT}}{2\lambda}$, it suffices to take

$$\begin{aligned} \frac{6E}{s'-4E} &\leq \frac{\alpha}{2} \\ s' &\geq \frac{12E}{\alpha} + 4E \\ s &\geq \frac{12E}{\alpha} + 6E + 1, \end{aligned}$$

or $s \geq 18E/\alpha$. In this case, the algorithm terminates with at most $\frac{\alpha \mathrm{OPT}}{\lambda}$ unsatisfied bidders, as desired.

On the other hand, suppose the algorithm does not terminate early, the bid count increasing by at least $Q-2E$ every round. By our conditioning, this means there are at least $Q-4E$ bids each round; let us bound the number of possible bids.

Since bidders only bid if they have valuation greater than λ for a good, and since the maximum weight matching has total valuation OPT, at most OPT/λ bidders can be matched. Like before, we say goods are under-demanded or over-demanded: they either have final price 0, or positive final price.

There are at most OPT/λ true bids on the goods of the first type; this is because bidders are never rejected from these goods. Like before, write $s' = s - m$. Each counter of a over-demanded good shows s' people matched, so at least $s'-2E$ bidders end up matched. Thus, there are at most

$$\frac{\mathrm{OPT}}{\lambda(s'-2E)}$$

over-demanded goods. Each such good takes at most $s'+2E$ bids at each of $1/\alpha$ price levels. Putting these two estimates together, the total number of bids B is upper bounded by

$$B \leq \frac{\mathrm{OPT}}{\lambda} \cdot \left(1 + \frac{s'+2E}{s'-2E}\right) \leq \frac{6\,\mathrm{OPT}}{\lambda\alpha}$$

if $s' \geq 4E$, which holds since we are already assuming $s' \geq 4E + \frac{12E}{\alpha}$. Hence, we know the

number of bids is at most

$$T \cdot (Q - 4E) \le B \le \frac{6\,\mathrm{OPT}}{\lambda\alpha}$$
$$T \le \frac{6\,\mathrm{OPT}}{\lambda} \cdot \left(\frac{2\lambda}{\alpha\,\mathrm{OPT} - 8\lambda E}\right).$$

Assuming $\alpha\,\mathrm{OPT} \ge 16\lambda E$, we find $T \le 24/\alpha^2$.

With this choice of T, the supply requirement is

$$s \ge \frac{18E}{\alpha} = \Omega\left(\frac{1}{\alpha^3\epsilon} \cdot \mathrm{polylog}\left(n, k, \frac{1}{\alpha}, \frac{1}{\gamma}\right)\right). \tag{4.2}$$

Likewise, the requirement on OPT is

$$\mathrm{OPT} \ge \frac{16\lambda E}{\alpha} = \Omega\left(\frac{\lambda}{\alpha^3\epsilon} \cdot \mathrm{polylog}\left(n, k, \frac{1}{\alpha}, \frac{1}{\gamma}\right)\right).$$

Now, we can follow the analysis from Theorem 4.3.5 to bound the welfare. Suppose the algorithm produces a matching μ, and consider any other matching μ^*. For each bidder who is matched to an α-approximate favorite good,

$$v_{i\mu(i)} - p_{\mu(i)} \ge v_{i\mu^*(i)} - p_{\mu^*(i)} - \alpha.$$

Each such bidder is matched to a good with value at least λ, so there are at most OPT/λ such bidders. Summing over these bidders (call them S),

$$\sum_{i \in S} v_{i\mu(i)} - p_{\mu(i)} \ge \sum_{i \in S} v_{i\mu^*(i)} - p_{\mu^*(i)} - \frac{\alpha\,\mathrm{OPT}}{\lambda}.$$

Letting N_j, N_j^* be the number of goods of type j matched in μ, μ^* and rearranging,

$$\sum_{i \in S} v_{i\mu^*(i)} - v_{i\mu(i)} \le \sum_{j \in S} p_j (N_j^* - N_j) + \frac{\alpha\,\mathrm{OPT}}{\lambda}.$$

Exactly the same as in Theorem 4.3.5, each over-demanded good ($p_j > 0$) clears except for at most $\beta = 4E+1$ supply. Since at most $\frac{\text{OPT}}{\lambda}$ bidders can be matched, the number of goods with $p_j > 0$ is at most

$$\frac{\text{OPT}}{\lambda(s-\beta)}.$$

Like before, $N_j^* - N_j \leq \beta$. Since there are at most $\alpha\,\text{OPT}/\lambda$ bidders not in S and each has valuation in $[0,1]$, when summing over all bidders,

$$\sum_i v_{i\mu^*(i)} - v_{i\mu(i)} \leq \frac{\text{OPT}\,\beta}{\lambda(s-\beta)} + \frac{\alpha\,\text{OPT}}{\lambda} + \frac{\alpha\,\text{OPT}}{\lambda}.$$

The first term is at most $\alpha\,\text{OPT}/\lambda$ for $s \geq \beta(1+1/\alpha)$, when the algorithm calculates a matching with weight $O((1-\alpha/\lambda)\text{OPT})$. Since $\beta = 4E+1$, this reduces to the supply constraint Equation (4.2). □

Remark 4.3.12. *For a comparison with Theorem 4.3.5 and* PMatch, *consider the "unweighted" case where bidders have valuations in* $\{0,1\}$ *(i.e.,* $\lambda = 1$*). Note that both* PMatch *and the multiplicative version require the same lower bound on supply. Ignoring log factors,* PMatch *requires* $n = \tilde{\Omega}(1/\alpha^3\epsilon)$ *for an additive* αn *approximation, while Theorem 4.3.11 shows* $\text{OPT} = \tilde{\Omega}(1/\alpha^3\epsilon)$ *is necessary for a multiplicative* α*, hence additive* $\alpha\,\text{OPT}$*, approximation. Hence, Theorem 4.3.11 gives a stronger guarantee if* $\text{OPT} = \tilde{o}(n)$ *in the unweighted case, ignoring log factors.*

4.4. Extension to Gross Substitute Valuations

While Kelso and Crawford's algorithm is simplest in the unit demand setting, it can also compute allocations when bidders have gross substitutes valuations. Before we discuss our analogous extension, we will first introduce some notation for gross substitutes valuations. Unlike unit demand valuations, bidders with gross substitute valuations may demand more than one good. Let $\Omega = 2^G$ denote the space of bundles (i.e., subsets of goods). Like previous sections, let k be number of types of goods, and let s be the supply

of each type of good. Let d denote the *market size*—the total number of goods, including identical goods, so $d = ks$.[5] We assume that each bidder has a valuation function on bundles, $v_i : \Omega \to [0,1]$, and that this valuation satisfies the gross substitutes condition (Definition 4.2.1).

Like before, we simulate k ascending price auctions in rounds. Bidders now maintain a bundle of goods that they are currently allocated to, and bid on one new good each round. For each good in a bidder's bundle, the bidder keeps track of the count of bids on that good when it was added to the bundle. When the current count ticks past the supply, the bidder knows that they have been outbid.

The main subtlety is in how bidders decide which goods to bid on. Namely, each bidder treat goods in their bundle as fixed in price (i.e., bidders ignore the price increment of at most α that might have occurred after winning the item). Goods outside of their bundle (even if identical to goods in their bundle) are evaluated at the true price. We call these prices the bidder's *effective* prices, so each bidder bids on an arbitrary good in his most-preferred bundle at the effective prices. The full algorithm is given in Algorithm 6.

Privacy is very similar to the case for matchings.

Theorem 4.4.1. $\mathsf{PAlloc}(\alpha, \rho, \varepsilon)$ *satisfies ε-joint differential privacy.*

Proof. Essentially the same proof as Theorem 4.3.4. □

Theorem 4.4.2. *Let $0 < \alpha < n/d$, and g be the allocation computed by $\mathsf{PAlloc}(\alpha/3, \alpha/3, \varepsilon)$, and let* OPT *be the optimum max welfare. Then, if $d \geq n$ and*

$$s \geq \frac{12E' + 3}{\alpha} = O\left(\frac{1}{\alpha^3 \epsilon} \cdot \mathrm{polylog}\left(n, k, \frac{1}{\alpha}, \frac{1}{\gamma}\right)\right),$$

[5] In general, goods may have different supplies, if s denotes the *minimum* supply of any good. Hence, d is not necessarily dependent on s.

Algorithm 6 PAlloc$(\alpha, \rho, \varepsilon)$ (with Gross Substitute Valuations)

Input: Bidders' gross substitute valuations on the bundles $\{v_i : \Omega \to [0,1]\}$
Initialize: for bidder i **and good** j**,**

$$T = \frac{10}{\alpha\rho}, \qquad \epsilon' = \frac{\epsilon}{2T}, \qquad E = \frac{2\sqrt{2}}{\epsilon'}(\log nT)^{5/2}\log\left(\frac{4k}{\gamma}\right)+1, \qquad m = 2E+1,$$

$$\text{counter}_0 = \text{Counter}(\epsilon', nT), \qquad \text{counter}_j = \text{Counter}(\epsilon', nT), \qquad p_j = c_j = 0, \qquad d_g = 0,$$

$$g(i) = \{\emptyset\} \qquad \text{for every bidder } i$$

Propose T times; **Output:** prices p and allocation g.

Propose:
for all bidders i **do**
  for all goods $g \in g(i)$ **do**
    if $c_{type(g)} - d_g \geq s - m$ **then**
      Remove $g(i) := g(i) \setminus g$
  Let p_0 be the original cost of $g(i)$.
  Let $\omega^* \in \underset{\omega \supsetneq g(i)}{\text{argmax}}\, v_i(\omega) - p(\omega \setminus g(i)) - p_0$ arbitrary.
  if $v_i(\omega^*) - p(\omega \setminus g(i)) - p_0 \geq v_i(g(i)) - p_0$ **then**
    Let $j \in \omega^* \setminus g(i)$ arbitrary.
    Save $d_j := c_{type(j)}$
    Add $g(i) := g(i) \cup j$ and **Bid**$(\mathbf{e_j})$
  else Bid$(\mathbf{0})$
CountUnsatisfied

Bid: On input bid vector $\mathbf{b}$
for all goods j **do**
  Feed $\mathbf{b}_j$ to counter$_j$.
  Update count $c_j :=$ counter$_j$.
  if $c_j \geq (p_j/\alpha + 1)(s - m)$ **then**
    Update $p_j := p_j + \alpha$.

CountUnsatisfied:
for all bidders i **do**
  if i wants continue bidding **then**
    Feed 1 to counter$_0$.
  else Feed 0 to counter$_0$.
if counter$_0$ increases by less than $\rho d - 2E$ **then**
  Halt and output μ.

the allocation g has social welfare at least

$$\sum_{i=1}^{n} v_i(g(i)) \geq \text{OPT} - \alpha d$$

with probability at least $1-\gamma$, where

$$E' = \frac{360\sqrt{2}}{\alpha^2 \varepsilon}\left(\log\left(\frac{90n}{\alpha^2}\right)\right)^{5/2} \log\left(\frac{4k}{\gamma}\right) + 1.$$

Remark 4.4.3. *In comparison with Theorem 4.3.5, Theorem 4.4.2 requires a similar constraint on supply but promises welfare* $\text{OPT} - \alpha d$ *rather than* $\text{OPT} - \alpha n$. *Since* $\text{OPT} \leq n$ *this guarantee is only non-trivial for* $\alpha \leq n/d$, *so the supply has a polynomial dependence on the total size of the market* d. *In contrast, Theorem 4.3.5 guarantees good welfare when the supply has a* logarithmic *dependence on the total number of goods in the market.*

We note that if bidders demand bundles of size at most b, *then we can improve the above welfare bound to* $\text{OPT} - \alpha n b$. *Note that this is independent of the market size* d *and smoothly generalizes the matching case where* $b = 1$.

Similar to Definition 4.3.1, we define an *approximate allocation equilibrium* as a prerequisite for showing our welfare guarantee.

Definition 4.4.4. *A price vector* $p \in [0,1]^k$ *and an assignment* $g\colon [n] \to \Omega$ *of bidders to goods is an* (α, β, ρ)-approximate allocation equilibrium *if*

1. *for all but* ρd *bidders,* $v_i(g(i)) - p(g(i)) \geq \max_{\omega \in \Omega} v_i(\omega) - p(\omega) - \alpha|g(i)|$;
2. *the number of bidders assigned to any good is at most* s; *and*
3. *each over-demanded good clears except for at most* β *supply.*

The following lemmas show that our algorithm finds an approximate allocation equilibrium. We prove the last two requirements first.

Lemma 4.4.5. *Assume all counters have error at most* E *throughout the run of* $\mathsf{PAlloc}(\alpha, \rho, \varepsilon)$.

Then, the number of bidders assigned to any good is at most s and each over-demanded good clears except for at most β supply, where

$$\beta = 4E + 1 = O\left(\frac{1}{\alpha\rho\epsilon} \cdot \text{polylog}\left(n, k, \frac{1}{\alpha}, \frac{1}{\rho}, \frac{1}{\gamma}\right)\right).$$

Proof. Consider any good j. If it is under-demanded, the counter corresponding to j never rise above $s - m$. Hence by our conditioning, at most $s - m + E < s$ bidders are assigned to j. If j is over-demanded, the same reasoning as in Lemma 4.3.10 shows that the number of bidders matched to j lies in the range $[s - m - 2E, s - m + 2E + 1]$. By the choice of m, the upper bound is at most s. Likewise, at least $s - m + E = s - (4E + 1)$ bidders are assigned to j. Setting $\beta = 4E + 1$ gives the desired bound. □

Lemma 4.4.6. *We call a bidder who wants to bid more* unsatisfied*; otherwise, a bidder is* satisfied*. At termination of* PAlloc$(\alpha, \rho, \varepsilon)$*, all satisfied bidders are matched to a bundle $g(i)$ that is an $\alpha \cdot |g(i)|$-most preferred bundle.*

Proof. We first show that a bidder's bundle $g(i)$ remains a subset of their most preferred bundle at the effective prices, i.e., with prices of goods in $g(i)$ set to their price at time of assignment, and all other goods taking current prices.

This claim follows by induction on the number of timesteps (ranging from 1 to nT). The base case is clear. Now, assume that the claim holds up to time t. There are three possible cases:

1. If the price of a good outside $g(i)$ is increased, $g(i)$ remains part of a most-preferred bundle by the gross substitutes condition.

2. If the price of a good in $g(i)$ is increased, some goods may be removed from the bundle leading to a new bundle $g'(i)$. The only goods that experience an effective price increase lie outside of $g'(i)$, so $g'(i)$ remains a subset of a most-preferred bundle at the effective prices.

3. If a bidder adds to their bundle, $g(i)$ is a subset of the most-preferred bundle by definition.

Hence, a bidder becomes satisfied precisely when $g(i)$ is equal to the most-preferred bundle at the effective prices. The true price is at most α more than the effective price, so the bidder must have an $\alpha|g(i)|$-most preferred bundle at the true prices. □

Lemma 4.4.7. *Suppose all counters have error at most E throughout the run of* $\mathsf{PAlloc}(\alpha, \rho, \varepsilon)$. *Then at termination, all but ρd bidders are satisfied if*

$$n \le d \quad \textit{and} \quad d \ge \frac{8E}{\rho} = \Omega\left(\frac{1}{\alpha\rho^2\epsilon} \cdot \mathrm{polylog}\left(n, k, \frac{1}{\alpha}, \frac{1}{\rho}, \frac{1}{\gamma}\right)\right).$$

Proof. Note that as long as the algorithm does not halt, at least $\rho d - 4E$ bidders are unsatisfied at the beginning of the round. They may not actually bid when their turn comes, because the prices may have changed. Let the number of bids among all bidders be B, and suppose we run for R rounds. We expect at least $\rho d - 4E$ bids per round, so $R(\rho d - 4E) - B$ is a lower bound on the number of times a bidder is unsatisfied but fails to bid.

In the matching case, if a bidder is unsatisfied at the beginning of the round but fails to bid during their turn, this must be because the prices have risen too high. Since prices are monotonic increasing, such a bidder will never be unsatisfied again.

In contrast, the gross substitutes case is slightly more subtle. Bidders who are unsatisfied at the beginning of a round and don't bid on their turn may later become unsatisfied again. Clearly, this happens only when the bidder loses at least one good after they decline to bid: if they don't lose any goods, then the prices can only increase after they decline to bid. Thus, they will have no inclination to bid in the future.

There are at most n cases of the bidder dropping out entirely. Thus, the number of times bidders report wanting to reenter the bidding is at least $R(\rho d - 4E) - n - B$. Since a bidder loses at least one good each time they reenter, the number of reentries is at most the

number of bids B. Hence, the number of bids in R rounds is at least

$$B \geq \frac{R(\rho d - 4E) - n}{2}. \tag{4.3}$$

Now, let $s' = s - m = s - (2E + 1)$ be the effective supply and consider how many bids are possible. Each of the k types of goods will accept at most $s' + 2E = s + 1$ bids at each of $1/\alpha$ price levels, so there are at most $k(s+1)/\alpha = (d+k)/\alpha$ possible bids. Setting the left side of Equation (4.3) equal to $(d+k)/\alpha$, we find

$$R \leq \frac{1}{\alpha}\left(\frac{2(d+k) + \alpha n}{\rho d - 4E}\right) := T_0,$$

so taking $T \geq T_0$ suffices to ensure that the algorithm halts with no more than ρd bidders unsatisfied. Assuming $\rho d \geq 8E$ and $d \geq n$,

$$T_0 \leq \frac{10d}{\alpha \rho d} = \frac{10}{\alpha \rho} = T.$$

The requirement on n and d is then

$$d \geq \frac{8E}{\rho} = \Omega\left(\frac{1}{\alpha \rho^2 \epsilon} \cdot \text{polylog}\left(n, k, \frac{1}{\alpha}, \frac{1}{\rho}, \frac{1}{\gamma}\right)\right) \quad \text{and} \quad n \leq d,$$

as desired. □

Lemma 4.4.8. *With probability at least $1 - \gamma$, $\mathsf{PAlloc}(\alpha, \rho, \varepsilon)$ computes an (α, β, ρ)-approximate allocation equilibrium where*

$$\beta = O\left(\frac{1}{\alpha \rho \epsilon} \cdot \text{polylog}\left(n, k, \frac{1}{\alpha}, \frac{1}{\rho}, \frac{1}{\gamma}\right)\right),$$

so long as

$$d \geq \frac{8E}{\rho} = \Omega\left(\frac{1}{\alpha \rho^2 \epsilon} \cdot \text{polylog}\left(n, k, \frac{1}{\alpha}, \frac{1}{\rho}, \frac{1}{\gamma}\right)\right) \text{and } n \leq d.$$

Proof. Condition on the error for each counter being at most E throughout the run of the

algorithm. By Theorem 4.2.4, this holds for any single counter with probability at least $1 - \gamma/2k$. By a union bound, this holds for all counters with probability at least $1 - \gamma$. The theorem follows by Lemmas 4.4.5 to 4.4.7. □

Now, it is straightforward to prove the welfare theorem (Theorem 4.4.2).

Proof. The proof follows the matching case (Theorem 4.3.5) closely. By Lemma 4.4.8, (g, p) is a $(\alpha/3, \beta, \alpha/3)$-approximate allocation equilibrium, where $\beta = 4E' + 1$. Then all but $\alpha d/3$ bidders are satisfied and get a bundle $g(i)$ that is $\alpha|g(i)|$ optimal; let this set of bidders be B. Note that $\sum_i |g(i)| \le d$. Let g^* be any other allocation. Then,

$$\sum_{i \in B} v_i(g(i)) - p(g(i)) \ge \sum_{i \in B} v_i(g^*(i)) - p(g^*(i)) - \frac{\alpha}{3}|g(i)|$$

$$\sum_{i \in B} v_i(g^*(i)) - v_i(g(i)) \le \sum_{i \in B} p(g^*(i)) - p(g(i)) + \alpha d/3 = \sum_{j \in G} p_j(N_j^* - N_j) + \alpha d/3$$

where the N_j is the number of good j sold in g and N_j^* is the number of good j sold in g^*. If $p_j > 0$, we know $N_j \ge s - \beta$, hence $N_j^* - N_j \le \beta \le \alpha s/3$. Since $p_j \le 1$ for each good j, we have

$$\sum_{j \in G} p_j(N_j^* - N_j) \le \sum_j p_j(N_j^* - N_j) \le \alpha \sum_j s = \alpha d/3.$$

Furthermore, at most $\alpha d/3$ bidders are left unsatisfied in the end; these bidders contribute at most $\alpha d/3$ welfare to the optimal matching since valuations are bounded by 1. Putting it all together,

$$\sum_i v_i(g^*(i)) - v_i(g(i)) \le \alpha d/3 + \alpha d/3 + \alpha d/3 = \alpha d.$$

The stated supply bound s follows directly from Lemma 4.4.8. □

4.5. Lower Bounds

Our lower bounds all reduce to a basic database reconstruction lower bound for differential privacy.

Theorem 4.5.1. *Let mechanism $\mathcal{M}\colon \{0,1\}^n \to \{0,1\}^n$ be (ϵ,δ)-differentially private, and suppose that for all database D, with probability at least $1-\beta$, $\|\mathcal{M}(D)-D\|_1 \le \alpha n$. Then,*

$$\alpha \ge 1 - \frac{e^\epsilon + \delta}{(1+e^\epsilon)(1-\beta)} := \theta(\epsilon,\delta,\beta).$$

Proof. Fix a database $D \in \{0,1\}^n$ and sample an index i uniformly at random from $[n]$. Let D' be a neighboring database of D that differs at the i-th bit. By assumption, we have that with probability at least $1-\beta$

$$\|\mathcal{M}(D)-D\|_1 \le \alpha n, \qquad \|\mathcal{M}(D')-D'\|_1 \le \alpha n.$$

Since i is chosen uniformly, we then have

$$\Pr[\mathcal{M}(D)_i = D_i] \ge (1-\alpha)(1-\beta), \qquad \Pr[\mathcal{M}(D')_i = D'_i] \ge (1-\alpha)(1-\beta).$$

It follows that $\Pr[\mathcal{M}(D')_i = D_i] \le 1-(1-\alpha)(1-\beta)$ because $D_i \ne D'_i$. By definition of (ϵ,δ)-differential privacy, we get

$$(1-\alpha)(1-\beta) \le \Pr[\mathcal{M}(D)_i = D_i] \le e^\epsilon \Pr[\mathcal{M}(D')_i = D_i] + \delta \le e^\epsilon(1-(1-\alpha)(1-\beta)) + \delta.$$

Then we have

$$1-\alpha \le \frac{e^\epsilon + \delta}{(1+e^\epsilon)(1-\beta)}$$

as desired. □

In other words, no (ϵ,δ)-private mechanism can reconstruct more than a fixed constant

fraction of its input database. For ϵ, δ, β small, $\theta(\epsilon, \delta, \beta) \sim 1/2$. Informally, this theorem states that a private reconstruction mechanism can't do much better than guessing a random database. Note that this holds even if the adversary doesn't know which fraction was correctly reconstructed.

Our lower bounds will all be proved using the following pattern.

- First, we describe how to convert a database $D \in \{0,1\}^n$ to a market, by specifying the bidders, the goods, and the valuations $v_{ij} \in [0,1]$ on goods.
- Next, we analyze how these valuations change when a single bit in D is changed. This will control how private the matching algorithm is with respect to the original database, when applied to this market.
- Finally, we show how to output a database guess $\hat{D}$ from the matching produced by the private matching algorithm.

This composition of three steps will be a private function from $\{0,1\}^n \to \{0,1\}^n$, so we can apply Theorem 4.5.1 to lower bound the error, implying a lower bound on the error of the matching algorithm.

4.5.1. Standard Differential Privacy

Note that Algorithm 4 produces market clearing prices under standard differential privacy. We will first show that this is not possible if each good has unit supply. Recall that prices correspond to an (α, β, ρ)-*approximate matching equilibrium* if all but ρ bidders can be allocated to a good such that their utility is within α of their favorite good (Definition 4.3.1). We will ignore the β parameter, which controls how many goods are left unsold.

Theorem 4.5.2. *Let n bidders have valuations $v_{ij} \in [0,1]$ for n goods. Suppose that mechanism $\mathcal{M}$ is (ϵ, δ)-differentially private, and calculates prices corresponding to an (α, β, ρ)-approximate*

matching equilibrium for $\alpha < 1/2$ and some β with probability $1-\gamma$. Then,

$$\rho \geq \frac{1}{2}\theta(2\epsilon, \delta(1+e^{\epsilon}), \gamma).$$

Note that this is independent of α.

Proof. Let $D \in \{0,1\}^{n/2}$ be a private database and construct the following market. For each bit i we construct the following gadget, consisting of two goods $\mathbf{0}_i, \mathbf{1}_i$ and two bidders, $b_i, \bar{b_i}$. Both bidders have valuation D_i for good $\mathbf{1}_i$, $1-D_i$ for good $\mathbf{0}_i$, and valuation 0 for the other goods. Evidently, there are n bidders and n goods.

Note that changing a bit i in D changes the valuation of exactly two bidders in the market: b_i and $\bar{b_i}$. Therefore, mechanism $\mathcal{M}$ is $(2\epsilon, \delta(1+e^{\epsilon}))$-differentially private with respect to D. Let the prices be p_{0i}, p_{1i}. To guess the database $\hat{D}$, we let $\hat{D}_i = 1$ if $p_{1i} > 1/2$, otherwise $\hat{D}_i = 0$.

By assumption, $\mathcal{M}$ produces prices corresponding to an (α, β, ρ)-approximate matching equilibrium with probability $1-\gamma$. We do not have access to the matching, but we know the prices must correspond to *some* matching μ. Then, for all but ρn gadgets, μ matches both bidders to their α-approximate favorite good and both goods are matched to bidders who receive α-approximate favorite goods.

Consider such a gadget i. We will show that exactly one of p_{0i} or p_{1i} is greater than $1/2$, and this expensive good corresponds to bit D_i. Consider one of the bidders in this gadget, and suppose she prefers good g_+ with price p_+, while he received good g_- with price p_-. Since she receives an α-approximate favorite good,

$$(1-p_+) - (0-p_-) \leq \alpha, \qquad \text{so} \qquad p_+ - p_- \geq 1-\alpha > 1/2.$$

So $p_+ > 1/2$ and $p_- < 1/2$. Note that good g_+ is in the gadget, while good g_- may not be. So, one of the goods in the gadget has price strictly greater than $1/2$. The other good in

the gadget is an α-approximate favorite good for some bidder. All bidders have valuation 0 for the good, hence its price must be strictly less than 1/2.

Thus, the reconstruction procedure will correctly produce bit for each such gadget, and so will miss at most ρn bits with probability at least $1-\gamma$. The combined reconstruction algorithm is a map from $\{0,1\}^{n/2} \to \{0,1\}^{n/2}$, and $(2\epsilon, \delta(1+e^{\epsilon}))$-differentially private. By Theorem 4.5.1,

$$2\rho \geq \theta(2\epsilon, \delta(1+e^{\epsilon}), \gamma).$$

This completes the proof. □

4.5.2. Separation Between Standard and Joint Differential Privacy

While we can compute an approximate maximum-weight matching under joint privacy when the supply of each good is large (Lemma 4.3.10), this is not possible under standard differential privacy even with infinite supply.

Theorem 4.5.3. *Let n bidders have valuations $v_{ij} \in \{0,1\}$ for 2 goods with infinite supply. Suppose that mechanism $\mathcal{M}$ is (ϵ, δ)-differentially private, and computes a matching with weight at least* OPT $-\alpha n$ *with probability $1-\gamma$. Then,*

$$\alpha \geq \theta(\epsilon, \delta, \gamma).$$

Proof. Let $D \in \{0,1\}^n$. We assume two goods, **0** and **1**. We have one bidder b_i for each bit $i \in [n]$, who has valuation D_i for **1**, and valuation $1-D_i$ for **0**. Since changing a bit changes a single bidder's valuation, applying $\mathcal{M}$ to this market is (ϵ, δ)-private with respect to D. To guess the database $\hat{D}$, we let $\hat{D}_i$ be 0 if b_i is matched to **0**, 1 if b_i is matched to **1**, and arbitrary otherwise.

Note that the maximum welfare matching assigns each b_i the good corresponding to D_i, and achieves social welfare OPT $= n$. If $\mathcal{M}$ computes an matching with welfare OPT $-\alpha n$, it must give all but an α fraction of bidders b_i the good corresponding to D_i. So, the recon-

structed database will miss at most αn bits with probability $1-\gamma$, and by Theorem 4.5.1,

$$\alpha \geq \theta(\epsilon, \delta, \gamma).$$

□

Note that this gives a separation: under joint differential privacy, Algorithm 4 can release a matching with welfare $\text{OPT} - \alpha n$ for any α, provided supply s is large enough (by Theorem 4.3.5), while this is not possible under standard differential privacy even with *infinite* supply.

4.5.3. Lower Bounds for Joint Differential Privacy

Finally, we show that a large supply assumption is necessary in order to compute an additive α maximum welfare matching under joint differential privacy.

Theorem 4.5.4. *Let n bidders have valuations $v_{ij} \in [0,1]$ for k types of goods with supply s each. Suppose mechanism $\mathcal{M}$ is (ϵ, δ)-joint differentially private for $\epsilon, \delta < 0.1$, and calculates a matching with welfare at least $\text{OPT} - \alpha n$ with probability $1-\gamma$ for $\gamma < 0.01$, and all n, k, s. Then, $s = \Omega(\sqrt{1/\alpha})$.*

Proof. Let $k = n/(s+1)$. Given a private database $D \in \{0,1\}^k$, construct the following market. For each bit i, we construct a gadget with two goods $\mathbf{0}_i, \mathbf{1}_i$, each with supply s. Each gadget has a distinguished bidder b_i and s identical bidders, all labeled $\bar{b_i}$. Let bidder b_i, who we call the *real bidder*, have valuation D_i for 1_i, and $1 - D_i$ for 0_i. Bidders $\bar{b_i}$, which we call the *spy bidders*, all have the same valuation: $\eta = \frac{1}{4s}$ for $\mathbf{0}_i$ or $\mathbf{1}_i$ drawn at random, and 0 for all other goods (in and out of the gadget). We say a bidder *prefers* a good if they have positive valuation for the good.

Note that changing a bit in D changes a single bidder's valuation. Also note that the spy bidders' valuations do not depend on D. Hence, by joint differential privacy of $\mathcal{M}$, the function that maps the above market through $\mathcal{M}$ to the allocation of just the spy bidders

is (ϵ, δ)-differentially private with respect to an entry change in D.

We will describe how to guess $\hat{D}$ based on just the spy bidders' joint view, i.e., the goods they are assigned. This reconstruction procedure will then be (ϵ, δ)-differentially private, and we can apply Theorem 4.5.1 to lower bound the error of $\mathcal{M}$. For every bit $i \in [k]$, let $\hat{D}_i$ be 1 if the spy bidders in gadget i are all assigned to $\mathbf{0}_i$, 0 if the spy bidders in gadget i are all assigned to $\mathbf{1}_i$, and uniformly random otherwise.

We'll say that a gadget *agrees* if the spy bidders and real bidder prefer the same good. Gadgets that don't agree, *disagree*. Let w be the number of gadgets that agree. By construction, gadgets agree independently with probability 1/2 each. Hence, Hoeffding's inequality gives

$$\Pr\left[\left|w - \frac{k}{2}\right| \leq \lambda k\right] \geq 1 - 2\exp(-2\lambda^2 k)$$

for some λ to be specified later; condition on this event. With probability at least $1 - \gamma$, mechanism $\mathcal{M}$ computes a matching with welfare at least $\text{OPT} - \alpha n$; condition on this event as well. Note that the optimum welfare is $1 + (s-1)\eta$ for gadgets that agree, and $1 + s\eta$ for gadgets that disagree, hence $\text{OPT} = w(1 + (s-1)\eta) + (k-w)(1 + s\eta)$ in total.

For each gadget, there are several possible allocations. Intuitively, an assignment gives social welfare, but may also lead to a bit being reconstructed. Let $RB(\mu) = \|D - \hat{D}\|_1$ be the error of the reconstruction when the matching is μ. We'll argue that any matching μ with nearly optimal social welfare must result in large expected reconstruction $\mathbb{E}[RB(\mu)]$. By linearity,

$$\mathbb{E}[RB(\mu)] = \sum_{i \in [k]} \Pr\left[D_i = \hat{D}_i\right],$$

so it suffices to focus on gadget at a time.

First, suppose the gadget i agrees. The matching μ can give the preferred good to the bidder, the spies, or neither. If the preferred good goes to the bidder, this gives at most

$1 + (s-1)\eta$ social welfare. Not all the spies get the same good, so

$$\Pr\left[D_i = \hat{D}_i\right] = \frac{1}{2}.$$

If the preferred good goes to the spies, then this contributes $s\eta$ to social welfare, and

$$\Pr\left[D_i = \hat{D}_i\right] = 0.$$

Note that it doesn't matter whether the bidder is assigned in μ, since the social welfare is unchanged and the reconstruction algorithm doesn't have access to the bidder's allocation. There are other possible allocations, but they are dominated by these two choices since they get less social welfare for higher reconstruction probability.

Now, suppose gadget i disagrees. There are several possible allocations. First, both the bidder and the spies may get their favorite good. This gives $1 + s\eta$ welfare, and

$$\Pr\left[D_i = \hat{D}_i\right] = 1.$$

Second, the bidder may be assigned their favorite good, and at most $s-1$ spies may be assigned their favorite good. This leads to $1 + (s-1)\eta$ welfare, with

$$\Pr\left[D_i = \hat{D}_i\right] = \frac{1}{2}.$$

Again, there are other possible allocations, but they lead to less social welfare or higher reconstruction probability. We say the four allocations above are *optimal*.

Let a_1, a_2 be the fractions of agreeing gadgets with the two optimal agreeing allocations, and d_1, d_2 be the fractions of disagreeing gadgets with the two optimal disagreeing allocations. Let t be the fraction of agreeing pairs. The following linear program minimizes $(1/k)\mathbb{E}[RB(\mu)]$ over all matchings μ achieving an α-approximate maximum welfare match-

ing for supply s.

$$
\begin{aligned}
LP_s := \qquad & \text{minimize: } \frac{1}{2}a_1 + d_1 + \frac{1}{2}d_2 \\
& \text{such that: } a_1 + a_2 \le t \\
& \qquad d_1 + d_2 \le 1 - t \\
& \qquad \frac{1}{2} - \lambda \le t \le \frac{1}{2} + \lambda \\
& \qquad (1 + (s-1)\eta)a_1 + s\eta a_2 + (1 + s\eta)d_1 + (1 + (s-1)\eta)d_2 \\
& \qquad \ge t(1 + (s-1)\eta) + (1-t)(1 + s\eta) - \frac{\alpha n}{k}
\end{aligned}
$$

The last constraint is the welfare requirement, the second to last constraint is from conditioning on the number of agreeing gadgets, and the objective is $(1/k)\mathbb{E}[RB(\mu)]$.

Plugging in $\eta = \frac{1}{4s}, \lambda = 1/128, \alpha = \frac{k}{16ns}$ and solving, we find

$$(a_1, a_2, d_1, d_2, t) = \left(\frac{65}{128}, 0, \frac{31}{128}, \frac{1}{4}, \frac{65}{128}\right)$$

is a feasible solution for all s with objective $\alpha' = 159/256$. To show that this is optimal, consider the dual problem:

$$
\begin{aligned}
DUAL_s := \qquad & \text{maximize: } -\rho_2 + \left(\frac{1}{2} - \lambda\right)\rho_3 - \left(\frac{1}{2} + \lambda\right)\rho_4 + \left(1 + s\eta - \frac{\alpha n}{k}\right)\rho_5 \\
& \text{such that: } -\rho_1 + (1 + (s-1)\eta)\rho_5 \le \frac{1}{2} \\
& \qquad -\rho_1 + s\eta\rho_5 \le 0 \\
& \qquad -\rho_2 + (1 + s\eta)\rho_5 \le 1 \\
& \qquad -\rho_2 + (1 + (s-1)\eta)\rho_5 \le \frac{1}{2} \\
& \qquad \rho_1 - \rho_2 + \rho_3 - \rho_4 + \eta\rho_5 \le 0
\end{aligned}
$$

We can directly verify that

$$(\rho_1, \rho_2, \rho_3, \rho_4, \rho_5) = \left(\frac{5}{2}s - 1, \frac{5}{2}s - 1, 0, \frac{1}{2}, 2s\right)$$

is a dual feasible solution with objective $\alpha' = 159/256$.

We know that $\mathcal{M}$ calculates an additive α-approximate maximum welfare matching. While the allocations to each gadget may not be an optimal allocation, suboptimal allocations all have less social welfare and larger RB. So, we know the objective of LP_m is a lower bound for $RB(\mathcal{M})$.

Thus, $\mathbb{E}[RB(\mathcal{M})] \geq k\alpha'$ for any supply s. Since RB is the sum of k independent, 0/1 random variables, another Hoeffding bound yields

$$\Pr\left[RB(\mathcal{M})/k \geq \alpha' - \lambda'\right] \geq 1 - 2\exp(-2\lambda'^2 k).$$

Set $\lambda' = 1/256$, and condition on this event. All together, any matching mechanism $\mathcal{M}$ which finds a matching with weight at least $\mathrm{OPT} - \alpha n$ failing with at most γ probability gives an (ϵ, δ)-private mechanism mapping D to $\hat{D}$ such that

$$\frac{1}{k} \cdot \|D - \hat{D}\|_1 \geq \alpha' - \lambda' = 79/128.$$

with probability at least $1 - \gamma - 2\exp(-2\lambda^2 k) - 2\exp(-2\lambda'^2 k)$.

For $\epsilon, \delta < 0.1$ and $\gamma < 0.01$, this contradicts Theorem 4.5.1 for large k. Note that the failure probability and accuracy do not depend directly on s since $\lambda, \lambda', \alpha'$ are constants. Hence

$$\alpha \gg \frac{k}{16ns} = \frac{1}{16s(s+1)}$$

uniformly for all s, and $s = \Omega(\sqrt{1/\alpha})$ as desired. □

4.6. Privacy Analysis for Counters

Chan et al. [2011] show that Counter(ϵ, T) is ϵ-differentially private with respect to single changes in the input stream, when the stream is generated non-adaptively. For our application we require privacy to hold for a large number of streams whose joint-sensitivity can nevertheless be bounded, and whose entries can be chosen adaptively. To show that Counter is also private in this setting (when ϵ is set appropriately), we first introduce some differential privacy notions.

4.6.1. Adaptive Composition

We give a slight generalization of Theorem 2.2.4.

Lemma 4.6.1 (Generalization of Theorem 2.2.4)**.** *Let $\Delta_1 \geq 0$. The class of $\frac{\epsilon}{\Delta_1}$-private mechanisms satisfies ϵ-differential privacy under adaptive composition, if the adversary always selects databases satisfying*

$$\sum_{t=1}^{T} \left| D^{t,0} - D^{t,1} \right| \leq \Delta_1.$$

In other words, the privacy parameter of each mechanism should be calibrated for the total distance between the databases over the whole composition (the ℓ_1 *sensitivity*).

4.6.2. Binary mechanism

We reproduce the Binary mechanism here in order to refer to its internal workings in our privacy proof. First, it is worth explaining the intuition of the Counter. Given a bit stream $\sigma : [T] \to \{0,1\}$, the algorithm releases the counts $\sum_{i=1}^{t} \sigma(i)$ for each t by maintaining a set of partial sums $\sum[i,j] := \sum_{t=i}^{j} \sigma(t)$. More precisely, each partial sum has the form $\Sigma[2^i + 1, 2^i + 2^{i-1}]$, corresponding to powers of 2.

In this way, we can calculate the count $\sum_{i=1}^{t} \sigma(i)$ by summing at most $\log t$ partial sums: let

$i_1 < i_2 \ldots < i_m$ be the indices of non-zero bits in the binary representation of t, so that

$$\sum_{i=1}^{t} \sigma(i) = \sum [1, 2^{i_m}] + \sum [2^{i_m} + 1, 2^{i_m} + 2^{i_{m-1}}] + \ldots + \sum [t - 2^{i_1} + 1, t].$$

Therefore, we can view the algorithm as releasing partial sums of different ranges at each time step t and computing the counts is simply a post-processing of the partial sums. The core algorithm is presented in Algorithm 7.

Algorithm 7 Counter(ε, T)

Input: A stream $\sigma \in \{0,1\}^T$
Output: $B(t)$ as estimate for $\sum_{i=1}^{t} \sigma(i)$ for each time $t \in [T]$
for all $t \in [T]$ **do**
  Express $t = \sum_{j=0}^{\log t} 2^j \text{Bin}_j(t)$.
  Let $i \leftarrow \min_j \{\text{Bin}_j(t) \neq 0\}$
  $a_i \leftarrow \sum_{j<i} a_j + \sigma(t)$, $(a_i = \sum [t - 2^i + 1, t])$
  for $0 \leq j \leq i - 1$ **do**
    Let $a_j \leftarrow 0$ and $\hat{a_j} \leftarrow 0$
  Let $\hat{a_j} = a_j + \text{Lap}(\log(T)/\varepsilon)$
  Let $B(t) = \sum_{i:\text{Bin}_i(t)\neq 0} \hat{a_i}$

4.6.3. Counter Privacy Under Adaptive Composition

We can now show that the prices released by our mechanism satisfy ϵ-differential privacy.

Theorem 4.3.3. *The sequence of prices and counts of unsatisfied bidders released by* PMatch$(\alpha, \rho, \varepsilon)$ *satisfies ϵ-differential privacy.*

Proof. Chan et al. [2011] show this for a single sensitivity 1 counter for a non-adaptively chosen stream. We here show the generalization to multiple counters run on adaptively chosen streams with bounded ℓ_1 sensitivity, and bound the ℓ_1 sensitivity of the set of streams produced by our algorithm. We will actually show that the sequence of noisy partial sums released by Counter satisfy ϵ-differential privacy. This is only stronger: the running counts are computed as a function of these noisy partial sums.

To do so, we first take the view of an adversary in the adaptive composition experiment (Theorem 2.2.4) and then show that the view of this adversary is precisely the sequence of noisy partial sums. The composition theorem (Lemma 4.6.1) will then show that the sequence of noisy partial sums are differentially private with respect to a change in a bidder's valuation.

Let the two runs $b = 0, 1$ correspond to any two neighboring valuations (v_i, v_{-i}) and (v_i', v_{-i}) that differ only in bidder i's valuation. We first analyze the view on all of the counter(j) for $j = 1, \ldots, k$.

The adversary will operate in phases. There are two kinds of phases, which we label P_t and P_t': one phase per step of the good counters, and one phase per step of the halting condition counter. Both counters run from time 1 to nT, so there are $2nT$ phases in total.

At each point in time, the adversary maintains histories $\{b_i\}, \{b_i'\}$ of all the bids prior to the current phase and histories $\{e_i\}, \{e_i'\}$ of all prior reports to the halting counter counter_0, when bidder i has valuation v_i, v_i' respectively.

Let us consider the first kind of phase. One bidder bids per step of the counter, so one bidder bids in each of these phases. Each step of the experiment the adversary will observe a partial sum. Suppose the adversary is in phase P_t. Having observed the previous partial sums, the adversary can simulate the action of the current bidder q from the histories of previous bids by first computing the prices indicated by the previous partial sums. The adversary will compute q's bid when the valuations are (v_i, v_{-i}), and when the valuations are (v_i', v_{-i}). Call these two bids b_t, b_t' (which may be $\perp$ if q is already matched in one or both of the histories).

Note that for bidders $q \neq i$, it is always the case that $b_t = b_t'$. This holds by induction: it is clearly true when no one has bid, and bidder q's decision depends only on her past bids, the prices, and her valuation. Since these are all independent of bidder i's valuation, bidder q behaves identically.

After the adversary calculates b_t, b'_t, the adversary simulates update and release of the counters. More precisely, the adversary spends phase P_t requesting a set of partial sums

$$\Sigma = \{\sigma_I^j \mid j \in [k], I \in S_t\},$$

where $S_t \subseteq [1, nT]$ is a set of intervals ending at t, corresponding to partial sums that Counter releases at step t.

For each $\sigma_I^j \in \Sigma$, $D^0, D^1 \in \{0,1\}_I$ are defined by

$$D_k^0 = \begin{cases} 1 & : \text{if } b_k = j \\ 0 & : \text{otherwise} \end{cases}$$

and similarly for D^1, with bid history $\{b'_i\}$. Informally, a database D for σ_I^j encodes whether a bidder bid on good j at every timestep in I. The adversary will define $\mathcal{M}$ to sum the bits in the database and add noise $Lap(1/\epsilon_0)$, an ϵ_0-differentially private operation. Once the partial sums for P_t are released, the adversary advances to the next phase.

Now, suppose the adversary is in the second kind of phase, say P'_t. This corresponds to a step of the halting condition counter. We use exactly the same construction as above: the adversary will request the partial sums corresponding to each timestep. The adversary will simulate each bidder's action by examining the history of bids and prices. Now suppose the two runs differ in bidder i's valuation. Following the same analysis, the reports to this halting condition counter differ only in bidder i's reports.

With this definition, the view of the adversary on database $\{D^0\}$ and $\{D^1\}$ is precisely the noisy partial sums when the valuations are (v_i, v_{-i}) and (v'_i, v_{-i}), respectively. So, it suffices to show that these views have almost the same probability.

We apply Lemma 4.6.1 by bounding the distance between the databases for counter(1) to counter(k). Note that the sequence of databases $\{D^0\}, \{D^1\}$ chosen correspond to streams

of bids that differ only in bidder i's bid, or streams of reports to counter(0) that differ only in bidder i's report. This is because the bid histories $\{b_t\}, \{b'_t\}$ and report histories $\{e_t\}, \{e'_t\}$ differ only on timesteps where i acts. Thus, it suffices to focus on bidder i when bounding the distance between these databases.

Consider a single good j, and suppose c_j of i's bids on good j differ between the histories. Each of bidder i's bids on j show up in $\log(nT)$ databases, so

$$\sum |D_j^0 - D_j^1| \leq c_j \log nT,$$

where the sum is taken over all databases corresponding to good j. The same is true for the halting condition counter: if there are c_0 reports that differ between the histories, then

$$\sum |D_0^0 - D_0^1| \leq c_0 \log nT.$$

Since we know that a bidder can bid at most T times over T proposing rounds, and will report at most T times, we have ℓ_1 sensitivity bounded by

$$\Delta_1 \leq c_0 \log nT + \sum_j c_j \log nT \leq 2T \log nT.$$

By Lemma 4.6.1, setting

$$\epsilon_0 = \frac{\epsilon}{2T \log nT}$$

suffices for ϵ-differential privacy, and this is precisely running each Counter with privacy level $\epsilon' = \epsilon/2T$. □

CHAPTER 5

Jointly Private Convex Programming

5.1. Introduction

In Chapter 4, we provide an algorithm for computing approximately max-weight matchings under joint differential privacy. The algorithm implements an ascending price auction in a deferred-acceptance style, and also uses prices (which are the dual variables in the matching problem) to coordinate the allocation. As a result, the techniques rather specific to matchings, which don't easily generalize even to the k-demand allocation problem (when agents have general preferences over bundles of k goods), and certainly not to general convex optimization problems. In fact, simply solving the allocation problem beyond gross substitutes valuations was stated as the main open problem in the original paper by Hsu et al. [2014a]; our results solve this problem as a special case. Our algorithm also yields approximate truthfulness, which is described in Chapter 6.

Our main contribution is a technique for solving a large family of convex optimization problems under joint differential privacy. Concretely, consider any convex optimization problem that can be written in the following *separable* form:

$$\begin{aligned} \underset{x}{\text{maximize}} \quad & \textstyle\sum_{i=1}^{n} v^{(i)}(x^{(i)}) + v^{(0)}(x^{(0)}) \\ \text{subject to} \quad & x^{(i)} \in S^{(i)} && \text{(for all } i\text{)} \\ & \textstyle\sum_{i=1}^{n} c_j^{(i)}(x^{(i)}) + c_j^{(0)}(x^{(0)}) \le b_j && \text{(for all } j\text{).} \end{aligned}$$

Here, $x^{(i)}$ denotes the variables that form agent i's portion of the output, for $i = 1, \ldots, n$. We also allow data for a special "agent 0" to model auxiliary variables and constraints that don't depend on private data. The functions $\{v^{(i)}\}_i$ in the objective are concave, while the

constraint functions $\{c_j^{(i)}\}_{i,j}$ are convex; both can depend on the private data of agent i. The compact and convex sets $S^{(i)}$ model the feasible region for a single agent; they are naturally private data.

There are two types of constraints in the above convex program. The first type of constraints ($x^{(i)} \in S^{(i)}$) involves *only* the variables of a single agent (for example, the flow conservation constraints between an agent's private source and destination in the multi-commodity flow problem). These are the "easy" constraints from a privacy perspective—if these were the only kinds of constraints, then each agent could separately optimize her own portion of the objective subject to these easy constraints. This is trivially jointly differentially private: an agent's output would be entirely independent of the other agents' data.

The second type of constraints involves variables from different agents (for example, the capacity constraints in multi-commodity flow, or resource constraints in multi-dimensional knapsack). These constraints are the "hard" constraints from the perspective of privacy: they couple different agents, forcing them to somehow coordinate their solution.

We give a general method for solving *separable* convex programs so that we approximately optimize the objective and satisfy the constraints, subject to joint differential privacy:

Theorem 5.1.1 (Informal, some important parameters missing)**.** *There is an ϵ-jointly differentially private algorithm which can solve linearly separable convex programs while (1) exactly satisfying the personal constraints, (2) obtaining objective value at least* OPT $-\alpha$*, and (3) guaranteeing that the sum violation over the k coupling constraints is at most*

$$\alpha \approx \tilde{O}\left(\frac{k\sigma}{\varepsilon}\right),$$

where σ is a measure of the sensitivity of the convex program. For packing problems, we can also guarantee no *violation of the coupling constraints at a small additional cost in the objective. For a broad class of problems, we can also round to integral solutions with a small additional loss.*

Our technique is based on the *dual decomposition* method in distributed optimization.[1] While the original motivation for such techniques was to solve large optimization problems on distributed networks of computers, distributed solvers are conceptually attractive from a privacy standpoint: they solve problems with distributed data, while minimizing communication between players.

Specifically, we construct a "partial Lagrangian" by bringing the hard (coupling) constraints into the objective, while leaving the easy (personal) constraints to constrain the primal feasible region. Doing this leaves us with the following equivalent minimax problem:

$$\underset{\lambda \geq 0}{\text{minimize}} \; \underset{x : x^{(i)} \in S^{(i)}}{\text{maximize}} \; \mathcal{L}(x, \lambda) := \sum_i v^{(i)}(x^{(i)}) - \sum_j \lambda_j \left(\sum_i c_j^{(i)}(x_i) - b_j \right).$$

This partial Lagrangian can be viewed as the payoff of zero-sum game between the primal player (the maximization player, controlling the x variables) and the dual player (the minimization player, controlling the λ variables). Under this interpretation, equilibrium strategies (x^*, λ^*) form the optimal primal and dual solutions. Further, approximate equilibria correspond to approximately optimal primal-dual solution pairs (in a sense we will make precise).

Without privacy, a standard way to find an approximate equilibrium is via repeated play: Repeatedly simulate play of the game, letting one player (for us, the dual player) update his variables using a no-regret algorithm (for us, gradient descent), while letting the other player (for us, the primal player) repeatedly best respond to his opponent's actions. The time averaged play of this simulation converges quickly to an approximate minimax equilibrium of the game [Freund and Schapire, 1996].[2]

However, differential privacy—and joint differential privacy, in particular—complicates this picture. First, the best response of the primal player is a candidate solution, whose various components depend strongly on the private information of agents (say, their pri-

[1] See Boyd et al. [2011] for an overview of this and related techniques.
[2] Recall that we use the same result to construct DualQuery in Chapter 3.

vate feasible region). Second, for the dual player to run a no-regret algorithm, she must have access to the losses of the candidate solution. In our situation, this is the also private information.

To get around these difficulties, we take advantage of the separable structure of convex program. Crucially, the form of the Lagrangian allows us to compute a primal best response by having each of the n agents best respond *individually*, without coordination: Given the current dual variables $\lambda_j^{(t)}$, agent i solves the problem

$$\begin{aligned} \underset{x^{(i)}}{\text{maximize}} \quad & v^{(i)}(x^{(i)}) - \textstyle\sum_j \lambda_j^{(t)} c_j^{(i)}(x^{(i)}) \\ \text{subject to} \quad & x^{(i)} \in S^{(i)}, \end{aligned}$$

which is an optimization problem that is *independent* of the private data of other players $j \neq i$.

Due to the form of the Lagrangian, the combination of the individual agents' best responses forms a primal best response in the optimization problem, and the time-averaged strategies of individual agents form our near-optimal primal solution. Since each agent i's solution depends on the other agents' data only through the dual player's actions, a standard argument shows that we can guarantee joint-differential privacy for the solution by ensuring that the dual actions satisfy *standard* differential privacy. To privatize these plays, we implement privacy preserving gradient descent, by adding Gaussian noise to the gradient of the Lagrangian at each step.

5.1.1. Related Work

Distributed optimization techniques date back to the 1950s, with the original goal of solving large optimization problems with networks of computationally limited machines. The area is very rich with mathematically elegant algorithms, many of which are used in practice today.

The method we develop in this chapter is based on a simple technique called dual decomposition. For a more comprehensive overview, the reader can consult the excellent survey by Boyd et al. [2011].

In the previous chapter, we give an algorithm for computing approximately max-weight matchings under joint differential privacy. The previous algorithm implements an ascending price auction in a deferred-acceptance style, and also uses prices (which are the dual variables in the matching problem) to coordinate the allocation. The techniques in the previous chapter is rather specific to matchings, which don't easily generalize even to the k-demand allocation problem (when agents have general preferences over bundles of k goods), and certainly not to general convex optimization problems.

For related work in private optimization, Hsu et al. [2014b] consider how to solve various classes of linear programs under (standard) differential privacy. Their work contains many negative results because many natural linear programs cannot be solved under the standard differential privacy, while we give broadly positive results under the looser notion of joint differential privacy. Nissim et al. [2007] consider combinatorial optimization problems, also under the standard constraint of differential privacy. Our work is also related to private empirical risk minimization, which involves *unconstrained* convex minimization subject to the standard differential privacy [Chaudhuri et al., 2011, Jain et al., 2011, Kifer et al., 2012, Jain and Thakurta, 2014, Bassily et al., 2014]. Many of these papers use privacy preserving variants of gradient descent (and other optimization algorithms), which we use as part of our algorithm.

5.2. Preliminaries

5.2.1. Zero-Sum Games and No-Regret Dynamics

We now present a continuous version of the no-regret dynamics for two-player zero sum game. (Previously in Chapter 3, we present the version for normal form games, where the

payoffs are given by a matrix.)

Consider a two-player zero-sum game with payoff function $A : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. The maximization player selects an action $x \in \mathcal{X}$, with the goal of maximizing $A(x, y)$. Simultaneously, the minimization player selects an action $y \in \mathcal{Y}$ with the goal of minimizing $A(x, y)$.

We will consider games such that:

- for all $x, y \in \mathcal{X} \times \mathcal{Y}$ the payoff $A(x, y)$ is bounded;
- $\mathcal{X}$ and $\mathcal{Y}$ are closed, compact, convex sets;
- fixing $y \in \mathcal{Y}$, $A(x, y)$ is a concave function in x; and
- fixing $x \in \mathcal{X}$, $A(x, y)$ is a convex function in y.

It is known that any such game has a *value* V: the maximization player has an action $x^* \in \mathcal{X}$ such that $A(x^*, y) \geq V$ for all $y \in \mathcal{Y}$, and the minimization player has action y^* such that $A(x, y^*) \leq V$ for all $x \in \mathcal{X}$ [Kneser, 1952]. We can define *approximate* minimax equilibria of such games as follows:

It is known that any such game has a *value* V: the maximization player has an action $x^* \in \mathcal{X}$ such that $A(x^*, y) \geq V$ for all $y \in \mathcal{Y}$, and the minimization player has action y^* such that $A(x, y^*) \leq V$ for all $x \in \mathcal{X}$ [Kneser, 1952]. We can define *approximate* minimax equilibria of such games as follows:

Definition 5.2.1. *Let $\alpha \geq 0$. A pair of strategies $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ form an* α-approximate minimax equilibrium *if*

$$A(x, y) \geq \max_{x' \in \mathcal{X}} A(x', y) - \alpha \geq V - \alpha \quad \text{and} \quad A(x, y) \leq \min_{y' \in \mathcal{Y}} A(x, y') + \alpha \leq V + \alpha.$$

To compute an approximate minimax equilibrium, we will use the following T-round no-regret dynamics: one player plays online gradient descent [Zinkevich, 2003] as an

no-regret algorithm, while the other player selects a *best-response* action in each round. We will let the min player be the no-regret learner, who produces a sequence of actions $\{y_1, \ldots, y_T\}$ against max player's best responses $\{x_1, \ldots, x_T\}$. For each $t \in [T]$,

$$y_{t+1} = \Pi_{\mathcal{Y}}\left[y_t - \eta \cdot \nabla_y A(x_t, y_t)\right] \qquad \text{and} \qquad x_t = \arg\max_x A(x, y_t),$$

where $\Pi_{\mathcal{Y}}$ is the Euclidean projection map onto the set $\mathcal{Y}$: $\min_{y'} \|y - y'\|_2$, η is the step size.

In the end, denote the minimization player's regret as

$$\mathcal{R}_y \equiv \frac{1}{T}\sum_{t=1}^{T} A(x_t, y_t) - \frac{1}{T}\min_{y \in \mathcal{Y}} \sum_{t=1}^{T} A(x_t, y).$$

Now consider the average actions for both players in this dynamics: $\bar{x} = \frac{1}{T}\sum_{t=1}^{T} x_t$ and $\bar{y} = \frac{1}{T}\sum_{t=1}^{T} y_t$. Freund and Schapire [1996] showed that the average plays form an approximate equilibrium:

Theorem 5.2.2 (Freund and Schapire [1996])**.** *$(\bar{x}, \bar{y})$ forms a $\mathcal{R}_y$-approximate minimax equilibrium.*

In order to construct the approximate equilibrium privately, we also develop a noisy and private version of the online gradient descent algorithm, described in the following section.

5.2.2. Private Online Linear Optimization

In this section we consider a private version of the online linear optimization problem. The techniques we use to solve this problem are relatively standard (and are similar to solutions in e.g. Kearns et al. [2014] and Bassily et al. [2014]), but we work in a somewhat different setting, so we provide proofs here for completeness.

The learner has action set $\mathcal{P} \subset \mathbb{R}^k$, and the adversary has action space $\mathcal{X} \subset [-X, X]^k$. Given any action p of the learner, and action x of the adversary, the (linear) loss function for the

learner is $\ell(p,x) = \langle p,x\rangle$. For any sequence of T actions from the adversary $\{x_1,\ldots,x_T\}$, the learner's goal is to minimize the regret defined as

$$\mathcal{R}_T = \frac{1}{T}\sum_{t=1}^{T}\langle p_t, x_t\rangle - \min_{p\in\mathcal{P}}\frac{1}{T}\sum_{t=1}^{T}\langle p, x_t\rangle.$$

For privacy reason, the learner only get to observe noisy and private versions of the adversary's actions. In particular, we can think of the loss vectors over T rounds as a $(T\times k)$-dimensional statistics $x = (x_1,\ldots,x_T)$ some underlying sensitive population D. Suppose we add noise sampled from the Gaussian distribution $\mathcal{N}(0,\sigma^2)$ on every entry of x. We will determine the scale of σ in the privacy analysis, and we will use the following concentration bound of Gaussian distribution.

Fact 5.2.3 (Gaussian Tails)**.** *Let Y be a random variable sampled from the distribution $\mathcal{N}(0,\sigma^2)$ and $a = \ln 2/(2\pi)$, then for all $\lambda > 0$*

$$\Pr[|Y| > \lambda] \le 2\exp\left(-a\lambda^2/\sigma^2\right).$$

Fact 5.2.4 (Gaussian Sums)**.** *Let $Y_1,\ldots,Y_n$ be independent variables with distribution $\mathcal{N}(0,\sigma^2)$. Let $Y = \sum_i Y_i$. Then, the random variable $Y \sim \mathcal{N}(0,n\sigma^2)$, and so*

$$\Pr[|Y| > \lambda] \le 2\exp\left(-a\lambda^2/(n\sigma^2)\right).$$

Let $\{\hat{x}_t\}$ be the noisy loss vectors. The learner will update the action p_t using projected gradient descent

$$p_{t+1} = \Pi_{\mathcal{P}}\left[p_t - \eta\hat{x}_t\right],$$

where $\Pi_{\mathcal{P}}$ is the Euclidean projection map onto the set $\mathcal{P}$: $\min_{p'}\|p-p'\|_2$, η is the step size.

Before we show the regret bound for our noisy gradient descent, we here include the no-regret result for standard online gradient descent (with no noise).

Lemma 5.2.5 ([Zinkevich, 2003])**.** *For any actions and losses space $\mathcal{P}$ and $\mathcal{X}$, the gradient descent algorithm: $p_{t+1} = \Pi_{\mathcal{P}}[p_t - \eta x_t]$ has regret*

$$\mathcal{R}_T \leq \frac{\|\mathcal{P}\|^2}{2\eta T} + \frac{\eta \|\mathcal{X}\|^2}{2}.$$

We are now ready to show the following regret bound for this noisy gradient descent.

Theorem 5.2.6. *Let $\|\mathcal{P}\| = \max_{p\in\mathcal{P}} \|p\|$ and $\|\mathcal{X}\|_2 = \max_{x\in\mathcal{X}} \|x\|_2$, then with probability at least $1-\beta$,*

$$\mathcal{R}_T = O\left(\frac{\|\mathcal{P}\|_2\sqrt{k}}{\sqrt{T}}\left(X + \sigma \log\left(\frac{Tk}{\beta}\right)\right)\right).$$

Proof. Let ν_t denote the noise vector we have in round t, we can decompose the regret into several parts

$$\begin{aligned}
\mathcal{R}_T &= \frac{1}{T}\sum_{t=1}^{T}\langle p_t, x_t\rangle - \frac{1}{T}\min_{p\in\mathcal{P}}\sum_{t=1}^{T}\langle p, x_t\rangle \\
&= \frac{1}{T}\sum_{t=1}^{T}\langle p_t, \hat{x}_t\rangle - \frac{1}{T}\sum_{t=1}^{T}\langle p_t, \nu_t\rangle - \frac{1}{T}\left[\min_{p\in\mathcal{P}}\sum_{t=1}^{T}\langle p, x_t\rangle - \min_{\hat{p}\in\mathcal{P}}\sum_{t=1}^{T}\langle \hat{p}, \hat{x}_t\rangle\right] - \frac{1}{T}\min_{\hat{p}\in\mathcal{P}}\sum_{t=1}^{T}\langle \hat{p}, \hat{x}_t\rangle \\
&= \left[\frac{1}{T}\sum_{t=1}^{T}\langle p_t, \hat{x}_t\rangle - \frac{1}{T}\min_{\hat{p}\in\mathcal{P}}\sum_{t=1}^{T}\langle \hat{p}, \hat{x}_t\rangle\right] - \frac{1}{T}\sum_{t=1}^{T}\langle p_t, \nu_t\rangle - \frac{1}{T}\left[\min_{p\in\mathcal{P}}\sum_{t=1}^{T}\langle p, x_t\rangle - \min_{\hat{p}\in\mathcal{P}}\sum_{t=1}^{T}\langle \hat{p}, \hat{x}_t\rangle\right] \\
&= \hat{\mathcal{R}}_T - \frac{1}{T}\sum_{t=1}^{T}\langle p_t, \nu_t\rangle - \frac{1}{T}\left[\min_{p\in\mathcal{P}}\sum_{t=1}^{T}\langle p, x_t\rangle - \min_{\hat{p}\in\mathcal{P}}\sum_{t=1}^{T}\langle \hat{p}, \hat{x}_t\rangle\right] \\
&\leq \hat{\mathcal{R}}_T - \frac{1}{T}\min_{p\in\mathcal{P}}\sum_{t=1}^{T}\langle p, \nu_t\rangle - \frac{1}{T}\left[\min_{p\in\mathcal{P}}\sum_{t=1}^{T}\langle p, x_t\rangle - \min_{\hat{p}\in\mathcal{P}}\sum_{t=1}^{T}\langle \hat{p}, \hat{x}_t\rangle\right].
\end{aligned}$$

We will bound the three terms separately. By the no-regret guarantee of online gradient descent in Lemma 5.2.5, we have the following the regret guarantee w.r.t the noisy losses if we set $\eta = \frac{\|\mathcal{P}\|}{\sqrt{T}\|\hat{\mathcal{X}}\|}$

$$\hat{\mathcal{R}}_T = \frac{1}{T}\sum_{t=1}^{T}\langle p_t, \hat{x}_t\rangle - \min_{p\in\mathcal{P}}\frac{1}{T}\sum_{t=1}^{T}\langle p, \hat{x}_t\rangle \leq \frac{\|\mathcal{P}\|^2}{2\eta T} + \frac{\eta\|\hat{\mathcal{X}}\|^2}{2} = \frac{\|\mathcal{P}\|\|\hat{\mathcal{X}}\|}{\sqrt{T}},$$

where $\|\mathcal{P}\|$ and $\|\hat{\mathcal{X}}\|$ denote the bound on the ℓ_2 norm of the vectors $\{p_t\}$ and $\{\hat{x}_t\}$ respectively.

Recall that for any random variable Y sampled from the Gaussian distribution $\mathcal{N}(0,\sigma^2)$, we know from Fact 5.2.3

$$\Pr[|Y| \geq d \cdot \sigma] \leq 2\exp(-ad^2).$$

For each noise vector ν_t and any coordinate i, we have with probability except β/Tk that $|\nu_t(i)| \leq \sigma\sqrt{\frac{1}{a}\cdot\log\left(\frac{2Tk}{\beta}\right)}$. By union bound, we have with probability except β that

$$\max_t \max_i |\nu_t(i)| \leq \sigma\sqrt{\frac{1}{a}\cdot\log\left(\frac{2Tk}{\beta}\right)} \quad \text{and so for all } t, \quad \|\nu_t\|_2 \leq \sigma\sqrt{\frac{k}{a}\cdot\log\left(\frac{2Tk}{\beta}\right)}$$

Since $\hat{x}_t = x_t + \nu_t$, we know that

$$\|\hat{\mathcal{X}}\| \leq \sqrt{k}\left(X + \sigma\sqrt{\frac{1}{a}\cdot\log\left(\frac{2Tk}{\beta}\right)}\right).$$

Now we bound the second and third term. From Fact 5.2.4, we know with probability at least $1-\beta/k$, for each coordinate i,

$$\sum_t \nu_t(i) \leq \sigma\sqrt{\frac{T}{a}\cdot\log\left(\frac{2k}{\beta}\right)}.$$

By a union bound, we know with probability at least $1-\beta$

$$\left\|\sum_t \nu_t/T\right\|_\infty \leq \frac{\sqrt{\frac{\sigma^2}{a}\log(k/\beta)}}{\sqrt{T}} \qquad \text{and so} \qquad \left\|\sum_t \nu_t/T\right\|_2 \leq \frac{\sqrt{\frac{k\sigma^2}{a}\log(k/\beta)}}{\sqrt{T}}.$$

Now we could use Holder's inequality to bound the second term

$$-\frac{1}{T}\min_{p\in\mathcal{P}}\sum_{t=1}^{T}\langle p,\nu_t\rangle \leq \left|\min_{p\in\mathcal{P}}\langle p,\sum_{t=1}^{T}\nu_t/T\rangle\right| \leq \|\mathcal{P}\|\frac{\sqrt{k\sigma^2\log(k/\beta)}}{\sqrt{aT}}.$$

Let $p^* \in \arg\min_{p\in\mathcal{P}} \sum_{t=1}^{T} \langle p, x_t\rangle$ and $\hat{p}^* \in \arg\min_{\hat{p}\in\mathcal{P}} \sum_{t=1}^{T} \langle \hat{p}, \hat{x}_t\rangle$. We can rewrite the third term as

$$
\begin{aligned}
-\frac{1}{T}\left[\min_{p\in\mathcal{P}} \sum_{t=1}^{T} \langle p, x_t\rangle - \min_{p\in\mathcal{P}} \sum_{t=1}^{T} \langle \hat{p}, \hat{x}_t\rangle\right] &= -\langle p^*, \sum_t x_t/T\rangle + \langle \hat{p}^*, \sum_t \hat{x}_t/T\rangle \\
&= -\langle p^*, \sum_t \hat{x}_t/T\rangle + \langle \hat{p}^*, \sum_t \hat{x}_t/T\rangle - \left\langle p^*, \frac{1}{T}\left(\sum_t x_t - \sum_t \hat{x}_t\right)\right\rangle \\
\text{(Holder's inequality)} &\leq \langle p^*, \frac{1}{T}\sum_t (v_t)\rangle \leq \|\mathcal{P}\| \frac{\sqrt{k\sigma^2 \log(k/\beta)}}{\sqrt{aT}}.
\end{aligned}
$$

In the end, we have that

$$
\begin{aligned}
\mathcal{R}_T &\leq \frac{\|\mathcal{P}\|\sqrt{k}}{\sqrt{T}}\left(\left(X + \sigma\sqrt{\frac{1}{a}\cdot\log\left(\frac{2Tk}{\beta}\right)}\right) + 2\frac{\sqrt{\sigma^2\log(k/\beta)}}{\sqrt{a}}\right) \\
&\leq \frac{\|\mathcal{P}\|\sqrt{k}}{\sqrt{T}}\left(X + 2\sigma\sqrt{\frac{1}{a}\cdot\log\left(\frac{2Tk}{\beta}\right)}\right) \\
&= \frac{\|\mathcal{P}\|\sqrt{k}}{\sqrt{T}}\left(X + \left(\sqrt{\frac{8\pi}{\log 2}}\right)\sigma\sqrt{\log\left(\frac{2Tk}{\beta}\right)}\right) \\
&= O\left(\frac{\|\mathcal{P}\|\sqrt{k}}{\sqrt{T}}\left(X + \sigma\log\left(\frac{Tk}{\beta}\right)\right)\right).
\end{aligned}
$$

This completes the proof. □

5.3. Private Dual Decomposition

Let's consider the electricity scheduling problem, which will be our running example as we present our algorithm. Suppose we have n agents, who need power for T intervals. Each interval is subdivided into Q slots, and agents have different valuations $v \in [0,1]$ for different slots. There is a maximum amount of electricity c available for each (interval, slot) pair. Finally, agents demand some total amount of electricity $d_t \in [0, Q]$ during each interval, and at most d_{max} in total over all intervals. The demand may be zero for some intervals, say, if the agent is not home. Translating the description in the introduction, we

have the following linear program:

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{n}\sum_{t=1}^{T}\sum_{q=1}^{Q} v_{tq}^{(i)} x_{tq}^{(i)} \\
\text{subject to} \quad & \sum_{i=1}^{n} x_{tq}^{(i)} \le c_{tq} && (\text{for } t \in [T], q \in [Q]) \\
& \sum_{q=1}^{Q} x_{tq}^{(i)} \ge d_t^{(i)}, \quad \sum_{q=1}^{Q}\sum_{t=1}^{T} x_{tq}^{(i)} \le d_{max}, \quad \text{and} \quad x_{tq}^{(i)} \in [0,1] && (\text{for } i \in [n], t \in [T], q \in [Q]).
\end{aligned}$$

We consider each agent's valuations $v^{(i)}$ and demands $d^{(i)}$ to be private. Agent i will receive variables $x^{(i)}$, which indicate when she is getting electricity. With this in mind, notice that this LP has a particularly nice structure: the objective and first constraints are sums of terms that (term by term) depend only a single agent's data and variables, while the second constraints only constrain a single agent's data. This LP is an instance of what we call *linearly separable convex programs*.

Definition 5.3.1. *Let the data universe be $\mathcal{X}$, and suppose there are n individuals. We map each database $D \in \mathcal{X}^n$ to a* linearly separable convex optimization problem *$\mathcal{O}$, which consists of the following data: for each agent i,*

- *a compact convex set $S^{(i)} \subseteq \mathbb{R}^l$,*
- *a concave objective function $v^{(i)} : S^{(i)} \to \mathbb{R}$,*
- *and k convex constraint functions $c_j^{(i)} : S^{(i)} \to \mathbb{R}$ (indexed by $j = 1 \dots k$);*

all defined by D_i. $\mathcal{O}$ also includes the following data, independent of D:

- *a compact convex set $S^{(0)} \subseteq \mathbb{R}^l$,*
- *a concave objective function $v^{(0)} : S^{(0)} \to \mathbb{R}$,*
- *k convex constraint functions $c_j^{(0)} : S^{(0)} \to \mathbb{R}$ (indexed by $j = 1 \dots k$),*

- *and a vector* $b \in \mathbb{R}^k$.

The convex optimization problem is:

$$\begin{aligned} \text{maximize} \quad & \sum_{i=0}^{n} v^{(i)}(x^{(i)}) \\ \text{subject to} \quad & \sum_{i=0}^{n} c_j^{(i)}(x^{(i)}) \leq b_j \quad (\textit{for } j = 1 \dots k) \\ & x^{(i)} \in S^{(i)} \qquad\qquad (\textit{for } i = 0 \dots n). \end{aligned}$$

We call the set of all such optimization problems the class of problems associated to $\mathcal{X}$ *or simply a* class of problems, *if* $\mathcal{X}$ *is unimportant.*

Intuitively, the variables, objective, and constraints indexed by i belong to agent i for $i \in \{1, \dots, n\}$. The constraints and variables for $i = 0$ are shared—it is sometimes useful to have auxiliary variables in a convex program that do not correspond to any player's part of the solution. We call the first kind of constraints the *coupling constraints*, since they involve multiple agents' variables and data. We call the second kind of constraints the *personal constraints*, since they only involve a single agent's variables and data. We write $S = S^{(0)} \times \cdots \times S^{(n)}$ for the portion of the feasible region defined only by the personal constraints.

Example 5.3.2. *The coupling constraints are the power supply constraints on each time slot, and the personal constraints are the agent's demand constraints for different intervals.*

5.3.1. Algorithm

To solve the convex program, we will work extensively with the *Lagrangian*:

$$\mathcal{L}(x, \lambda) = \sum_{i=0}^{n} v^{(i)}(x^{(i)}) - \sum_{j=1}^{k} \lambda_j \left(\sum_{i=0}^{n} c_j^{(i)}(x^{(i)}) - b_j \right).$$

For clarity of presentation, we will assume in the following that our optimization problems

are feasible and that the *strong duality* condition holds, i.e.,

$$\max_{x \in S} \min_{\lambda \in \mathbb{R}^k_+} \mathcal{L}(x, \lambda) = \min_{\lambda \in \mathbb{R}^k_+} \max_{x \in S} \mathcal{L}(x, \lambda).$$

If our problems are not even approximately feasible, this will be easy to detect, and the assumption that strong duality holds is without loss of generality in our setting, since our goal is only to approximately satisfy the coupling constraints.[3]

We will interpret the Lagrangian objective as the payoff function of a zero-sum game between a primal player (controlling the x variables), and a dual player (controlling the λ variables). We will privately construct an approximate equilibrium of this game by simulating repeated play of the game. At each step, the primal player will best-respond to the dual player by finding x maximizing the Lagrangian subject to the dual player's λ. The dual player in turn will run a no-regret algorithm to update the λ variables, using losses defined by the primal player's choice of x. By a standard result about repeated play Theorem 5.2.2, the average of each player's actions converges to an approximate equilibrium.

We will first detail how to privately construct this approximate equilibrium. Then, we will show that an approximate equilibrium is an approximately optimal primal-dual pair for the original problem; in particular, the primal player's point will be approximately feasible and optimal. Throughout, let the problem be $\mathcal{O} = (S, v, c, b)$.

Remark 5.3.3. *Before we begin, we want to clarify one point. We will sometimes say "Agent i plays ..." or "Agent i solves ...". These descriptions sound natural, but are slightly misleading: Our algorithms will not be online or interactive in any sense, and all computation is done by our algorithm, not by the agents. Agents will submit their private data, and will receive a single output. Our algorithm will* simulate *the agent's behavior, be it selecting actions to play, or solving smaller optimization problems, or rounding.*

[3] Strong duality is guaranteed in particular by *Slater's condition* [Slater, 1950]: there exists some point $x \in S$ such that for all $j \in [k]$, $\sum_{i=0}^{n} c_j^{(i)}(x) < b_j$. If this is not already the case, it can be guaranteed by simply relaxing our constraints by a tiny amount. Since our solutions will already only *approximately* satisfy the coupling constraints, this is without loss.

Let's start with the primal player's best-response. Rewriting the Lagrangian and fixing λ, at each round, the primal player plays

$$\operatorname*{argmax}_{x \in S} \mathcal{L}(x) = \sum_{i=0}^{n} \left(v^{(i)}(x^{(i)}) - \sum_{j=1}^{k} \lambda_j c_j^{(i)}(x^{(i)}) \right) + \sum_{j=1}^{k} \lambda_j b_j, \tag{5.1}$$

which can be computed by solving

$$x^{(i)} \in BR^{(i)}\left(\mathcal{O}, \{\lambda_j\}\right) := \operatorname*{argmax}_{x \in S^{(i)}} \left(v^{(i)}(x) - \sum_{j=1}^{k} \lambda_j c_j^{(i)}(x) \right)$$

independently for each agent i, and combining the solutions to let $x := (x^{(1)}, \ldots, x^{(n)})$.

For the dual player, we will use the online gradient descent algorithm due to Zinkevich [2003] for a suitably chosen target set Λ, together with noise addition to guarantee differential privacy. At each time step we feed in a perturbed version of the loss vector $l \in \mathbb{R}^k$ defined to be the gradient of the Lagrangian with respect to λ:

$$l_j := \frac{\partial \mathcal{L}}{\partial \lambda_j} = \sum_{i=0}^{n} c_j^{(i)}(x^{(i)}) - b_j.$$

To obtain privacy properties, we further add Gaussian noise the to above gradient and update the dual variables according to the noisy gradients.

Putting everything together, our algorithm Private Dual Decomposition (PrivDude) presented in Algorithm 8 solves linearly separable convex programs under joint differential privacy.

5.3.2. Privacy

Next, we establish the privacy properties of PrivDude. We will first argue that the dual variables λ satisfy (standard) differential privacy, because the algorithm adds Gaussian

Algorithm 8 Joint Differentially Private Convex Solver: PrivDude($\mathcal{O}, \sigma, \tau, w, \varepsilon, \delta, \beta$)

Input: Convex problem $\mathcal{O} = (S, v, c, b)$ with n agents and k coupling constraints, gradient sensitivity bounded by σ, a dual bound τ, width bounded by w, and privacy parameters $\varepsilon > 0, \delta \in (0,1)$, confidence parameter $\beta \in (0,1)$.

Initialize:

$$\lambda_j^{(1)} := 0 \text{ for } j \in [k], \qquad T := w^2, \qquad \varepsilon' := \frac{\varepsilon\sigma}{\sqrt{8T\ln(2/\delta)}}, \qquad \delta' := \frac{\delta}{2T},$$

$$\eta := \frac{2\tau}{\sqrt{T}\left(w + \frac{1}{\varepsilon'}\log\left(\frac{Tk}{\beta}\right)\right)}, \qquad \Lambda := \{\lambda \in \mathbb{R}_+^k \mid \|\lambda\|_\infty \le 2\tau\}.$$

for iteration $t = 1 \dots T$

  for each agent $i = 0 \dots n$

    Compute personal best response:

$$x_t^{(i)} := \operatorname*{argmax}_{x \in S^{(i)}} v^{(i)}(x) - \sum_{j=1}^{k} \lambda_j^{(t)} c^{(i)}(x).$$

  for each constraint $j = 1 \dots k$

    Compute noisy gradient:

$$\hat{\ell}_j^{(t)} := \left(\sum_{i=0}^{n} c^{(i)}(x_t^{(i)})\right) - b_j + \mathcal{N}\left(0, \frac{2\sigma^2 \log(1.25/\delta')}{\varepsilon'^2}\right),$$

  Do gradient descent update:

$$\lambda^{(t+1)} := \Pi_\Lambda\left(\lambda^{(t)} + \eta\hat{\ell}^{(t)}\right).$$

Output: $\bar{x}^{(0)} := \frac{1}{T}\sum_{t=1}^{T} x_t^{(0)}$ and $\bar{\lambda} := \frac{1}{T}\sum_{t=1}^{T} \lambda^{(t)}$ to everyone, $\bar{x}^{(i)} := \frac{1}{T}\sum_{t=1}^{T} x_t^{(i)}$ to agents $i \in [n]$.

noise to the gradients. Then, we will argue that the primal solution satisfies joint differential privacy, because each agent's best-response depends only on her own private data and the dual variables.

First, we define neighboring convex problems.

Definition 5.3.4. *Let $D, D' \in \mathcal{X}^n$ be two neighboring databases. We say the associated convex programs $\mathcal{O}, \mathcal{O}'$ are* neighboring *problems.*

By looking at how much the gradient l_j may change in neighboring instances, we can determine how much noise to add to ensure differential privacy.

Definition 5.3.5. *A problem $\mathcal{O}$ has* gradient sensitivity *bounded by σ if*

$$\max \sum_{j=1}^{k} \left| c_j^{(i)}(x^{(i)}) - c_j^{\prime(i)}(x^{\prime(i)}) \right|^2 \leq \sigma^2,$$

where the maximum is taken over agents i, dual variables $\{\lambda_j\} \subseteq \mathbb{R}_+^k$, neighboring problems $\mathcal{O}$ and $\mathcal{O}'$, and

$$x^{(i)} \in BR^{(i)}\left(\mathcal{O}, \{\lambda_j\}\right) \qquad \text{and} \qquad x^{\prime(i)} \in BR^{(i)}\left(\mathcal{O}', \{\lambda_j\}\right).$$

Example 5.3.6. *We can bound the gradient sensitivity of the electricity scheduling LP. By changing her private data, an agent may change her demand vector by at most d_{max} in each of T time slots. Since the coupling constraints are simply the total demand over all agents for each time slot, the gradient sensitivity is at most $\sigma = 2\sqrt{d_{max}}$.*

The gradient sensitivity σ is the key parameter for guaranteeing privacy. By definition, it is a bound on the ℓ_2 sensitivity of the gradient vector l. By Theorem 2.3.2, releasing the noisy vector $\hat{l}$ by adding independent Gaussian noise drawn from the distribution $\mathcal{N}(0, 2\sigma^2 \log(1.25/\delta)/\varepsilon^2)$ to each coordinate satisfies (ε, δ)-(standard) differential privacy.

Thus, the following theorem shows privacy of the dual variables in PrivDude.

Theorem 5.3.7. *Let $\epsilon > 0, \delta \in (0, 1/2)$ be given. The sequence of dual variables $\lambda^{(1)}, \ldots, \lambda^{(T)}$ and the public variables $x_1^{(0)}, \ldots, x_T^{(0)}$ produced by* PrivDude *satisfy (ϵ, δ)-differential privacy.*

Proof. By Theorem 2.3.2 and Theorem 2.2.4. □

To show joint differential privacy of the primal variables, note that agent i's best response function $BR^{(i)}(\mathcal{O}, \{\lambda_j\})$ (defined in Equation (5.1)) is a function of i's personal data and the current dual variables λ_j, which satisfy standard differential privacy by Theorem 5.3.7. So, we can use the *billboard lemma* (Lemma 4.3.2) to show the sequence of best-responses satisfies *joint*-differential privacy.

Theorem 5.3.8. *Let $\epsilon > 0, \delta \in (0, 1/2)$ be given. Releasing the sequence of private variables $x_1^{(i)} \dots x_T^{(i)}$ to agent i satisfies (ϵ, δ)-joint differential privacy.*

Proof. By Theorem 5.3.7 and Lemma 4.3.2. □

5.3.3. Accuracy

Now, let us turn to accuracy. We first argue that the exact equilibrium of the game in which we restrict the dual player's strategy space corresponds to an optimal primal-dual pair for the original game. While the original game allows the dual variables to lie anywhere in $\mathbb{R}^k_+$, we need to restrict the dual action space to a bounded subset Λ, in order to use gradient descent. We first show that if Λ is a large enough set, restricting the dual player to play in Λ will not change the equilibrium strategy and value of the game. We next show that PrivDude computes an approximate equilibrium of the Lagrangian game. Finally, we show that the approximate equilibrium strategy of the primal player must be an approximately feasible and optimal point of the original convex program.

For the first step, we observe that the optimal primal and dual solutions (x^*, λ^*) achieve the value of the unrestricted game, which is the optimal objective value OPT of the original convex program.

Lemma 5.3.9. *Let (x^*, λ^*) achieve*

$$\operatorname*{argmax}_{\lambda \in \mathbb{R}^k_+} \min_{x \in S} \mathcal{L}(x, \lambda).$$

Then,

- x^* *is a feasible solution to the original convex program, and*
- $\mathcal{L}(x^*, \lambda^*) = \text{OPT}$, *the optimal objective value of the original convex program.*

Proof. Follows directly from strong duality of the convex problem. □

We now reason about the *restricted* game, in which the dual player plays in a subset $\Lambda \subseteq \mathbb{R}^k_+$.

We first define a key parameter for the accuracy analysis, which measures how much the objective can be improved beyond OPT by *infeasible* solutions, as a function of how much the infeasible solution violates the constraints.

Definition 5.3.10. *Consider all instances $\mathcal{O} = (S, v, c, b)$ in a class of problems. We call $\tau > 0$ a* dual bound *for the class if for all $\delta \geq 0$, i, and $x \in S$ such that*

$$\sum_{i=0}^{n} \sum_{j=1}^{k} \left(c_j^{(i)}(x^{(i)}) - b_j \right)_+ \leq \delta, \qquad \textit{we have} \qquad \sum_{i=0}^{n} v^{(i)}(x^{(i)}) \leq \text{OPT}(\mathcal{O}) + \tau\delta,$$

where $\text{OPT}(\mathcal{O})$ *is the optimum objective for $\mathcal{O}$. (We will frequently elide $\mathcal{O}$, and just say* OPT *when the convex program is clear.)*

Example 5.3.11. *The coupling constraints are power supply constraints. Violating these constraints by δ in total will increase the objective by at most $\delta v_{max} \leq \delta$, so $\tau = 1$ is a dual bound for this problem.*

Intuitively, the dual bound indicates how much the objective can increase beyond the optimal value by slightly violating the feasibility constraints. It will control how large the dual action space must be, in order to discourage the primal player from playing an infeasible point at equilibrium. More precisely, we can show that the game with dual actions restricted to Λ still has the optimal and feasible point as the equilibrium strategy for the primal player, if Λ is large enough.

Lemma 5.3.12. *Suppose τ is a dual bound for a class of convex optimization problems. Then*

if we restrict the dual action space to be $\Lambda = \{\lambda \in \mathbb{R}_+^k \mid \|\lambda\|_2 \leq 2\tau\sqrt{k}\}$*, there is a dual strategy* $\lambda^\bullet \in \Lambda$ *such that* $(x^*, \lambda^\bullet)$ *is an equilibrium of the restricted game.*

Proof. By strong duality, there exists (x^*, λ^*) an equilibrium of the Lagrangian game with value OPT. Played against any strategy in Λ, x^* gets value at least OPT since it is an optimal, feasible solution of the convex program. We first show that restricting the dual player's action set to Λ leaves the value of the game at OPT.

Consider any other primal action x. If it doesn't violate any coupling constraints, then the dual player can set all dual variables to 0. Thus, x has payoff at most OPT in the worst case over all dual player strategies in Λ.

On the other hand, suppose x violates some constraints:

$$\delta := \sum_{j=1}^{k} \left(\sum_{i=0}^{n} c_j^{(i)}(x^{(i)}) - b_j \right)_+ > 0.$$

We can construct a dual player action $\lambda' \in \Lambda$ to give the primal player payoff strictly less than OPT:

1. For constraints j where x violates the constraints

$$\sum_{i=0}^{n} c_j^{(i)}(x^{(i)}) > b_j,$$

   set $\lambda'_j = 2\tau$.

2. For constraints j where x satisfies the constraint:

$$\sum_{i=0}^{n} c_j^{(i)}(x^{(i)}) \leq b_j,$$

   set $\lambda'_j = 0$.

Note that λ' is a valid dual action in the restricted game, since $\lambda' \in \Lambda$. Now, let's bound

the payoff $\mathcal{L}(x, \lambda')$ by comparing it to $\mathcal{L}(x^*, \lambda^*)$. By assumption, the objective term increases by at most $\tau\delta$. While $\mathcal{L}(x^*, \lambda^*)$ has no penalty since all constraints are satisfied, $\mathcal{L}(x, \lambda')$ has penalty $2\tau\delta$ since there is δ total constraint violation. Thus,

$$\mathcal{L}(x, \lambda') \leq \mathcal{L}(x^*, \lambda^*) - 2\tau\delta + \tau\delta < \text{OPT}.$$

Thus, any infeasible x gets payoff at most OPT in the worst case over dual strategies Λ. Since x^* is a primal play achieving payoff OPT, the value of the game must be exactly OPT. In particular, x^* is a maxmin strategy.

Now, since the primal player and the dual player play in compact sets S, Λ, the minmax theorem [Kneser, 1952] states that the restricted game has an equilibrium $(x^\bullet, \lambda^\bullet)$. Thus, $\lambda^\bullet \in \Lambda$ is a minmax strategy, and $(x^*, \lambda^\bullet)$ is the claimed equilibrium. □

In the remainder, we will always work with the restricted game: the dual player will have action set $\Lambda = \{\lambda \in \mathbb{R}^k_+ \mid \|\lambda\|_2 \leq 2\tau\sqrt{k}\}$.

To show that PrivDude computes an approximate equilibrium, we want to use the no-regret guarantee Theorem 5.2.6. We define the second key parameter for accuracy.

Definition 5.3.13. *Consider all problem instances $\mathcal{O} = (S, v, c, b)$ for a class of convex optimization problems. The class has* width bounded by w *if*

$$\max \left| \sum_{i=0}^{n} c_j^{(i)}(x^{(i)}) - b_j \right| \leq w,$$

where the max is taken over all instances $\mathcal{O}$, and coupling constraint j, and $x \in S$.

Example 5.3.14. *The coupling constraints are of the form*

$$\sum_{i=1}^{n} x_{tq}^{(i)} \leq c_{tq},$$

and the x variables lie in $[0, d_{max}]$. Let $c_{max} = \max_{t,q} c_{tq}$ be the maximum capacity over all slots.

If we assume there are a large number of agents so $nd_{max} \gg c_{max}$, then the width is bounded by $w = nd_{max}$.

The width controls how fast online gradient descent converges. However, although the convergence time depends polynomially on w, our accuracy bound depends only on $\log(w)$.

Applying Theorem 5.2.6 gives the following regret guarantee for the dual player.

Lemma 5.3.15. *Suppose w is the width for a class of convex optimization problems. Then with probability at least $1-\beta$, running* PrivDude *for $T = w^2$ iterations yields a sequence of dual plays $\lambda^{(1)}, \ldots, \lambda^{(T)} \in \Lambda$ with regret $\mathcal{R}_p$ to any point in Λ, against the sequence of best responses $x_1, \ldots, x_T$, where*

$$\mathcal{R}_p = O\left(\frac{k\tau\sigma \log^{1/2}(w/\delta)}{\epsilon} \log \frac{wk}{\beta}\right).$$

As above, σ is the gradient sensitivity and τ is the dual bound.

Proof. We add Gaussian noise with variance

$$\frac{2\sigma^2 \log(1.25T/\delta)}{\varepsilon'^2}$$

to each gradient, where

$$\epsilon' = \frac{\epsilon}{\sqrt{8kT\log(1/\delta)}}.$$

Furthermore, the target space Λ has ℓ_2 diameter $2\tau\sqrt{k}$. So by Theorem 5.2.6, the regret to the sequence of unnoised best responses $x_1, \ldots, x_T$ is at most

$$\begin{aligned}
\mathcal{R}_p &\leq \frac{\tau\sqrt{k}}{\sqrt{T}}\left(w + 4\sqrt{\frac{\pi\log(1.25T/\delta)}{\log 2}}\frac{\sigma}{\epsilon'}\log\left(\frac{2Tk}{\beta}\right)\right) \\
&\leq \tau\sqrt{k}\left(1 + \frac{20\sigma\sqrt{8k}}{\epsilon}\log\left(\frac{2Tk}{\beta}\right)\log^{1/2}\left(\frac{T}{\delta}\right)\right) \\
&\leq \frac{40\sqrt{8}k\tau\sigma}{\epsilon}\log\left(\frac{2w^2k}{\beta}\right)\log^{1/2}\left(\frac{w^2}{\delta}\right) \\
&= O\left(\frac{k\tau\sigma}{\epsilon}\log\frac{wk}{\beta}\log^{1/2}\frac{w}{\delta}\right)
\end{aligned}$$

as desired. □

By applying Theorem 5.2.2, we immediately know that PrivDude computes an approximate equilibrium.

Corollary 5.3.16. *With probability at least* $1-\beta$*, the output* $(\bar{x}, \bar{\lambda})$ *of* PrivDude *is an* $\mathcal{R}_p$*-approximate equilibrium for*

$$\mathcal{R}_p = O\left(\frac{k\tau\sigma \log^{1/2}(w/\delta)}{\epsilon} \log \frac{wk}{\beta}\right).$$

Finally, we show that PrivDude generates an approximately feasible and optimal point. For feasibility, the intuition is simple: since the dual player can decrease the Lagrangian value by putting weight 2τ on every violated constraint, there can't be too many violated constraints: $\bar{x}$ is an approximate equilibrium strategy for the primal player, and hence achieves nearly OPT even against a best response from the dual player.

Proving approximate optimality is similar. We think of the primal player as deriving payoff in two ways: from the original objective, and from any over-satisfied constraints with positive dual variable. The dual player can always set the dual variables for over-satisfied constraints to zero and decrease the Lagrangian value. If the primal player derives large payoff from these over-satisfied constraints, then the best response by the dual player will substantially reduce the payoff of the primal player and lead to a contradiction with the approximate maxmin condition. More formally, we have the following theorem.

Theorem 5.3.17. *Suppose a convex program* $\mathcal{O} = (S, v, c, b)$ *has width bounded by* w *and gradient sensitivity bounded by* σ*, and dual bound* τ*. With probability at least* $1-\beta$*,* PrivDude *produces a point* $\bar{x} \in S$ *such that:*

- *the total violation of coupling constraints is bounded by*

$$\sum_{j=1}^{k} \sum_{i=0}^{n} \left(c_j^{(i)}(\bar{x}^{(i)}) - b_j\right)_+ = O\left(\frac{k\sigma \log^{1/2}(w/\delta)}{\epsilon} \log \frac{wk}{\beta}\right), \qquad \textit{and}$$

- *the objective satisfies*

$$\sum_{i=0}^{n} v^{(i)}(\bar{x}^{(i)}) \geq \mathrm{OPT} - \alpha, \qquad \textit{for} \qquad \alpha = 2\mathcal{R}_p = O\left(\frac{k\tau\sigma \log^{1/2}(w/\delta)}{\epsilon} \log \frac{wk}{\beta}\right).$$

Proof. By Lemma 5.3.15, PrivDude computes an $\mathcal{R}_p$ approximate equilibrium $(\bar{x}, \bar{\lambda})$ with

$$\mathcal{R}_p = O\left(\frac{k\tau\sigma \log^{1/2}(w/\delta)}{\epsilon} \log \frac{wk}{\beta}\right),$$

with probability at least $1 - \beta$.

Let us consider first consider feasibility. Since S is convex and each best response lies in S, $\bar{x} \in S$. Suppose $\bar{x}$ violates the coupling constraints by

$$\Delta_1 := \sum_{j=1}^{k} \left(\sum_{i=0}^{n} c^{(i)}(\bar{x}^{(i)}) - b_j \right)_+ .$$

Define $\lambda' \in \Lambda$ as

$$\lambda' = \begin{cases} 2\tau & \text{if } \sum_{i=0}^{n} c^{(i)}(\bar{x}^{(i)}) > b_j \\ 0 & \text{otherwise.} \end{cases}$$

Let's compare the payoff $\mathcal{L}(\bar{x}, \bar{\lambda})$ to $\mathcal{L}(\bar{x}, \lambda')$. By the equilibrium property, we have

$$\mathrm{OPT} - \mathcal{R}_p \leq \mathcal{L}(\bar{x}, \bar{\lambda}) \leq \mathrm{OPT} + \mathcal{R}_p$$

and

$$\mathcal{L}(\bar{x}, \lambda') \geq \mathrm{OPT} - 2\mathcal{R}_p. \tag{5.2}$$

For $\mathcal{L}(\bar{x}, \lambda')$, since the dual bound is τ, we know

$$\sum_{i=0}^{n} v^{(i)}(\bar{x}^{(i)}) \leq \mathrm{OPT} + \tau\Delta_1.$$

At the same time, the penalty is at least

$$\sum_{j=1}^{k} \lambda'_j \left(\sum_{i=0}^{n} c_j^{(i)}(\bar{x}^{(i)}) - b_j \right) \geq 2\tau\Delta_1.$$

So,

$$\mathcal{L}(\bar{x}, \lambda') \leq \text{OPT} - \tau\Delta_1$$

and by Equation (5.2),

$$\Delta_1 \leq \frac{2\mathcal{R}_p}{\tau} = O\left(\frac{k\sigma \log^{1/2}(w/\delta)}{\epsilon} \log \frac{wk}{\beta} \right)$$

as desired.

To show optimality, let the convex program have optimal objective OPT. Suppose $\bar{x}$ has objective value:

$$\sum_{i=0}^{n} v^{(i)}(\bar{x}^{(i)}) = \text{OPT} - \alpha.$$

and say the penalty against $\bar{\lambda}$ is

$$\rho = \sum_{j=1}^{k} \bar{\lambda}_j \left(\sum_{i=0}^{n} c_j^{(i)}(\bar{x}^{(i)}) - b_j \right)$$

for total Lagrangian value $\mathcal{L}(\bar{x}, \bar{\lambda}) = \text{OPT} - \alpha - \rho$. Consider the deviation of the dual variables $\lambda' \in \Lambda$:

$$\lambda'_j = \begin{cases} 0 & \text{if coupling constraint } j \text{ loose} \\ \bar{\lambda}_j & \text{otherwise.} \end{cases}$$

Then $\mathcal{L}(\bar{x}, \lambda') = \text{OPT} - \alpha$. But since $(\bar{x}, \bar{\lambda})$ is an $\mathcal{R}_p$-approximate equilibrium,

$$\mathcal{L}(\bar{x}, \lambda') \geq \text{OPT} - 2\mathcal{R}_p$$

so $\alpha \leq 2\mathcal{R}_p$ as desired. □

Remark 5.3.18. *We stress that Theorem 5.3.17 bounds the sum of the violations over all coupling constraints. In Section 5.5, we discuss how to modify the solution to instead give a stronger bound on the* maximum *violation over any coupling constraint at a small cost to the objective value.*

Applying Theorem 5.3.17 to the electricity scheduling LP, we immediately have the following result.

Corollary 5.3.19. *With probability at least* $1-\beta$, PrivDude *run on the electricity scheduling LP produces an electricity schedule* $\bar{x}$ *that*

- *satisfies all demand constraints exactly,*
- *exceeds the power supply constraints by*

$$O\left(\frac{QT\sqrt{d_{max}\log(nd_{max}/\delta)}}{\epsilon}\log\frac{nd_{max}TQ}{\beta}\right)$$

  in total, over all time slots, and

- *achieves welfare at least* $\mathrm{OPT}-\alpha$ *for*

$$\alpha = O\left(\frac{QT\sqrt{d_{max}\log(nd_{max}/\delta)}}{\epsilon}\log\frac{nd_{max}TQ}{\beta}\right),$$

  where OPT *the optimal objective value.*

5.4. Examples

In this section, we illustrate the general bounds for PrivDude by instantiating them on several example problems. For each example, we present the problem description and the relevant parameters (gradient sensitivity, dual bounds, and width), and then state the guarantee we get on the quality of the solution produced by PrivDude.

While some examples are combinatorial optimization problems, our instantiations of PrivDude

Problems	Relevant Parameters	Welfare / Cost (OPT $\pm$)	Constraint Violation
d-demand Allocation	d: max bundle size; k: # goods	$\tilde{O}(kd/\varepsilon)$	$\tilde{O}(kd/\varepsilon)$
Multi-commodity Flow	L: longest path; m: # edges	$\tilde{O}\left(m\sqrt{L}/\varepsilon\right)$	$\tilde{O}\left(m\sqrt{L}/\varepsilon\right)$
Multi-dimensional Knapsack	v_i: value; $\{w_{ij}\}$: weights k: # resources	$\tilde{O}\left(k^{3/2}\max_{i,j}\frac{v_i}{w_{ij}}/\varepsilon\right)$	$\tilde{O}\left(k^{3/2}/\varepsilon\right)$
Allocation with shared resources	m: # projects; k: # resources d: # resources a project needs	$\tilde{O}\left(md^{3/2}/\varepsilon\right)$	$\tilde{O}\left(md^{3/2}/\varepsilon\right)$
Aggregative Games Equilibrium LP	k: dimension of aggregator; γ: sensitivity of aggregator	N/A	$\tilde{O}\left(k^{3/2}\gamma/\varepsilon\right)$

Table 2: Instantiations of PrivDude to different optimization problems.

only produce fractional solutions. To simplify the presentation, we will not discuss rounding techniques here. In Section 5.5, we present a extension of PrivDude to privately round the fractional solution with a small additional loss.

5.4.1. The d-Demand Allocation Problem

Consider a market with n agents, a collection of goods G of k different types, and let s_j be the supply of good j. We assume that the agents have *d-demand valuations* over bundles of goods, i.e., they demand bundles of size no more than d. Let $\mathcal{B} = \{S \subseteq G \mid |S| \leq d\}$ denote the set of all bundles with size no more than d. For each $S \in \mathcal{B}$ and $i \in [n]$, we write $v^{(i)}(S)$ to denote agent i's valuation on S; we assume that $v^{(i)}(S) \in [0,1]$. We are interested in computing a welfare-maximizing allocation:

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{n}\sum_{S\in\mathcal{B}} v^{(i)}(S)\cdot x^{(i)}(S) \\
\text{subject to} \quad & \sum_{i=1}^{n}\sum_{S\ni j} x^{(i)}(S) \leq s_j \qquad\qquad (\text{for } j\in[k]) \\
& \sum_{S\subseteq\mathcal{B}} x^{(i)}(S) \leq 1, \quad x^{(i)}(S)\geq 0 \quad (\text{for } i\in[n], S\in\mathcal{B}).
\end{aligned}$$

The private data lies in agents' valuations over the bundles, and two problem instances are neighboring if they differ by any agent i's valuations. Since each agent demands at most d items, the gradient sensitivity σ is at most $\sqrt{2}d$, and the width is bounded by nd. Also, the problem has a dual bound $\tau = 1$, as each agent's valuation is bounded by 1.

Corollary 5.4.1. *With probability at least* $1-\beta$, $\mathsf{PrivDude}(\cdot, \sqrt{2}d, 1, nd, \varepsilon, \delta, \beta)$ *computes a fractional allocation* $\bar{x}$ *such that the total supply violation is bounded by*

$$\alpha = O\left(\frac{kd \log^{1/2}(nd/\delta)}{\varepsilon} \log \frac{ndk}{\beta}\right) \quad \textit{with welfare} \quad \sum_{i=1}^{n} \sum_{S \subseteq G} v^{(i)}(S) \cdot \bar{x}^{(i)}(S) \geq \mathrm{OPT} - \tau\alpha = \mathrm{OPT} - \alpha.$$

Note that the *average* violation per good is $\tilde{O}(d/\epsilon)$. See Section 5.5 for a method that solves this problem with *no* constraint violations, at a small cost to the objective.

Remark 5.4.2. *Our result gives an affirmative answer to the open problem posed by Hsu et al. [2014a] (Chapter 4): can the allocation problem be solved privately for more general valuation functions beyond the class of gross substitutes (GS)? Our welfare and supply violation bounds are incomparable with the ones in their work, which assume GS valuations. For a detailed discussion, see Section 5.5.*

Note that there are exponentially many primal variables (in d), but our error bound is independent of the number of variables. Moreover, we can implement PrivDude *efficiently given a* demand oracle *for each player, which, for any given item prices* $\{\lambda_j\}$*, returns a bundle* $B \in \arg\max_{S \in \mathcal{B}}(v^{(i)}(S) - \sum_{j \in S} \lambda_j)$ *for each agent* i*.*

5.4.2. Multi-Commodity Flow

While a broad class of combinatorial problems are instantiations of the d-demand allocation problem, some problems have special structure and more compact representations. For example, consider the following multi-commodity flow problem over a network $G(V, E)$. There are m edges and n agents, and each agent needs to route 1 unit of flow from its source s_i to its destination t_i. We assume that for any agent i, any path from s_i to t_i has length bounded by L. For each edge $e \in E$, there is an associated cost $c_e^{(i)}$ if an agent i uses

that edge, and we assume that $c_e^{(i)} \in [0,1]$ for all e and i. Also, each edge e has a *capacity constraint*: the amount of flow on edge e should be no more than q_e. The problem can be written as the following LP:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} \sum_{e \in E} c_e^{(i)} \cdot x_e^{(i)} \\ \text{subject to} \quad & \sum_{i=1}^{n} x_e^{(i)} \le q_e && \text{(for each } e \in E\text{)} \\ & x^{(i)} \text{ forms a } (s_i, t_i)\text{-flow} && \text{(for each } i \in [n]\text{).} \end{aligned}$$

The private data lies in each agent's costs on the edges and its source and destination. Since each agent only routes 1 unit of flow, the gradient sensitivity of the problem is bounded by $\sigma = \sqrt{2L}$.

The problem has a dual bound $\tau = 1$, and its width is bounded by n.

Corollary 5.4.3. *With probability at least* $1 - \beta$, $\mathsf{PrivDude}(\cdot, \sqrt{2L}, 1, n, \varepsilon, \delta, \beta)$ *computes a fractional flow* $\bar{x}$ *such that the total capacity violation is bounded by*

$$\alpha = O\left(\frac{m\sqrt{L} \log^{1/2}(n/\delta)}{\varepsilon} \log \frac{nm}{\beta} \right),$$

and the resulting fractional flow has total cost at most $\mathrm{OPT} + \alpha\tau = \mathrm{OPT} + \alpha$.

Note again that this is the *total* violation summed over all edges. The *average* violation per edge is smaller by a factor of m: $\tilde{O}(\sqrt{L}/\epsilon)$. See Section 5.5 for a method that solves this problem with *no* constraint violations, at a small cost to the objective.

5.4.3. Multi-Dimensional Knapsack

In a multi-dimensional knapsack problem, there are a set of n items with values $v_i \in [0,1]$ and k different resources with capacities $c_i > 0$. Each item i requires an amount $w_{ij} \in [0,1]$ of each resource j. The goal is to select a subset of items to maximize the sum value while

satisfying the resource constraints:

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} \sum_{e \in E} v_i \cdot x^{(i)} \\ \text{subject to} \quad & \sum_{i=1}^{n} w_{ij}\, x^{(i)} \le c_j \quad \text{(for each } j \in [k]) \\ & x^{(i)} \in [0,1] \qquad \text{(for each } i \in [n]). \end{aligned}$$

Each agent's private data consists of both the value of her job v_i and its resource demands $\{w_{ij}\}$. The gradient sensitivity is bounded by $\sigma = \sqrt{k}$, because each item can consume at most 1 unit of each resource. The problem has a dual bound $\tau = \max_{i,j} \frac{v_i}{w_{ij}}$, and the width is bounded by n.

Corollary 5.4.4. *With probability at least* $1-\beta$, $\mathsf{PrivDude}(\cdot, \sqrt{k}, 1, n, \varepsilon, \delta, \beta)$ *computes a fractional assignment such that the total violation in the resource constraints is bounded by*

$$\alpha = O\left(\frac{k\sqrt{k} \log^{1/2}(n/\delta)}{\varepsilon} \log \frac{nk}{\beta} \right),$$

and has total profit at least

$$\mathrm{OPT} - \tau\alpha = \mathrm{OPT} - \alpha \cdot \max_{i,j} \frac{v_i}{w_{ij}}.$$

5.4.4. Allocations with Shared Resources

We now give an example with auxiliary decision variables that are not associated with private data. Suppose we have n agents, m projects, and k different resources. Each agent i has private valuations $\{v_{ij}\}$ over the projects. Each project requires a set of resources in R_j, but the resources can be shared between different projects. A unit of resource r has cost c_r, and for any project j with e_j enrolled agents, we require at least e_j units of resources r for every $r \in R_j$. We also assume that each project requires at most d distinct resources, and so the number of coupling constraints is bounded by md. We further assume that $v_{ij}, c_r \in [0,1]$ for all i, j and r. Our goal is to match people to projects so that the welfare of the agents

minus the total cost of the resources is maximized, as illustrated by the following linear program:

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{n}\sum_{j=1}^{m} v_{ij}\, x_{ij} - \sum_{r=1}^{k} c_r\, y_r \\
\text{subject to} \quad & \sum_{i=1}^{n} x_{ij} \le y_r && (\text{for } j \in [k] \text{ and } r \in R_j) \\
& \sum_{j=1}^{k} x_{ij} \le 1, \quad x_{ij} \ge 0 && (\text{for } i \in [n], j \in [k]).
\end{aligned}$$

The private data lies in the valuations of the agents over the projects, and two problem instances are neighboring if they only differ in some agent i's preferences over the projects. To fit this problem into our general framework, we interpret the variables $\{y_r\}$ as the "public" variables, controlled by "agent 0". The gradient sensitivity is bounded by $\sigma = \sqrt{2d}$. The problem has a dual bound $\tau = 1$, and the width of the problem is bounded by $w = n$.

Corollary 5.4.5. *With probability at least* $1-\beta$, $\mathsf{PrivDude}(\cdot, \sqrt{2d}, 1, n, \varepsilon, \delta, \beta)$ *computes a fractional allocation* $\bar{x}$ *such that the total resource violation (resource shortage across all projects) is bounded by*

$$\alpha = O\left(\frac{md\sqrt{d}\log^{1/2}(n/\delta)}{\varepsilon}\log\frac{nmd}{\beta}\right),$$

and the project matching along with the resource allocation gives welfare at least $\mathrm{OPT} - \alpha\tau = \mathrm{OPT} - \alpha$.

5.4.5. Equilibrium Computation in Aggregative Games

A recent paper by Cummings et al. [2015] showed that mixed strategy equilibria in *aggregative games*[4] can be computed using an algorithm that repeatedly solves a feasibility

[4] They consider multi-dimensional aggregative games, a broad class of games that generalizes both anonymous games and weighted congestion games. For more details, see Cummings et al. [2015].

linear programs of the following form:

$$\begin{aligned}
\sum_{i=1}^{n}\sum_{\ell=1}^{m} c_{i\ell}^{j} \cdot x_{i\ell} &\le \hat{s} \quad \text{(for all } j \in [k]\text{)} \\
-\sum_{i=1}^{n}\sum_{\ell=1}^{m} c_{i\ell}^{j} \cdot x_{i\ell} &\le -\hat{s} \quad \text{(for all } j \in [k]\text{)} \\
x^{(i)} &\in B_i \quad \text{(for all } i \in [n]\text{).}
\end{aligned}$$

Each agent i controls the variables $x^{(i)} = (x_{i1}, \ldots, x_{im})$, which forms a probability distribution over m actions. We assume that each coefficient in the coupling constraint is bounded by $|c_{i\ell}^{j}| \le \gamma$. Note that we can add the objective $\min_x 0$ the LP without changing the problem, so it can be framed as a linearly separable convex program. In particular, the gradient sensitivity is bounded by $\sqrt{k}\gamma$. Since the objective function is a constant, any positive number is a dual bound for this problem, so we could use $\tau = 1$ as a dual bound. Also, the width is bounded by γn.

Corollary 5.4.6. *With probability at least* $1-\beta$, $\mathsf{PrivDude}(\cdot, \sqrt{k}\gamma, 1, \gamma n, \varepsilon, \delta, \beta)$ *outputs a mixed strategy profile that has total violation across all of the constraints bounded by*

$$\alpha = O\left(\frac{k\sqrt{k}\gamma \log^{1/2}(1/\delta)}{\varepsilon} \log \frac{nk\gamma}{\beta}\right).$$

Remark 5.4.7. *This leads to an improvement over the private LP solver in Cummings et al. [2015], which gives a violation bound of* $\tilde{O}\left(\frac{\sqrt{n}\gamma}{\sqrt{\varepsilon}}\right)$. *Since* n *is large and* k *is a constant in their setting, our violation bound is considerably better with no the polynomial dependence on* n.

5.5. Achieving Exact Feasibility

When the convex program has additional structure, we can extend PrivDude to achieve additional guarantees. In this section, we'll discuss two extensions to PrivDude: guaranteeing exact feasibility.

We will continue to work with packing linear programs with a null action (Definition 6.3.1), assuming one more thing.

Assumption 5.5.1. *We will consider classes of packing linear programs $\mathcal{O} = (S, v, c, b)$ with one extra condition: Each $S^{(i)}$ is a polytope such that at each vertex x, for all j, $c_j^{(i)}(x) = 0$ or $c_j^{(i)}(x) \geq L > 0$. This parameter L is valid for the entire class; in particular, it does not depend on private data.*

Our modification to the solution of PrivDude will work in two steps. Instead of tightening the constraints like TightDude, we run PrivDude on the original linear program. Similar to TightDude, each agent will then round her solution by selecting a uniformly random best response from her set of best responses $x_1^{(i)}, \ldots, x_T^{(i)}$. To handle the constraint violation, we will maintain a differentially private flag on each constraint, which is raised when the constraint goes tight. In order, we will take agent i's rounded solution if the flag is down for every constraint she contributes to, i.e., for every j with $\left\langle c_j^{(i)}, x^{(i)} \right\rangle > 0$. Otherwise, she goes *unserved*: we give her solution 0. The full code is in Algorithm 9.

Let's consider the first step. By almost the same analysis as for TightDude, we can show that the rounding procedure degrades the objective and violates the constraints by only a small amount (past what was guaranteed by Theorem 5.3.17).

Theorem 5.5.2. *Let $\beta > 0$ be given. Suppose each agent i independently and uniformly at random selects $\tilde{x}^{(i)}$ from $x_1^{(i)}, \ldots, x_T^{(i)}$. Then with probability at least $1 - \beta$,*

- *the objective satisfies $\langle v, \tilde{x} \rangle \geq \mathrm{OPT} - \alpha$, for*

$$\alpha = O\left(\left(\frac{k\tau\sigma}{\epsilon} + \sqrt{V \cdot \mathrm{OPT}}\right)\log\left(\frac{wk}{\beta}\right)\log^{1/2}\left(\frac{w}{\delta}\right)\right);$$

  and

Algorithm 9 RoundDude$(\mathcal{O}, \sigma, \tau, w, \varepsilon, \delta, \beta)$

Input: Packing linear program $\mathcal{O} = (S, v, c, b)$ with n agents and k coupling constraints, gradient sensitivity bounded by σ, a dual bound τ, width bounded by w, and privacy parameters $\varepsilon > 0, \delta \in (0, 1/2)$, and confidence parameter $\beta \in (0, 1)$.

Initialize:

$$\delta' := \frac{\delta}{2}, \qquad \varepsilon' := \frac{\varepsilon}{2\sqrt{8T \ln(1/\delta')}}, \qquad \zeta := \frac{8C_\infty(\log n + \log(3k/\beta))}{\epsilon'},$$

$$T_j := b_j - \zeta, \quad F_j := \mathsf{Sparse}(\varepsilon', T_j) \text{ for } j \in [k].$$

Run PrivDude:

$$(\bar{x}, \bar{\lambda}) := \mathsf{PrivDude}(\mathcal{O}, \sigma, \tau, w, \epsilon/2, \delta/2, \beta/3).$$

for each agent $i = 0 \dots n$:

Select $\tilde{x}^{(i)}$ uniformly at random from best responses $x_1^{(i)}, \dots, x_T^{(i)}$.

if $F_j = \top$ for some constraint with $c_j^{(i)}(\tilde{x}^{(i)}) > 0$:

Set $\hat{x}^{(i)} := 0$.

else Set $\hat{x}^{(i)} := \tilde{x}^{(i)}$.

Query each sparse vector j with

$$q_j^{(i)} := \sum_{l=0}^{n} c_j^{(l)}(\hat{x}^{(l)}).$$

Output: $\hat{x}^{(i)}$ to agents $i \in [n]$.

- *the total constraint violation is bounded by*

$$\sum_{j=1}^{k} \left(\left\langle c_j, \tilde{x} \right\rangle - b_j \right)_+ = O\left(\sqrt{kC_\infty \|b\|_1 \log\left(\frac{k}{\beta}\right)} \right),$$

as long as

$$\|b\|_1 = \Omega\left(\frac{k\sigma}{\epsilon} \log^2\left(\frac{wk}{\beta}\right) \log\left(\frac{w}{\delta}\right) \max\left\{ \frac{\sigma}{\epsilon}, 1 \right\} \right).$$

Proof. Let $\bar{x}$ be the output from the $(\epsilon/2, \delta/2)$-private PrivDude. Note that the objective $\langle v, \bar{x} \rangle$ is the sum of i independent random variables, each bounded in $[0, V]$. By Theo-

rem 5.3.17, we can lower bound the expected objective:

$$\mathbb{E}[\langle v, \tilde{x}\rangle] = \langle v, \tilde{x}\rangle \geq \mathrm{OPT} - \frac{160\sqrt{8}k\tau\sigma}{\epsilon}\log\left(\frac{4w^2k}{\beta}\right)^2\log\left(\frac{2}{\delta}\right),$$

with probability at least $1-\beta/2$. Applying the concentration bound (Theorem 6.3.4), we have

$$\begin{aligned}\langle v, \tilde{x}\rangle &> \mathrm{OPT} - \frac{160\sqrt{8}k\tau\sigma}{\epsilon}\log\left(\frac{4w^2k}{\beta}\right)^2\log\left(\frac{2}{\delta}\right) - \sqrt{2V\cdot\mathrm{OPT}\log\left(\frac{2(k+1)}{\beta}\right)} \\ &= \mathrm{OPT} - O\left(\left(\frac{k\tau\sigma}{\epsilon} + \sqrt{V\cdot\mathrm{OPT}}\right)\log\left(\frac{wk}{\beta}\right)\log^{1/2}\left(\frac{w}{\delta}\right)\right)\end{aligned}$$

with probability at least $1-\beta/2(k+1)$.

For the constraints, define $y_j = (\langle c_j\tilde{x}\rangle - b_j)_+$ to be the constraint violation for j, if any. For each constraint, we can bound the expected left-hand side of each constraint for $\tilde{x}$:

$$\mathbb{E}\left[\langle c_j, \tilde{x}\rangle\right] = \langle c_j, \tilde{x}\rangle \leq b_j + y_j.$$

Since $\langle c_j, \tilde{x}\rangle$ is the sum of i independent random variables in $[0, C_\infty]$, applying Theorem 6.3.4 gives:

$$\langle c_j, \tilde{x}\rangle < b_j + y_j + \sqrt{3(b_j+y_j)C_\infty\log\left(\frac{2(k+1)}{\beta}\right)}$$

with probability at least $\beta/2(k+1)$. So, the total constraint violation is bounded by

$$\begin{aligned}\sum_{j=1}^{k}\left(\langle c_j, \tilde{x}\rangle - b_j\right)_+ &\leq \sum_{j=1}^{k} y_j + \sqrt{3(b_j+y_j)C_\infty\log\left(\frac{2(k+1)}{\beta}\right)} \\ &\leq \left(\sum_{j=1}^{k} y_j\right) + \sqrt{3kC_\infty\log\left(\frac{2(k+1)}{\beta}\right)\left(\|b\|_1 + \sum_{j=1}^{k} y_j\right)},\end{aligned}$$

where the second step is by Jensen's inequality. The sum of y_j is the total constraint viola-

tion of $\bar{x}$, which is bounded by Theorem 5.3.17:

$$\sum_{j=1}^{k} y_j \leq \frac{160\sqrt{8}k\sigma}{\epsilon} \log\left(\frac{4w^2k}{\beta}\right) \log^{1/2}\left(\frac{2w}{\delta}\right) \ll \|b\|_1,$$

where we have assumed that the constraint violation guarantees for PrivDude are non-trivial.[5] Then,

$$\begin{aligned}\sum_{j=1}^{k} \left(\left\langle c_j, \tilde{x}\right\rangle - b_j\right)_+ &\leq \frac{160\sqrt{8}k\sigma}{\epsilon} \log\left(\frac{4w^2k}{\beta}\right) \log^{1/2}\left(\frac{2w}{\delta}\right) + \sqrt{6kC_\infty\|b\|_1 \log\left(\frac{2(k+1)}{\beta}\right)} \\ &= O\left(\sqrt{kC_\infty\|b\|_1 \log\left(\frac{k}{\beta}\right)}\right)\end{aligned}$$

with probability at least $1 - \beta/2(k+1)$; the last step holds since we have assumed

$$\frac{k\sigma^2}{\epsilon^2} \log^2\left(\frac{wk}{\beta}\right) \log\left(\frac{w}{\delta}\right) \ll \|b\|_1.$$

Taking a union bound over the $k+1$ Chernoff bounds, everything holds with probability at least $1 - \beta/2$. Since PrivDude succeeds with probability $1 - \beta/2$, the total failure probability is $1 - \beta$. □

Now, let's consider the second step. To maintain the flags on each constraint, we will use k copies of the *sparse vector mechanism* [Dwork and Roth, 2014]. Recall that This standard mechanism from differential privacy takes a numeric threshold and a sequence of (possibly adaptively chosen) queries. Sparse vector outputs $\perp$ while the current query has answer substantially less than the threshold, and outputs $\top$ and halts when the query has answer near or exceeding the threshold. See Section 2.3 for the details.

Our queries will measure how large each constraint is for the users who have already submitted their solution $x^{(i)}$, and the threshold will be slightly less than the constraint

[5] If the constraint violation for PrivDude itself is already too big, then there is no hope for getting non-trivial constraint violation for RoundDude.

bound b_j. We want to argue two things: (a) sparse vector doesn't halt while the constraint is at least α away from being tight (for some α to be specified), and (b) each constraint ends up feasible.

The following result shows show how much objective RoundDude loses in order to guarantee exact feasibility.

Theorem 5.5.3. *Let $\beta \in (0,1)$ be given. With probability at least $1-\beta$,* RoundDude *run on a packing linear program produces a solution exactly satisfying all the constraints, and with objective at least* $\mathrm{OPT} - \alpha$*, for*

$$\alpha = O\left(\sqrt{V}\log\left(\frac{k}{\beta}\right)\left(\sqrt{\mathrm{OPT}} + \frac{\sqrt{V}}{L}\left(\frac{C_\infty \log n}{\epsilon} + \sqrt{kC_\infty \|b\|_1}\right)\right)\right),$$

as long as

$$\|b\|_1 = \Omega\left(\frac{k\sigma}{\epsilon}\log^2\left(\frac{wk}{\beta}\right)\log\left(\frac{w}{\delta}\right)\max\left\{\frac{\sigma}{\epsilon}, 1\right\}\right).$$

Proof. Let $\hat{x}$ be the output. Since we make at most n queries to every flag and each query is C_∞-sensitive, by Lemma 2.3.4 and a union bound over all k flags, the final left-hand side of each constraint is at most

$$\left\langle c_j, \hat{x}\right\rangle \le T_j + \frac{32\sqrt{2}C_\infty(\log n + \log(3k/\beta)\sqrt{\log(2/\delta)})}{\epsilon} = b_j,$$

with probability at least $1 - \beta/3$; in particular, $\hat{x}$ is strictly feasible. Furthermore, each constraint with raised flag satisfies

$$\left\langle c_j, \hat{x}\right\rangle \ge T_j - \frac{32\sqrt{2}C_\infty(\log n + \log(3k/\beta)\sqrt{\log(2/\delta)})}{\epsilon} = b_j - 2\zeta.$$

Now, an agent is only unserved if she contributes to a violated constraint. Since the best-response problem of each agent is to maximize a linear function over a polytope $S^{(i)}$, all best-responses are vertices. So, an unserved agent contributes at least $L > 0$ to violated constraints, and each unserved bidder reduces the total constraint violation by at least L.

By Theorem 5.5.2, with probability at least $1 - 2\beta/3$, the total constraint violation of $\tilde{x}$ is:

$$\sum_{j=1}^{k} \left(\left\langle c_j, \tilde{x} \right\rangle - b_j \right)_+ \leq 3\sqrt{kC_\infty \|b\|_1 \log\left(\frac{k}{\beta}\right)}.$$

Now the final output $\hat{x}$ has no constraint violation, and has reduced the right-hand side of each constraint by at most 2ζ. So, the number of agents who are unserved is at most

$$U \leq \frac{1}{L}\left(2\zeta + +3\sqrt{kC_\infty \|b\|_1 \log\left(\frac{k}{\beta}\right)}\right).$$

Since each unserved agent contributes at most V to the objective, the final output $\hat{x}$ reduces the objective of $\tilde{x}$ by at most UV, so

$$\begin{aligned}
\langle v, \hat{x} \rangle &\geq \mathrm{OPT} - \frac{160\sqrt{8}k\tau\sigma}{\epsilon} \log\left(\frac{6w^2k}{\beta}\right) \log^{1/2}\left(\frac{2w}{\delta}\right) \\
&\quad - \sqrt{2V \cdot \mathrm{OPT} \log\left(\frac{3(k+1)}{\beta}\right)} \\
&\quad - \frac{V}{L}\left(\frac{64\sqrt{2}C_\infty(\log n + \log(3k/\beta)\sqrt{\log(2/\delta)})}{\epsilon} + 3\sqrt{kC_\infty \|b\|_1 \log\left(\frac{k}{\beta}\right)}\right) \\
&= \mathrm{OPT} - O\left(\sqrt{V}\log\left(\frac{k}{\beta}\right)\left(\sqrt{\mathrm{OPT}} + \frac{\sqrt{V}}{L}\left(\frac{C_\infty \log n}{\epsilon} + \sqrt{kC_\infty \|b\|_1}\right)\right)\right).
\end{aligned}$$

With probability at least $1 - \beta$, sparse vector is accurate, the rounding succeeds, and PrivDude succeeds. □

To show privacy, we use a result about the privacy of sparse vector.

Theorem 5.5.4 (see e.g., Dwork and Roth [2014] for a proof)**.** *Let $\epsilon > 0$.* Sparse(ϵ, T) *is ϵ-differentially private.*

Privacy of RoundDude follows directly.

Theorem 5.5.5. *Let $\epsilon > 0, \delta \in (0, 1/2)$. Then,* RoundDude *satisfies (ϵ, δ)-joint differential privacy.*

Proof. By Theorem 5.5.4 and Theorem 2.2.4, the flags are $(\epsilon/2, \delta/2)$-differentially private. Theorem 5.3.7 shows that the result of PrivDude is $(\epsilon/2, \delta/2)$-jointly differentially private. By the billboard lemma (Lemma 4.3.2) and composition (Theorem 2.2.4), RoundDude is (ϵ, δ)-jointly differentially private. □

Comparison with Chapter 4 Like for TightDude, we can compare the welfare guarantee of RoundDude to the welfare guarantee of PAlloc on the d-demand allocation problem. As discussed there in Corollary 5.4.1, $\tau = 1$ is a dual bound, $\sigma = d\sqrt{2}$ bounds the gradient sensitivity, and $w = n$ bounds the width. The maximum welfare for any agent is $V = 1$ and each agent contributes at most $C_\infty = 1$ towards each coupling constraint. Each agent's feasible set is simply the simplex $\{x : \mathbb{R}^m \mid \sum_{i=1}^m x_i \leq 1\}$, so the minimum non-zero coordinate at any vertex is $L = 1$. Applying Theorem 5.5.3, we can lower bound the welfare of RoundDude on a d-demand problem.

Corollary 5.5.6. *With high probability,* RoundDude *run on a d-demand problem with supply s for each good finds a solution with welfare at least*

$$\text{OPT} - \tilde{O}\left(\frac{k\sqrt{s}}{\epsilon}\right)$$

and exactly meets the supply constraints, as long as $s = \Omega(d^2/\epsilon^2)$, ignoring logarithmic factors.

Proof. Note that the sum of the scalars $\|b\|_1$ in the d-demand problem is simply the total number of items sk. The welfare theorem for RoundDude (Theorem 5.5.3) holds as long as

$$sk = \|b\|_1 \geq \tilde{\Omega}\left(\frac{k\sigma}{\epsilon} \max\left\{\frac{\sigma}{\epsilon}, 1\right\}\right).$$

As discussed in Section 5.4, the gradient sensitivity for the d-demand problem is bounded by $\sigma = d\sqrt{2}$. So, we need $s \geq d^2/\epsilon^2$. The welfare guarantee follows from Theorem 5.5.3, plugging in parameters $V = L = C_\infty = 1$ and noting that the maximum welfare OPT is at most the total number of goods sk, so $\text{OPT} \leq \|b\|_1$. □

While the welfare guarantees are somewhat incomparable (see discussion for TightDude), we can try to make a rough comparison. Take $\alpha \approx (d\epsilon)^{1/3}$, and say the supply $s \approx d^3/\alpha^3\epsilon \approx d^2/\epsilon^2$ (the minimum needed for both Lemma 4.4.8 and Corollary 5.5.6 to apply). Then by Corollary 5.5.6, RoundDude gets welfare at least

$$\mathrm{OPT} - \tilde{O}\left(\frac{kd}{\epsilon^2}\right)$$

versus $\mathrm{OPT} - \alpha n \approx \mathrm{OPT} - n(d\epsilon)^{1/3}$. Thus, RoundDude improves for larger n:

$$n \gg \frac{kd^{2/3}}{\epsilon^{7/3}}.$$

CHAPTER 6

Privacy as a Tool for Mechanism Design

6.1. Introduction

In this chapter, we will continue to study the same type of distributed optimization problems in Chapter 5, but under the setting where the agents have preferences over their parts of the final solution, and the objective of the convex program is to maximize the total value of all agents (*social welfare*). We can guarantee asymptotic truthfulness at little additional cost to the approximation factor, with only minor modifications to the algorithm PrivDude. By taking our approximately optimal primal/dual solution pair and making a small modification (losing a bit more in approximate feasibility), we can treat the dual variables as prices. Then, every agent is allocated her favorite set of primal variables given her constraints and the prices.[1]

In general, this property would not be enough to guarantee truthfulness: while no agent would prefer a different solution at the given prices, agents may be able to *manipulate* the prices to their advantage (say, by lowering the price on the goods they want). However, in our case, the dual variables (and hence, the prices) are computed under differential privacy; they can't be substantially manipulated by any single agent. As we show, this makes truth telling an approximate dominant strategy for all players.

The connection between differential privacy and approximate truthfulness was first made by McSherry and Talwar [2007], and later extended in many papers (e.g., by Nissim et al. [2012] and Chen et al. [2013]). Huang and Kannan [2012] showed a generic (and typically computationally inefficient) method to make welfare maximization problems exactly

[1] Similar to our algorithm in Chapter 4, the allocation and prices can also be seen as an *Walrasian equilibrium* [Gul and Stacchetti, 1999].

truthful subject to standard differential privacy, but their techniques cannot solve most natural auction problems (in which each agent receives some allocation) with non-trivial social welfare guarantees.

In contrast, we show a generic method of obtaining *joint* differential privacy and approximate truthfulness for any problem that can be posed as a linearly separable convex program, which covers relaxations of combinatorial auctions as a special case. Our method is also efficient when the problem can be solved efficiently without privacy. In particular, the connection between joint differential privacy and truthfulness is more subtle. In order for a jointly differentially private computation to be approximately truthful, it must in some sense be computing an equilibrium of an underlying game [Kearns et al., 2014, Rogers and Roth, 2014, Cummings et al., 2015, Rogers et al., 2015, Kannan et al., 2015]. In our case, this corresponds to computing dual variables which serve as equilibrium prices. Using joint differential privacy circumvents criticisms of standard differential privacy as a solution concept by Nissim et al. [2012], Xiao [2013], who note that *all* strategies under standard differential privacy are approximate dominant strategies, so it does not really matter what an agent chooses to play. Under joint differential privacy, this is not the case: not *all* of an agent's strategies are approximate dominant strategies, even though truthful reporting can be made dominant.

6.2. Achieving Approximate Truthfulness

In the course of computing an approximate (primal) solution to the convex program, PrivDude also computes an approximate dual solution, which has a standard interpretation as *prices* (e.g, see Boyd and Vandenberghe [2004]). Informally, if we think of each constraint as modeling a finite resource that is divided between the variables, the dual variable for the constraint corresponds to how much each variable should "pay" for using that resource. If each agent has a real-valued *valuation function* for their portion of the solution, and the objective of the convex program is the sum of the valuation functions (a

social welfare objective), then we can make the prices interpretation precise: Each agent's solution will approximately maximize her valuation less the prices charged for using each constraint, where the prices are the approximate dual solution produced by PrivDude.

Since the dual solution (and hence the final price vector) is computed under standard differential privacy, we can also guarantee *approximate truthfulness*: an agent can't substantially increase her expected utility by misreporting her private data. Informally, this is because agents can only influence the prices to a small degree, so since the algorithm is maximizing their utility function subject to the final prices (which they have little influence on), agents are incentivized to report their true utility. One technical difficulty is that since we are computing only an approximate primal and dual solution, there may be a small number of agents who are not getting their approximately utility maximizing allocation. For approximate truthfulness, we will modify their allocations to assign them to their favorite solution at the dual prices. This may further violate some primal constraints, but only by a small amount.

Let us first define the class of optimization problems we will solve truthfully.

Definition 6.2.1. *Let the data universe be $\mathcal{X}$, and suppose there are n individuals. A class of* welfare maximization problems *is a class of convex programs $\mathcal{O} = (S, v, c, b)$ associated to $\mathcal{X}$, with the following additional properties:*

- *Bounded welfare: $v^{(i)}(x^{(i)}) \leq V$ for all agents i and points $x^{(i)} \in S$, for all feasible sets S for agent i in the class.*
- *Bounded constraints: $\sum_{j=1}^{k} c_j^{(i)}(x^{(i)}) \leq C_1$ for all agents i points $x^{(i)} \in S$, for all feasible sets S for agent i in the class.*
- *Null action: for each agent i, there exists $x^{(i)} \in S^{(i)}$ such that $v^{(i)}(x^{(i)}) = c_j^{(i)}(x^{(i)}) = 0$ for all constraints j. We call such a point a* null action for i.

Now, we can define the personal data and utility function for each player. As is typical in

the literature, agents' utilities will be quasilinear in money.

Definition 6.2.2. *Recall that agent i has a compact* feasible set $S^{(i)} \subseteq \mathbb{R}^d$, valuation function $v^{(i)} : \mathbb{R}^d \to \mathbb{R}$, *and* constraint functions $c_j^{(i)} : \mathbb{R}^d \to \mathbb{R}$. *We assume that $v^{(i)}(x^{(i)}) = 0$ for any $x^{(i)} \notin S^{(i)}$. An agent's* utility *for solution $x^{(i)}$ and price $p^{(i)}(x^{(i)}) \in \mathbb{R}$ is $v^{(i)}(x^{(i)}) - p^{(i)}(x^{(i)})$.*

Our truthful modification to PrivDude will assign each constraint a price $\lambda_j \in \mathbb{R}$, and charge each agent i price

$$p^{(i)}(x^{(i)}) = \sum_{j=1}^{k} \lambda_j c_j^{(i)}(x^{(i)}),$$

where agent i's solution is $x^{(i)}$. To guarantee truthfulness, we want every agent to have a solution that is approximately maximizing her utility given the fixed prices on constraints. This motivates the following definition:

Definition 6.2.3. *Let $\alpha \geq 0$ and prices $\lambda \in \mathbb{R}^k$ be given. An agent i with solution $x^{(i)}$ is* α-satisfied *with respect to these prices if*

$$v^{(i)}(x^{(i)}) - \sum_{j=1}^{k} \lambda_j c_j^{(i)}(x^{(i)}) \geq \max_{x \in S^{(i)}} v^{(i)}(x) - \sum_{j=1}^{k} \lambda_j c_j^{(i)}(x) - \alpha.$$

We are now ready to present our approximately truthful mechanism for welfare maximization. The idea is to run PrivDude, obtaining solution $\bar{x}$ and approximate dual solution $\bar{\lambda}$, which we take to be the prices on constraints. For α to be specified later, we change the allocation for each agent i who is not α-satisfied to a primal allocation $\tilde{x}^{(i)}$ that maximizes her utility at prices $\bar{\lambda}$. Combining $\tilde{x}$ with $\bar{x}$ for α-satisfied agents gives the final solution. We call this algorithm TrueDude, formally described in Algorithm 10.

Let's first show that the final allocation is approximately feasible, and approximately maximizing the objective (the *social welfare*). Both proofs follow by bounding the number of α-unsatisfied agents, and arguing that since the intermediate solution $(\bar{x}, \bar{\lambda})$ is an approximate equilibrium (by Corollary 5.3.16), changing the allocation of the unsatisfied agents will only degrade the feasibility and optimality a bit more (beyond what is guaranteed by

Algorithm 10 TrueDude$(\mathcal{O}, \sigma, \tau, w, \varepsilon, \delta, \beta)$

Input: Welfare maximization problem $\mathcal{O} = (S, v, c, b)$ with n agents and k coupling constraints, gradient sensitivity bounded by σ, a dual bound τ, width bounded by w, and privacy parameters $\varepsilon > 0, \delta \in (0, 1/2)$, confidence parameter $\beta \in (0, 1)$, and truthfulness parameter α.

Run PrivDude:

$$(\bar{x}, \bar{\lambda}) := \text{PrivDude}(\mathcal{O}, \sigma, \tau, w, \epsilon, \delta, \beta).$$

for each agent $i = 0 \dots n$:

Let the price of the solution be:

$$p^{(i)}(x) := \sum_{j=1}^{k} \bar{\lambda}_j c_j^{(i)}(x).$$

if $\bar{x}^{(i)}$ does not satisfy

$$v^{(i)}(\bar{x}^{(i)}) - p^{(i)}(x^{(i)}) \geq \max_{x \in S^{(i)}} v^{(i)}(x) - p^{(i)}(x) - \alpha,$$

Set

$$\bar{x}^{(i)} := \operatorname*{argmax}_{x \in S^{(i)}} v^{(i)}(x) - p^{(i)}(x).$$

Output: $\bar{x}^{(i)}$ and price $p^{(i)}(x^{(i)})$ to agents $i \in [n]$.

Theorem 5.3.17).

Lemma 6.2.4. *Let $\alpha > 0$ be given, and let $(\bar{x}, \bar{\lambda})$ be an $\mathcal{R}_p$-approximate equilibrium of the Lagrangian game. Then, the number of bidders who are not α-satisfied is at most $\mathcal{R}_p/\alpha$.*

Proof. Note that the Lagrangian can be written as the sum of agent utilities plus a constant:

$$\mathcal{L}(x, \lambda) = \sum_{i=0}^{n} \left(v^{(i)}(x^{(i)}) - \sum_{j=1}^{k} \lambda_j c_j^{(i)}(x^{(i)}) \right) + \sum_{j=1}^{k} \lambda_j b_j = \sum_{i=0}^{n} u^{(i)}(x^{(i)}, p^{(i)}) + \sum_{j=1}^{k} \lambda_j b_j,$$

so every α-unsatisfied agent that deviates to her favorite solution at prices λ increases the Lagrangian by at least α; suppose that there are m such bidders. Since $(\bar{x}, \bar{\lambda})$ is an $\mathcal{R}_p$-approximate equilibrium, we can bound the change in the Lagrangian by

$$\alpha m \leq \mathcal{L}(x, \bar{\lambda}) - \mathcal{L}(\bar{x}, \bar{\lambda}) \leq \mathcal{R}_p.$$

So, at most $m \le \mathcal{R}_p/\alpha$ agents can be α-unsatisfied. □

This immediately gives us approximate feasibility and optimality for the output of TrueDude:

Corollary 6.2.5. *Let $\beta > 0$. Running* TrueDude *on a welfare optimization problem $\mathcal{O} = (S, v, c, b)$ with gradient sensitivity bounded by σ, a dual bound τ, and width bounded by w, produces a point $\bar{x}$ such that all agents are α-satisfied, and with probability at least $1 - \beta$:*

- *the maximum violation of any constraint is at most $\mathcal{R}_p(2/\tau + C_1/\alpha)$; and*
- *the welfare $\sum_{i=0}^{n} v^{(i)}(\bar{x}^{(i)})$ is at least $\mathrm{OPT} - \mathcal{R}_p(2 + V/\alpha)$, where OPT is the optimal welfare.*

As above,

$$\mathcal{R}_p = O\left(\frac{k\tau\sigma \log^{1/2}(w/\delta)}{\varepsilon} \log \frac{wk}{\beta}\right).$$

Proof. Follows directly from Theorem 5.3.17, since any unsatisfied agent changes the welfare by at most V and has total contribution to all constraints at most C_1, and by Lemma 6.2.4 there are at most $\mathcal{R}_p/\alpha$ unsatisfied agents. □

Additionally, TrueDude is approximately *individually rational*: no agent will have large negative utility. More precisely:

Lemma 6.2.6. *Suppose* TrueDude *compute solution $x^{(i)}$ and payment $p^{(i)}(x^{(i)})$ for agent i. Then,*

$$v^{(i)}(x^{(i)}) - p^{(i)}(x^{(i)}) \ge -\alpha.$$

Proof. The null action in agent i's feasible region has utility equal to 0: the valuation is 0, and the constraint functions are all 0, so the payment is 0. The claim follows, since

$$v^{(i)}(x^{(i)}) - p^{(i)}(x^{(i)}) \ge \max_{x \in S^{(i)}} v^{(i)}(x) - p^{(i)}(x) - \alpha.$$

□

Now, let us turn to showing approximate truthfulness. We now suppose that agents may

misreport their feasible set $S^{(i)}$ and valuation function $v^{(i)}$, but not their constraint functions $c_j^{(i)}$; we want to show that agents can't gain much in expected utility by misreporting their inputs.[2]

More formally, we want to show *approximate truthfulness*. We will give a combined multiplicative and additive guarantee:

Definition 6.2.7. *Fix a class of welfare maximization problems. Consider a randomized function f that takes in a convex optimization problem $\mathcal{O} = (S, v, c, b)$, and outputs a point $x^{(i)} \in S^{(i)}$ and a price $p^{(i)}(x^{(i)}) \in \mathbb{R}$ to each agent i. We say f is* (ρ, γ)-approximately truthful *if every agent i with true feasible set $S^{(i)}$ and valuation function $v^{(i)}$ has expected utility satisfying*

$$\mathop{\mathbb{E}}_{f(\mathcal{O}')}\left[v^{(i)}(x^{(i)}) - p^{(i)}(x^{(i)})\right] \leq \rho \mathop{\mathbb{E}}_{f(\mathcal{O})}\left[v^{(i)}(x^{(i)}) - p^{(i)}(x^{(i)})\right] + \gamma,$$

where $\mathcal{O} = (S^{(i)}, S^{(-i)}, v^{(i)}, v^{(-i)}, c, b)$ is the problem when agent i truthfully reports, and $\mathcal{O}' = (S'^{(i)}, S^{(-i)}, v'^{(i)}, v^{(-i)}, c, b)$ is the problem when agent i deviates to any report $S'^{(i)}, v'^{(i)}$ in the class.

Remark 6.2.8. *We note that the inequality must hold for any data from other agents $S^{(-i)}, v^{(-i)}$, not necessarily their true data. Put another way, truthful reporting should be an approximate dominant strategy (in expectation), not just an approximate Nash equilibrium.*

We use a standard argument for achieving truthfulness via differential privacy. First, we show that conditioning on the prices being fixed, an agent can't gain much utility by changing her inputs; this holds deterministically, since agents are getting their approximately utility maximizing solution given the prices and their reported utility.

Lemma 6.2.9. *Condition on* TrueDude *computing a fixed sequence of dual variables prices $\lambda^{(1)}, \ldots, \lambda^{(T)}$, and let $\bar{\lambda} = (1/T)\sum_{t=1}^{T} \lambda^{(t)}$ be the final prices. Suppose an agent i has true feasible set $S^{(i)}$ and valuation $v^{(i)}$. Let $u^{(i)}(S, v, \{\lambda^{(t)}\})$ be i's utility (computed according to her true feasible set, valuation, and final prices $\bar{\lambda}$) for the output $\bar{x}^{(i)}$* TrueDude *computes when the*

[2] More precisely, we can allow agents to misreport their constraint functions as long as the functions are *verifiable*: they must feel their true contribution to the constraint (e.g., in terms of payments) when computing their utility.

sequence of dual variables is $\{\lambda^{(t)}\}$ and i reports feasible set S and valuation v. Then,

$$u^{(i)}(S', v', \{\lambda^{(t)}\}) \leq u^{(i)}(S^{(i)}, v^{(i)}, \{\lambda^{(t)}\}) + \alpha$$

for every set S' and valuation v'.

Proof. By Corollary 6.2.5, we know all agents are α-satisfied, so

$$u^{(i)}(S^{(i)}, v^{(i)}, \{\lambda^{(t)}\}) \geq \max_{x \in S^{(i)}} v^{(i)}(x) - \sum_{j=1}^{k} \bar{\lambda}_j c_j^{(i)}(x) - \alpha.$$

Now, we claim that

$$\max_{x \in S^{(i)}} v^{(i)}(x) - \sum_{j=1}^{k} \bar{\lambda}_j c_j^{(i)}(x) \geq v^{(i)}(x') - \sum_{j=1}^{k} \bar{\lambda}_j c_j^{(i)}(x')$$

for any x'. This is clear for $x' \in S^{(i)}$. For $x' \notin S^{(i)}$, we know $v^{(i)}(x') = 0$ so the right hand side is at most 0. Since there is a null action in $S^{(i)}$, the left hand side is at least 0, so the inequality is true for $x' \notin S^{(i)}$.

In particular, letting x' be i's solution when reporting (S', v') against dual variables $\{\lambda^{(t)}\}$, the right hand side is precisely $u^{(i)}(S', v', \{\lambda^{(t)}\})$, and we have

$$u^{(i)}(S^{(i)}, v^{(i)}, \{\lambda^{(t)}\}) \geq u^{(i)}(S', v', \{\lambda^{(t)}\}) - \alpha$$

as desired. □

Now, we use differential privacy. Since the sequence of dual variables (and final dual prices) are computed under standard differential privacy, any agent misreporting her input only has a limited effect on the prices. More formally, we will use the following standard lemma about the expected value of a differentially private mechanism.

Lemma 6.2.10 (McSherry and Talwar [2007])**.** *Suppose we have a non-negative real valued*

mechanism $\mathcal{M} : \mathcal{D} \to \mathbb{R}^+$ that is (ϵ, δ)-differentially private, and suppose that the output is bounded by $\mathcal{M}(D) \le B$ for all inputs $D \in \mathcal{D}$. Then, if $D, D' \in \mathcal{D}$ are neighboring inputs,

$$\mathbb{E}[\mathcal{M}(D)] \le e^{\epsilon}\mathbb{E}[\mathcal{M}(D')] + \delta B.$$

This is enough to argue that TrueDude is approximately truthful in expectation.

Theorem 6.2.11. *Suppose we run* TrueDude *on a class of welfare optimization problems with a dual bound τ. Then,* TrueDude *is (ρ, γ)-approximately truthful for*

$$\rho = e^{\varepsilon}, \qquad \gamma = \alpha(2e^{\varepsilon} - 1) + \delta \max\{V, C_1 \tau \sqrt{k}\}.$$

Proof. Fix valuations and feasible sets of $n-1$ agents, the constraint functions of all agents, and consider a possibly deviating agent i. We first note that since the prices $\bar{\lambda}$ have norm $\|\bar{\lambda}\|_2 \le \tau\sqrt{k}$, the maximum price charged is

$$\left\langle c^{(i)}(x), \bar{\lambda} \right\rangle \le \|c^{(i)}(x)\|_2 \|\bar{\lambda}\|_2 \le C_1 \tau \sqrt{k}$$

by Cauchy-Schwarz. Since the valuation is at most V, the utility of agent i is bounded by $\max\{V, C_1\tau\sqrt{k}\}$. As above, let $u^{(i)}(S, v, \{\lambda^{(t)}\})$ be the true utility of agent i for the outcome produced by TrueDude when reporting feasible set S and valuation v, against dual variables $\lambda^{(t)}$. By approximate individual rationality (Lemma 6.2.6), $u^{(i)}(S, v, \{\lambda^{(t)}\}) + \alpha \ge 0$. By Lemma 6.2.9,

$$u^{(i)}(S, v, \{\lambda^{(t)}\}) \le u^{(i)}(S^{(i)}, v^{(i)}, \{\lambda^{(t)}\}) + \alpha$$

for every sequence of dual variables $\{\lambda^{(t)}\}$. Let $g(\mathcal{O})$ be the sequence of dual variables produced by TrueDude on problem $\mathcal{O}$. Noting that $u^{(i)}(S, v, -)$ is a deterministic function of the sequence of dual variables, which are differentially private, Lemma 6.2.10 shows

that

$$\begin{aligned}\mathop{\mathbb{E}}_{\{\lambda^{(t)}\}\sim g(\mathcal{O}')}\left[u^{(i)}(S'^{(i)}, v'^{(i)}, \{\lambda^{(t)}\})\right] &\leq \mathop{\mathbb{E}}_{\{\lambda^{(t)}\}\sim g(\mathcal{O})}\left[u^{(i)}(S^{(i)}, v^{(i)}, \{\lambda^{(t)}\}) + \alpha\right] \\ &\leq e^{\varepsilon}\mathop{\mathbb{E}}_{\{\lambda^{(t)}\}\sim g(\mathcal{O})}\left[u^{(i)}(S^{(i)}, v^{(i)}, \{\lambda^{(t)}\}) + 2\alpha\right] + \delta\max\{V, C_1\tau\sqrt{k}\} \\ &\leq e^{\varepsilon}\mathop{\mathbb{E}}_{\{\lambda^{(t)}\}\sim g(\mathcal{O})}\left[u^{(i)}(S^{(i)}, v^{(i)}, \{\lambda^{(t)}\})\right] + \alpha(2e^{\varepsilon} - 1) + \delta\max\{V, C_1\tau\sqrt{k}\},\end{aligned}$$

where $\mathcal{O} = (S^{(i)}, S^{(-i)}, v^{(i)}, v^{(-i)}, c, b)$ is the problem when agent i truthfully reports, and $\mathcal{O}' = (S'^{(i)}, S^{(-i)}, v'^{(i)}, v^{(-i)}, c, b)$ is the problem when agent i deviates to any report $S'^{(i)}, v'^{(i)}$ in the class. □

Finally, it is straightforward to show that TrueDude is private.

Theorem 6.2.12. TrueDude *satisfies* (ϵ, δ)*-joint differential privacy.*

Proof. By the privacy of PrivDude (Theorems 5.3.7 and 5.3.8) and the billboard lemma (Lemma 4.3.2). □

6.3. Adding Exact Feasibility

For the more restricted class of *packing linear programs*, not only can we achieve approximate truthfulness, but we can also achieve exact feasibility. In fact, we can also round each agent's solution to a vertex in their feasible region S_i (which is integral for many problems); this can lead to more natural solutions for many problems (e.g., matchings and flows).

Let's first define the linear programs we will consider.

Definition 6.3.1. *A class of* packing linear programs *is a class of convex programs* $\mathcal{O} = (S, v, c, b)$ *with the following additional properties:*

- *Objective functions are linear and bounded:* $0 \leq v^{(i)}(x^{(i)}) \leq V$ *for all* $x^{(i)} \in S^{(i)}$.
- *Constraint functions are linear and bounded:* $0 \leq c^{(i)}(x^{(i)}) \leq C_\infty$ *for all* $x^{(i)} \in S^{(i)}$.

- *For each i, $0 \in S^{(i)}$.*

The parameters V, C_∞ should hold for the whole class (i.e., they are independent of the private data). To emphasize that the objective and constraints are linear, we will often write $\left\langle v^{(i)}, x^{(i)} \right\rangle$ for $v^{(i)}(x^{(i)})$, and likewise for $c_j^{(i)}$.

Our algorithm TightDude will first *tighten* the constraints, by reducing the scalars b_j by an amount ξ; we think of this step as "reserving some constraint". Next, we will run PrivDude on the reduced problem. Like TrueDude, we will let the final dual variables be the prices for the constraints, and we will reassign agents who are very unsatisfied to their favorite good at the final prices; this will guarantee approximate truthfulness. Finally, satisfied agents will *round*: they will each select one of their best responses $x_1^{(i)}, \ldots, x_T^{(i)}$ uniformly at random. Note that for packing linear problems, each best response is a vertex of an agent's private feasible region (if agents break ties by selecting vertices).

Since each agent's favorite good at the prices is also a vertex of the private feasible region, all agents will end up playing at a vertex. We will choose the amount ξ to cover the potential increase in each constraint from the unsatisfied agents and from the rounding, thereby giving exact feasibility.

We present the algorithm TightDude in Algorithm 11.

To analyze the welfare and the constraint violation, let OPT_{red} be the optimal objective of the reduced problem, and let $\kappa = \max_j \xi / b_j$ be the largest fraction of constraint we are reducing. We assume that the problem is feasible, so OPT_{red} is defined and $\kappa < 1$. We can immediately lower bound OPT_{red}.

Lemma 6.3.2. $\mathrm{OPT}_{red} \geq (1-\kappa)\mathrm{OPT}$.

Proof. For any optimal solution x^* of the original problem, $(1-\kappa)x^*$ is a feasible solution of the reduced problem with objective $(1-\kappa)\mathrm{OPT}$. The personal feasibility constraints aren't violated since $0 \in S^{(i)}$. □

We'll first briefly argue approximate truthfulness; the argument for TrueDude carries over unchanged. Note that the final output $\tilde{x}^{(i)}$ for any unsatisfied bidder is $\bar{x}^{(i)}$ in expectation, while any unsatisfied bidder gets her favorite good at the final prices. Thus, truthfulness is clear by Theorem 6.2.11.

Corollary 6.3.3. *Suppose we run* TightDude *on a class of packing linear programs with a dual bound τ. Then,* TightDude *is (ρ, γ)-approximately truthful for*

$$\rho = e^{\varepsilon}, \qquad \gamma = \alpha(2e^{\varepsilon} - 1) + \delta \max\{V, C\tau\sqrt{k}\}.$$

Next, let's look at the welfare guarantee. We can bound the possible welfare loss from reassigning unsatisfied bidders by the same argument form TrueDude, and we can use a standard Chernoff-Hoeffding bound to bound the possible welfare loss from rounding.

Theorem 6.3.4 (see e.g., Dubhashi and Panconesi [2009])**.** *Suppose $X = \sum_i X_i$ is a finite sum of of independent, bounded random variables $0 \le X_i \le M$, and $\mu_L \le \mathbb{E}[X] \le \mu_H$. Then, for any $\beta > 0$,*

$$\Pr\left[X > \mu_H + \sqrt{3M\mu_M \log(1/\beta)}\right] \le \beta \quad \textit{and} \quad \Pr\left[X < \mu_L - \sqrt{2M\mu_L \log(1/\beta)}\right] \le \beta$$

Theorem 6.3.5. *Let $\beta > 0$ be given. Then with probability at least $1 - \beta$,* TightDude *run on a packing linear program $\mathcal{O} = (S, v, c, b)$ with gradient sensitivity bounded by σ, a dual bound τ, and width bounded by w has objective satisfying $\langle v, \tilde{x}\rangle \ge \mathrm{OPT} - \alpha$, for*

$$\alpha = O\left(\kappa \cdot \mathrm{OPT} + \left(\frac{k\tau\sigma}{\epsilon}\left(1 + \frac{V}{\alpha}\right) + \sqrt{V \cdot \mathrm{OPT}}\right)\log\left(\frac{wk}{\beta}\right)\log^{1/2}\left(\frac{w}{\delta}\right)\right),$$

for

$$\kappa = \max_j \xi / b_j < 1,$$

so

$$\min_j b_j \gg \xi = \Omega\left(\|b\|_\infty^{1/2} + \frac{k\tau\sigma C_\infty}{\epsilon}\log^2\left(\frac{wk}{\beta}\right)\log^{1/2}\left(\frac{w}{\delta}\right)\left(\frac{1}{\tau} + \frac{C_\infty k}{\alpha}\right)\right).$$

Proof. Let $\bar{x}$ be the output from the $(\epsilon/2, \delta/2)$-private PrivDude on the reduced problem $\mathcal{O}_{red}$. Note that the objective $\langle v, \tilde{x} \rangle$ is the sum of i independent random variables, each bounded in $[0, V]$. We can lower bound the expected welfare with by Corollary 6.2.5; reassigning the unsatisfied bidders leads to welfare at least

$$\mathbb{E}[\langle v, \tilde{x} \rangle] = \langle v, \bar{x} \rangle \geq \mathrm{OPT}_{red} - \frac{80\sqrt{8}k\tau\sigma}{\epsilon} \log\left(\frac{4w^2k}{\beta}\right) \log^{1/2}\left(\frac{2w^2}{\delta}\right)\left(2 + \frac{V}{\alpha}\right).$$

with probability at least $1 - \beta/2$. Applying the concentration bound (Theorem 6.3.4), we have

$$\begin{aligned}
\langle v, \tilde{x} \rangle &> \mathrm{OPT}_{red} - \frac{80\sqrt{8}k\tau\sigma}{\epsilon} \log\left(\frac{4w^2k}{\beta}\right) \log^{1/2}\left(\frac{2w^2}{\delta}\right)\left(2 + \frac{V}{\alpha}\right) - \sqrt{2V \cdot \mathrm{OPT}_{red} \log\left(\frac{2}{\beta}\right)} \\
&\geq (1 - \kappa)\,\mathrm{OPT} - \frac{80\sqrt{8}k\tau\sigma}{\epsilon} \log\left(\frac{4w^2k}{\beta}\right) \log^{1/2}\left(\frac{2w^2}{\delta}\right)\left(2 + \frac{V}{\alpha}\right) - \sqrt{2V \cdot \mathrm{OPT} \log\left(\frac{2}{\beta}\right)} \\
&= \mathrm{OPT} - O\left(\kappa \cdot \mathrm{OPT} + \left(\frac{k\tau\sigma}{\epsilon}\left(1 + \frac{V}{\alpha}\right) + \sqrt{V \cdot \mathrm{OPT}}\right) \log\left(\frac{wk}{\beta}\right) \log^{1/2}\left(\frac{w}{\delta}\right)\right)
\end{aligned}$$

with probability at least $1 - \beta/2$, so everything holds with probability at least $1 - \beta$. □

Finally, let's look at the constraint violation.

Theorem 6.3.6. *Let $\beta > 0$ be given. Then with probability at least $1 - \beta$,* TightDude *run on a packing linear program $\mathcal{O} = (S, v, c, b)$ with gradient sensitivity bounded by σ, a dual bound τ, and width bounded by w has produces an exactly feasible solution as long as*

$$\kappa = \max_j \xi / b_j < 1,$$

so

$$\min_j b_j \gg \xi = \Omega\left(\|b\|_\infty^{1/2} + \frac{k\tau\sigma C_\infty}{\epsilon} \log^2\left(\frac{wk}{\beta}\right) \log^{1/2}\left(\frac{w}{\delta}\right)\left(\frac{1}{\tau} + \frac{C_\infty k}{\alpha}\right)\right).$$

Proof. Let $\bar{x}$ be the output from the $(\epsilon/2, \delta/2)$-private PrivDude on the reduced problem $\mathcal{O}_{red}$. For each constraint,note that the objective $\left\langle c_j, \tilde{x} \right\rangle$ is the sum of i independent random variables, each bounded in $[0, C_\infty]$. We can bound the expected left-hand side of each con-

straint for $\tilde{x}$ by Corollary 6.2.5; reassigning the unsatisfied bidders makes the constraints at most

$$\mathbb{E}\left[\left\langle c_j, \tilde{x}\right\rangle\right] = \left\langle c_j, \bar{x}\right\rangle \le b_j - \xi + \frac{80\sqrt{8}k\tau\sigma}{\epsilon}\log\left(\frac{4w^2k^2}{\beta}\right)\log^{1/2}\left(\frac{2w}{\delta}\right)\left(\frac{2}{\tau} + \frac{C_\infty k}{\alpha}\right)$$

with probability at least $1 - \beta/2k$, since the total amount any agent contributes to the constraints is $C_1 = C_\infty k$. Applying the concentration bound (Theorem 6.3.4), we have

$$\begin{aligned}\left\langle c_j, \tilde{x}\right\rangle &\le b_j - \xi + \frac{80\sqrt{8}k\tau\sigma}{\epsilon}\log\left(\frac{4w^2k^2}{\beta}\right)\log^{1/2}\left(\frac{2w}{\delta}\right)\left(\frac{2}{\tau} + \frac{C_\infty k}{\alpha}\right) \\ &\quad + \sqrt{3\left(b_j - \xi + \frac{80\sqrt{8}k\tau\sigma}{\epsilon}\log\left(\frac{4w^2k^2}{\beta}\right)\log^{1/2}\left(\frac{2w}{\delta}\right)\left(\frac{2}{\tau} + \frac{C_\infty k}{\alpha}\right)\right)C_\infty\log\left(\frac{2k}{\beta}\right)} \\ &\le b_j - \xi + \sqrt{3b_j} + \frac{160\sqrt{8}k\tau\sigma C_\infty}{\epsilon}\log^2\left(\frac{4w^2k^2}{\beta}\right)\log^{1/2}\left(\frac{2w}{\delta}\right)\left(\frac{2}{\tau} + \frac{C_\infty k}{\alpha}\right) \le b_j\end{aligned}$$

with probability at least $1-\beta/2k$, so taking a union bound over all k constraints, everything holds with probability at least $1 - \beta$. □

Remark 6.3.7. *While we have presented* TightDude *as achieving approximate truthulness, we can also run* TightDude *just for the rounding and exact feasibility by letting the truthfulness parameter α be large; the welfare and constraint violation bounds degrade gracefully.*

Finally, it is straightforward to show that TightDude is private.

Theorem 6.3.8. TightDude *satisfies (ϵ, δ)-joint differential privacy.*

Proof. By the privacy of PrivDude (Theorems 5.3.7 and 5.3.8) and the billboard lemma (Lemma 4.3.2). □

Comparison with Chapter 4 In Chapter 4, we give an algorithm for the d-demand allocation problem (a packing linear program) we considered in Section 5.4, but assuming additionally the *gross substitutes condition* [Gul and Stacchetti, 1999] from the economics literature on agent valuations. In this setting, the algorithm PAlloc in Chapter 4 is $(\varepsilon, 0)$-

joint differentially private algorithm with the following welfare.

While we do not have algorithms specific to the gross substitutes case, we can consider the d-demand problem (a packing linear program) with general valuations, rather than gross substitutes. As discussed in Corollary 5.4.1, $\tau = 1$ is a dual bound, $\sigma = d\sqrt{2}$ bounds the gradient sensitivity, and $w = n$ bounds the width. The maximum welfare for any agent is $V = 1$ and each agent contributes at most $C_\infty = 1$ towards any coupling constraint. While we can also achieve approximate truthfulness (unlike PAlloc), we will take α to be large; this makes TightDude non-truthful. Applying Theorem 6.3.5, we can lower bound the welfare of TightDude on a d-demand problem.

Corollary 6.3.9. *With high probability,* TightDude *run on a d-demand problem with supply s for each good finds a solution with welfare at least*

$$\mathrm{OPT} - \tilde{O}\left(\left(\frac{1}{\sqrt{s}} + \frac{kd}{s\epsilon}\right) \cdot \mathrm{OPT} + \frac{kd}{\epsilon}\right)$$

and exactly meets the supply constraints, as long as $s = \tilde{\Omega}(kd/\epsilon)$, ignoring logarithmic factors.

Proof. Note that the max scalar $\|b\|_\infty$ in the d-demand problem is simply supply for each item s. The welfare theorem for TightDude (Theorem 6.3.5) holds as long as

$$\min_j b_j = s = \tilde{\Omega}\left(\sqrt{s} + \frac{k\sigma}{\epsilon}\right).$$

As discussed in Section 5.4, the gradient sensitivity for the d-demand problem is bounded by $\sigma = d\sqrt{2}$. So, we need $s \gg kd/\epsilon$. The welfare guarantee follows from Theorem 5.5.3, plugging in parameters $V = C_\infty = 1$ and noting that $\kappa = O\left(1/\sqrt{s} + kd/s\varepsilon\right)$. □

The two algorithms are somewhat incomparable, for several reasons:

- The welfare of TightDude (Corollary 5.5.6) depends on k (but not n), while the welfare guarantee in Lemma 4.4.8 depends on n (but not k).

- The algorithm PAlloc requires the gross substitutes condition while TightDude does not.

- The algorithm PAlloc satisfies pure $(\varepsilon, 0)$-joint differential privacy, while TightDude satisfies (ϵ, δ)-joint differential privacy only for $\delta > 0$.

Nevertheless, we can try to make a rough comparison. For the algorithm PAlloc, take $\alpha \approx (d^2/k)^{1/3}$, and say the supply $s \approx d^3/\alpha^3\epsilon \approx kd/\epsilon$ (the minimum needed for both utility guarantee Lemma 4.4.8 and Corollary 5.5.6 to apply). Then by Corollary 6.3.9, TightDude gets welfare at least

$$(1 - \tilde{O}(1)) \cdot \mathrm{OPT} - \tilde{O}\left(\frac{kd}{\epsilon}\right)$$

versus $\mathrm{OPT} - \alpha n \approx \mathrm{OPT} - n(d^2/k)^{1/3}$. Thus, TightDude improves when n is large compared to k and OPT:

$$n \gg \frac{k^{4/3}}{d\epsilon} + \frac{k^{1/3}}{d^{2/3}} \cdot \mathrm{OPT}.$$

Algorithm 11 TightDude$(\mathcal{O}, \sigma, \tau, w, \varepsilon, \delta, \beta)$

Input: Packing linear program $\mathcal{O} = (S, v, c, b)$ with n agents and k coupling constraints, gradient sensitivity bounded by σ, a dual bound τ, width bounded by w, and privacy parameters $\varepsilon > 0, \delta \in (0, 1/2)$, and confidence parameter $\beta \in (0, 1)$.

Initialize:

$$\xi := \sqrt{3b_j} + \frac{160\sqrt{8}k\tau\sigma C_\infty}{\epsilon} \log^2\left(\frac{4w^2k^2}{\beta}\right) \log^{1/2}\left(\frac{2w}{\delta}\right)\left(\frac{2}{\tau} + \frac{C_\infty k}{\alpha}\right),$$

$$\mathcal{O}_{red} := \mathcal{O} \text{ with scalars } b_j - \xi.$$

Run PrivDude:

$$(\bar{x}, \bar{\lambda}) := \text{PrivDude}(\mathcal{O}_{red}, \sigma, \tau, w, \epsilon/2, \delta/2, \beta/2).$$

for each agent $i = 0 \dots n$:

Let the price of the solution be:

$$p^{(i)}(x) := \sum_{j=1}^{k} \bar{\lambda}_j c_j^{(i)}(x).$$

if $\bar{x}^{(i)}$ does not satisfy

$$v^{(i)}(\bar{x}^{(i)}) - p^{(i)}(x^{(i)}) \geq \max_{x \in S^{(i)}} v^{(i)}(x) - p^{(i)}(x) - \alpha,$$

Set

$$\tilde{x}^{(i)} := \operatorname*{argmax}_{x \in S^{(i)}} v^{(i)}(x) - p^{(i)}(x).$$

else

Select $\tilde{x}^{(i)}$ uniformly at random from best responses $x_1^{(i)}, \dots, x_T^{(i)}$.

Output: $\tilde{x}^{(i)}$ and price $p^{(i)}(x^{(i)})$ to agents $i \in [n]$.

CHAPTER 7

Privacy for the Protected (Only)

7.1. Introduction

The tension between the useful or essential gathering and analysis of data about citizens, and the privacy rights of those citizens, is at an historical peak. Perhaps the most striking and controversial recent example is the revelation that U.S. intelligence agencies systemically engage in "bulk collection" of civilian "metadata" detailing telephonic and other types of communication and activities, with the alleged purpose of monitoring and thwarting terrorist activity [Greenwald, 2013]. Other compelling examples abound, including in medicine (patient privacy vs. preventing epidemics), marketing (consumer privacy vs. targeted advertising), and many other domains.

Debates about (and models for) data privacy often have an "all or nothing" flavor: privacy guarantees are either provided to every member of a population, or else privacy is deemed to be a failure. This dichotomy is only appropriate if all members of the population have an equal right to, or demand for, privacy. Few would argue that actual terrorists should have such rights, which leads to difficult questions about the balance between protecting the rights of ordinary citizens, and using all available means to prevent terrorism.[1] A major question is whether and when the former should be sacrificed in service of the latter. Similarly, in the medical domain, epidemics (such as the recent international outbreak of Ebola [Reuters, 2014]) have raised serious debate about the clear public interest in controlling contagion versus the privacy rights of the infected and those that care for them.

[1] A recent National Academies study [Council, 2015] reached the conclusion that there are not (yet) technological alternatives to bulk collection and analysis of civilian metadata, in the sense that such data is essential in current counterterrorism practices.

The model and results in this paper represent a step towards explicit acknowledgments of such trade-offs, and algorithmic methods for their management. The scenarios sketched above can be broadly modeled by a population divided into two types. There is a *protected subpopulation* that enjoys (either by law, policy, or choice) certain privacy guarantees. For instance, in the examples above, these protected individuals might be non-terrorists, or uninfected citizens (and perhaps informants and health care professionals). They are to be contrasted with the "unprotected" or *targeted* subpopulation, which does not share those privacy assurances. A key assumption of the model we will introduce is that the protected or targeted status of individual subjects is not known, but can be discovered by (possibly costly) measures, such as surveillance or background investigations (in the case of terrorism) or medical tests (in the case of disease). Our overarching goal is to allow parties such as intelligence or medical agencies to identify and take appropriate actions on the targeted subpopulation, while also providing privacy assurances for the protected individuals who are not the specific targets of such efforts — all while limiting the cost and extent of the background investigations needed.

As a concrete example of the issues we are concerned with, consider the problem of using social network data (for example, telephone calls, emails and text messages between individuals) to search for candidate terrorists. One natural and broad approach would be to employ common graph search methods: beginning from known terrorist "seed" vertices in the network, neighboring vertices are investigated, in an attempt to grow the known subnetwork of targets.[2] A major concern is that such search methods will inevitably encounter protected citizens, and that even taking action against only discovered targeted individuals may compromise the privacy of the protected.

In order to rigorously study the trade-offs between privacy and societal interests discussed

[2] This general practice is sometimes referred to as "contact chaining": *"Communications metadata, domestic and foreign, is used to develop contact chains by starting with a target and using metadata records to indicate who has communicated with the target (1 hop), who has in turn communicated with those people (2 hops), and so on. Studying contact chains can help identify members of a network of people who may be working together; if one is known or suspected to be a terrorist, it becomes important to inspect others with whom that individual is in contact who may be members of a terrorist network."* Section 3.1 of Council [2015].

above, our work introduces a formal model for privacy of network data that provides provable assurances *only* to the protected subpopulation, and gives algorithms that allow effective investigation of the targeted population. These algorithms are deliberately "noisy" and are privacy-preserving versions of the widely used graph search methods mentioned above, and as such represent only mild (but important) departures from commonly used approaches. At the highest level, one can think of our algorithms as outputting a list of confirmed targeted individuals discovered in the network, for whom any subsequent action (e.g. publication in a most-wanted list, further surveillance or arrest in the case of terrorism; medical treatment or quarantine in the case of epidemics) will not compromise the privacy of the protected.

The key elements of our model include the following:

1. Network data collected over a population of individuals and consisting of pairwise contacts (physical, social, electronic, financial, etc.). The contacts or links of each individual comprise the private data they desire to protect. We assume a third party (such as an intelligence agency or medical organization) has direct access to this network data, and would like to discover and act upon targeted individuals.

2. For each individual, an immutable *status bit* that determines their membership status in the targeted subpopulation (such as terrorism or infection). These status bits can be discovered by the third party, but only at some nontrivial cost (such as further surveillance or medical testing), and thus there is a *budget* limiting the number of status bits that an algorithm can reveal. One might assume or hope that in practice, this budget is sufficient to investigate a number of individuals that is of the order of the targeted subpopulation size (so they can all be discovered), but considerably less than that needed to investigate every member of the general population.

3. A mathematically rigorous notion of individual data privacy based on differential privacy that provides guarantees of privacy for the network data of only the pro-

tected individuals, while allowing the discovery of targeted individuals. Informally, this notion guarantees that compared to a counterfactual world in which any protected individual *arbitrarily* changed any part of their data, or even removed themselves entirely from the computation, their risk (measured with respect to the probability of *arbitrary* events) has not substantially increased.

We emphasize two important points about our model. First, we assume that the process of investigating an individual to determine their status bit is *unobservable*, and leaks no information itself. This assumption is justified in some settings — for example, when the investigation involves secretly intercepting digital communications, like emails and phone calls, or when it involves performing tests on materials (like blood samples) or information already obtained. However, our model does not fit situations in which the investigations themselves are observable — for example, if the investigation requires interviewing an individual's family, friends and colleagues — because the very fact that an individual was chosen for an investigation (regardless of its outcome) might disclose their private data. The second point is that "privacy" is a word that has many meanings, and it is important to distinguish between the *types* of privacy that we aim to protect (see e.g. Solove's taxonomy of privacy [Solove, 2006]).

Our goal is to quantify *informational privacy* — that is, how much information about a protected individual can be deduced from the output of an analysis. However, it is important to note that the status bit investigations our algorithms make, even if unobservable, are a privacy loss that Solove calls "intrusion." Our results can be viewed as providing a quantitative trade-off between informational privacy and intrusion: using our algorithms, it is possible to guarantee more informational privacy at the cost of a higher degree of intrusion, and vice-versa.

Our main results are:

1. The introduction of a broad class of graph search algorithms designed to find and

identify targeted individuals. This class of algorithms is based on a general notion of a *statistic of proximity* — a network-based measure of how "close" a given individual v is to a certain set of individuals S. For instance, one such closeness measure is the number of short paths in the network from v to members of S. Our (necessarily randomized) algorithms add noise to such statistics in order to prioritize which status bits to query (and thus how to spend the budget).

2. A theoretical result proving a quantitative privacy guarantee for this class of algorithms, where the level of privacy depends on a measure of the *sensitivity* of the statistic of proximity to small changes in the network.

3. Extensive computational experiments in which we demonstrate the effectiveness of our privacy-preserving algorithms on real social network data. These experiments demonstrate that in addition to the privacy guarantees, our algorithms are also useful, in the sense that they find almost as many members of the targeted subpopulation as their non-private counterparts. The experiments allow us to quantify the loss in effectiveness incurred by the gain in privacy.

To our knowledge, our formal framework is the first to introduce explicit protected and targeted subpopulations with qualitatively differing privacy rights,[3] and our algorithms are the first to provide mathematically rigorous privacy guarantees for the protected while still allowing effective discovery of the targeted. More generally, we believe our work makes one of the first steps towards richer privacy models that acknowledge and manage the tensions between different levels of privacy guarantees to different subgroups.

[3]This is in contrast to the quantitative distinction proposed by Dwork and McSherry Dwork and McSherry [2010], which still does not allow for the explicit discovery of targeted individuals. We also note that our definition of privacy can be expressed in the "Blowfish privacy" framework He et al. [2014] (although this had not previously been done).

7.2. Preliminaries

7.2.1. Computational Model

We study graph search algorithms which operate on graphs $G = (V, E)$ defined over a vertex set V and edge set $E \subseteq V \times V$. The vertex set V partitioned into two fixed subsets $V = \mathcal{T} \cup \mathcal{P}$, where $\mathcal{T}$ represents the *targeted* subpopulation, and $\mathcal{P}$ represents the *protected* subpopulation. The algorithms we consider are initially given a single *seed vertex* $s \in \mathcal{T}$ (or several such vertices), and the goal of the algorithm will be to find as many other members of the targeted subpopulation $\mathcal{T}$ as possible.

The algorithm cannot directly observe which subpopulation a particular vertex $v \in V$ belongs to since otherwise the problem is trivial, but it has the ability to make a query on a vertex $v \in V$ to determine its subpopulation membership. We model this ability formally by giving the algorithm access to an *identity oracle* $\mathcal{I} : V \to \{0, 1\}$, defined such that $\mathcal{I}(v) = 1$ if and only if $v \in \mathcal{T}$. A call to this oracle is the abstraction we use to represent the possibly costly operation (instantiated in our example applications by e.g. surveillance, or medical tests) which determines whether a particular member of the population is protected or not. Because we view these operations as expensive, we want our algorithm to operate by making as few calls to this oracle as possible. Hence, the algorithm must use the network data represented by the graph G to *guide its search* for which vertices to query. This creates a source of privacy tension since the edges in this network are what we view as private information.

Thus our goal is to give algorithms which discover members of the targeted population using the edges in the network to guide their search. We wish to protect the privacy of the protected individuals: we do not want the outcome of the search to reveal too much about the edge set incident to any protected individual. However, we want to exploit the edges incident to targeted individuals in ways that will not necessarily be privacy preserving.

Formally, our network analysis algorithms take as input a network and a method by which they may query whether vertices v are members of the protected population $\mathcal{P}$ or not. The class of algorithms we consider are network search algorithms — they aim to identify some subset of the targeted population. Our formal model is agnostic as to what action is ultimately taken on the identified members (for example, in a medical application they might be quarantined, in a security application they might be arrested, etc.) From the perspective of informational privacy, all that is relevant is that which members of the targeted population we ultimately identify is observable. Hence, without loss of generality we can abstract away the action taken and simply view the output of the mechanism to be an ordered list of individuals who are confirmed to be targeted.

Any practical algorithm must limit the number of status bits that are examined. Our goal is a total number of status bit examinations that is on the order of the size of the targeted subpopulation $\mathcal{T}$, which may be much smaller than the size of the protected population $\mathcal{P}$. This is the source of the tension we study — because the number of calls to the identity oracle is limited, it is necessary to exploit the private edge set to guide our search (i.e. we cannot simply investigate the entire population), but we wish to do so in a way that does not reveal much about the edges incident to any specific protected individual.

7.2.2. Protected Differential Privacy

The privacy guarantee we provide is a variant of *differential privacy*, an algorithmic definition of data privacy. It formalizes the requirement that arbitrary changes to a single individual's private data should not significantly affect the output distribution of the data analysis procedure, and so guarantees that the analysis leaks little information about the private data of any single individual. We introduce the definition of differential privacy specialized for the network setting. We treat networks as a collection of vertices representing individuals, each represented as a list of its edges (which form the private data of each vertex). For a network G and a vertex v, let $D_v(G)$ be the set of edges incident to the vertex v in G. Let $\mathcal{G}_n$ be the family of all n-vertex networks.

Definition 7.2.1 (Vertex Differential Privacy Dwork et al. [2006], Hay et al. [2009])**.** *The networks G, G' in $\mathcal{G}_n$ are* neighboring[4] *if one can be obtained from the other by an (arbitrary) rewiring of the edges incident to a single vertex v — i.e. if for some vertex v, $D_u(G) \setminus \{(u,v)\} = D_u(G') \setminus \{(u,v)\}$ for all $u \neq v$. An algorithm $\mathcal{A}\colon \mathcal{G}_n \to \mathcal{O}$ satisfies (ε, δ)-differential privacy if for every event $S \subseteq \mathcal{O}$ and all neighboring networks $G, G' \in \mathcal{G}_n$,*

$$\Pr[\mathcal{A}(G) \in S] \leq e^{\varepsilon} \Pr[\mathcal{A}(G') \in S] + \delta.$$

Remark 7.2.2. *This definition is also known as vertex differential privacy, and is the strongest version of differential privacy for networks that is used in the literature (cp. edge differential privacy). It is a variant of a slightly more general original definition of differential privacy [Dwork et al., 2006]. Vertex differential privacy was first defined by Hay et al. [2009] and later studied by Kasiviswanathan et al. [2013] and Blocki et al. [2013].*

Recall that differential privacy promises the following: simultaneously for every individual i, and simultaneously for any event S that they might be concerned about, event S is *almost* no more likely to occur given that individual i's data is used in the computation, compared to if it were replaced by an arbitrarily different entry. Here, "almost no more likely" means that the probability that the bad event S occurs has increased by a multiplicative factor of at most e^{ϵ}, which we term the *risk multiplier*. As the privacy parameter ϵ approaches 0, the value of the risk multiplier approaches 1, meaning that individual i's data has no effect at all on the probability of a bad outcome. The smaller the risk multiplier, the more meaningful the privacy guarantee. It will be easier for us to reason directly about the privacy parameter ϵ in our analyses, but semantically it is the risk multiplier e^{ϵ} that measures the quality of the privacy guarantee, and it is this quantity that we report in our experiments.

[4] Because it is standard terminology in both graph theory and differential privacy, for us the term "neighbor" has two distinct and separate meanings: two vertices can be neighbors within a network, and two networks can be neighbors in the sense we are defining here. Which definition is intended will be clear from the context.

Differential privacy promises the same protections for *every* individual in a network, which is incompatible with our setting. We want to be able to identify members of the targeted population, and to do so, we want to be able to make arbitrary inferences from their network data. Nevertheless, we want to give strong privacy guarantees to members of the protected subpopulation. This motivates our variant of differential privacy, which redefines the neighboring relation between networks: for any protected and targeted subpopulations $\mathcal{P}$ and $\mathcal{T}$, two networks G and G' are *neighboring* if G' can be obtained by only changing a single *protected* node's edges in G. Specifically, $G = (V, E)$ and $G' = (V, E')$ are *neighbors* with respect to a partition $V = \mathcal{P} \cup \mathcal{T}$ if there exists a $v \in \mathcal{P}$ such that for all $v' \neq v$: $D_{v'}(G) \cup \{(v, v')\} = D_{v'}(G') \cup \{(v, v')\}$. Note that for neighboring graphs G and G', the edge sets in the subgraph induced on the vertices $\mathcal{T}$ must also be the same.[5]

In the following, we denote the set of all possible networks over the vertices V by $\mathcal{G}$, and denote the set of all possible outcomes of an algorithm by $\mathcal{O}$.

What this means is that we are offering no guarantees about what an observer can learn about either the status bit of an individual (protected vs. targeted), or the set of edges incident to targeted individuals. However, we are still promising that no observer can learn much about the set of edges incident to any member of the protected subpopulation. This naturally leads us to the following definition:

Definition 7.2.3 (Protected Differential Privacy)**.** *An algorithm $\mathcal{A}\colon \mathcal{G} \to \mathcal{O}$ satisfies* (ε, δ)-protected differential privacy *if for every partition of n vertices V into sets $\mathcal{P}$ and $\mathcal{T}$, for every pair of graphs G, G' that are neighbors with respect to the partition $(\mathcal{P}, \mathcal{T})$, and for any set of outcomes $S \subseteq \mathcal{O}$*

$$\Pr[\mathcal{A}(G) \in S] \leq \exp(\varepsilon) \Pr[\mathcal{A}(G') \in S] + \delta.$$

[5] The "Blowfish privacy" framework gives a general definition of privacy with different neighboring relations [He et al., 2014]. Our definition can be seen as an instantiation of this general framework. This is in contrast to other kinds of relaxations of differential privacy, which relax the worst-case assumptions on the prior beliefs of an attacker as in Bassily et al. [2013], or the worst-case collusion assumptions on collections of data analysts as in Kearns et al. [2014]. Several works have also proposed assigning different differential privacy parameters to different individuals (see e.g. Alaggan et al. [2015]) – but this is not compatible with identifying members of a targeted population.

If $\delta = 0$, we say $\mathcal{A}$ satisfies ε-protected differential privacy.

One way to interpret protected differential privacy is differential privacy applied to an appropriately defined input. Let the algorithm have two inputs: the set of edges incident to the protected vertices in $\mathcal{P}$, and the edges in $E(\mathcal{T}) = \{(u,v) \mid u,v \in \mathcal{T}\}$ (i.e. all of the other edges)[6]. In this view, protected differential privacy only requires the algorithm to be differentially private in its first argument, and not in its second. This view, formalized in the following lemma, will allow us to apply some of the basic tools of differential privacy in order to achieve protected differential privacy.

Lemma 7.2.4. *Given a graph $G = (V,E)$ and a partition its edges into $E_2 = E(\mathcal{T})$ and $E_1 = E \setminus E_2$ an algorithm $\mathcal{A}_{\mathcal{I}}(G) := \mathcal{A}_{\mathcal{I}}(E_1, E_2)$ satisfies (ε,δ)-protected differential privacy if it is (ε,δ)-differentially private in its first argument.*

To visualize the difference between standard differential privacy and protected differential privacy, see Figure 4. In particular, for algorithms satisfying standard DP (left), the addition of a single edge (dashed blue) can alter the output distribution by only a small amount. PDP is similar, except we introduce a targeted subpopulation (highlighted in red). If the added edge is between two targeted individuals, the output distribution may change arbitrarily, reflecting the fact that the targeted parties may not enjoy privacy protection. The formal definitions are stronger, in that privacy for protected individuals must be preserved even if any number of edges to them is added or deleted.

Our privacy definition promises that what an observer learns about an individual "Alice" (e.g. that Alice is in contact with a particular individual Bob, or an entire class of individuals — like members of a religious group) is almost independent of Alice's connections, so long as Alice is not herself a member of the targeted population. On the other hand, it does not prevent an observer from learning that Alice exists at all. This models a setting in which (e.g.) a national government has access to an index of all of its citizens (through

[6] It is crucial here that such a simplification can only be made for the purposes of the analysis only. Since all our algorithms are only given access to a membership oracle $\mathcal{I}$ there is no way for them to explicitly construct these two inputs without incurring a cost associated with oracle queries.

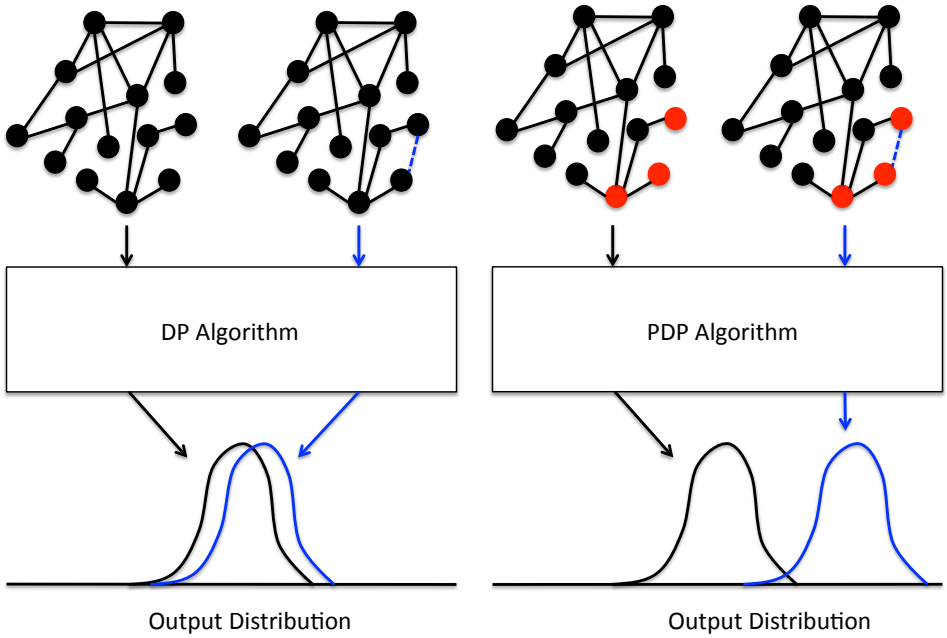

Figure 4: Informal illustration of standard Differential Privacy (DP) versus Protected Differential Privacy (PDP).

birth and immigration records), but nevertheless would like to protect information about their interactions with each other.

Remark 7.2.5. *A careful reader may already have noticed that there is a trivial graph search algorithm that achieves* 0*-protected differential privacy while outputting the entire set of targeted individuals* $\mathcal{T}$ *— it simply queries* $\mathcal{I}(v)$ *for every* $v \in V$*, and outputs every* v *such that* $\mathcal{I}(v) = 1$*. This algorithm satisfies perfect (i.e. with* $\epsilon = 0$*) protected differential-privacy because it operates* independently *of the private network* G*. The problem with this approach is that it requires querying the status of every vertex* $v \in V$*, which can be impractical both because of cost (the query might itself require a substantial investment of resources) and because of societal norms (it may not be defensible to subject every individual in a population to background checks). Hence, here we aim to design algorithms that use the graph data* G *to effectively guide the search for which vertices* v *to query. This is what leads to the tension with privacy, and our goal is to effectively trade off the privacy parameter* ϵ *with the number of queries to* $\mathcal{I}$ *that the*

algorithm must make.

7.3. Algorithmic Framework

7.3.1. Statistics of Proximity (SoP)

The key element in our algorithmic framework is the notion of a *Statistic of Proximity* (SoP), a network-based measure of how close an individual is to another set of individuals in a network.

Formally, a statistic of proximity is a function f that maps a network G, a node v, and a set of nodes $S \subseteq \mathcal{T}$ to a real number. Since the value $f(G, v, S)$ can reveal information about the links in the network, we will often need to perturb the values of these statistics with noise, calibrated with scale proportional to the *targeted sensitivity* — the maximum change in any *targeted* node's SoP relative to any set S when a *protected node*'s adjacency list is changed.

Definition 7.3.1 (Targeted Sensitivity)**.** *Let $f : \mathcal{G} \times V \times 2^{\mathcal{T}} \to \mathbb{R}$ be a statistic of proximity. The* targeted sensitivity *of f is*

$$\Delta(f) = \max_{G \sim G', t \in \mathcal{T}, S \subseteq \mathcal{T}} \left| f(G, t, S) - f(G', t, S) \right|,$$

where $G \sim G'$ indicates that G and G' are neighboring graphs in $\mathcal{G}$ relative to a fixed partition of V into $\mathcal{P}$ and $\mathcal{T}$.

Note that when computing the *targeted sensitivity* we are not concerned with the effect that a change in the edges incident on vertices in $\mathcal{T}$ has on the statistic, nor on the effect of any change on the statistic computed on vertices $v \in \mathcal{P}$.

Another quantity of interest is *impact cardinality* — the maximum *number* of nodes whose SoP's can change as the result of a change to the adjacency list of a single node $v \in \mathcal{P}$:

Definition 7.3.2 (Impact Cardinality)**.** *Let $f : \mathcal{G} \times V \times 2^{\mathcal{T}} \to \mathbb{R}$ be a statistic of proximity. The*

impact cardinality *of* f *is*

$$\mathrm{IC}(f) = \max_{G \sim G', S \subseteq \mathcal{T}} \left| \{ v \in V \mid f(G, v, S) \neq f(G', v, S) \} \right|.$$

We include some examples of candidate SoPs and their sensitivities. A desirable property for good statistics is that they should have low sensitivity (relative to the scale of the statistic) and small impact cardinality (relative to the target number of queries to the identity oracle), which will allow us to achieve protected differential privacy by adding only small amounts of noise to the various parts of our computations.

- $\mathsf{Flow}_k(G, v, S)$: the value of the maximum flow that can be routed between node v and the nodes in S using only paths of length at most k;
- $\mathsf{Path}_k(G, v, S)$: the number of paths from v to nodes in S with length at most k;
- $\mathsf{Triangle}(G, v, S) = |\{\{a, b\} \subseteq S \mid a, b, v \text{ forms a triangle in } G\}|$, the number of triangles formed by the vertex v in G;
- $\mathsf{CN}(G, v, S) = |\{u \mid (v, u) \in E \text{ and } (u, v') \in E \text{ for some } v' \in S\}|$, the number of common neighbors v has with vertices in S.

In graphs with maximum degree d, the sensitivity of these SoPs are as follows:

- $\Delta(\mathsf{Flow}_k) \leq d$ since a vertex can only affect the size of the flow by at most d.
- $\Delta(\mathsf{Path}_k) \leq (k-1)d^{k-1}$ since the total number of paths from v to S on which a vertex $u \in \mathcal{P}$ might lie is at most $\sum_{j=1}^{k-1} d^{j-1} d^{k-j} = (k-1)d^{k-1}$. Here we used the index j to denote the index of u along the path starting from v together with the fact that the total number of different paths of length ℓ from u is at most d^ℓ.
- $\Delta(\mathsf{Triangle}) \leq d$ since each triangle is associated with an edge and the total number of edges affected is at most d.

- $\Delta(\mathsf{CN}) \leq 1$ since a single vertex can change the count of common neighbors by at most 1.

Note that $\mathsf{Path}_1(G, v, S)$, which simply counts the number of edges between v and $S \subseteq \mathcal{T}$ actually has targeted sensitivity *zero*. This is because, since $S \subseteq \mathcal{T}$, if $v \in \mathcal{T}$ is also a member of the targeted population, then the statistic is a function only of $E(\mathcal{T})$, the edge set of the subgraph defined over the targeted sub-population $\mathcal{T}$. Since $E(\mathcal{T})$ is identical on all neighboring graphs, and because targeted sensitivity only measures the sensitivity of the SoP evaluated on *targeted* nodes to changes in *protected* nodes, we get zero sensitivity. This will be important to our analysis.

7.3.2. Non-Private Search Algorithm

We will first describe the non-private version of our search algorithm $\mathsf{Target}(k, f)$[7]. For any fixed SoP f, Target proceeds in k rounds, each corresponding to the identification of a new connected component in the subgraph induced by $\mathcal{T}$. The algorithm must be started with a "seed vertex" — a pre-identified member of the targeted population. Each round of the algorithm consists of two steps:

1. *Statistic-First Search*: Given a seed targeted vertex, the algorithm iteratively grows a discovered component of targeted vertices, by examining, in order of their SoP values (computed with respect to the set S of individuals already identified as being members of the targeted population), the vertices that neighbor the previously discovered targeted vertices. This continues until every neighbor of the discovered members of the targeted population has been examined, and all of them have been found to be members of the protected population. We note that this procedure discovers every member of the targeted population that is part of the same *connected component* as the seed vertex, in the subgraph induced by only the members of the

[7] Our motivation in choosing this particular algorithm is simplicity: it is the most straightforward type of "contact chaining" algorithm that ignores privacy entirely, and simply uses the given SoP to prioritize investigations.

targeted population. The formal description is presented in Algorithm 12.

Algorithm 12 $\mathsf{SFS}(G, t)$ Statistic-First Search

Input: known targeted t in a network G
Initialize:

$$\tilde{T} = \{t\} \qquad I = \{t\} \qquad \mathcal{N} = \text{ neighbors of } t$$

while $\mathcal{N} \setminus I \neq \emptyset$
Let

$$v' = \underset{x \in \mathcal{N} \setminus I}{\operatorname{argmax}} \mathsf{Path}_1(G, x, \tilde{T})$$

Query $\mathcal{I}(v')$ to determine v''s targeted status.
$I = I \cup \{v'\}$
if $\mathcal{I}(v') = 1$ **then** $\tilde{T} = \tilde{T} \cup \{v'\}$ and $\mathcal{N} =$ neighbors of $\tilde{T}$
Output: list $\tilde{T}$

2. *Search for a New Component*: Following the completion of statistic-first search, the algorithm must find a new vertex in the targeted population to serve as an initial vertex to begin a new round of statistic-first search. To do this, the algorithm computes the value of the SoP for all vertices whose status bit has not already been examined, using as the input set S the set of already discovered members of the targeted population. It then sorts all of the vertices in decreasing order of their SoP value, and begins examining their status bits in this order. The first vertex that is found to be a member of the targeted population is then used as the seed vertex in the next round. We can also specify a budget K, so that the sub-routine will not examine more than K nodes.

The algorithm outputs discovered targeted individuals as they are found, and so its output can be viewed as being an ordered list of individuals who are confirmed to be from the targeted population. We will defer the full description of the non-private algorithm Target to Section 7.7.

7.3.3. Private Search Algorithm

The private version of the targeting algorithm $\mathsf{PTarget}(k, f, \varepsilon)$, is a simple variant of the non-private version. The statistic-first search stage remains unchanged, and only the search for a new component is modified via randomization. In the private variant, when the algorithm computes the value of the SoP f on each unexamined vertex, it then perturbs each of these values independently with noise sampled from the Laplace distribution $\mathrm{Lap}(\triangle(f)/\varepsilon)$, where ε is a parameter. The algorithm then examines the vertices in sorted order of their *perturbed* SoP values. See Algorithm 13 for a formal description of the private version of this subroutine.

Private Stopping Rule: In the non-private version of SearchCom, the algorithm will stop if no targeted node is found in the first K examinations. In the private version, we need to develop a private stopping rule: we will first use the exponential mechanism to compute a threshold f_K such that the number of unexamined nodes with SoP values above f_K is nearly K and; then we insert a *fake* targeted node in the network, and stop the algorithm whenever we examine the fake node.[8]

The formal description of the full algorithm in presented in Algorithm 14, and its formal privacy guarantee is in Theorem 7.3.3.

Theorem 7.3.3. *Fix any* $0 < \delta < 1$. $\mathsf{PTarget}(\cdot,\cdot,\cdot,\cdot,k,\cdot,\varepsilon)$ *satisfies* ε_1*-protected differential privacy for*

$$\varepsilon_1 = (k-1)\varepsilon,$$

and satisfies (ε_2, δ)*-protected differential privacy for*

$$\varepsilon_2 = 2\sqrt{2(k-1)\ln(1/\delta)}\varepsilon.$$

[8] The implementation of the exponential mechanism depends on the range of the SoP values. We provide a concrete example Section 7.7.

Algorithm 13 SearchCom$(G, \tilde{T}, I, f, \varepsilon, K)$ Search for a New Component

Input: identified members of the targeted population $\tilde{T} \subseteq \mathcal{T}$ in a network G, the set of investigated nodes I, SoP f, privacy parameter ε, and stopping threshold K
Initialize:
Compute a SoP value f_K using exponential mechanism with privacy parameter $\varepsilon/2$, sensitivity parameter IC(f) and quality score

$$q(f) = |(|\{v \in V \setminus I \mid f(v) \geq f\}| - K)| \tag{7.1}$$

Create a "fake targeted node" v_f with SoP value f_K
Let $V' = V \bigcup \{v_f\}$
for each $v \in V' \setminus I$:
  let $\hat{f}(v) = f(G, v, \tilde{T}) + \zeta_v$ where $\zeta_v \sim \text{Lap}(4\Delta(f)/\varepsilon)$
while $(V' \setminus I) \neq \emptyset$
  Let

$$v' = \underset{x \in V \setminus I}{\text{argmax}}\, \hat{f}(v)$$

  If $v' = v_f$ then **Halt**
  Query $\mathcal{I}(v')$ to determine if $v' \in \mathcal{T}$.
  Let $I = I \cup \{v'\}$
  if $\mathcal{I}(v') = 1$ **then return** $\{v'\}$
return $\emptyset$

There are two important things to note about this theorem. First, we obtain a privacy guarantee despite the fact that the statistic-first search portion of our algorithm is not randomized — only the search for new components employs randomness.[9] Second, the privacy cost of the algorithm grows only with k, the number of disjoint connected components of targeted individuals (disjoint in the subgraph defined on targeted individuals), and *not* with the total number of individuals examined, or even the total number of targeted individuals identified. Hence, the privacy cost can be very small on graphs in which the targeted individuals lie only in a small number of connected components or "cells". Both of these features are unusual when compared with typical guarantees that one can obtain under the standard notion of differential privacy.

[9] Intuitively, the reason that statistic-first search can remain unmodified and deterministic is that as long as we remain with a connected component of targeted vertices, we will eventually output only those vertices, and thus we are not compromising the privacy of protected vertices. It is only when we search for a new targeted component via protected vertices and the SoP that we must randomize — for instance to provide privacy to protected "bridge" vertices between targeted components.

Algorithm 14 Private Search Algorithm: PTarget$(G, \mathcal{S}, f, k, N, \varepsilon)$

Input: A network G, a seed node $\mathcal{S} \in \mathcal{T}$, a SoP f for SearchCom, a target number of components k, a stopping threshold N for SearchCom, and privacy parameter ε
Initialize: Use set I to keep track of the set of investigated nodes. Initially $I = \{\mathcal{S}\}$.
Let list $\tilde{T} = \mathsf{SFS}(G, \mathcal{S})$
For $k-1$ rounds:
  let $a = \mathsf{SearchCom}(G, \tilde{T}, I, f, \varepsilon, N)$
  if $a = \emptyset$
    Output $\tilde{T}$
  else
    let $\tilde{T} = \tilde{T} \cup \mathsf{SFS}(G, a)$
Output $\tilde{T}$

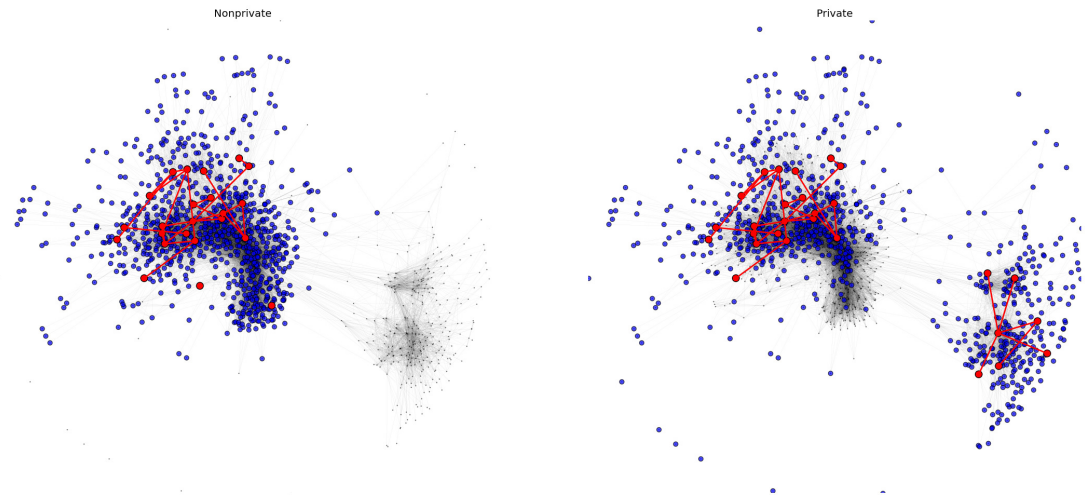

Figure 5: Visual comparison of the non-private algorithm Target (left panel) and the private algorithm PTarget (right panel) on a small portion of the IMDB network (see Experimental Evaluation for more details).

Because PTarget adds randomness for privacy, it results in examining a different set of vertices as compared to Target. Figure 5 provides a sample visualization of the contrasting behavior of the two algorithms. For each algorithm, blue indicates protected vertices that have been examined, red indicates targets that have been examined, and gray vertices have not been examined yet. Both algorithms begin with the same seed target vertex, and by directed statistic-first search discover a subnetwork of targeted individuals (central red edges). As a consequence, many protected vertices are discovered and examined as well. Due to the added noise, PTarget explores the network in a more diffuse fashion, which in

this case permits it to find an additional subnetwork of targets towards the right side of the network. The primary purpose of the noise, however, is for the privacy of protected vertices.

While theorems comparing the utility of Target and PTarget are possible, they require assumptions ensuring that the chosen SoP is sufficiently "informative", in the sense of separating the targeted from the protected by a wide enough margin. In particular, one needs to rule out cases in which all unexplored targeted vertices are deemed closer to the current set than all protected vertices, but only by an infinitesimal amount, in which case the noise added by PTarget eradicates all signal. In general such scenarios are unrealistic, so instead of comparing utility theoretically, we now provide an extensive empirical comparison.

7.4. Privacy Analysis: Proof of Theorem 7.3.3

We will first establish the simple but remarkable privacy guarantee of SFS — the algorithm can often identify a targeted connected component free of privacy cost.

Lemma 7.4.1. *The graph search algorithm* SFS *satisfies* 0*-protected differential privacy.*

Proof. Let G and G' be two neighboring networks in $\mathcal{G}$ with respect to the same partition $(\mathcal{P}, \mathcal{T})$. We know that both networks have the same set of targeted nodes $\mathcal{T}$ and targeted links $E(\mathcal{T})$. Since we know that $\Delta(\mathsf{Path}_1) = 0$, and SFS only branches on the evaluations of f on nodes $v \in \mathcal{T}$, the behavior of SFS depends only on $\mathcal{T}$ and $E(\mathcal{T})$, and hence $\mathsf{SFS}(G, v, f)$ and $\mathsf{SFS}(G', v, f)$ always produce the same output. □

To establish the privacy guarantee of SearchCom, we will introduce the subroutine of *Report Noisy Max* mechanism: given a database $D \in \mathcal{X}^n$ and a collection of k functions $f_1, f_2, \ldots, f_k$ each with sensitivity at most γ, Report Noisy Max performs the following computation: compute the noisy estimate of each function evaluated on D: $\hat{f}_i := f_i(D) + \nu_i$ where $\nu_i \sim \mathrm{Lap}(\gamma/\varepsilon)$; then output the index $i^* = \arg\max_i \hat{f}_i$.

Lemma 7.4.2 ((see e.g. Dwork and Roth [2014])). *The Report Noisy Max mechanism is 2ε-differentially private.*

Lemma 7.4.3. *The targeting algorithm* SearchCom *instantiated with privacy parameter ε satisfies ε-protected differential privacy.*

Proof. Let G and G' be neighboring networks in $\mathcal{G}$ and f be the SoP of choice. First suppose that $\tilde{T} = \mathcal{T}$. In this case, SearchCom will output $\emptyset$ with probability 1 on both inputs G and G', and hence satisfy 0-protected differential privacy. Hence, for the remainder of the argument, we can assume that there exists a vertex $v \in \mathcal{T} \setminus \tilde{T}$. In this case, the algorithm can equivalently be viewed as a composition of the following procedures:

- Compute a noisy threshold f_K using the exponential mechanism such that the number of unchecked nodes with SoP's above f_K is nearly K;
- Use the Report Noisy Max algorithm to output the index of the targeted node t which maximizes $\hat{f}(t)$.

When the input to the algorithm $G = (V, E)$ is viewed as the pair of edge sets $(E \backslash E(\mathcal{T}), E(\mathcal{T}))$, we show below the procedure satisfies ε-differential privacy with respect to its first argument. By Lemma 7.2.4 the algorithm SearchCom satisfies ε-protected differential privacy.

The first step is an instantiation of the exponential mechanism. Since the quality score defined in Equation (7.1) has sensitivity $\mathrm{IC}(f)$, the instantiation satisfies $\varepsilon/2$-differential privacy with respect to the entire edge set E (and therefore the edge set $(E \setminus E(\mathcal{T}))$).

The second step is simply an instantiation of the Report Noisy Max that selects the targeted node with the top noisy SoP. It satisfies $\varepsilon/2$-differential privacy with respect to the edge set $E \setminus E(\mathcal{T})$ by our choice of the parameter and also Lemma 7.4.2. □

Putting all the pieces together, we can then prove Theorem 7.3.3.

Proof of Theorem 7.3.3. The algorithm is a composition of at most k instantiations of SFS

and $(k-1)$ instantiations of SearchCom with privacy parameter ε. Recall that each call to SFS is 0-differentially private, and each call to SearchCom is ε-differentially private with respect to the edges incident on vertices in $\mathcal{P}$. By the composition theorem, we know that the algorithm is $(k-1)\varepsilon$-differentially private by Theorem 2.2.2, and at the same time $(\sqrt{8k\ln(1/\delta)}, \delta)$-differentially private for any $\delta \in (0,1)$ by Theorem 2.2.4. Our result then easily follows from Lemma 7.2.4. □

7.5. Experimental Evaluation

In this section we empirically demonstrate the utility of our private algorithm PTarget by comparing its performance to its non-private counterpart Target.

We report on computational experiments performed on real social network data drawn from two sources — the paper coauthorship network of DBLP (the "Digital Bibliography and Library Project") DBLP [2014], and the co-appearance network of film actors of IMDB (the "Internet Movie Database") IMDB [2005], whose macroscopic properties are described in Table 7.5.

Network	# vertices	# edges	Edge relation
DBLP	956,043	3,738,044	scientific paper co-authorship
IMDB	235,710	4,587,715	movie co-appearance

Table 3: Social network datasets for PTarget.

These data sources provide us with naturally occurring networks, but not a targeted subpopulation. While one could attempt to use communities within each network (e.g. all co-authors within a particular scientific subtopic), our goal was to perform large-scale experiments in which the component structure of targeted vertices (which we shall see is the primary determinant of performance) could be more precisely controlled.

We thus used a simple parametric stochastic diffusion process (described in Section 7.7) to generate the targeted subpopulation in each network. We then evaluate our private search

algorithm PTarget on these networks, and compare its performance to the non-private variant Target. For brevity we shall describe our results only for the IMDB network; results for the DBLP network are quite similar.

In our experiments, we fix a particular SoP: the sum of the number of common neighbors shared between the vertex v and any vertex w that is among the subset of vertices S representing the already discovered members of the targeted population. This SoP has sensitivity 1, and so can be used in our algorithm while adding only a small amount of noise. In particular, the private algorithm PTarget adds noise sampled from the Laplace distribution Lap(20) to the SoP when performing new component search. By Theorem 7.3.3, such an instantiation of PTarget guarantees $((k-1)/20)$-protected differential privacy if it finds k targeted components.

The main trade-off we explore is the number of members of the targeted population that are discovered by the algorithms (the y-axis in the ensuing plots), as a function of the budget, or number of status bits that have been investigated so far (the x-axis in the ensuing plots).

In each plot, the parameters of the diffusion model described above were fixed and used to stochastically generate targeted subpopulations of the fixed networks given by our social network data. By varying these parameters, we can investigate performance as a function of the underlying component structure of the targeted subnetwork. As we shall see, in terms of relative performance, there are effectively three different regimes of the diffusion model (i.e. targeted subpopulation) parameter space. In all of them PTarget compares favorably with Target, but to different extents and for different reasons that we now discuss. We also plot the growth of the risk multiplier for PTarget, which remains less than 2 in all three regimes.

On each plot, there is a single blue curve showing the performance of the (deterministic) algorithm Target, and multiple red curves showing the performance across 200 runs of

our (randomized) algorithm PTarget.

- The first regime (Figure 6) occurs when the largest connected component of the targeted subnetwork is much larger than all the other components. In this regime, if both algorithms begin at a seed vertex inside the largest component, there is effectively no difference in performance, as both algorithms remain inside this component for the duration of their budget and find identical sets of targeted individuals. More generally, if the algorithms begin at a seed outside the largest component, relative performance is a "race" to find this component; the private algorithm lags slightly due to the added noise, but is generally quite competitive. In the left panel, we show the number of targeted vertices found as a function of the budget used for both the (deterministic) non-private algorithm Target (blue), and for several representative runs of the randomized private algorithm PTarget (red). Colored circles indicate points at which the corresponding algorithm has first discovered a new targeted component. In the right panel, we show average performance over 200 trials for the private algorithm with 1-standard deviation error bars. We also show the private algorithm risk multiplier with error bars. In this regime, after a brief initial flurry of small component discovery, both algorithms find the dominant component, so the private performance closely tracks non-private, and the private algorithm's risk multiplier quickly levels off at around only 1.17.

- The second regime (Figure 7) occurs when the component sizes are more evenly distributed, but there remain a few significantly larger components. In this setting both algorithms spend more of their budget outside the targeted subpopulation "searching" for these components. Here the performance of the private algorithm lags more significantly — since both algorithms behave the same when inside of a component, the smaller the components are, the more detrimental the noise is to the private algorithm (though again we see particular runs in which the randomness of the private algorithm permits it to actually outperform the non-private). Figure 7 has the

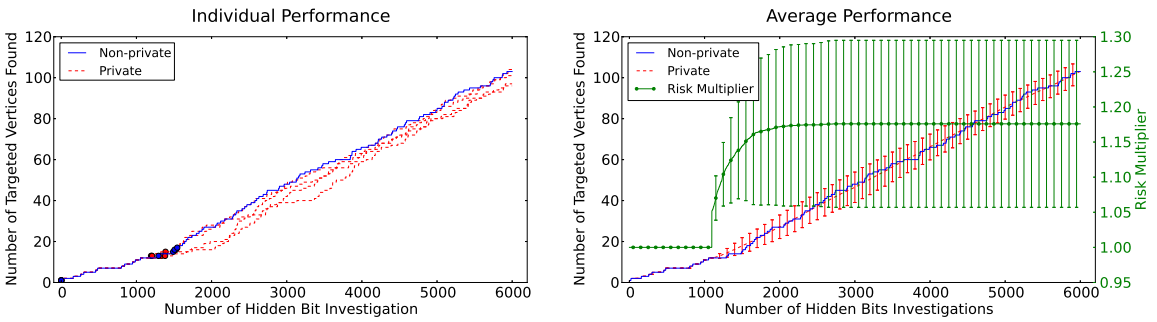

Figure 6: Performance for the case in which there is a dominant component in the targeted subpopulation.

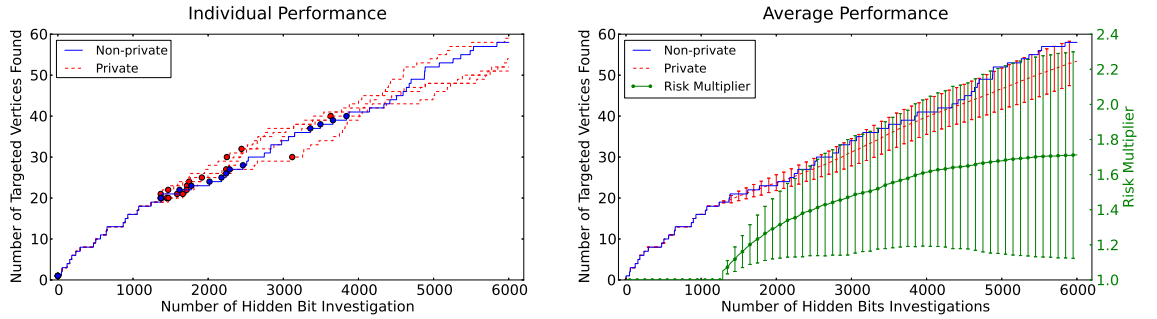

Figure 7: Performance for the case where the component sizes are more evenly distributed, but still relatively large.

same format as in Figure 6. As we can see, the performance of both algorithms is hampered by longer time spent investigating non-targeted vertices (note the smaller scale of the y-axis compared to Figure 6). Targeted component discovery is now more diffuse. The private algorithm remains competitive but lags slightly, and as per Theorem 7.3.3 the risk multiplier grows (but remains modest) as more targeted components are discovered.

- The third regime (Figure 8) occurs when all the targeted components are small, and thus both algorithms suffer accordingly, discovering only a few targeted individuals; but again the private algorithm compares favorably with the non-private, finding only a few less targeted vertices.

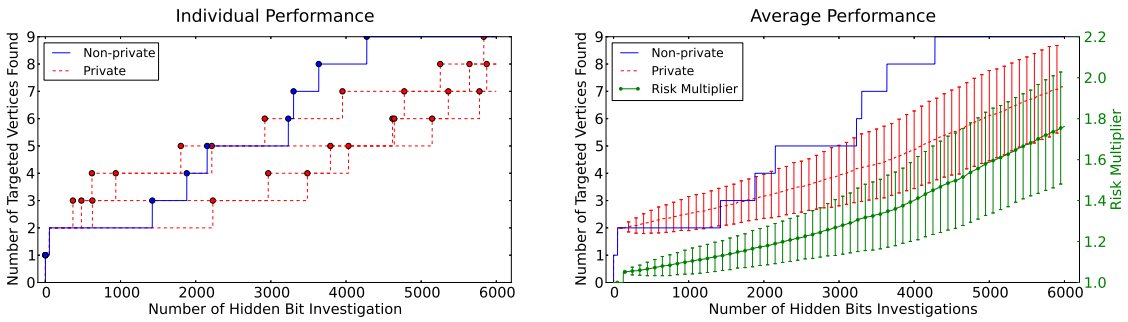

Figure 8: Performance for the case with a highly fragmented targeted subpopulation.

7.6. Conclusion

We view the work presented here as a proof of concept: despite the fact that using network analysis to identify members of a targeted population is intrinsically contrary to the privacy of the targeted individuals, we have shown that there is no inherent reason why informational privacy guarantees cannot be given to individuals who are not members of the targeted population, and that these privacy guarantees need not severely harm our ability to find targeted individuals. Our work is of course not a complete solution to the practical problem, which can differ from our simple model in many ways. Here we highlight just one interesting modeling question for future work: Is it possible to give rigorous privacy guarantees to members of the protected population when membership in the targeted population is defined as a function of the individuals' private data? In our model, we avoid this question by endowing the algorithm with a costly "investigation" operation which we assume can infallibly determine an individual's targeted status — but it would be interesting to extend our style of analysis to situations in which this kind of investigation is not available.

7.7. Missing Details

7.7.1. Subpopulation Construction in the Experiments

Pre-processing step on the networks: We sparsify the IMDB and DBLP networks by removing a subset of the edges. This will allow us to generate multiple targeted components more easily. In both networks, there is a natural notion of weights for the edges. In the case of DBLP, the edge weights correspond to the number of papers the individuals have co-written. In the case of IMDB, the edge weights correspond to the number of movies two actors have co-starred in. In our experiments, we only remove edges with weights less than 2.

However, the networks we use do not have an identified partition of the vertices into a targeted and protected subpopulation. Instead, we generate the targeted subpopulation synthetically using the following diffusion process. We use the language of "infection", which is natural, but we emphasize that this process is not specific to our motivating example of the targeted population representing people infected with a dangerous disease. The goal of the infection process is to generate a targeted subpopulation $\mathcal{T}$ such that:

1. The subnetwork restricted to $\mathcal{T}$ has multiple distinct connected components (so that the search problem is algorithmically challenging, and isn't solved by a single run of statistic-first search), and

2. The connected components of $\mathcal{T}$ are close to one another in the underlying network G, so that the network data is useful in identifying new members of $\mathcal{T}$.

The process $\mathsf{infect}(G, s, p, q, k)$ takes as input a seed infected node s, two values $p, q \in (0, 1)$, and a number of rounds k, and proceeds with two phases:

1. **Infection phase:** Initially, only the node s is in the infected set $\tilde{\mathcal{I}}$. Then in each of the k rounds, each neighbor v of the infected nodes $\tilde{\mathcal{I}}$ becomes infected independently

with probability p.

2. **Immune phase:** After the infection process above, we will set some of the infected nodes as immune. For each node i in the infected node set $\tilde{\mathcal{I}}$, let i become "immune" (non-infected) with probability q.

The infection phase above is common in the literature on contagion in networks; e.g. relatively recently it was considered as the "independent cascades model" of Kempe et al. [2003]. The immune phase is introduced in order to permit control of the component structure while still keeping targeted vertices proximate to each other in the resulting network. We include a formal description of the algorithm in Algorithm 15.

Algorithm 15 infect(G, s, p, q, k)

Input: a network G, a seed node s in G, infection probability p, and immune probability q

Initially the infected population contains only the seed node:

$$\tilde{I} = \{s\}$$

for $t = 1, \ldots, k$:
  for each node v that is neighbor to $\tilde{I}$:
    Let ν be a uniformly random number from $[0, 1]$
    if $\nu \leq p$ **then** $\tilde{I} = \tilde{I} \cup \{v\}$
let $\mathcal{T} = \emptyset$
for each node $v' \in \tilde{I}$:
  let ν be a uniformly random number from $[0, 1]$
  if $\nu > q$ **then** $\mathcal{T} = \mathcal{T} \cup \{v\}$
Output: $\mathcal{T}$ as the targeted subpopulation

7.7.2. Non-Private Benchmark Target

We here give a formal description of the non-private version of our graph search algorithm in Algorithm 16.

Algorithm 16 Non-Private Targeting Algorithm: Target$(G, \mathcal{S}, f, k, N)$

Input: A network G, a seed node $\mathcal{S} \in \mathcal{T}$, a SoP f for SearchCom, a target number of components to find k, and a stopping threshold N for SearchCom
Initialize: Use I to keep track of the set of investigated nodes. Initially $I = \{\mathcal{S}\}$.
Let list $\tilde{T} = \mathsf{SFS}(G, \mathcal{S})$
For $k-1$ rounds:
  let count = 0 and $a = \emptyset$
  while $(V \setminus I) \neq \emptyset$ and count$\leq N$
    Let

$$v' = \underset{x \in V \setminus I}{\operatorname{argmax}} f_2(G, v, \tilde{T})$$

    Query $\mathcal{I}(v')$ to determine if $v' \in \mathcal{T}$.
    Let $I = I \cup \{v'\}$ and count = count +1
    if $\mathcal{I}(v') = 1$ **then let** $a = v'$ and **break**
  if $a = \emptyset$ **then Output** $\tilde{T}$
  else let $\tilde{T} = \tilde{T} \cup \mathsf{SFS}(G, a, f_1)$
Output $\tilde{T}$

7.7.3. SoP Instantiation

In our experiments, we will use the SoP CN for the SearchCom subroutine, which is the number of common neighbors between the node v and the subset of nodes S representing the already discovered members of the targeted population. The targeted sensitivity of CN is bounded by 1.

Lemma 7.7.1. *The SoP* CN *has targeted sensitivity* $\triangle(\mathsf{CN})$ *bounded by 1.*

Proof. Let G and G' be two neighboring networks over the same protected and targeted populations $\mathcal{P}$ and $\mathcal{T}$. Let $t \in \mathcal{T}$ be a targeted node and $S \subseteq V$ be a subset of nodes. Since G and G' only differ by the edges associated with a protected node i, we know that the neighbor sets of both t and S can differ by at most one node between G and G'. Note that the $\mathsf{CN}(G, t, S)$ computes the cardinality of the intersection between these two sets, and the intersection sets of these two networks can differ by at most one node. It follows that $\triangle(\mathsf{CN}) \leq 1$. □

Claim 7.7.2. *Suppose each graph in the class* $\mathcal{G}$ *has maximum degree no more than* d*, then the SoP* CN *has impact cardinality* $\mathrm{IC}(\mathsf{CN}) \leq d + 1$.

Implementation of the Stopping Rule with CN Suppose that each graph in the class $\mathcal{G}$ has maximum degree no more than d, then the SoP values of CN will be integer values in the set $D = \{0, 1, \ldots, d\}$. Then we can compute the stopping threshold f_K by using the exponential mechanism to select a value from the discrete set D.

BIBLIOGRAPHY

Mohammad Alaggan, Sébastien Gambs, and Anne-Marie Kermarrec. Heterogeneous differential privacy. *arXiv preprint arXiv:1504.06998*, 2015.

Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

K. Bache and M. Lichman. UCI machine learning repository, 2013.

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 *ACM SIGACT–SIGMOD–SIGART Symposium on Principles of Database Systems (PODS), Beijing, China*, pages 273–282, 2007.

Michael Barbaro and Tom Zeller. A face is exposed for aol searcher no. 4417749, August 2006.

Raef Bassily, Adam Groce, Jonathan Katz, and Adam Smith. Coupled-worlds privacy: Exploiting adversarial uncertainty in statistical data privacy. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 439–448. IEEE, 2013.

Raef Bassily, Adam Smith, and Abhradeep Guha Thakurta. Differentially private empirical risk minimization: Efficient algorithms and tight error bounds. 2014.

Jeremiah Blocki, Avrim Blum, Anupam Datta, and Or Sheffet. Differentially private data analysis of social networks via restricted sensitivity. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 87–96. ACM, 2013.

Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the sulq framework. In *Proceedings of the Twenty-fourth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 13-15, 2005, Baltimore, Maryland, USA*, pages 128–138, 2005.

Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to noninteractive database privacy. *J. ACM*, 60(2):12:1–12:25, 2013.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004. ISBN 0521833787.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Transactions on Information and System Security*, 14(3):26, 2011.

Kamalika Chaudhuri and Staal A. Vinterbo. A stability-based validation procedure for differentially private machine learning. In *Advances in Neural Information Processing*

Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States., pages 2652–2660, 2013.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12:1069–1109, 2011.

Yiling Chen, Stephen Chong, Ian A Kash, Tal Moran, and Salil Vadhan. Truthful mechanisms for agents that value privacy. In *ACM SIGecom Conference on Economics and Computation (EC), Philadelphia, Pennsylvania*, pages 215–232, 2013.

National Research Council. *Bulk Collection of Signals Intelligence: Technical Options*. The National Academies Press, Washington, DC, 2015. ISBN 978-0-309-32520-2.

Rachel Cummings, Michael Kearns, Aaron Roth, and Zhiwei Steven Wu. Privacy and truthful equilibrium selection for aggregative games. In *Web and Internet Economics - 11th International Conference, WINE 2015, Amsterdam, The Netherlands, December 9-12, 2015, Proceedings*, pages 286–299, 2015.

Rachel Cummings, Katrina Ligett, Jaikumar Radhakrishnan, Aaron Roth, and Zhiwei Steven Wu. Coordination complexity: Small information coordinating large populations. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, Cambridge, MA, USA, January 14-16, 2016*, pages 281–290, 2016.

DBLP. DBLP computer science bibliography, 2014. Avaialable at `http://dblp.uni-trier.de/xml/`http://dblp.uni-trier.de/xml/.

Devdatt P Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.

Cynthia Dwork and Frank D McSherry. Selective privacy guarantees, October 19 2010. US Patent 7,818,335.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, pages 265–284, 2006.

Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil P. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 381–390, 2009.

Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *ACM SIGACT Symposium on Theory of Computing (STOC), Cambridge, Massachusetts*, pages 715–724, 2010a.

Cynthia Dwork, Guy N. Rothblum, and Salil P. Vadhan. Boosting and differential privacy. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 51–60, 2010b.

Cynthia Dwork, Moni Naor, and Salil Vadhan. The privacy of the analyst and the power of the state. In *IEEE Symposium on Foundations of Computer Science (FOCS), New Brunswick, New Jersey*, pages 400–409, 2012.

Cynthia Dwork, Aleksandar Nikolov, and Kunal Talwar. Using convex relaxations for efficiently and privately releasing marginals. In *30th Annual Symposium on Computational Geometry, SOCG'14, Kyoto, Japan, June 08 - 11, 2014*, page 261, 2014.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Leon Roth. Preserving statistical validity in adaptive data analysis. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 117–126, 2015.

Cynthia Dwork, Adam Smith, Thomas Steinke, and Jonathan Ullman. Exposed! a survey of attacks on private data. *Annual Review of Statistics and Its Application*, (0), 2017.

Yoav Freund and Robert E. Schapire. Game theory, on-line prediction and boosting. In *Proceedings of the Ninth Annual Conference on Computational Learning Theory, COLT 1996, Desenzano del Garda, Italy, June 28-July 1, 1996.*, pages 325–332, 1996.

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 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 1170–1178, 2014.

Glenn Greenwald. *NSA Collecting Phone Records of Millions of Verizon Customers Daily*. *The Guardian*, June 6 2013.

Faruk Gul and Ennio Stacchetti. Walrasian equilibrium with gross substitutes. *Journal of Economic Theory*, 87(1):95–124, 1999.

Anupam Gupta, Katrina Ligett, Frank McSherry, Aaron Roth, and Kunal Talwar. Differentially private combinatorial optimization. In *ACM–SIAM Symposium on Discrete Algorithms (SODA), Austin, Texas*, pages 1106–1125, 2010.

Anupam Gupta, Aaron Roth, and Jonathan Ullman. Iterative constructions and private data release. In *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, pages 339–356, 2012.

Anupam Gupta, Moritz Hardt, Aaron Roth, and Jonathan Ullman. Privately releasing conjunctions and the statistical query barrier. *SIAM J. Comput.*, 42(4):1494–1520, 2013.

Moritz Hardt and Guy N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 61–70, 2010.

Moritz Hardt, Katrina Ligett, and Frank McSherry. A simple and practical algorithm for differentially private data release. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 2348–2356, 2012.

Michael Hay, Chao Li, Gerome Miklau, and David Jensen. Accurate estimation of the degree distribution of private networks. In *Ninth IEEE International Conference on Data Mining. ICDM'09*, pages 169–178. IEEE, 2009.

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*, pages 1447–1458. ACM, 2014.

Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V Pearson, Dietrich A Stephan, Stanley F Nelson, and David W Craig. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS Genet*, 4(8):e1000167, 2008.

Justin Hsu, Aaron Roth, and Jonathan Ullman. Differential privacy for the analyst via private equilibrium computation. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 341–350, 2013.

Justin Hsu, Zhiyi Huang, Aaron Roth, Tim Roughgarden, and Zhiwei Steven Wu. Private matchings and allocations. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 21–30, 2014a.

Justin Hsu, Aaron Roth, Tim Roughgarden, and Jonathan Ullman. Privately solving linear programs. pages 612–624, 2014b.

Justin Hsu, Zhiyi Huang, Aaron Roth, and Zhiwei Steven Wu. Jointly private convex programming. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 580–599, 2016.

Zhiyi Huang and Sampath Kannan. The exponential mechanism for social welfare: Private, truthful, and nearly optimal. In *IEEE Symposium on Foundations of Computer Science (FOCS), New Brunswick, New Jersey*, pages 140–149, 2012.

IMDB. Internet movie database, 2005. Available at `http://www3.ul.ie/gd2005/dataset.html`http://www3.ul.ie/gd2005/dataset.html.

Prateek Jain and Abhradeep Guha Thakurta. (Near) dimension independent risk bounds for differentially private learning. pages 476–484, 2014.

Prateek Jain, Pravesh Kothari, and Abhradeep Guha Thakurta. Differentially private online learning. *arXiv preprint arXiv:1109.0105*, 2011.

Sampath Kannan, Jamie Morgenstern, Aaron Roth, and Zhiwei Steven Wu. Approximately stable, school optimal, and student-truthful many-to-one matchings (via differential privacy). In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1890–1903, 2015.

Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Analyzing graphs with node differential privacy. In *TCC*, pages 457–476, 2013.

Michael Kearns, Mallesh M. Pai, Aaron Roth, and Jonathan Ullman. Mechanism design in large games: incentives and privacy. In *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*, pages 403–410, 2014.

Michael Kearns, Aaron Roth, Zhiwei Steven Wu, and Grigory Yaroslavtsev. Private algorithms for the protected in social network search. *Proceedings of the National Academy of Sciences*, 113(4):913–918, 2016.

Michael J. Kearns. Efficient noise-tolerant learning from statistical queries. *J. ACM*, 45(6): 983–1006, 1998.

Alexander Kelso and Vincent Crawford. Job matching, coalition formation, and gross substitutes. *Econometrica*, 50(6):22, 1982.

David Kempe, Jon Kleinberg, and Eva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146, 2003.

Daniel Kifer, Adam Smith, and Abhradeep Guha Thakurta. Private convex empirical risk minimization and high-dimensional regression. *Journal of Machine Learning Research*, 1: 41, 2012.

Hellmuth Kneser. Sur un théoreme fondamental de la théorie des jeux. *Comptes Rendus de l'Acadèmie des Science*, 234:2418–2420, 1952.

Chao Li and Gerome Miklau. Lower bounds on the error of query sets under the differentially-private matrix mechanism. *Theory Comput. Syst.*, 57(4):1159–1201, 2015.

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

Frank McSherry and Ilya Mironov. Differentially private recommender systems: building privacy into the net. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), Paris, France*, pages 627–636, 2009.

Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*, pages 94–103, 2007.

Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (S&P 2008), 18-21 May 2008, Oakland, California, USA*, pages 111–125, 2008.

Netflix. Netflix prize.

Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *ACM SIGACT Symposium on Theory of Computing (STOC), San Diego, California*, pages 75–84, 2007.

Kobbi Nissim, Rann Smorodinsky, and Moshe Tennenholtz. Approximately optimal mechanism design via differential privacy. pages 203–213, 2012.

Paul Ohm. Broken promises of privacy: Responding to the surprising failure of anonymization. *UCLA Law Review, Vol. 57, p. 1701, 2010*, 2009.

Reuters. *U.S. Nurse Quarantined Over Ebola Calls Treatment "Frenzy of Disorganization"*. *The New York Times*, October 25 2014.

Ryan M Rogers and Aaron Roth. Asymptotically truthful equilibrium selection in large congestion games. In *ACM SIGecom Conference on Economics and Computation (EC), Palo Alto, California*, pages 771–782, 2014.

Ryan M. Rogers, Aaron Roth, Jonathan Ullman, and Zhiwei Steven Wu. Inducing approximately optimal flow using truthful mediators. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation, EC '15, Portland, OR, USA, June 15-19, 2015*, pages 471–488, 2015.

Aaron Roth and Tim Roughgarden. Interactive privacy via the median mechanism. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 765–774, 2010.

Morton Slater. Lagrange multipliers revisited. Technical report, Cowles Commission Discussion Paper, Mathematics, 1950.

Daniel J Solove. A taxonomy of privacy. *University of Pennsylvania law review*, pages 477–564, 2006.

Justin Thaler, Jonathan Ullman, and Salil P. Vadhan. Faster algorithms for privately releasing marginals. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, pages 810–821, 2012.

Jonathan Ullman. Answering $n^{\{2+o(1)\}}$ counting queries with differential privacy is hard. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 361–370, 2013.

Jonathan Ullman and Salil P. Vadhan. Pcps and the hardness of generating private synthetic data. In *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings*, pages 400–416, 2011.

David Xiao. Is privacy compatible with truthfulness? In *ACM SIGACT Innovations in Theoretical Computer Science (ITCS), Berkeley, California*, pages 67–86, 2013.

Grigory Yaroslavtsev, Graham Cormode, Cecilia M. Procopiuc, and Divesh Srivastava. Accurate and efficient private release of datacubes and contingency tables. In *29th IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, April 8-12, 2013*, pages 745–756, 2013.

Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: private data release via bayesian networks. In *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, pages 1423–1434, 2014.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. pages 928–936, 2003.