Model Checking Differentially Private Properties

Depeng Liu[1,2], Bow-Yaw Wang[3], and Lijun Zhang[1,2,4]

[1] State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences *
[2] University of Chinese Academy of Sciences *
[3] Institute of Information Science, Academia Sinica †
[4] Institute of Intelligent Software, Guangzhou *

Abstract. We introduce the branching time temporal logic dpCTL* for specifying differential privacy. Several differentially private mechanisms are formalized as Markov chains or Markov decision processes. Using our formal models, subtle privacy conditions are specified by dpCTL*. In order to verify privacy properties automatically, model checking problems are investigated. We give a model checking algorithm for Markov chains. Model checking dpCTL* properties on Markov decision processes however is shown to be undecidable.

1 Introduction

In the era of data analysis, personal information is constantly collected and analyzed by various parties. Privacy has become an important issue for every individual. In order to address such concerns, the research community has proposed several privacy preserving mechanisms over the years (see [18] for a slightly outdated survey). Among these mechanisms, differential privacy has attracted much attention from theoretical computer science to industry [16, 24, 34].

Differential privacy formalizes the tradeoff between privacy and utility in data analysis. Intuitively, a randomized data analysis mechanism is differentially private if it behaves similarly on similar input datasets [15, 17]. Consider, for example, the Laplace mechanism where analysis results are perturbed by random noises with the Laplace distribution [16]. Random noises hide the differences of analysis results from similar datasets. Clearly, more noises give more privacy but less utility in released perturbed results. Under the framework of differential privacy, data analysts can balance the tradeoff rigorously in their data analysis mechanisms [16, 24].

Designing differentially private mechanisms can be tedious for sophisticated data analyses. Privacy leak has also been observed in data analysis programs implementing differential privacy [25, 30]. This calls for formal analysis of differential privacy on both designs and implementations. In this paper, we propose the logic dpCTL* for specifying differential privacy and investigate their model checking problems. Data analysts can automatically verify their designs and implementations with our techniques.

*Partially supported by the National Natural Science Foundation of China (Grants No. 61532019, 61761136011, 61472473).

†Partially supported by the Academia Sinica Thematic Project: Socially Accountable Privacy Framework for Secondary Data Usage.

Most interestingly, our techniques can be adopted easily by existing probabilistic model checkers. Privacy checking with existing tools is attainable with minimal efforts. More interaction between model checking [12, 2] and privacy analysis hopefully will follow.

In order to illustrate applicability of our techniques, we give detailed formalizations of several data analysis mechanisms in this paper. In differential privacy, data analysis mechanisms are but randomized algorithms. We follow the standard practice in probabilistic model checking to formalize such mechanisms as Markov chains or Markov decision processes [27]. When a data analysis mechanism does not interact with its environment, it is formalized as a Markov chain. Otherwise, its interactions are formalized by non-deterministic actions in Markov decision processes. Our formalization effectively assumes that actions are controlled by adversaries. It thus considers all privacy attacks from adversaries in order to establish differential privacy as required.

Two ingredients are introduced to specify differentially private behaviors. A reflexive and symmetric user-defined binary relation over states is required to formalize similar datasets. We moreover add the path quantifier $\mathcal{D}_{\epsilon,\delta}$ for specifying similar behaviors. Informally, a state satisfies $\mathcal{D}_{\epsilon,\delta}\phi$ if its probability of having path satisfying ϕ is close to those of similar states. Consider, for instance, a data analysis mechanism computing the likelihood ($high$ or low) of an epidemic. A state satisfying $\mathcal{D}_{\epsilon,\delta}(\mathsf{F}high) \wedge \mathcal{D}_{\epsilon,\delta}(\mathsf{F}low)$ denotes similar states have similar probabilities on every outcomes.

We moreover extend the standard probabilistic model checking algorithms to verify dpCTL* properties automatically. For Markov chains, states satisfying a subformula $\mathcal{D}_{\epsilon,\delta}\phi$ are computed by a simple variant of the model checking algorithm for Markov chains. The time complexity of our algorithm is the same as those of PCTL* for Markov chains. The logic dpCTL* obtains its expressiveness essentially for free. For Markov decision processes, checking whether a state satisfies $\mathcal{D}_{\epsilon,\delta}\phi$ is undecidable.

Related Work. An early attempt on formal verification of differential privacy is [31]. The work formalizes differential privacy in the framework of information leakage. The connection between differential privacy and information leakage is investigated in [1, 20]. Type systems for differential privacy have been developed in [33, 19, 29]. A lightweight technique for checking differential privacy can be found in [35]. Lots of formal Coq proofs about differential privacy are reported in [6, 9, 10, 4, 7, 8, 5]. This work emphasizes on model checking differential privacy. We develop a framework to formalize and analyze differential privacy in Markov chains and Markov decision processes.

Contributions. Our main contributions are threefold.

1. We introduce the logic dpCTL* for reasoning about differential privacy. The logic is able to to express subtle and generalized differentially private properties;
2. We model several differentially private mechanisms in Markov chains or Markov decision processes; and
3. We show that the model checking problem for Markov chains is standard. For Markov decision processes, we show that it is undecidable.

Organization of the paper. Preliminaries are given in Section 2. In Section 3 we discuss how offline differentially private mechanisms are modeled as Markov chains. The logic dpCTL* and its syntax are presented in Section 4. The semantics over Markov chains and its model checking algorithm are given in Section 5. Section 6 discusses differential privacy properties using dpCTL*. More examples of online differentially private

mechanisms as Markov decision processes are given in Section 7. The semantics over Markov decision processes and undecidability of model checking is given in Section 8. Finally, Section 9 concludes our presentation.

2 Preliminaries

Let $\mathbb{Z}$ and $\mathbb{Z}^{\geq 0}$ be the sets of integers and non-negative integers respectively. We briefly review the definitions of differential privacy, Markov chains, and Markov decision processes [27]. For differential privacy, we follow the standard definition in [16, 21, 22]. Our definitions of Markov chains and Markov decision processes are adopted from [2].

2.1 Differential Privacy

We denote the data universe by $\mathcal{X}$; $x \in \mathcal{X}^n$ is a *dataset* with n *rows* from the data universe. Two datasets x and x' are *neighbors* (denoted by $d(x, x') \leq 1$) if they are identical except at most one row. A *query* f is a function from $\mathcal{X}^n$ to its range $R \subseteq \mathbb{Z}$. The *sensitivity* of the query f (written $\Delta(f)$) is $\max_{d(x,x')\leq 1} |f(x) - f(x')|$. For instance, a *counting* query counts the number of rows with certain attributes (say, female). The sensitivity of a counting query is 1 since any neighbor can change the count by at most one. We only consider queries with finite ranges for simplicity. A *data analysis mechanism* (or *mechanism* for brevity) M_f for a query f is a randomized algorithm with inputs in $\mathcal{X}^n$ and outputs in $\tilde{R}$. A mechanism may not have the same output range as its query, that is, $\tilde{R} \neq R$ in general. A mechanism M_f for f is *oblivious* if $\Pr[M_f(x) = \tilde{r}] = \Pr[M_f(x') = \tilde{r}]$ for every $\tilde{r} \in \tilde{R}$ when $f(x) = f(x')$. In words, outputs of an oblivious mechanism depend on the query result $f(x)$. The order of rows, for instance, is irrelevant to oblivious mechanisms. Let x, x' be datasets and $\tilde{r} \in \tilde{R}$. The probability of the mechanism M_f outputting $\tilde{r}$ on x is (ϵ, δ)-*close* to those on x' if

$$\Pr[M_f(x) = \tilde{r}] \leq e^{\epsilon} \Pr[M_f(x') = \tilde{r}] + \delta.$$

A mechanism M_f is (ϵ, δ)-*differentially private* if for every $x, x' \in \mathcal{X}^n$ with $d(x, x') \leq 1$ and $\tilde{r} \in \tilde{R}$, the probability of M_f outputting $\tilde{r}$ on x is (ϵ, δ)-close to those on x'.

The non-negative parameters ϵ and δ quantify mechanism behaviors probabilistically; the smaller they are, the behaviors are more similar. Informally, a differentially private mechanism has probabilistically similar behaviors on neighbors. It will have similar output distributions when any row is replaced by another in a given dataset. Since the output distribution does not change significantly with the absence of any row in a dataset, individual privacy is thus preserved by differentially private mechanisms.

2.2 Markov Chains and Markov Decision Processes

Let AP be the set of *atomic propositions*. A *(finite) discrete-time Markov chain* $K = (S, \wp, L)$ consists of a non-empty finite set S of *states*, a *transition probability function* $\wp : S \times S \to [0, 1]$ with $\sum_{t \in S} \wp(s, t) = 1$ for every $s \in S$, and a *labeling function* $L : S \to 2^{AP}$. A *path* in K is an infinite sequence $\pi = \pi_0 \pi_1 \cdots \pi_n \cdots$ of states with $\wp(\pi_i, \pi_{i+1}) > 0$ for all $i \geq 0$. We write $\pi[j]$ for the suffix $\pi_j \pi_{j+1} \cdots$.

A *(finite) Markov decision process* (MDP) [¶] $M = (S, Act, \wp, L)$ consists of a finite set of *actions* Act, a *transition probability function* $\wp : S \times Act \times S \to [0,1]$ with $\sum_{t \in S} \wp(s, \alpha, t) = 1$ for every $s \in S$ and $\alpha \in Act$. S and L are as for Markov chains. A *path* π in M is an infinite sequence $\pi_0 \alpha_1 \pi_1 \cdots \pi_n \alpha_{n+1} \cdots$ with $\wp(\pi_i, \alpha_{i+1}, \pi_{i+1}) > 0$ for all $i \geq 0$. Similarly, we write $\pi[j]$ for the suffix $\pi_j \alpha_{j+1} \pi_{j+1} \cdots$ of π.

Let $M = (S, Act, \wp, L)$ be an MDP. A (history-dependent) *scheduler* for M is a function $\mathfrak{S} : S^+ \to Act$. A *query scheduler* for M is a function $\mathfrak{Q} : S^+ \to Act$ such that $\mathfrak{Q}(\sigma) = \mathfrak{Q}(\sigma')$ for any $\sigma, \sigma' \in S^+$ of the same length. Intuitively, decisions of a query scheduler depend only on the length of the history. A path $\pi = \pi_0 \alpha_1 \pi_1 \cdots \pi_n \alpha_{n+1} \cdots$ is an $\mathfrak{S}$-*path* if $\alpha_{i+1} = \mathfrak{S}(\pi_0 \pi_1 \cdots \pi_i)$ for all $i \geq 0$. Note that an MDP with a scheduler $\mathfrak{S}$ induces a Markov chain $M_{\mathfrak{S}} = (S^+, \wp_{\mathfrak{S}}, L')$ where $L'(\sigma s) = L(s)$, $\wp_{\mathfrak{S}}(\sigma s, \sigma s t) = \wp(s, \mathfrak{S}(\sigma s), t)$ for $\sigma \in S^*$ and $s, t \in S$.

3 Differentially Private Mechanisms as Markov Chains

To model differentially private mechanisms by Markov chains, we formalize inputs (such as datasets or query results) as states. Randomized computation is modeled by probabilistic transitions. Atomic propositions are used to designate intended interpretation on states (such as inputs or outputs). We demonstrate these ideas in examples.

3.1 Survey Mechanism

Consider the survey question: have you been diagnosed with the disease X? In order to protect privacy, each surveyee answers the question as follows. The surveyee first flips a coin. If it is tail, she answers the question truthfully. Otherwise, she randomly answers 1 or 0 uniformly (Figure 1a) [16].

Let us analyze the mechanism briefly. The data universe $\mathcal{X}$ is $\{+, -\}$. The mechanism M is a randomized algorithm with inputs in $\mathcal{X}$ and outputs in $\{1, 0\}$. For any $x \in \mathcal{X}$, we have $\frac{1}{4} \leq \Pr[M(x) = 1] \leq \frac{3}{4}$. Hence $\Pr[M(x) = 1] \leq \frac{3}{4} = 3 \cdot \frac{1}{4} \leq e^{\ln 3} \Pr[M(x') = 1]$ for any neighbors $x, x' \in \mathcal{X}$. Similarly, $\Pr[M(x) = 0] \leq e^{\ln 3} \Pr[M(x') = 0]$. The survey mechanism is hence $(\ln 3, 0)$-differentially private. The random noise boosts the probability of answering 1 or 0 to at least $\frac{1}{4}$ regardless of diagnoses. Inferences on individual diagnosis can be plausibly denied.

Figure 1b shows the corresponding Markov chain. In the figure, the states $+$ and $-$ denote positive or negative diagnoses respectively; the states s and t denote answers to the survey question and hence $out_1 \in L(s)$ and $out_0 \in L(t)$. States $+$ and $-$ are neighbors. Missing transitions (such as those from s and t) lead to a special state † with a self-loop. We omit such transitions and the state † for clarity.

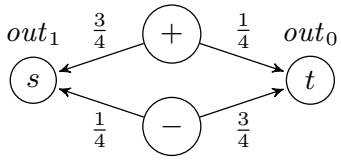
3.2 Truncated α-Geometric Mechanism

More sophisticated differentially private mechanisms are available. Consider a query $f : \mathcal{X}^n \to \{0, 1, \ldots, m\}$. Let $\alpha \in (0, 1)$. The α-*geometric mechanism* outputs $f(x) + Y$

[¶] The MDP we consider is *reactive* in the sense that all actions are enabled in every state.

input \ *output*	1	0
+	$\frac{3}{4} = \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot 1$	$\frac{1}{4} = \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot 0$
−	$\frac{1}{4} = \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot 0$	$\frac{3}{4} = \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot 1$

(a) Survey Mechanism



(b) Corresponding Markov Chain

Fig. 1: Survey Mechanism with $\ln 3$-Differential Privacy

on a dataset x where Y is a random variable with the geometric distribution [21, 22] :

$$\Pr[Y = y] = \frac{1-\alpha}{1+\alpha}\alpha^{|y|} \text{ for } y \in \mathbb{Z}$$

The α-geometric mechanism is oblivious since it has the same output distribution on any inputs x, x' with $f(x) = f(x')$. It is $(-\Delta(f) \ln \alpha, 0)$-differentially private for any query f with sensitivity $\Delta(f)$. Observe that the privacy guarantee $(-\Delta(f) \ln \alpha, 0)$ depends on the sensitivity of the query f. To achieve $(\epsilon, 0)$-differential privacy using the α-geometric mechanism, one first decides the sensitivity of the query and then computes the parameter $\alpha = e^{-\epsilon/\Delta(f)}$.

The range of the mechanism is $\mathbb{Z}$. It may give nonsensical outputs such as negative integers for non-negative queries. The *truncated α-geometric mechanism over* $\{0, 1, \ldots, m\}$ outputs $f(x) + Z$ where Z is a random variable with the distribution:

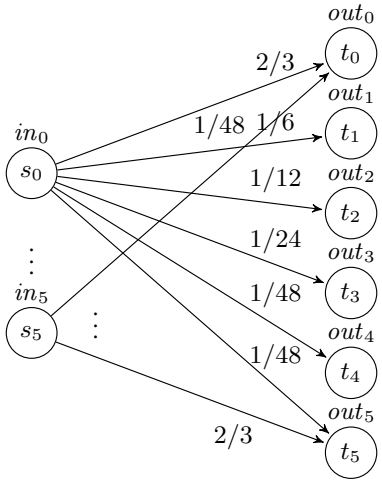
$$\Pr[Z = z] = \begin{cases} 0 & \text{if } z < -f(x) \\ \frac{\alpha^{f(x)}}{1+\alpha} & \text{if } z = -f(x) \\ \frac{1-\alpha}{1+\alpha}\alpha^{|z|} & \text{if } -f(x) < z < m - f(x) \\ \frac{\alpha^{m-f(x)}}{1+\alpha} & \text{if } z = m - f(x) \\ 0 & \text{if } z > m - f(x) \end{cases}$$

Note the range of the truncated α-geometric mechanism is $\{0, 1, \ldots, m\}$. The truncated α-geometric mechanism is again oblivious; it is also $(-\Delta(f) \ln \alpha, 0)$-differentially private for any query f with sensitivity $\Delta(f)$. The truncated $\frac{1}{2}$-geometric mechanism over $\{0, 1, \ldots, 5\}$ is given in Figure 2a.

Similar to the survey mechanism, it is straightforward to model the truncated $\frac{1}{2}$-geometric mechanism as a Markov chain. One could naïvely take datasets as inputs in the formalization, but it is unnecessary. Recall that the truncated $\frac{1}{2}$-geometric mechanism is oblivious. The mechanism depends on query results but not datasets. It hence suffices to consider the range of query f as inputs. Let the state s_k and t_l denote the input k and output l respectively. Define $S = \{s_k, t_k : k \in \{0, 1, \ldots, m\}\}$. The probability transition $\wp(s_k, t_l)$ is the probability of the output l on the input k as defined in the mechanism. Moreover, we have $in_k \in L(s_k)$ and $out_k \in L(t_k)$ for $k \in \{0, 1, \ldots, n\}$. If $\Delta(f) = 1$, $|f(x) - f(x')| \leq 1$ for every neighbors $x, x' \in \mathcal{X}^n$. Subsequently, s_k and s_l are neighbors iff $|k - l| \leq 1$ in our model. Figure 2b gives the Markov chain for the truncated $\frac{1}{2}$-geometric mechanism over $\{0, 1, \ldots, 5\}$.

input \ *output*	0	1	2	3	4	5
0	2/3	1/6	1/12	1/24	1/48	1/48
1	1/3	1/3	1/6	1/12	1/24	1/24
2	1/6	1/6	1/3	1/6	1/12	1/12
3	1/12	1/12	1/6	1/3	1/6	1/6
4	1/24	1/24	1/12	1/6	1/3	1/3
5	1/48	1/48	1/24	1/12	1/6	2/3

(a) $\frac{1}{2}$-Geometric Mechanism



(b) Markov Chain

Fig. 2: A Markov Chain for $\frac{1}{2}$-Geometric Mechanism

3.3 Subsampling Majority

The sensitivity of queries is required to apply the (truncated) α-geometric mechanism. Recall that the sensitivity is the maximal difference of query results on any two neighbors. Two practical problems may arise for mechanisms depending on query sensitivity. First, sensitivity of queries can be hard to compute. Second, the sensitivity over arbitrary neighbors can be too conservative for the actual dataset in use. One therefore would like to have mechanisms independent of query sensitivity.

Subsampling is a technique to design such mechanisms [16]. Concretely, let us consider $\mathcal{X} = \{R, B\}$ (for red and blue team members) and a dataset $d \in \mathcal{X}^n$. Suppose we would like to ask which team is the majority in the dataset while respecting individual privacy. This can be achieved as follows (Algorithm 1). The mechanism first samples m sub-datasets $\hat{d}_1, \hat{d}_2, \ldots, \hat{d}_m$ from d (line 3). It then computes the majority of each sub-dataset and obtains m sub-results. Let $count_R$ and $count_B$ be the number of sub-datasets with the majority R and B respectively (line 4). Since there are m sub-datasets, we have $count_R + count_B = m$. To ensure differential privacy, the mechanism makes sure the difference $|count_R - count_B|$ is significantly large after perturbation. In line 6, $Lap(p)$ denotes the continuous random variable with the probability density function $f(x) = \frac{1}{2p}e^{-|x|/p}$ of the Laplace distribution. If the perturbed difference is sufficiently large, the mechanism reports 1 if the majority of the m sub-results is R or 0 if it is B (line 7). Otherwise, no information is revealed (line 9).

Fix the dataset size n and privacy parameters ϵ, δ, the subsampling majority mechanism can be modeled by a Markov chain. Figure 3 gives a sketch of the Markov chain for $n = 3$. The leftmost four states represent all possible datasets. Given a dataset, m samples are taken with replacement. Outcomes of these samples are denoted by $(count_R, count_B)$. There are only $m + 1$ outcomes: $(m, 0), (m - 1, 1), \ldots, (0, m)$. Each outcome is represented by a state in Figure 3. From each dataset, the proba-

Algorithm 1 Subsampling Majority

1: **function** SUBSAMPLINGMAJORITY(d, f)
Require: $d \in \{R, B\}^n$, $f : \{R, B\}^* \to \{R, B\}$
2: $q, m \leftarrow \frac{\epsilon}{64 \ln(1/\delta)}, \frac{\log(n/\delta)}{q^2}$
3: Subsample m data sets $\hat{d}_1, \hat{d}_2, \ldots, \hat{d}_m$ from d where each row of d is chosen with probability q
4: $count_R, count_B \leftarrow |\{i : f(\hat{d}_i) = R\}|, |\{i : f(\hat{d}_i) = B\}|$
5: $r \leftarrow |count_R - count_B|/(4mq) - 1$
6: **if** $r + Lap(\frac{1}{\epsilon}) > \ln(1/\delta)/\epsilon$ **then**
7: **if** $count_R \geq count_B$ **then return** 1 **else return** 0
8: **else**
9: **return** $\perp$
10: **end function**

bility distribution on all outcomes gives the transition probability. Next, observe that $|count_R - count_B|$ can have only finitely many values. The values of r (line 5) hence belong to a finite set $\{r_m, \ldots, r_M\}$ with the minimum r_m and maximum r_M. For instance, both outcomes $(m, 0)$ and $(0, m)$ transit to the state $r_M = 1/(4q) - 1$ with probability 1. For each $r \in \{r_m, \ldots, r_M\}$, the probability of having $r + Lap(\frac{1}{\epsilon}) > \ln(1/\delta)/\epsilon$ (line 6) is equal to the probability of $Lap(\frac{1}{\epsilon}) > \ln(1/\delta)/\epsilon - r$. This is equal to $\int_{\ln(1/\delta)/\epsilon - r}^{\infty} \frac{\epsilon}{2} e^{-\epsilon|x|} dx$. From each state $r \in \{r_m, \ldots, r_M\}$, it hence goes to the state $\top$ with probability $\int_{\ln(1/\delta)/\epsilon - r}^{\infty} \frac{\epsilon}{2} e^{-\epsilon|x|} dx$ and to the state $\perp$ with probability $1 - \int_{\ln(1/\delta)/\epsilon - r}^{\infty} \frac{\epsilon}{2} e^{-\epsilon|x|} dx$. Finally, the Markov chain moves from the state $\top$ to 1 if $count_R \geq count_B$; otherwise, it moves to 0. Two dataset states are neighbors if they differ at most one member. For example, rrb is a neighbor of rrr and rbb but not bbb.

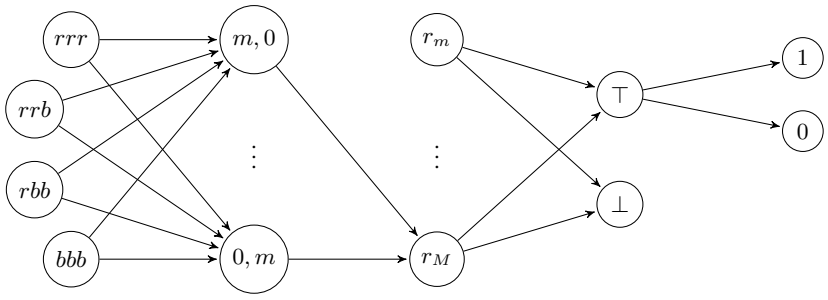


Fig. 3: Markov Chain for Subsampling Majority

4 The Logic dpCTL*

The logic dpCTL* is designed to specify differentially private mechanisms. We introduce the differentially private path quantifier $\mathcal{D}_{\epsilon,\delta}$ and neighborhood relations for

neighbors in dpCTL*. For any path formula ϕ, a state s in a Markov chain K satisfies $\mathcal{D}_{\epsilon,\delta}\phi$ if the probability of having paths satisfying ϕ from s is close to the probabilities of having paths satisfying ϕ from its neighbors.

4.1 Syntax

The syntax of dpCTL* state and path formulae is given by:

$$\Phi ::= p \mid \neg\Phi \mid \Phi \wedge \Phi \mid \mathbb{P}_J\phi \mid \mathcal{D}_{\epsilon,\delta}\phi$$
$$\phi ::= \Phi \mid \neg\phi \mid \phi \wedge \phi \mid \mathsf{X}\phi \mid \phi \,\mathsf{U}\, \phi$$

A *state* formula Φ is either an atomic proposition p, the negation of a state formula, the conjunction of two state formulae, the *probabilistic* operator $\mathbb{P}_J$ with J an interval in $[0,1]$ followed by a path formula, or the *differentially private* operator $\mathcal{D}_{\epsilon,\delta}$ with two non-negative real numbers ϵ and δ followed by a path formula. A *path* formula ϕ is simply a linear temporal logic formula, with temporal operator *next* (X) followed by a path formula, and *until* operator (U) enclosed by two path formulae. We define $F\phi \equiv true\ \mathsf{U}\ \phi$ and $G\phi \equiv \neg F(\neg\phi)$ as usual.

As in the classical setting, we consider the sublogic dpCTL by allowing only path formulae of the form $\mathsf{X}\Phi$ and $\Phi\,\mathsf{U}\,\Phi$. Moreover, one obtains PCTL [23] and PCTL* [11] from dpCTL and dpCTL* by removing the differentially private operator $\mathcal{D}_{\epsilon,\delta}$.

5 dpCTL* for Markov Chains

Given a Markov chain $K = (S, \wp, L)$, a *neighborhood relation* $N_S \subseteq S \times S$ is a reflexive and symmetric relation on S. We will write sN_St when $(s,t) \in N_S$. If sN_St, we say s and t are *neighbors* or t is a *neighbor* of s. For any Markov chain K, neighborhood relation N on S, $s \in S$, and a path formula ϕ, define

$$\Pr_N^K(s,\phi) = \Pr[\{\pi : K, N, \pi \models \phi \text{ with } \pi_0 = s\}].$$

That is, $\Pr_N^K(s,\phi)$ denotes the probability of paths satisfying ϕ from s on K with N. Define the satisfaction relation $K, N_S, s \models \Phi$ as follows.

$$\begin{aligned}
K, N_S, s &\models p \text{ if } p \in L(s)\\
K, N_S, s &\models \neg\Phi \text{ if } K, N_S, s \not\models \Phi\\
K, N_S, s &\models \Phi_0 \wedge \Phi_1 \text{ if } K, N_S, s \models \Phi_0 \text{ and } K, N_S, s \models \Phi_1\\
K, N_S, s &\models \mathbb{P}_J\phi \text{ if } \Pr_{N_S}^K(s,\phi) \in J\\
K, N_S, s &\models \mathcal{D}_{\epsilon,\delta}\phi \text{ if for every } t \text{ with } sN_St, \Pr_{N_S}^K(s,\phi) \le e^\epsilon \Pr_{N_S}^K(t,\phi) + \delta \text{ and}\\
&\qquad \Pr_{N_S}^K(t,\phi) \le e^\epsilon \Pr_{N_S}^K(s,\phi) + \delta
\end{aligned}$$

Moreover, the relation $K, N_S, \pi \models \phi$ is defined as in the standard linear temporal logic formulae [12]. We only recall the semantics for the temporal operators X and U:

$$\begin{aligned}
K, N_S, \pi &\models \mathsf{X}\phi \text{ if } K, N_S, \pi[1] \models \phi\\
K, N_S, \pi &\models \phi\,\mathsf{U}\,\psi \text{ if there is a } j \ge 0 \text{ such that } K, N_S, \pi[j] \models \psi \text{ and}\\
&\qquad K, N_S, \pi[k] \models \phi \text{ for every } 0 \le k < j
\end{aligned}$$

Other than the differentially private operator, the semantics of dpCTL* is standard [2]. To intuit the semantics of $\mathcal{D}_{\epsilon,\delta}\phi$, recall that $\Pr_N^K(s,\phi)$ is the probability of having paths satisfying ϕ from s. A state s satisfies $\mathcal{D}_{\epsilon,\delta}\phi$ if the probability of having paths satisfying ϕ from s is (ϵ,δ)-close to those from every neighbor of s. Informally, it is probabilistically similar to observe paths satisfying ϕ from s and from its neighbors.

5.1 Model Checking

We describe the model checking algorithm for dpCTL. The algorithm follows the classical algorithms for PCTL by computing the states satisfying sub state-formulae inductively [23, 2]. It hence suffices to consider the inductive step where the states satisfying the subformula $\mathcal{D}_{\epsilon,\delta}(\phi)$ are to be computed.

In the classical PCTL model checking algorithm for Markov chains, states satisfying the subformula $\mathbb{P}_J\phi$ are obtained by computing $\Pr_{N_S}^K(s,\phi)$ for $s \in S$. These probabilities can be obtained by solving linear equations or through iterative approximations. We summarize it in the following theorem (details see [2]):

Lemma 1. *Let $K = (S, \wp, L)$ be a Markov chain, N_S a neighborhood relation on S, $s \in S$, and $B, C \subseteq S$. The probabilities $\Pr_{N_S}^K(s, \bigcirc B)$ and $\Pr_{N_S}^K(s, B \mho C)$ are computable within time polynomial in $|S|$.*

In Lemma 1, we abuse the notation slightly to admit path formulae of the form $\bigcirc B$ (next B) and $B \mho C$ (B until C) with $B, C \subseteq S$ as in [2]. They are interpreted by introducing new atomic propositions B and C for each $s \in B$ and $s \in C$ respectively.

In order to determine the set $\{s : K, N_S, s \models \mathcal{D}_{\epsilon,\delta}\phi\}$, our algorithm computes the probabilities $p(s) = \Pr_{N_S}^K(s,\phi)$ for every $s \in S$ (Algorithm 2). For each $s \in S$, it then compares the probabilities $p(s)$ and $p(t)$ for every neighbor t of s. If there is a neighbor t such that $p(s)$ and $p(t)$ are not (ϵ,δ)-close, the state s is removed from the result. Algorithm 2 returns all states which are (ϵ,δ)-close to their neighbors. The algorithm requires at most $O(|S|^2)$ additional steps. We hence have the following results:

Proposition 1. *Let $K = (S, \wp, L)$ be a Markov chain, N_S a neighborhood relation on S, and ϕ a dpCTL path formula. $\{s : K, N_S, s \models \mathcal{D}_{\epsilon,\delta}\phi\}$ is computable within time polynomial in $|S|$ and $|\phi|$.*

Corollary 1. *Let $K = (S, \wp, L)$ be a Markov chain, N_S a neighborhood relation on S, and Φ a dpCTL formula. $\{s : K, N_S, s \models \Phi\}$ is computable within time polynomial in $|S|$ and $|\Phi|$.*

The model checking algorithm for dpCTL* can be treated as in the classical setting [2]: all we need is to compute the probability $\Pr_{N_S}^K(s,\phi)$ with general path formula ϕ. For this purpose one first constructs a deterministic ω-automaton R for ϕ. Then, the probability reduces to a reachability probability in the product Markov chain obtained from K and R. There are more efficient algorithms without the product construction, see [13, 14, 3] for details.

Algorithm 2 SAT(K, N_S, ϕ)

procedure SAT(K, N_S, ϕ)
  match ϕ **with** ▷ by Lemma 1
    case $\mathsf{X}\Psi$:
      $B \leftarrow \mathrm{SAT}(K, N_S, \Psi)$
      $p(s) \leftarrow \Pr^K_{N_S}(s, \bigcirc B)$ for every $s \in S$
    case $\Psi \,\mathsf{U}\, \Psi'$:
      $B \leftarrow \mathrm{SAT}(K, N_S, \Psi)$
      $C \leftarrow \mathrm{SAT}(K, N_S, \Psi')$
      $p(s) \leftarrow \Pr^K_{N_S}(s, B \,\mho\, C)$ for every $s \in S$
  $R \leftarrow S$
  for $s \in S$ **do**
    for t with sN_St **do**
      if $p(s) \not\leq e^\epsilon p(t) + \delta$ or $p(t) \not\leq e^\epsilon p(s) + \delta$ **then** remove s from R
  return R
end procedure

6 Specifying Properties in dpCTL*

In this section we describe how properties in the differential privacy literature can be expressed using dpCTL* formulae.

Differential Privacy. Consider the survey mechanism (Section 3.1). For v with uNv, we have $\Pr^K_N(u, \mathsf{X}out_1) \leq 3\Pr^K_N(v, \mathsf{X}out_1)$ for the probabilities of satisfying $\mathsf{X}out_1$ from u and v. The formula $\mathcal{D}_{\ln 3,0}(\mathsf{X}out_1)$ holds in state u and similarly for $\mathcal{D}_{\ln 3,0}(\mathsf{X}out_0)$. Recall that differential privacy requires similar output distributions on neighbors. The formula $\mathcal{D}_{\ln 3,0}(\mathsf{X}out_1) \wedge \mathcal{D}_{\ln 3,0}(\mathsf{X}out_0)$ thus specifies differential privacy for states $+$ and $-$. The survey mechanism is $(\ln 3, 0)$-differentially private.

For the $\frac{1}{2}$-geometric mechanism (Section 3.2), define the formula $\psi = \mathcal{D}_{\ln 2,0}(\mathsf{X}out_0) \wedge \mathcal{D}_{\ln 2,0}(\mathsf{X}out_1) \wedge \cdots \wedge \mathcal{D}_{\ln 2,0}(\mathsf{X}out_5)$. If the state s_k satisfies ψ for $k = 0, \ldots, 5$, then the $\frac{1}{2}$-geometric mechanism is $(\ln 2, 0)$-differentially private. For the subsampling majority mechanism (Section 3.3), consider the formula $\psi = \mathcal{D}_{\epsilon,\delta}(\mathsf{F}0) \wedge \mathcal{D}_{\epsilon,\delta}(\mathsf{F}1)$. If a state satisfies ψ, its probability of outputting is (ϵ, δ)-close to those of its neighbor for every outcomes. The subsampling majority mechanism is (ϵ, δ)-differentially private.

Compositionality. Compositionality is one of the building blocks for differential privacy. For any (ϵ_1, δ_1)-differentially private mechanism M_1 and (ϵ_2, δ_2)-differentially private mechanism M_2, their combination $(M_1(x), M_2(x))$ is $(\epsilon_1+\epsilon_2, \delta_1+\delta_2)$-differentially private by the compositional theorem [16, Theorem 3.16]. The degradation is rooted in the repeated releases of information. To illustrate this property, we consider the extended survey mechanism which allows two consecutive queries. In this mechanism, an input is either $+$ or $-$; but outputs are $out_1\,out_1$, $out_1\,out_0$, $out_0\,out_1$, or $out_0\,out_0$. The model is depicted in Figure 4.

Consider the formula $\mathcal{D}_{\ln 9,0}(\mathsf{X}(out_1 \wedge \mathsf{X}out_1))$. A path satisfies $\mathsf{X}(out_1 \wedge \mathsf{X}out_1)$ if the second state satisfies out_1 and the third state satisfies out_1 as well. We verify

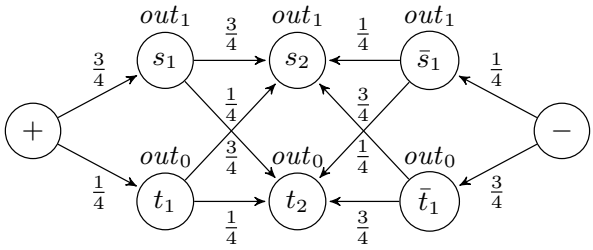


Fig. 4: Markov Chain of Double Surveys

that this formula is satisfied for states $+$ and $-$. Moreover, the bound $\epsilon = \ln 9$ is tight since the probability of satisfying $\mathsf{X}(out_1 \wedge \mathsf{X}out_1)$ from states $+$ and $-$ are $\frac{9}{16}$ and $\frac{1}{16}$ respectively. Finally, the formula $\wedge_{a_1,a_2}\mathcal{D}_{\ln 9,0}(\mathsf{X}(a_1 \wedge \mathsf{X}a_2))$ specifies differential privacy for the model, where a_1, a_2 range over atomic propositions $\{out_1, out_0\}$.

Let us consider two slightly different formulae for comparison:

- $\mathcal{D}_{\ln 3,0}(\mathsf{XX}out_1)$. In this case we claim there is no privacy loss, even though there are two queries. The reason is that the output of the first query is not observed at all. It is easy to verify that it is indeed satisfied by $+$ and $-$.
- $\mathcal{D}_{\ln 3,0}(\mathsf{X}(out_1 \wedge \mathcal{D}_{\ln 3,0}(\mathsf{X}out_1)))$. This is a nested dpCTL formula, where the inner state formula $\mathcal{D}_{\ln 3,0}(\mathsf{X}out_1)$ specifies the one-step differential privacy. Observe the inner formula is satisfied by all states. The outer formula has no privacy loss.

Tighter Privacy Bounds for Composition. An advantage of applying model checking is that we may get tighter bounds for composition. Consider the survey mechanism, and the property $\mathcal{D}_{0,.5}(\mathsf{X}out_1)$. Obviously, it holds in states $+$ and $-$ since $\Pr^K_N(u, out_1) = \frac{3}{4}, \frac{1}{4}$ for $u = +, -$ respectively (Figure 1). A careful check infers that one cannot decrease $\delta_1 = .5$ without increasing ϵ. Now consider the formula $\mathcal{D}_{\epsilon_2,\delta_2}(\mathsf{X}(out_1 \wedge \mathsf{X}out_1))$ in Figure 4. Applying the compositional theorem, one has $\epsilon_2 = 2\epsilon_1 = 0$ and $\delta_2 = 2\delta_1 = 1$. However, we can check easily that one gets better privacy parameter $(0, .5)$ using the model checking algorithm because $\Pr^K_N(u, out_1) = \frac{9}{16}, \frac{1}{16}$ for $u = +, -$ respectively. In general, compositional theorems for differential privacy only give asymptotic upper bounds. Privacy parameters ϵ and δ must be calculated carefully and often pessimistically. Our algorithm allows data analysts to choose better parameters.

7 Differentially Private Mechanisms as Markov Decision Processes

In differential privacy, an *offline* mechanism releases outputs only once and plays no further role; an *online* (or *interactive*) mechanism allows analysts to ask queries adaptively based on previous responses. The mechanisms considered previously are offline mechanisms. Since offline mechanisms only release one query result, they are relatively easy to analyze. For online mechanisms, one has to consider all possible adaptive queries. We therefore use MDPs to model these non-deterministic behaviors. Specifically, adaptive queries are modeled by actions. Randomized computation associated with different queries is modeled by distributions associated with actions.

Consider again the survey mechanism. Suppose we would like to design an interactive mechanism which adjusts random noises on surveyors' requests. When the surveyor requests low-accuracy answers, the surveyee uses the survey mechanism in Section 3.1. When high-accuracy answers are requested, the surveyee answers 1 with probability $\frac{4}{5}$ and 0 with probability $\frac{1}{5}$ when she has positive diagnosis. She answers 1 with probability $\frac{1}{5}$ and 0 with probability $\frac{4}{5}$ when she is not diagnosed with the disease X. This gives an interactive mechanism corresponding to the MDP shown in Figure 5.

In the figure, the states $+$, $-$, s, and t are interpreted as before. The actions L and H denote low- and high-accuracy queries respectively. Note that the high-accuracy survey mechanism is $(\ln 4, 0)$-differentially private. Unlike non-interactive mechanisms, the privacy guarantees vary from queries with different accuracies.

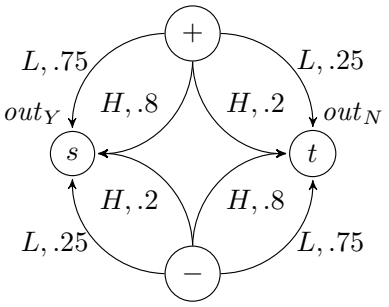


Fig. 5: Markov Decision Process

7.1 Above Threshold Mechanism

Below we describe an online mechanism from [16]. Given a threshold and a series of adaptive queries, we care for the queries whose results are above the threshold; queries below the threshold only disclose minimal information and hence is irrelevant. Let us assume the mechanism will halt on the first such query result for simplicity. In [16], a mechanism is designed for continuous queries by applying the Laplace mechanism. We will develop a mechanism for discrete bounded queries using the truncated geometric mechanism.

Assume that we have a threshold $t \in \{0, 1, \ldots, 5\}$ and queries $\{f_i : \Delta(f_i) = 1\}$. In order to protect privacy, our mechanism applies the truncated $\frac{1}{4}$-geometric mechanism to obtain a perturbed threshold t'. For each query f_i, the truncated $\frac{1}{2}$-geometric mechanism is applied to its result $r_i = f_i(x)$. If the perturbed result r'_i is not less than the perturbed threshold t', the mechanism halts with the output $\top$. Otherwise, it outputs $\perp$ and continues to the next query (Algorithm 3). The above threshold mechanism outputs a sequence of the form $\perp^*\top$. On similar datasets, we want to show that the above threshold mechanism outputs the same sequence with similar probabilities.

It is not hard to model the above threshold mechanism as a Markov decision process (Figure 6). In the figure, we sketch the model where the threshold and query results are in $\{0, 1, 2\}$. The model simulates two computation in parallel: one for the dataset, the other for its neighbor. The state $t_i r_j$ represents the input threshold i and the first query result j; the state $t'_i r'_j$ represents the perturbed threshold i and the perturbed query result j. Other states are similar. Consider the state $t_0 r_1$. After applying the truncated $\frac{1}{4}$-geometric mechanism, it goes to one of the states $t'_0 r_1$, $t'_1 r_1$, $t'_2 r_1$ accordingly. From the state $t'_1 r_1$, for instance, it moves to one of $t'_1 r'_0$, $t'_1 r'_1$, $t'_1 r'_2$ by applying the truncated $\frac{1}{2}$-geometric mechanism to the query result. If it arrives at $t'_1 r'_1$ or $t'_1 r'_2$, the perturbed query result is not less than the perturbed threshold. The model halts with the output $\top$ by entering the state with a self loop. Otherwise, it moves to one of $t'_1 r_0$, $t'_1 r_1$, or $t'_1 r_2$

Algorithm 3 Input: private database d, queries $f_i : d \to \{0, 1, \ldots, 5\}$ with sensitivity 1, threshold $t \in \{0, 1, \ldots, 5\}$; Output: $a_1, a_2, \ldots$

procedure ABOVETHRESHOLD(d, $\{f_1, f_2, \ldots\}$, t)
  match t **with** ▷ obtain t' by $\frac{1}{4}$-geometric mechanism
    case 0: $t' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{4}{5}, \frac{3}{20}, \frac{3}{80}, \frac{3}{320}, \frac{3}{1280}, \frac{1}{1280}$ respectively
    case 1: $t' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{1}{5}, \frac{3}{5}, \frac{3}{20}, \frac{3}{80}, \frac{3}{320}, \frac{1}{320}$ respectively
    case 2: $t' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{1}{20}, \frac{3}{20}, \frac{3}{5}, \frac{3}{20}, \frac{3}{80}, \frac{1}{80}$ respectively
    case 3: $t' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{1}{80}, \frac{3}{80}, \frac{3}{20}, \frac{3}{5}, \frac{3}{20}, \frac{1}{20}$ respectively
    case 4: $t' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{1}{320}, \frac{3}{320}, \frac{3}{80}, \frac{3}{20}, \frac{3}{5}, \frac{1}{5}$ respectively
    case 5: $t' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{1}{1280}, \frac{3}{1280}, \frac{3}{320}, \frac{3}{80}, \frac{3}{20}, \frac{4}{5}$ respectively
  for each query f_i **do**
    $r_i \leftarrow f_i(d)$
    match r_i **with** ▷ obtain r_i' by $\frac{1}{2}$-geometric mechanism
      case 0: $r_i' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{2}{3}, \frac{1}{6}, \frac{1}{12}, \frac{1}{24}, \frac{1}{48}, \frac{1}{48}$ respectively
      case 1: $r_i' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{1}{3}, \frac{1}{3}, \frac{1}{6}, \frac{1}{12}, \frac{1}{24}, \frac{1}{24}$ respectively
      case 2: $r_i' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{1}{6}, \frac{1}{6}, \frac{1}{3}, \frac{1}{6}, \frac{1}{12}, \frac{1}{12}$ respectively
      case 3: $r_i' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{1}{12}, \frac{1}{12}, \frac{1}{6}, \frac{1}{3}, \frac{1}{6}, \frac{1}{6}$ respectively
      case 4: $r_i' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{1}{24}, \frac{1}{24}, \frac{1}{12}, \frac{1}{6}, \frac{1}{3}, \frac{1}{3}$ respectively
      case 5: $r_i' \leftarrow 0, 1, 2, 3, 4, 5$ with probability $\frac{1}{48}, \frac{1}{48}, \frac{1}{24}, \frac{1}{12}, \frac{1}{6}, \frac{2}{3}$ respectively
    if $r_i' \geq t'$ **then halt** with $a_i = \top$ **else** $a_i = \bot$
end procedure

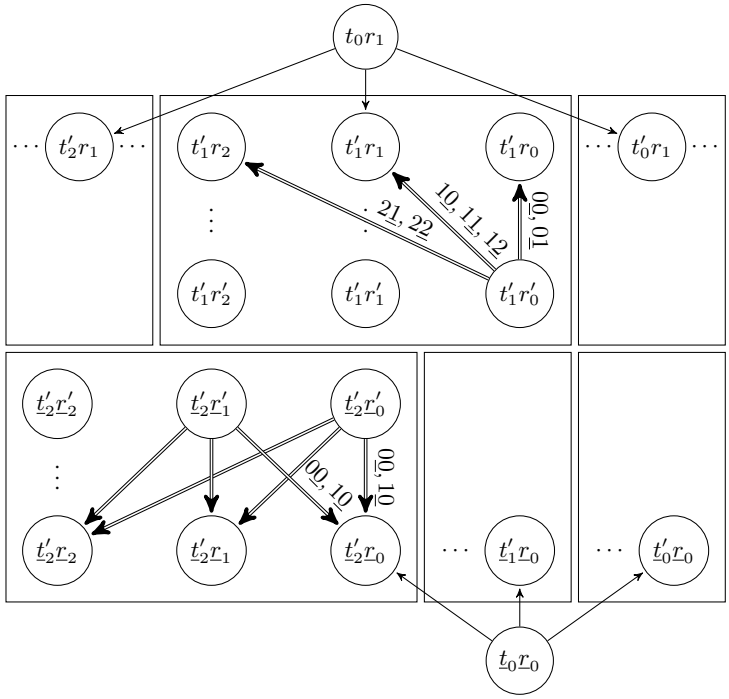


Fig. 6: Markov Decision Process for Above Threshold

non-deterministically (double arrows). The computation of its neighbor is similar. We just use the underlined symbols to represent threshold and query results. For instance, the state $\underline{t'_2}\underline{r'_1}$ represents the perturbed threshold 2 and the perturbed query result 1 in the neighbor.

Now, the non-deterministic choices in the two computation cannot be independent. Recall that the sensitivity of each query is 1. If the top computation moves to the state, say, $t'_1 r_0$, it means the next query result on the dataset is 0. Subsequently, the bottom computation can only move to $\underline{t'_j}\underline{r}_0$ or $\underline{t'_j}\underline{r}_1$ depending on its perturbed threshold. This is where actions are useful. Define the actions $\{m\underline{n} : |m-n| \leq 1\}$. The action $m\underline{n}$ represents that the next query result for the dataset and its neighbor are m and n respectively. For instance, the non-deterministic choice from $t'_1 r'_0$ to $t'_1 r_0$ is associated with two actions $0\underline{0}$ and $0\underline{1}$ (but not $0\underline{2}$). Similarly, the choice from $\underline{t'_2}\underline{r'_1}$ to $\underline{t'_2}\underline{r}_0$ is associated with the actions $0\underline{0}$ and $1\underline{0}$ (but not $2\underline{0}$). Assume the perturbed thresholds of the top and bottom computation are i and j respectively. On the action $0\underline{0}$, the top computation moves to $t'_i r_0$ and the bottom computation moves to $\underline{t'_j}\underline{r}_0$. Actions make sure the two computation of neighbors is modeled properly. Now consider the action sequence $-, -, 0\underline{1}, -, 2\underline{2}, -, 2\underline{1}$ from the states $t_0 r_1$ and $\underline{t_0}\underline{r}_0$ ("$-$" represents the purely probabilistic action). Together with the first query results, it denotes four consecutive query results 1, 0, 2, 2 on the top computation, and 0, 1, 2, 1 on the bottom computation. Each action sequence models two sequences of query results: one on the top, the other on the bottom computation. Moreover, the difference of the corresponding query results on the two computation is at most one by the definition of the action set. Any sequence of adaptive query results is hence formalized by an action sequence in our model.

It remains to define the neighborhood relation. Recall the sensitivity is 1. Consider the neighborhood relation $\{(t_i r_m, t_i r_m), (\underline{t_i}\underline{r_n}, \underline{t_i}\underline{r_n}), (t_i r_m, \underline{t_i}\underline{r_n}), (\underline{t_i}\underline{r_n}, t_i r_m) : |m-n| \leq 1\}$. That is, two states are neighbors if they represent two inputs of the same threshold and query results with difference at most one.

8 dpCTL* for Markov Decision Processes

The logic dpCTL* can be interpreted over MDPs. Let $M = (S, Act, \wp, L)$ be an MDP and N_S a neighborhood relation on S. Define the satisfaction relation $M, N_S, s \models \Phi$ for $\mathbb{P}_J \phi$ and $\mathcal{D}_{\epsilon,\delta}\phi$ as follows (others are straightforward).

$$M, N_S, s \models \mathbb{P}_J \phi \text{ if } \Pr_{N_S}^{M_{\mathfrak{S}}}(s, \phi) \in J \text{ for every scheduler } \mathfrak{S}$$

$$M, N_S, s \models \mathcal{D}_{\epsilon,\delta}\phi \text{ if for all } t \text{ with } sN_S t \text{ and query scheduler } \mathfrak{Q}, \Pr_{N_S}^{M_{\mathfrak{Q}}}(s, \phi) \leq e^{\epsilon} \cdot \Pr_{N_S}^{M_{\mathfrak{Q}}}(t, \phi) + \delta \text{ and } \Pr_{N_S}^{M_{\mathfrak{Q}}}(t, \phi) \leq e^{\epsilon} \cdot \Pr_{N_S}^{M_{\mathfrak{Q}}}(s, \phi) + \delta$$

Recall that $M_{\mathfrak{S}}$ is but a Markov chain. The semantics of $M_{\mathfrak{S}}, N_S, \pi \models \phi$ and hence the probability $\Pr_{N_S}^{M_{\mathfrak{S}}}(s, \phi)$ are defined as in Markov chains. The semantics of dpCTL* on MDPs is again standard except the differentially private operator $\mathcal{D}_{\epsilon,\delta}$. For any path formula ϕ, $\mathcal{D}_{\epsilon,\delta}\phi$ specifies states whose probability of having paths satisfying ϕ are (ϵ, δ)-close to those of all its neighbors for query schedulers. That is, no query scheduler can force any of neighbors to distinguish the specified path behavior probabilistically.

Justification of query schedulers. We use query schedulers in the semantics for the differentially private operator. A definition with history-dependent schedulers might be

$$M, N_S, s \models \mathcal{D}^{bad}_{\epsilon,\delta}\phi \text{ if for all } t \text{ with } sN_St \text{ and scheduler } \mathfrak{S}, \Pr^{M_{\mathfrak{S}}}_{N_S}(s,\phi) \leq e^{\epsilon} \cdot \Pr^{M_{\mathfrak{S}}}_{N_S}(t,\phi) + \delta \text{ and } \Pr^{M_{\mathfrak{S}}}_{N_S}(t,\phi) \leq e^{\epsilon} \cdot \Pr^{M_{\mathfrak{S}}}_{N_S}(s,\phi) + \delta.$$

A state satisfies $\mathcal{D}^{bad}_{\epsilon,\delta}\phi$ if no history-dependent scheduler can differentiate the probabilities of having paths satisfying ϕ from neighbors. Recall that a history-dependent scheduler chooses actions according to previous states. Such a definition would allow schedulers to take different actions from different states. Two neighbors could hence be differentiated by different action sequences. The specification might be too strong for our purposes. A query scheduler $\mathfrak{Q} : S^+ \to Act$, on the other hand, corresponds to a query sequence. A state satisfies $\mathcal{D}_{\epsilon,\delta}\phi$ if no query sequence can differentiate the probabilities of having paths satisfying ϕ from neighbors. Recall query schedulers only depend on lengths of histories. Two neighbors cannot be distinguished by the same action sequence of any length if they satisfy a differentially private subformula. Our semantics agrees with the informal interpretation of differential privacy for such systems. We therefore consider only query schedulers in our definition.

8.1 Model Checking

Given an MDP $M = (S, Act, \wp, L)$, a neighborhood relation N_S, $s \in S$, and a path formula ϕ, consider the problem of checking $M, N_S, s \models \mathcal{D}_{\epsilon,\delta}\phi$. Recall the semantics of $\mathcal{D}_{\epsilon,\delta}\phi$. Given s, t with sN_St and a path formula ϕ, we need to decide whether $\Pr^{M_{\mathfrak{Q}}}_{N_S}(s,\phi) \leq e^{\epsilon}\Pr^{M_{\mathfrak{Q}}}_{N_S}(t,\phi) + \delta$ for every query scheduler $\mathfrak{Q}$. When ϕ is $\bigcirc B$ with $B \subseteq S$, only the first action in the query sequence needs to be considered. This can also be easily generalized to nested next operators: one needs only to enumerate all actions query sequences of a fixed length. The problem however is undecidable in general.

Theorem 1. *The dpCTL* model checking problem for MDPs is undecidable.*

The proof is in Appendix. We discuss some decidable special cases. Consider the formula $\phi := FB$ with $B \subseteq S$ and assume that states in B with only self-loops. For the case $\epsilon = 0$, the condition reduces to $\Pr^{M_{\mathfrak{Q}}}_{N_S}(s, FB) - \Pr^{M_{\mathfrak{Q}}}_{N_S}(t, FB) \leq \delta$. If $\delta = 0$ it is the classical language equivalence problem for probabilistic automata [28], which can be solved in polynomial time. However, if $\delta > 0$, the problem becomes an approximate version of the language equivalence problem. To the best of our knowledge, its decidability is still open except for the special case where all states are connected [32].

Despite of the negative result in Theorem 1, a sufficient condition for $M, N_S, s \models \mathcal{D}_{\epsilon,\delta}\phi$ is available. To see this, observe that for $s \in S$ and query scheduler $\mathfrak{Q}$, we have

$$\min_{\mathfrak{S}} \Pr^{M_{\mathfrak{S}}}_{N_S}(s,\phi) \leq \Pr^{M_{\mathfrak{Q}}}_{N_S}(s,\phi) \leq \max_{\mathfrak{S}} \Pr^{M_{\mathfrak{S}}}_{N_S}(s,\phi)$$

where the minimum and maximum are taken over all schedulers $\mathfrak{S}$. Hence,

$$\Pr^{M_{\mathfrak{Q}}}_{N_S}(s,\phi) - e^{\epsilon} \cdot \Pr^{M_{\mathfrak{Q}}}_{N_S}(t,\phi) \leq \max_{\mathfrak{S}} \Pr^{M_{\mathfrak{S}}}_{N_S}(s,\phi) - e^{\epsilon} \cdot \min_{\mathfrak{S}} \Pr^{M_{\mathfrak{S}}}_{N_S}(t,\phi)$$

for any $s, t \in S$ and query scheduler $\mathfrak{Q}$. We have the following proposition:

Proposition 2. *Let $M = (S, Act, \wp, L)$ be an MDP, N_S a neighborhood relation on S. $M, N_S, s \models \mathcal{D}_{\epsilon,\delta}\phi$ if $\max_{\mathfrak{S}} \Pr^{M_{\mathfrak{S}}}_{N_S}(s,\phi) - e^{\epsilon} \cdot \min_{\mathfrak{S}} \Pr^{M_{\mathfrak{S}}}_{N_S}(t,\phi) \leq \delta$ and $\max_{\mathfrak{S}} \Pr^{M_{\mathfrak{S}}}_{N_S}(t,\phi) - e^{\epsilon} \cdot \min_{\mathfrak{S}} \Pr^{M_{\mathfrak{S}}}_{N_S}(s,\phi) \leq \delta$ for any $s, t \in S$ with sN_St.*

For $s \in S$, recall that $\max_{\mathfrak{S}} \Pr^{M_{\mathfrak{S}}}_{N_S}(s,\phi)$ and $\min_{\mathfrak{S}} \Pr^{M_{\mathfrak{S}}}_{N_S}(s,\phi)$ can be efficiently computed [2]. By Proposition 2, $M, N_S, s \models \mathcal{D}_{\epsilon,\delta}\phi$ can be checked soundly and efficiently.

We model the above threshold algorithm (Algorithm 3) and apply Proposition 2 to check whether the mechanism is differentially private using the classical PCTL model checking algorithm for MDPs. Since concrete values of the parameters ϵ and δ are computed, tighter bounds for specific neighbors can be obtained. For instance, for the state $t_3 r_5$ and its neighbor $\underline{t}_3 \underline{r}_4$, we verify the property $\bigwedge_{k \in \mathbb{Z}^{\geq 0}} \mathcal{D}_{0,0.17}((\mathsf{X}^k \bot)\top)$ is satisfied. Note the reachability probability goes to 0 as k goes to infinity. By repeating the computation, we verify that the property $\bigwedge_{k \in \mathbb{Z}^{\geq 0}} \mathcal{D}_{1,0.74}((\mathsf{X}^k \bot)\top)$ is satisfied for all neighbors. Subsequently, the above threshold mechanism in Algorithm 3 is $(1, 0.74)$-differentially private. Compared to the parameters for the neighbors $t_3 r_5$ and $\underline{t}_3 \underline{r}_4$, the parameter δ appears to be significantly large. It means that there are two neighbors with drastically different output distributions from our mechanism. Moreover, recall that Proposition 2 is a sufficient condition. It only gives an upper bound of privacy parameters. Tighter bounds may be computed by more sophisticated sufficient conditions.

9 Conclusions

We have introduced dpCTL* to reason about properties in differential privacy, and investigated its model checking problems. For Markov chains, the model checking problem has the same complexity as for PCTL*. The general MDP model checking problem however is undecidable. We have discussed some decidable special cases and a sufficient yet efficient condition to check differentially private subformulae. An interesting future work is to identify more decidable subclasses and sufficient conditions. As an example, consider the extended dpCTL* formula $\bigwedge_{k \in \mathbb{Z}^{\geq 0}} \mathcal{D}_{\epsilon,\delta}(\mathsf{X}^k \top)$. For the case $\epsilon = \delta = 0$, it reduces to a language equivalence problem for probabilistic automata. It is interesting to characterize other cases as well. Another interesting line of further works is to consider continuous perturbation (such as Laplace distribution used in [16]). We would need Markov models with continuous state space.

References

1. Alvim, M.S., Andrés, M.E., Chatzikokolakis, K., Degano, P., Palamidessi, C.: On the information leakage of differentially-private mechanisms. Journal of Computer Security 23(4), 427–469 (2015)
2. Baier, C., Katoen, J.P.: Principles of Model Checking. The MIT Press (2008)
3. Baier, C., Kiefer, S., Klein, J., Klüppelholz, S., Müller, D., Worrell, J.: Markov chains and unambiguous Büchi automata. In: CAV. LNCS, vol. 9779, pp. 23–42. Springer (2016)
4. Barthe, G., Danezis, G., Grégoire, B., Kunz, C., Zanella-Béguelin, S.: Verified computational differential privacy with applications to smart metering. In: CSF. pp. 287–301. IEEE (2013)

5. Barthe, G., Farina, G.P., Gaboardi, M., Arias, E.J.G., Gordon, A., Hsu, J., Strub, P.Y.: Differentially private Bayesian programming. In: CCS. pp. 68–79. ACM (2016)
6. Barthe, G., Fong, N., Gaboardi, M., Grégoire, B., Hsu, J., Strub, P.Y.: Advanced probabilistic couplings for differential privacy. In: CCS. pp. 55–67. ACM (2016)
7. Barthe, G., Gaboardi, M., Arias, E.J.G., Hsu, J., Kunz, C., Strub, P.Y.: Proving differential privacy in Hoare logic. In: CSF. pp. 411–424. IEEE (2014)
8. Barthe, G., Gaboardi, M., Arias, E.J.G., Hsu, J., Roth, A., Strub, P.: Higher-order approximate relational refinement types for mechanism design and differential privacy. In: POPL. pp. 68–79. ACM (2015)
9. Barthe, G., Gaboardi, M., Gregoire, B., Hsu, J., Strub, P.Y.: Proving differential privacy via probabilistic couplings. In: LICS. IEEE (2016)
10. Barthe, G., Köpf, B., Olmedo, F., Zanella-Béguelin, S.: Probabilistic relational reasoning for differential privacy. In: POPL. pp. 97–110. ACM (2012)
11. Bianco, A., de Alfaro, L.: Model checking of probabalistic and nondeterministic systems. In: FSTTCS. LNCS, vol. 1026, pp. 499–513. Springer (1995)
12. Clarke, E.M., Grumberg, O., Peled, D.: Model Checking. The MIT Press (1999)
13. Courcoubetis, C., Yannakakis, M.: The complexity of probabilistic verification. Journal of the ACM 42(4), 857–907 (1995)
14. Couvreur, J.M., Saheb, N., Sutre, G.: An optimal automata approach to LTL model checking of probabilistic systems. In: LPAR. LNCS, vol. 2850, pp. 361–375 (2003)
15. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC. LNCS, vol. 3876, pp. 265–284. Springer (06)
16. Dwork, C., Roth, A.: The algorithmic foundations of differential privacy. Foundations and Trends in Theoretical Computer Science 9(3–4), 211–407 (2014)
17. Dwork, C.: Differential privacy. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP. LNCS, vol. 4052, pp. 1–12. Springer (2006)
18. Fung, B.C.M., Wang, K., Chen, R., Yu, P.S.: Privacy-preserving data publish: A survey of recent developments. ACM Computing Surveys 42(4), 14:1–14:53 (June 2010)
19. Gaboardi, M., Haeberlen, A., Hsu, J., Narayan, A., Pierce, B.C.: Linear dependent types for differential privacy. In: POPL. pp. 357–370 (2013)
20. Gazeau, I., Miller, D., Palamidessi, C.: Preserving differential privacy under finite-precision semantics. Theoretical Computer Science 655, 92–108 (2016)
21. Ghosh, A., Roughgarden, T., Sundararajan, M.: Universally utility-maximizing privacy mechanisms. In: STOC. pp. 351–360. ACM, New York, NY, USA (2009)
22. Ghosh, A., Roughgarden, T., Sundararajan, M.: Universally utility-maximizing privacy mechanisms. SIAM Journal of Computing 41(6), 1673–1693 (2012)
23. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Formal Aspects of Computing 6(5), 512–535 (1994)
24. Ji, Z., Lipton, Z.C., Elkan, C.: Differential privacy and machine learning: a survey and review. CoRR abs/1412.7584 (2014), `http://arxiv.org/abs/1412.7584`
25. Mironov, I.: On significance of the least significant bits for differential privacy. In: Yu, T., Danezis, G., Gligor, V.D. (eds.) ACM CCS. pp. 650–661 (2012)
26. Paz, A.: Introduction to Probabilistic Automata (Computer Science and Applied Mathematics). Academic Press, Inc., Orlando, FL, USA (1971)
27. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. No. 594 in Wiley Series in Probability and Statistics, John Wiley & Sons, Inc. (2005)
28. Rabin, M.: Probabilistic automata. Information and Control 6(3), 230–245 (1963)
29. Reed, J., Pierce, B.C.: Distance makes the types grow stronger: A calculus for differential privacy. In: ICFP. pp. 157–168. ACM (2010)

30. Tang, J., Korolova, A., Bai, X., Wang, X., Wang, X.: Privacy loss in apple's implementation of differential privacy on MacOS 10.12. CoRR abs/1709.02753 (2017), http://arxiv.org/abs/1709.02753
31. Tschantz, M.C., Kaynar, D., Datta, A.: Formal verification of differential privacy for interactive systems (extended abstract). ENTCS, vol. 276, pp. 61–79 (2011)
32. Tzeng, W.: A polynomial-time algorithm for the equivalence of probabilistic automata. SIAM Journal on Computing 21(2), 216–227 (1992)
33. Winograd-Cort, D., Haeberlen, A., Roth, A., Pierce, B.C.: A framework for adaptive differential privacy. Proceedings of the ACM on Programming Languages 1(ICFP), 10:1–10:29 (2017)
34. WWDC: Engineering privacy for your users (2016), https://developer.apple.com/videos/play/wwdc2016/709/
35. Zhang, D., Kifer, D.: LightDP: Towards automating differential privacy proofs. In: POPL. pp. 888–901. ACM (2017)

A Proof of Theorem 1

Proof. The proof follows by a reduction from the *emptiness* problem for probabilistic automata. A *probabilistic automaton* [28] is a tuple $\mathcal{A} = (S, \Sigma, M, s_0, B)$ where

- S is a finite set of states,
- Σ is the finite set of input alphabet,
- $M : S \times \Sigma \times S \to [0, 1]$ such that $\sum_{t \in S} M(s, \alpha, t) = 1$ for all $s \in S$ and $\alpha \in \Sigma$,
- $s_0 \in S$ is the initial state,
- $B \subseteq S$ is a set of accepting states.

Each input alphabet α induces a stochastic matrix $M(\alpha)$ in the obvious way. Let λ denote the empty string. For $\eta \in \Sigma^*$ we define $M(\eta)$ inductively by: $M(\lambda)$ is the identity matrix, $M(x\eta') = M(x)M(\eta')$. Thus, $M(\eta)(s, s')$ denotes the probability of going from s to s' after reading η. Let v_B denote the characteristic row vector for the set B, and v_{s_0} denote the characteristic row vector for the set $\{s_0\}$. Then, the accepting probably of η by $\mathcal{A}$ is defined as $v_{s_0} \cdot M(\eta) \cdot (v_B)^c$ where $(v_B)^c$ denotes the transpose of v_B. The following *emptiness problem* is know to be undecidable [26]:

Emptiness problem: Given a probabilistic automaton $\mathcal{A} = (S, \Sigma, M, s_0, B)$, whether there exists $\eta \in \Sigma^*$ such that $v_{s_0} \cdot M(\eta) \cdot (v_B)^c > 0$?

Now we establish the proof by reducing the emptiness problem to our dpCTL* model checking problem. Given the probabilistic automaton $\mathcal{A} = (S, \Sigma, M, s_0, B)$, assume we have a *primed* copy $\mathcal{A}' = (S', \Sigma, M', s'_0, \emptyset)$.

Let $AP := \{at_B\}$. Now we construct our MDP $M = (S \cup S', \Sigma, \wp, L)$ where $\wp(s, a, t)$ equals to $M(s, a, t)$ if $s, t \in S$ and to $M'(s, a, t)$ if $s, t \in S'$. We define the neighbor relation $N_S := \{(s_0, s'_0), (s'_0, s_0)\}$ by relating states s_0, s'_0. The labelling function L is defined by $L(s) = \{at_B\}$ if $s \in B$ and $L(s) = \emptyset$ otherwise.

Now we consider the formula $\Phi = \mathcal{D}_{1,0}(Fat_B)$. For the reduction we prove $s_0 \models \mathcal{D}_{1,0}(Fat_B)$ iff for all $\eta \in \Sigma^*$ it holds $v_{s_0} \cdot M(\eta) \cdot (v_B)^c \leq 0$.

First we assume $s_0 \models \mathcal{D}_{1,0}(Fat_B)$. By dpCTL* semantics we have that for all query scheduler $\mathfrak{Q} \in \Sigma^\omega$, $\Pr^{M_\mathfrak{Q}}_{N_S}(s_0, Fat_B) \leq e \cdot \Pr^{M_\mathfrak{Q}}_{N_S}(s'_0, Fat_B)$. Since the set of accepting state in the primed copy is empty, we have $\Pr^{M_\mathfrak{Q}}_{N_S}(s'_0, Fat_B) = 0$, thus we have $\Pr^{M_\mathfrak{Q}}_{N_S}(s_0, Fat_B) \leq 0$. This implies $v_{s_0} \cdot M(\eta) \cdot (v_B)^c \leq 0$ for all $\eta \in \Sigma^*$.

For the other direction, assume that all $\eta \in \Sigma^*$ it holds $v_{s_0} \cdot M(\eta) \cdot (v_B)^c \leq 0$. We prove by contradiction. Assume that $s_0 \not\models \mathcal{D}_{1,0}(Fat_B)$. Since the relation $N_S = \{(s_0, s'_0), (s'_0, s_0)\}$, there exists (s_0, s'_0), and a query scheduler $\mathfrak{Q} \in \Sigma^\omega$ such that

$$\Pr^{M_\mathfrak{Q}}_{N_S}(s_0, Fat_B) \not\leq e \cdot \Pr^{M_\mathfrak{Q}}_{N_S}(s'_0, Fat_B)$$

which implies $\Pr^{M_\mathfrak{Q}}_{N_S}(s_0, Fat_B) > 0$. It is then easy to construct a finite sequence $\eta \in \Sigma^*$ with $v_{s_0} \cdot M(\eta) \cdot (v_B)^c > 0$, a contradiction.