Distributed Transition Systems with Tags for Privacy Analysis

Siva Anantharaman, Sabine Frittella, Benjamin Nguyen

Distributed Transition Systems with Tags for Privacy Analysis

Siva Anantharaman[1] Sabine Frittella[2]
Benjamin Nguyen[3]
[1] LIFO, Université d'Orléans (France), email: siva@univ-orleans.fr
[2] INSA, Centre Val de Loire (France), email: sabine.frittella@insa-cvl.fr
[3] INSA, Centre Val de Loire (France), email: benjamin.nguyen@insa-cvl.fr

Abstract

We present a logical framework that formally models how a given *private* information P stored on a given database D, can get captured progressively, by an agent/adversary querying the database repeatedly. Named DLTTS (*Distributed Labeled Tagged Transition System*), the framework borrows ideas from several domains: Probabilistic Automata of Segala, Probabilistic Concurrent Systems, and Probabilistic labelled transition systems. To every node on a DLTTS is attached a *tag* that represents the 'current' knowledge of the adversary, acquired from the responses of the answering mechanism of the DBMS to his/her queries, at the nodes traversed earlier, along any given run; this knowledge is completed at the same node, with further relational deductions, possibly in combination with 'public' information from other databases given in advance. A 'blackbox' mechanism is also part of a DLTTS. It is meant as an oracle, whose role is to tell if the private information has been deduced by the adversary at the current node, and if so terminate the run; an additional special feature is that the blackbox also gives information on how 'close', or how 'far', the knowledge of the adversary is, from the private information P at the current node. A value-wise *metric* is defined for that purpose, on the set of all 'type compatible' tuples from the given database, the data themselves being typed with the headers of the base. Despite the transition systems flavor of our framework, this metric is not 'behavioral' in the sense presented in some other works. It is exclusively database oriented, and allows to define new notions of *adjacency* and of ϵ-*indistinguishabilty* between databases, more generally than those usually based on the Hamming metric with a restricted notion of adjacency. Examples are given all along to illustrate how our framework works.

Keywords:
Database, Privacy, Transition System, Probability, Distribution.

1 Introduction

Data anonymization has been investigated for decades, and many privacy models have been proposed (k-anonymity, *differential privacy*, ...) whose goals are to protect sensitive information. Our goal in this paper is not to define a new anonymization model, but rather to propose a logical framework to formally model how the information stored in a database can get captured progressively by any agent repeatedly querying the database. This model can also be used to quantify reidentification attacks on a database.

The logical framework we propose below formally models how the information stored in a database can get captured progressively, by an agent/adversary querying repeatedly the database. The data can be of any following types: numerical, non-numerical,or literal. In practice however, some of the literals representing 'sensitive data' could be in a taxonomical relation; and part of the data could be presented, for 'anonymization' purposes, as finite intervals or sets, over the basic types. We shall therefore agree to consider the types of the data in an extended 'overloaded' sense. Cf. Example 1 below. (Only tree-structured taxonomies will be considered in this work.)

We assume given a data base D, with its attributes set $\mathcal{A}$, usually divided in three disjoint groups: the subgroup $\mathcal{A}^{(i)}$ of *identifiers*, $\mathcal{A}^{(qi)}$ of *quasi-identifiers*, and $\mathcal{A}^{(s)}$ of *sensitive attributes*. The tuples of the base D will be generally denoted as t, and their attributes denoted respectively as t^i, t^{qi}, and t^s in the three subgroups of $\mathcal{A}$. The attributes t^i on any tuple t of D are conveniently viewed as defining a 'user' or a 'client' of the database D. Quasi-identifiers[1] are informally defined in general, as a set of public attributes, which in combination with other attributes and/or external information, can allow to re-identify all or some of the users to whom the information refers. The base D itself could be 'distributed probabilistically' over a finite set (referred to then, as 'universe', and its elements named as possible 'worlds').

By a *privacy policy* $P = P_A(D)$ on D with respect to a given agent/adversary A is meant the stipulation that for a certain *given set* of tuples $\{t \in P \subset D\}$, the sensitive attributes t^s on any such t shall remain inaccessible ('even after further deduction' – see below) to A. It is assumed that A is not the user identified by the attributes t^i on these t's.

The logical framework we propose in this work, to model the evolution of the 'knowledge' that an adversary A can gain by repeatedly querying the given database D – with a view to get access to sensitive data meant to remain hidden for him/her under the given privacy policy P –, will be called *Distributed Labeled-Tagged Transition System* (DLTTS); The underlying logic for DLTTS is first-order, with countably many variables and finitely many constants (including certain usual dummy symbols like '$\star, \$, \#$'). In this work, the basic signature Σ for the framework is assumed to have no non-constant function symbols. By

[1] A formal definition of quasi-identifier attributes does not seem to be known. For our purposes, it suffices to see them as those that are not identifiers nor sensitive.

'knowledge' of A we shall mean the data that A retrieves as answers to his/her successive queries, as well as other data that can be deduced/derived, under relational operations on these answers; and in addition, some others derivable from these, using relational combinations with data (possibly involving some of the users of D) from finitely many *external DBs given in advance*, denoted as $B_1, \ldots, B_m$, to which the adversary A is assumed to have free access. These relational and querying operations are all assumed done with a well-delimited fragment of the language SQL ; *it is assumed that this fragment of SQL is part of the signature* Σ *underlying the DLTTSs.* In addition, if $n \geq 1$ is the length of the tuples forming the data in D, finitely many predicate symbols $\mathcal{K}_i, 1 \leq i \leq n$, each K_i of arity i, will be part of the signature Σ; in the work presented here they will be the only predicate symbols in Σ. The role of these symbols is to allow us to see any data tuple of length $r, 1 \leq r \leq n$, as a variable-free first-order formula with top symbol $\mathcal{K}_r$, with all arguments assumed typed implicitly (with the help of the headers of the base D). In practice however, we shall drop these top symbols $\mathcal{K}_i$, and see any data tuple that is not part of the given privacy policy $P_A(D)$, directly as a first-order variable-free formula over Σ; data tuples t that are elements of the policy $P_A(D)$ will in practice be just written as $\neg t$.

As we shall see, the DLTTS framework is well suited for capturing the ideas on acquiring knowledge and on policy violation, in an elegant and abstract setup. A preliminary definition of this framework (Section 2) considers only the case where the data, as well as the answers to the queries, do not involve any notion of 'noise'. (By 'noise' we shall mean the perturbation of data by some external random (probabilistic mechanism.) But we shall extend this definition in a later section, as an option to also handle noisy data.The notion of violation of any given privacy policy on a database can then be (optionally) extended into a notion of violation up to some given $\epsilon \geq 0$ (ϵ-violation, for short). In the first part of the work, we will be modeling the lookout for the sensitive attributes of certain given users, by a single adversary. In the second part of the work (Section 5 onwards), we propose a method for *comparing the evolution of knowledge* of an adversary at two different nodes on a given run, or on two different possible runs; the same method also applies for comparing the evolution of knowledge of two different adversaries A_1, A_2, both querying repeatedly (and independently) the given database.

But before formally defining the DLTTS, a couple of examples might help; they will also throw some light on how to delimit properly the fragment of SQL that we want included in our logical setup.

1.1 A couple of Examples

Example 1. Table 1 below is the record kept by the central Hospital of a Faculty, with three Departments, in a University, on recent consultations by the faculty staff. In this record, 'Name' is an identifier attribute, 'Ailment' is sensitive, the others are QI; 'Ailment' is categorical with 3 branches: Heart-Disease, Cancer, and Viral-Infection; this latter in turn is categorical too, with

Name	Age	Gender	Dept.	Ailment
Joan	24	F	Chemistry	Heart-Disease
Michel	46	M	Chemistry	Cancer
Aline	23	F	Physics	Flu
Harry	53	M	Maths	Flu
John	46	M	Physics	CoVid

Table 1: Hospital's 'secret' record

2 branches: Flu and CoVid. By convention., such taxonomical relations are assumed known to public, (For simplicity of the example, we assume that all Faculty staff are on the consultation list of the Hospital.)

The Hospital intends to keep 'secret' information concerning CoVid infected faculty members; the tuple $\neg(John, 46, M, \#, CoVid)$ therefore constitutes its privacy policy. The following Table 2 is then published for the public, where the 'Age' attributes have been anonymized as (integer) intervals, the 'Ailment' attribute is anonymized by an upward push in the taxonomy.

A certain person A, who met John at a faculty banquet, suspected John to have been infected with CoVid; (s)he thus decides to consult the published record of the hospital for information. Knowing that the 'John' (s)he met is a

Age	Gender	Dept.	Ailment	
ℓ_1	$[20-30[$	F	Chemistry	Heart-Disease
ℓ_2	$[40-50[$	M	Chemistry	Cancer
ℓ_3	$[20-30[$	F	Physics	Viral-Infection
ℓ_4	$[50-60[$	M	Maths	Viral-Infection
ℓ_5	$[40-50[$	M	Physics	Viral-Infection

Table 2: Hospital's published record

'man' and that the table 2 must contain John's health bulletin), A has as choice lines 2, 4 and 5 (ℓ_2, ℓ_4, ℓ_5) of Table 2. A being in the lookout for a 'CoVid-infected' man, this choice is reduced to the last two tuples of the table – a priori indistinguishable because of the 'anomymization' (as 'Viral-Infection'). Now, A had the impression that the John (s)he met 'was not too old', so feels that the last tuple is twice more likely; (s)he thus 'decides that John must be from the Physics Dept.', and goes to consult the CoVid-cases statement kept publicly visible at that Dept.; which reads:

Recent CoVid-cases in the Dept: Female 0 ; Male 1.

And that confirms A's suspicion concerning John.

In this case, the DLTTS framework would function as follows: At the starting state s a transition with three branches would a priori be possible, corresponding to the three ('M') lines 2, 4 and 5 of Table 2, which represent the knowledge that would be acquired respectively along these branches. Now A is on the lookout

for a possible CoVid case, so rules out the 'line 2 branch' (i.e., gives this branch probability 0). As for the remaining two branches (corresponding to lines 4 and 5 on Table 2), A chooses to go by the line 5 branch, considering it twice more likely to be successful, than the other (A had the impression that 'John was not too old'). That leads to the probability distribution $0, 1/3, 2/3$ assigned respectively on the three possible branches for the transition. If s_0, s_1, s_2 are the respective successor states for the transition considered, the privacy policy of the Hospital (concerning John's CoVid information) would thus be violated at state s_2 (with probability 2/3), it wouldn't be at s_1 (probability 1/3); no information deduced at state s_0.

As just seen, modeling an adversary's search for some specific information on a given data base D – as 'runs' on a suitable DLTTS and probability distributions over the successor steps along the runs –, depends in general on the nature(structure) of the information looked for. The probability distributions on the transitions along the runs would generally depend on some random mechanism, which could also reflect the choices the adversary might make. □

The role of our next example is to point out that specifying Privacy policy policies will in general have some serious side effects on the functioning of the primitives and aggregate procedures of SQL. If the policies are to have some 'content', operationally speaking, the DBMS may have to stipulate that the queries employing these primitives either should have 'void outputs' in certain contexts, or 'get filtered by the Privacy policy'.

Example 2. Table 3 below is an imaginary record D of a bank $\mathcal{L}$, containing a list of its clients: with client_ids, their names, and their monthly balances. (Client_id is the identifier attribute, Monthly–balance is sensitive.) The privacy policy P of the bank is that client *Jean*'s Monthly–balance should 'be invisible' to others; formally, the policy P is the negated formula $\neg(Jean, \geq 420)$.

On the other hand, the bank is obliged administratively to render public a monthly statement, on its minimum total Monthly–balance; that is Table 4.

Client_id	Name	Monthly-balance
1	Claude	320
2	Paul	270
3	Jean	420
4	Martin	150
5	Michel	420

Table 3: $\mathcal{L}$'s (secret) client record

Number of Clients	Minimum Total Monthly–balance
5	≥ 1580

Table 4: Bank $\mathcal{L}$'s Monthly public statement

An adversary A wants to know if Jean is a client of the bank, and if so, with

a monthly balance among the highest. So A first queries the bank to get the list of its clients with their Monthly-balances. The Bank-DBMS's answer to A's query will be, say, as in Table 5 below, where $\star$ stands for the anonymization of Jean's sensitive data, as a 'mask' or as an interval, say of the form $[330 - 450[$.

Name	Monthly-balance
Claude	320
Jean	$\star$
Paul	270
Michel	420
Martin	150

Table 5: DBMS's Answer to A's query

The external Table 4 is freely accessible to A; so, if the functionalities `COUNT` and `SUM` are applied 'without any filter', A can easily deduce that *Jean*'s Monthly–balance at $\mathcal{L}$ is ≥ 420; the bank's Privacy policy is thus violated. □

Remark 1: In the above example, if the external DB (Table 4) was unavailable to A, the DBMS could have answered his/her query with a Table 5' where the entire tuple on Jean is deleted; in such a case, the privacy policy P on D (concerning Jean) would a priori remain unviolated; *except* if we assume that the DBMS accepts queries with aggregate operations on the database D that 'do not explicitly look' for Jean's sensitive attribute: For instance A could first retrieve the SUM on the entire Monthly–balance column, then ask for SUM(Monthly–balance) where 'Name <> Jean'. A relational deduction then leads to the violation of the policy P. The above two Examples show that the violation of privacy policies needs, in general, some additional 'outside knowledge'. □

We may assume wlog that the given external bases $B_1, \ldots, B_m$ – to which A could resort, with relational operations for deducing additional information – are also of the same signature Σ as D; so all the knowledge A can deduce/derive from his/her repeated queries can be expressed as a first-order variable-free formula over the signature Σ.

2 Distributed Labeled-Tagged Transition Systems

The DLTTS framework presented in this section synthesizes ideas coming from various domains, such as the Probabilistic Automata of Segala ([13], Probabilistic Concurrent Systems, Probabilistic labelled transition systems ([3, 4]. Although the underlying signature for the DLTTS can be rich in general, for the purposes of our current work we shall be working with a limited first-order signature (as mentioned in the Introduction) denoted Σ, with countably many variables, finitely many constants (including some 'standard dummies'), no non-constant function symbols, and a finite limited set of predicate (propositional) symbols. Let $\mathcal{E}$ be the set of all variable-free formulas over Σ, and `Ext` a given subset of $\mathcal{E}$. We assume given a decidable procedure $\mathcal{C}$ whose role is to 'satu-

rate' any finite set G of variable-free formulas into a finite set $\overline{G}$, by adding a finite (possibly empty) set of variable-free formulas, using *relational operations* on G and `Ext`. This procedure $\mathcal{C}$ will be internal at every node on a DLTTS; in addition, *there will also be a 'blackbox' mechanism $\mathcal{O}$, acting as an oracle* telling if the given privacy policy on a given database is violated at the current node. More details will be given in Section 5 on the additional role the oracle will play in a privacy analysis procedure (for any querying sequence on a given DB), based on a novel data-based metric, which will be defined in that section.

Definition 1 *A Distributed Labeled-Tagged Transition System (DLTTS), over a given signature Σ, is formed of:*

- *a finite (or denumerable) set S of states, an 'initial' state $s_0 \in S$, and a special state $\otimes \in S$ named 'fail':*
- *a finite set Act of action symbols (disjoint from Σ), with a special action $\delta \in Act$ called 'violation';*
- *a (probabilistic) transition relation $\mathcal{T} \subset S \times Act \times Distr(S)$, where $Distr(S)$ is the set of all probability distributions over S, with finite support.*
- *a tag $\tau(s)$ attached to every state $s \in S \setminus \{\otimes\}$, formed of finitely many first-order variable-free formulas over Σ; the tag $\tau(s_0)$ at the initial state is the singleton set $\{\top\}$.*
- *at every state s a special action symbol $\iota = \iota_s \in Act$, said to be* internal *at s, completes/saturates $\tau(s)$ into a set $\overline{\tau}(s)$ with the procedure $\mathcal{C}$, by using relational operations between the formulas in $\tau(s)$ and* `Ext`.

A (probabilistic) transition $\mathfrak{t} \in \mathcal{T}$ will generally be written as a triple $(s, a, \mathfrak{t}(s))$; and $\mathfrak{t}$ will be said to be 'from' (or 'at') the state s, the states of $\mathfrak{t}(s)$ will be the 'successors' of s under $\mathfrak{t}$. The formulas in the tag $\overline{\tau}(s)$ attached to any state s will all be assigned the same probability as the state s in $Distr(S)$. If the set $\overline{\tau}(s)$ of formulas turns out to be inconsistent, then the oracle mechanism $\mathcal{O}$ will (intervene and) impose $(s, \delta, \otimes)$ as the only transition from s, standing for 'violation' and 'fail', by definition,

Nondeterminism of transitions can be defined without difficulty on DLTTS, as a nondeterministic choice between the possible probabilistic transitions at any given state. We shall assume that nondeterminism is managed by the choice of a suitable scheduler; and in addition, that at most one probabilistic transition is firable from any state $s \in S \setminus \{\otimes\}$, and none from the halting state $\otimes$.

DLTTS and Repeated queries on a database: The states of the DLTTS will stand for the various 'moments' of the querying sequence, while the tags attached to the states will stand for the knowledge A has acquired on the data of D 'thus far'. This knowledge consists partly in the answers to the queries (s)he made so far, then completed with additional knowledge using the internal

'saturation' procedure $\mathcal{C}$ of the framework. In the context of DBs, this procedure would consist in relational algebraic operations between the answers retrieved by A for his/her repeated queries on D, all seen as tuples (variable-free formulas), and suitable tuples from the given external databases $B_1, \ldots, B_m$. If the saturated knowledge of A at a current state s on the DLTTS (i.e., the tag $\overline{\tau}(s)$ attached to the current state s) is not inconsistent, then the transition from s to its successor states represents the probability distribution of the likely answers A would expect to get for his/her next query.

Note that we make no assumption on whether the repeated queries by A on D are treated *interactively, or non-interactively*, by the DBMS. It appears that the logical framework would function exactly alike, in both cases.

Remark 2: (a) Suppose $\mathfrak{t}$ is a transition from a state s, on the DLTTS corresponding to a querying sequence by an adversary A, and s' is one of the successors of s under $\mathfrak{t}$; then, by definition, the 'fresh' knowledge $\tau(s')$ of A at s' resulting from this transition, is the addition to A's saturated knowledge $\overline{\tau}(s)$ at s, the part of the response of the DBMS's answering mechanism for A's current query, represented by the branch of $\mathfrak{t}$ going from s to s'.

(b) As already mentioned, we assume that the relational operations needed for gaining further knowledge are done using a well delimited finite subset of the functionalities of SQL; and that 'no infinite set can get generated from a finite set' under these functionalities, assumed included in the signature Σ. (This corresponds to the *bounded inputs outputs* assumption, as in e.g., [1, 2].) □

Proposition 1 *Suppose given a database D, a finite sequence of repeated queries on D by an adversary A, and a first-order relational formula $P = P_A(D)$ over the signature Σ of D, expressing the privacy policy of D with respect to A. Let $\mathcal{W}$ be the DLTTS modeling the various queries of A on D, and the evolution of the knowledge of A on the data of D, resulting from these queries and the internal actions at the states of $\mathcal{W}$, as described above.*

(i) The given privacy policy $P_A(D)$ on D is violated if and only if the failure state $\otimes$ on the DLTTS $\mathcal{W}$ is reachable from the initial state of $\mathcal{W}$.

(ii) The satisfiabiliity of the set of formulas $\overline{\tau}(s) \cup \{\neg P\}$ is decidable, at any state s on the DLTTS, under the assumptions of Remark 2(b).

Proof: Assertion (i) is restatement. Observe now, that at any state s on $\mathcal{W}$, the tags $\tau(s)$, $\overline{\tau}(s)$ are both *finite sets of first-order variable-free formulas* over Σ, without non-constant function symbols. For, to start with, the knowledge of A consists of the responses received for his/her queries, in the form of a finite set of data tuples from the given databases, and some subtuples. And by our assumptions of Remark-2 (b), no infinite set can be generated by saturating this initial knowledge with procedure $\mathcal{C}$. Assertion (ii) follows then from the known result that the inconsistency of any given finite set of variable-free first-order Datalog formulas is decidable, e.g., by the analytic tableaux procedure. (Only the absence of variables is essential.) □

3 ϵ-indistinguishability, ϵ-local-differential privacy

Our objective now is to extend the result of Proposition 1 to the case when the violation to be considered can be *up to some given* $\epsilon \geq 0$, in a sense to be made precise. We stick to the same notation as above. The set $\mathcal{E}$ of all variable-free formulas over Σ is thus a disjoint union of subsets of the form $\mathcal{E} = \cup\{\mathcal{E}_i^{\mathcal{K}} \mid 0 < i \leq n, \mathcal{K} \in \Sigma\}$, the index i in $\mathcal{E}_i^{\mathcal{K}}$ standing for the common length of the formulas in the subset, and $\mathcal{K}$ for the common root symbol of its formulas; each set $\mathcal{E}_i^{\mathcal{K}}$ will be seen as a database of i-tuples.

We shall first look at the situation where the queries intend to capture certain (sensitive) values on a given tuple t in the database D. Two different tuples in $\mathcal{E}$ might correspond to two likely answers to such a query, but with possibly different probabilities in the distribution assigned for the transitions, by the probabilistic mechanism $\mathcal{M}$ (e.g., as in Example 1).

Given two such instances, and a real $\epsilon \geq 0$, we can also define a notion of their ϵ-local-indistinguishabilty, wrt the tuple t and the mechanism $\mathcal{M}$ answering the queries. This can be done in a slightly extended setup, where the answering mechanism may, *as an option*, also add 'noise' to certain numerical data values, for several reasons among which the safety of data. We shall then assume that the internal procedure $\mathcal{C}$ of the DLTTS at each of its states (meant to saturate the current knowledge of the adversary querying the database) incorporates the following three well-known noise adding mechanisms: the Laplace, Gauss, and exponential mechanisms. With the stipulation that this optional noise additions to numerical values can be done in a *bounded* fashion, so as to be from a finite prescribed domain around the values; it will then be assumed that tuples formed of such noisy data are also in $\mathcal{E}$.

Definition 2 *(i) Suppose that, while answering a given query on the base D, at two instances v, v', the probabilistic answering mechanism $\mathcal{M}$ outputs the same tuple $\alpha \in \mathcal{E}$. Given $\epsilon \geq 0$, these two instances are said to be ϵ-local-indistinguishable wrt α, if and only if:*

$$Prob[\mathcal{M}(v) = \alpha] \leq e^{\epsilon} Prob[\mathcal{M}(v') = \alpha] \text{ and}$$
$$Prob[\mathcal{M}(v') = \alpha] \leq e^{\epsilon} Prob[\mathcal{M}(v) = \alpha].$$

(ii) The probabilistic answering mechanism $\mathcal{M}$ is said to satisfy ϵ-local differential privacy (ϵ-LDP) *for $\epsilon \geq 0$, if and only if: For any two instances v, v' of $\mathcal{M}$* that lead to the same output*, and any set $\mathcal{S} \subset Range(\mathcal{M})$, we have*

$$Prob[\mathcal{M}(v) \in \mathcal{S}] \leq e^{\epsilon} Prob[\mathcal{M}(v') \in \mathcal{S}].$$

We shall also be needing the following notion of ϵ-indistinguishability (and of ϵ-distinguishability) of two different outputs of the mechanism $\mathcal{M}$: These definitions – as well that of ϵ-DP given below – are essentially reformulations of the same (or similar) notions defined in [7, 8].

Definition 3 *Given $\epsilon \geq 0$, two outputs α, α' of the probabilistic mechanism $\mathcal{M}$ answering the queries of an agent A, are said to be ϵ-indistinguishable, if and*

only if: For every pair v, v' of inputs for $\mathcal{M}$, such that $Prob[\mathcal{M}(v) = \alpha] = p$ and $Prob[\mathcal{M}(v') = \alpha'] = p'$, we must have: $p \leq e^{\epsilon}p'$ and $p' \leq e^{\epsilon}p$.

Otherwise, the outputs α, α' will be said to be ϵ-distinguishable.

Remark 3: Given an $\epsilon \geq 0$, one may assume as an option, that at every state on the DLTTS *the retrieval of answers to the current query (from the mechanism $\mathcal{M}$) is done up to ϵ-indistinguishabilty*; this will then be implicitly part of what was called the saturation procedure $\mathcal{C}$ at that state. The procedure thus enhanced for saturating the tags at the states, will then be denoted as $\epsilon\mathcal{C}$, when necessary (it will still be decidable, under the finiteness asumptions of Remark-2 (b)). Inconsistency of the set of formulas, in the '$\epsilon\mathcal{C}$-saturated' tag at any state, will be checked up to ϵ-indistinguishabilty, and referred to as ϵ-inconsistency, or ϵ-failure. The notion of privacy policy will not need to be modified; that of its violation will be referred to as ϵ-violation, Under these optional extensions of ϵ-failure and ϵ-violation, it must be clear that the statements of Proposition 1 continue to be valid. □

Two small examples of ϵ-local-indistinguishability, before closing this section.

(i) The two sub-tuples ([50–60], M, Maths) and ([40–50], M, Physics), from the last two tuples on the Hospital's published record in Example 1 (Table 2), both point to Viral–Infection as output; they can thus be seen as $log(2)$-local-indististinguishable, for the adversary A.

(ii) The 'Randomized Response' mechanism RR ([14]) can be modelled as follows. Input is (X, F_1, F_2) where X is a Boolean, and F_1, F_2 are flips of a coin (H or T). RR outputs X if $F_1 = H$, $True$ if $F_1 = T$ and $F_2 = H$, and $False$ if $F_1 = T$ and $F_2 = T$. This mechanism is $log(3)$-LDP : the instances $(True, H, H)$, $(True, H, T)$, $(True, T, H)$ and $(True, T, T)$ are $log(3)$-indistinguishable for output $True$. $(False, H, H)$, $(False, H, T)$, $(False, T, H)$and $(False, T, T)$ are $log(3)$-indistinguishable for output $False$.

4 ϵ-Differential Privacy

The notion of *ϵ-indistinguishability of two given databases D, D'* for an adversary, is more general than that of ϵ-local-indistinguishability (of pairs of instances of a probabilistic answering mechanism giving the same output, defined in the previous section). ϵ-indistinguishability is usually defined only for pairs of databases D, D' that are *adjacent* in a certain sense (cf. below).

There is no uniquely defined notion of adjacence on pairs of databases; in fact, several are known, and in use in the literature. Actually, a notion of adjacence can be defined in a generic parametrizable manner (as in e.g., [5]), as follows. We assume given a map $\mathbf{f}$ from the set $\mathcal{D}$ of all databases of m-tuples (for some given $m > 0$), into some given metric space (X, d_X). The binary relation on pairs of databases in $\mathcal{D}$, defined by $\mathbf{f}_{adj}(D, D') = d_X(\mathbf{f}(D), \mathbf{f}(D'))$ is then said to define a measure of *adjacence* on these databases. The relation $\mathbf{f}_{adj}$ is said to define an 'adjacency relation'.

Definition 4 *Let* $\mathbf{f}_{adj}$ *be a given adjacency relation on a set* $\mathcal{D}$ *of databases, and* $\mathcal{M}$ *a probabilistic mechanism answering queries on the databases in* $\mathcal{D}$*.*

- Two databases $D, D' \in \mathcal{D}$ *are said to be* $\mathbf{f}_{adj}$*-indistinguishable under* $\mathcal{M}$*, if and only if, for any possible output* $\mathcal{S} \subset Range(\mathcal{M})$*, we have*

$$Prob[\mathcal{M}(D) \in \mathcal{S}] \leq e^{\mathbf{f}_{adj}(D,D')} Prob[\mathcal{M}(D') \in \mathcal{S}].$$

- The mechanism $\mathcal{M}$ *is said to satisfy* $\mathbf{f}_{adj}$-differential privacy *(*$\mathbf{f}_{adj}$*-DP), if and only if the above condition is satisfied for* every pair of databases D, D' *in* $\mathcal{D}$*, and any possible output* $\mathcal{S} \subset Range(\mathcal{M})$*.*

Comments: (i) Given $\epsilon \geq 0$, the 'usual' notions of ϵ*-indistinguishability and* ϵ*-DP* correspond to the choice of adjacency $\mathbf{f}_{adj} = \epsilon d_h$, where d_h is the Hamming metric on databases – namely, the number of 'records' where D and D' differ, plus the assumption $d_h(D, D') \leq 1$ (cf. [5]).

(ii) In Section 6, we propose a more general notion of adjacency, based on a different metric defined 'value-wise', to serve other purposes as well.

(iii) On disjoint databases, one can work with different adjacency relations, using different maps to the same (or different) metric space(s),

(iv) The mechanism RR described above is actually $log(3)$-DP, not only $log(3)$-LDP. To check DP, we have to check all possible pairs of numbers of the form $(Prob[\mathcal{M}(x) = y], Prob[\mathcal{M}(x') = y])$, $(Prob[\mathcal{M}(x) = y'], Prob[\mathcal{M}(x') = y])$, $(Prob[\mathcal{M}(x) = y], Prob[\mathcal{M}(x') = y'])$, etc., where the x, x'.... are the input instances for RR, and $y, y', ...$ the outputs. The mechanism RR has 2^3 possible input instances for (X, F_1, F_2) and two outputs (*True, False*); thus 16 pairs of numbers, the distinct ones being $(1/4, 1/4), (1/4, 3/4), (3/4, 1/4), (3/4, 3/4)$; if (a, b) is any such pair, obviously $a \leq e^{log(3)} b$. Thus RR is indeed $log(3)$-DP. □

5 Comparing Two Nodes on one or more Runs

In the previous two sections, we looked at the issue of 'quantifying' the indistinguishability of two data tuples or databases, under repeated queries of an adversary A. In this section, our concern will be in a sense 'orthogonal': the issue will be that of quantifying how different the probabilistic mechanism's answers can be, at different moments of A's querying sequence. Remember that the knowledge of A, at any node on the DLTTS of the run corresponding to the query sequence, is represented as a set of tuples; and also that the data forming any tuple are assumed implicitly typed, 'labeled with' (i.e., under) the headers of the database D. To be able to compare two tuples of the same length, we shall assume that there is a natural, injective, *type-preserving* map from one of them onto the other; this map will remain implicit in general; two such tuples will be said to be *type-compatible*. If the two tuples are not of the same length, one of them will be projected onto (or restricted to) a suitable subtuple, so as to be type-compatible and comparable with the other; if this turns out to be impossible, the two tuples will be said to be uncomparable.

The quantification looked for will be based on a suitable notion of 'distance' between two *sets of type-compatible tuples*. For that, we shall first define 'distance' between any two type-compatible tuples; more precisely, define such a notion of distance between any two data values under every given header of D. As a first step, we shall therefore begin by defining, for every given header of D, a binary 'distance' function on the set of all values that get assigned to the *attributes under that header*, along the sequence of A's queries. This distance function to be defined will be a *metric*: non-negative, symmetric, and satisfying the so-called Triangle Inequality (cf. below). The '*direct-sum*' of these metrics, taken over all the headers of D, will then define a metric d on the set of all type-compatible tuples of data assigned to the various attributes, under all the headers of D, along the sequence of A's queries. The 'distance' $d(t, t')$, from any given tuple t in this set to another type-compatible tuple t', will be defined as the value of this direct-sum metric on the pair of tuples (t, t'); it will, by definition, be calculated 'column-wise' on the base D, and also on the intermediary databases along A's query sequence; note that it will give us a priori an m-tuple of numbers, where m is the number of headers (or columns) in the database D.

A single number can then be derived as the sum of the entries in the m-tuple $d(t, t')$. This sum will be denoted as $\overline{d}(t, t')$, and defined as the distance from the tuple t to the tuple t' in the database D. Finally, if S, S' are any two given finite sets of type-compatible tuples, of data that get assigned to the various attributes (along the queries), we shall define the distance from the set S to the set S' as the number $\rho(S, S') = min\{\, \overline{d}(t, t') \mid t \in S,\, t' \in S' \,\}$

Some preliminaries are needed before we can define the 'distance' function between the data values under every given header of D. We begin by dividing the headers of the base D into four classes classes, for clarity of presentation:

. 'Nominal': identities, names, attributes receiving literal data *not in any taxonomy* (e.g., gender, city, ...), finite sets of such data;

. 'Numerval' : attributes receiving numerical values, or bounded intervals of (finitely many) numerical values;

. 'Numerical': attributes receiving single numerical values (numbers).

. 'Taxoral': attributes receiving literal data in a taxonomy relation.

For defining the 'distance' between any two values v, v' assigned to an attribute under a given 'Nominal' header of D, for the sake of uniformity we agree to consider every value as a *finite set* of singleton values. (In particular, a singleton value 'x' will be seen as the set $\{x\}$.) Given two such values v, v', note first that the so-called *Jaccard Index* between them is the number $jacc(v, v') = |(v \cap v')/(v \cup v')|$, which is a 'measure of their similarity'; but this index is not a metric: the *triangle inequality* is not satisfied; however, the Jaccard metric $d_{Nom}(v, v') = 1 - jacc(v, v') = |(v \Delta v')/(v \cup v')|$ does satisfy that property, and will suit our purposes. Thus defined, $d_{Nom}(v, v')$ is a 'measure of the dissimilarity' between the sets v and v'.

Let $\mathcal{T}_{Nom}$ be the set of all data assigned to the attributes under the 'Nominal' headers of D, along the sequence of A's queries. Then the above defined binary function d_{Nom} extends to a metric on the set of all type-compatible data-tuples from $\mathcal{T}_{Nom}$, defined as the 'direct-sum' taken over the 'Nominal' headers of D.

If $\mathcal{T}_{Num}$ is the set of all data assigned to the attributes under the 'Numerval' headers along the sequence of queries by A, we also define a 'distance' metric d_{Num} on the set of all type-compatible data-tuples from $\mathcal{T}_{Num}$, in a similar manner. We first define d_{Num} on any couple of values u, v assigned to the attributes under a given 'Numerval' header of D, then extend it to the set of all type-compatible data-tuples from $\mathcal{T}_{Num}$ (as the direct-sum taken over the 'Numerval' headers of D). This will be done exactly as under the 'Nominal' headers: suffices to visualize any finite interval value as a particular way of presenting a set of numerical values (integers, usually). (In particular, a single value 'a' under a 'Numerval' header will be seen as the interval value $[a]$.) Thus defined the (Jaccard) metric distance $d_{Nom}([a, b], [c, d])$ is a measure of 'dissimilarity' between $[a, b]$ and $[c, d]$. .

Between numerical data x, x' under the 'Numerical' headers, the distance we shall work with is the euclidean metric $|x - x'|$, *normalized as*: $d_{eucl}(x, x') = |x - x'|/D$, where $D > 0$ is a fixed finite number, bigger than the maximal euclidean distance between the numerical data on the databases and on the answers to A's queries.

On the data under the 'Taxoral' headers, we choose as distance function the metric d_{wp}, defined in Lemma 1 (cf. *Appendix*) between the nodes of any Taxonomy tree.

Note that the 'datawise distance functions' defined above are *all with values in the real interval* $[0, 1]$. (This is also one reason for our choice of the distance metric on Taxonomy trees.) This fact is of importance, for comparing the metric ρ we defined above with the Hamming metric, cf. Section 6.

An additonal role for Oracle $\mathcal{O}$: In Section 5.1 below, we present a procedure for comparing the knowledge of an adversary A at different nodes of the DLTTS that models the 'distributed sequence' of A's queries on a given database D. The comparison can be with respect to any given 'target' dataset T (e.g., a privacy policy P on D). In operational terms, so to say, the oracle mechanism $\mathcal{O}$ of the DLTTS keeps the target dataset 'in store'; and as said earlier, a first role for the oracle $\mathcal{O}$ of the DLTTS is to keep a watch on the deduction of the target dataset by the adversary A at some node. The additional second role that we assign now to the oracle $\mathcal{O}$, is to publish information on the distance of A's saturated knowledge $\overline{\tau}(s)$, at any given node s, to the target dataset T. This distance is calculated wrt the distance ρ, defined above as the minimal distance $\overline{d}(t, t')$ between the tuples $t \in \overline{\tau}(s), t' \in T$, where $\overline{d}$ is the direct sum of the 'column-wise distances' between the data on the tuples.

Before presenting the comparison schema, here is an example to illustrate

how the notions developed above operate in practice.

Example 1 bis. We go back to the Hospital-CoVid example seen earlier, more particularly its Table 2, reproduced here:

Age	Gender	Dept.	Ailment	
ℓ_1	$[20-30[$	F	Chemistry	Heart-Disease
ℓ_2	$[40-50[$	M	Chemistry	Cancer
ℓ_3	$[20-30[$	F	Physics	Viral-Infection
ℓ_4	$[50-60[$	M	Maths	Viral-Infection
ℓ_5	$[40-50[$	M	Physics	Viral-Infection

Table 6: Hospital's public record recalled

'Gender' and 'Dept.'. are the 'Nominal' headers in this record, 'Age' is 'Numerval' and 'Ailment' is 'Taxoral'. We are interested in the second, fourth and fifth tuples on the record, respectively referred to as l_2, l_4, l_5. The 'target set' of (type-compatible) tuple in this example is taken as the (negation of the) privacy policy specified, namely the tuple $T = (John, 46, M, \#, CoVid)$.

We compute now the distance $\overline{d}$ between the target T, and the three tuples l_2, l_4, l_5. This involves only the subtuple $L = (46, M, \#, CoVid)$ of T:

. $\overline{d}(l_2, L) = d_{Num}(l_2, L) + d_{Nom}(l_2, L) + d_{wp}(L_2, L)$
$= (1 - 1/10) + 0 + (1 - 2/5) = 9/10 + 3/5 = 15/10$

. $\overline{d}(l_4, L) = d_{Num}(l_2, L) + d_{Nom}(l_4, L) + d_{wp}(L_4, L)$
$= (1 - 0) + 0 + (1 - 4/5) = 1 + 1/5 = 6/5$

. $\overline{d}(l_5, L) = d_{Num}(l_5, L) + d_{Nom}(l_5, L) + d_{wp}(L_5, L)$
$= (1 - 1/10) + 0 + (1 - 4/5) = 9/10 + 1/5 = 11/10$

The tuple l_2 is the farthest from the target, while l_5 is the closest. This 'explains' that the adversary can choose the branch on the transition that leads to a state where l_5 is added to his/her knowledge. This is more formally detailed in the procedure presented below. □

5.1 A (Non-Deterministic) Comparison Procedure

· Given: DLTTS associated with a querying sequence, by adversary A on given database D; and *a Target set* of tuples T.

· Given: Two states s, s' on the DLTTS, with respective *saturated* tags l, l', and probabilties p, p'. Target T assumed not in l or l': neither $\rho(l, T)$ nor $\rho(l', T)$ is 0. Also given:

- $config_1$: successor states $s_1, \ldots, s_n$ for a transition $\mathfrak{t}$ from s, with probability distribution $p_1, \ldots, p_n$; and respective tags $l_1, \ldots, l_n$, with the contribution from $\mathfrak{t}$ (cf. Remark 2(a)).

- $config_2$: successor states $s'_1, \ldots, s'_m$ for a transition $\mathfrak{t}'$ from s', with probability distribution $p'_1, \ldots, p'_m$; and respective tags $l'_1, \ldots, l'_m$, with the contribution from $\mathfrak{t}'$ (cf. Remark 2(a)).

· Objective: *Choose states to compare under s, s' (with probability measures not lower than p, p') in $config_1$, or in $config_2$, or from either.*

(i) Compute $d_i = \rho(l_i, T), i \in 1 \cdots n$, and $d'_j = \rho(l'_j, T), j \in 1 \cdots m$.

$$d_{min}(\mathfrak{t}, T) = min\{d_i \mid i \in 1 \cdots n\}, \quad d'_{min}(\mathfrak{t}', T) = min\{d'_j \mid j \in 1 \cdots m\}$$

(ii) Check IF *the following conditions are satisfied* by $config_1$:

$$d_{min}(\mathfrak{t}, T) \leq d'_{min}(\mathfrak{t}', T)$$

$$\exists \text{ an } i, 1 \leq i \leq n, \text{ such that } d_i = d_{min}(\mathfrak{t}, T),\ p_i \leq p,$$

$$\text{and} \quad p_i \geq p'_j \text{ for any } j,\ 1 \leq j \leq m, \text{ where } d'_j = d'_{min}(\mathfrak{t}', T)$$

(iii) IF YES, continue under s with $config_1$, else RETURN.

6 New Metric for Indistinguishability and DP

Given a randomized/probabilistic mechanism $\mathcal{M}$ answering the queries on databases, and an $\epsilon \geq 0$, recall that the ϵ-indistinguishability of any two given databases under $\mathcal{M}$, and the notion of ϵ-DP for $\mathcal{M}$, were both defined in Definition 4 (Section 4), based first on a hypothetical map $\mathbf{f}$ from the set of all the databases concerned, into some given metric space (X, d_X), and an 'adjacency relation' on databases defined as $\mathbf{f}_{adj}(D, D') = d_X(\mathbf{f}D, \mathbf{f}D')$, which was subsequently instantiated to $\mathbf{f}_{adj} = \epsilon d_h$, where d_h is the Hamming metric between databases. It must be observed here, that *the Hamming metric is defined only between databases with the same number of columns*, and usually only with all data of the same type.

In this subsection, our objective is to propose a more general notion of adjacency, based on the distance metric ρ defined above, between type-compatible tuples on databases with data of multiple types. In other words, our $\mathcal{D}$ here will be the set of all databases, *not necessarily all with the same number of columns, and with data of several possible types* as mentioned in the Introduction. We define then a binary relation $\mathbf{f}^{\rho}_{adj}(D, D')$ between D, D' in the set $\mathcal{D}$ by setting $\mathbf{f}^{\rho}_{adj}(D, D') = \rho(D, D')$, visualizing D, D' as sets of type-compatible data tuples.

Given ϵ, we can then define the notion of ϵ_ρ-indistinguishabilty of two databases D, D' under a (probabilistic) answering mechanism $\mathcal{M}$, as well as the notion of ϵ_ρ-DP for $\mathcal{M}$, exactly as in Definition 4, by replacing $\mathbf{f}_{adj}$ first with the relation $\mathbf{f}^{\rho}_{adj}$, and subsequently with $\epsilon\rho$. The notions thus defined are *more general* than those presented earlier in Section 4 with the choice $\mathbf{f}_{adj} = \epsilon d_h$. An example will illustrate this point.

Example 4. We go back to the 'Hospital's public record' of our previous example, with the same notation. For this example, we shall assume that the

mechanism $\mathcal{M}$ answering a query for 'ailment information involving men' on that record, returns the tuples l_2, l_4, l_5 with the probability distribution $0, 2/5, 3/5$, respectively. Let us look for the minimum value of $\epsilon \geq 0$, for which these three tuples will be ϵ_ρ-indistinguishable under the mechanism $\mathcal{M}$.

The output l_2, with probability 0, will be ϵ_ρ-distinguishable for any $\epsilon \geq 0$. Only the two other outputs l_4, l_5 need to be considered. We first compute the ρ-distances between these two tuples: $\overline{d}(l_4, l_5) = (1 - \frac{1}{20}) + 0 + 1 + 0 = 39/20$. The condition for l_4 and l_5 to be ϵ_ρ-indistinguishable under $\mathcal{M}$ is thus:

$$(2/5) \leq e^{(39/20)\epsilon} * (3/5) \;\; and \;\; (3/5) \leq e^{(39/20)\epsilon} * (2/5),$$

i.e., $\epsilon \geq (20/39) * ln(3/2)$. In other words, for any $\epsilon \geq (20/39) * ln(3/2)$, the two tuples l_4 and l_5 will be ϵ_ρ-indistinguishable; and for values of ϵ with $0 \leq \epsilon < (20/39) * ln(3/2)$, these tuples will be ϵ_ρ-distinguishable.

For the ϵ-indistinguishabilty of these tuples wrt the Hamming metric d_h, we proceed similarly: the distance $d_h(l_4, l_5)$ is by definition the number of 'records' where these tuples differ, so $d_h(l_4, l_5) = 2$. So the condition on $\epsilon \geq 0$ for their ϵ-indistinguishabilty wrt d_h is: $(3/5) \leq e^{2\epsilon} * (2/5)$, i.e., $\epsilon \geq (1/2) * ln(3/2)$.

In other words, if these two tuples are ϵ_ρ-indistinguishables wrt ρ under $\mathcal{M}$ for some ϵ, then they will be ϵ-indistinguishable wrt d_h for the same ϵ. But the converse is not true, since $(1/2) * ln(3/2) < (20/39) * ln(3/2)$. Said otherwise: *$\mathcal{M}$ ϵ-distinguishes more finely with ρ, than with d_h.* □

Remark 4: The statement¡'$\mathcal{M}$ ϵ-distinguishes more finely with ρ, than with d_h", is *always true* (not just in Example 4). For the following reasons: The records that differ 'at some given position' on two bases D, D' are always at distance 1 for the Hamming metric d_h, by definition, whatever be the type of data stored at that position. Now, if the data stored at that position 'happened to be' numerical, the usual euclidean distance between the two data could have been (much) bigger than their Hamming distance 1; precisely to avoid such a situation, our definition of the metric d_{eucl} on numerical data 'normalized' the euclidean distance, to ensure that their d_{eucl}-distance will not exceed their Hamming distance. Thus, all the 'record-wise' metrics we have defined above have their values in $[0, 1]$, as we mentioned earlier; so, whatever the type of data at corresponding positions on any two bases D, D', the ρ-distance between the records will never exceed their Hamming distance. That suffices to prove our statement above. The Proposition below formulates all this, more precisely:

Proposition 2 *Let $\mathcal{D}_m$ be the set of all databases with the same number m of columns, over a finite set of given data, and $\mathcal{M}$ a probabilistic mechanism answering queries on the bases in $\mathcal{D}$. Let ρ be the metric (defined above) and d_h the Hamming metric, between the databases in $\mathcal{D}$, and suppose given an $\epsilon \geq 0$.*

- If two databases $D, D' \in \mathcal{D}_m$ are ϵ_ρ-indistinguishable under $\mathcal{M}$ wrt ρ, then they are also ϵ-indistinguishable under $\mathcal{M}$ wrt d_h.

- If the mechanism $\mathcal{M}$ is ϵ_ρ-DP on the bases in $\mathcal{D}_m$ (wrt ρ), then it is also ϵ-DP (wrt d_h) on these bases.

The idea of 'normalizing' the Hamming metric between numerical databases (with the same number of columns) was already suggested in [5] for the same reasons. When only numerical databases are considered, the metric ρ that we have defined above is the same as the 'normalized Hamming metric' of [5]. Our metric ρ must actually be seen as a generalization of that notion, to directly handle bases with more general types of data: anonymized, taxonomies, ...

7 Related Work and Conclusion

A starting point for the work presented is the observation that databases could be distributed over several 'worlds' in general, so querying such bases leads to answers which would also be distributed; to such distributed answers one could conceivably assign probability distributions of relevance to the query. The probabilistic automata of Segala ([12, 13]) are among the first logical structures proposed to model such a vision, in particular with outputs. Distributed Transition Systems (DTS) appeared a little later, with as objective the behavioral analysis of the distributed transitions, based on traces or on simulation/bisimulation, using quasi- or pseudo- or hemi- metrics as in [3, 4, 6]. Our lookout in this work was for a syntax-based *metric in the mathematical sense,* that can directly handle data of 'mixed' types – which can be numbers or literals, but can also be 'anonymized' as intervals or sets; they can also be taxonomically related to each other in a tree structure. (The metric d_{wp} we have defined in the *Appendix* on the nodes of a taxonomy tree is novel.) Data-wise metrics as defined in our work can express more precisely, in a mathematical sense, the 'estimation errors' of an adversary wrt the given privacy policies on the database, at any point of his/her querying process. (In [10], such estimations are expressed in terms of suitably defined 'probability measures'.) Implementation and experimentation are part of our future work, where we also hope to define a 'divergence measure' between two given nodes on a DLTTS modeling a querying process, in terms of the knowledge distributions at the two nodes – independently of any notion of a given target data set.

References

[1] G. Barthe, B. Köpf, F. Olmedo, S.Z. Béguelin. "Probabilistic relational reasoning for differential privacy". In: Proceedings of POPL, ACM (2012)

[2] G. Barthe, R. Chadha, V. Jagannath, A. Prasad Sistla, M. Viswanathan. "Deciding Differential Privacy for Programs with Finite Inputs and Outputs". In: LICS'20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020.

[3] V. Castiglioni, K. Chatzikokolakis, C. Palamidessi. "A Logical Characterization of Differential Privacy via Behavioral Metrics". In: Formal Aspects of Component Software (FACS), Pohang, South Korea. pp. 75–96, Oct. 2018.

[4] V. Castiglioni, M. Loreti, S. Tini. "The metric linear-time branching-time spectrum on nondeterministic probabilistic processes". In: Theoretical Comp. Science, Vol. 813:20–69, 2020.

[5] K. Chatzikokolakis, M. Andrés, N. Bordenabe, C. Palamidessi. "Broadening the Scope of Differential Privacy Using Metrics". In: Privacy Enhancing Technologies Symposium (PETS), Bloomington, IND (US), pp. 82–102, 2013,

[6] L. de Alfaro, M. Faella, M. Stoelinga. "Linear and Branching System Metrics". In: IEEE Trans. on Software Engineering, Vol. 35(2):258–273, 2009.

[7] C. Dwork. "Differential privacy". In: Proceedings of ICALP 2006. LNCS (Springer–Verlag), Vol. 4052, pp. 1–12 (2006).

[8] C. Dwork. A. Roth. "The Algorithmic Foundations of Differential Privacy". In: Found. Trends Theor. Comput. Sci., Vol. 9:3-4, pp. 211–407, 2014.

[9] N. Holohan, S. Antonatos, S. Braghin, P. M. Aonghusa. "The Bounded Laplace Mechanism in Differential Privacy". In: Journal of Privacy and Confidentiality (Proc. TPDP 2018), Vol. 10 (1), 2020.

[10] D. Rebollo-Monedero, J. Parra-Arnau, C. Díaz, J. Forné. "On the measurement of privacy as an attacker's estimation error". In: Int. J. Inf. Sec. 12(2): 129–149 (2013).

[11] R. Segala. "Modeling and Verification of Randomized Distributed Real-Time Systems". Ph.D. thesis, MIT (1995).

[12] R. Segala. "A compositional trace-based semantics for probabilistic automata". In: Proc. CONCUR'95, 1995, pp. 234–248.

[13] R. Segala, N.A. Lynch. "Probabilistic simulations for probabilistic processes". In: Nord. J. Comput. 2(2):250–273, 1995.

[14] Stanley L. Warner. "Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias" In: Journal of the American Statistical Association Vol. 60(309), pp. 63–69, 1965.

[15] Z. Wu, M. Palmer. "Verb Semantics and Lexical selection". In: Proc. 32nd Annual meeting of the Associations for Comp. Linguistics, pp 133-138. 1994.

Appendix

Taxonomies are frequent in machine learning. Data mining and clustering techniques employ reasonings based on measures of symmetry, or on metrics, depending on the objective. The Wu-Palmer symmetry measure on tree-structured taxonomies is one among those in use; it is defined as follows ([15]): Let $\mathcal{T}$ be a given taxonomy tree. For any node x on $\mathcal{T}$, define its depth c_x as the number of

nodes from the root to x (both included), along the path from the root to x. For any pair x, y of nodes on $\mathcal{T}$, let c_{xy} be the depth of the common ancestor of x, y that is *farthest* from the root. The Wu-Palmer symmetry measure between the nodes x, y on $\mathcal{T}$ is then defined as $\mathrm{WP}(x,y) = \frac{2\,c_{xy}}{c_x+c_y}$. This measure, although considered satisfactory for many purposes, is known to have some disadvantages such as not being conform to semantics in several situations.

What we are interested in, for the purposes of our current paper, is a *metric* between the nodes of a taxonomy tree, which in addition will suit our semantic considerations. This is the objective of our Lemma below. (A result that seems to be unknown, to our knowledge.)

Lemma 1 *On any taxonomy tree $\mathcal{T}$, the binary function between its nodes defined by* $d_{wp}(x,y) = 1 - \frac{2\,c_{xy}}{c_x+c_y}$ (notation as above) *is a metric.*

Proof: We drop the suffix *wp* for this proof, and just write d. Clearly $d(x,y) = d(y,x)$; and $d(x,y) = 0$ if and only if $x = y$. We only have to prove the Triangle Inequality; i.e. show that $d(x,z) \leq d(x,y) + d(y,z)$ holds for any three nodes x, y, z on $\mathcal{T}$. A 'configuration' can be typically represented in its 'most general form' by the diagram below. The boldface characters X, Y, Z, a, h in the diagram stand for the *number of arcs* on the corresponding paths. Thus, for the depths of the nodes x, y, z, and of their farthest common ancestors on $\mathcal{T}$, we get:

$$c_x = X + h + 1, \;\; c_y = Y + h + a + 1, \;\; c_z = Z + h + a + 1,$$
$$c_{xy} = h + 1, \;\; c_{yz} = h + a + 1, \;\; c_{xz} = h + 1$$

The '+1' in these equalities is because the X, Y, Z, a, h stand for the *number of arcs* on the paths, whereas the depths are defined as the number of nodes. Also note that the X, Y, Z, a, h must all be integers ≥ 0.

For the Triangle Inequality on the three nodes x, y, z on $\mathcal{T}$, it suffices to prove the following two relations:

$$d(x,z) \leq d(x,y) + d(y,z) \quad \text{and} \quad d(y,z) \leq d(y,x) + d(x,z).$$

by showing that the following two algebraic inequalities hold:

(1) $1 - \frac{2*(h+1)}{(X+Y+2*h+a+2)} + 1 - \frac{2*(h+a+1)}{(Y+Z+2*h+2*a+2)} \geq 1 - \frac{2*(h+1)}{(X+Z+2*h+a+2)}$

(2) $1 - \frac{2*(h+1)}{(X+Y+2*h+a+2)} + 1 - \frac{2*(h+1)}{(X+Z+2*h+2*a+2)} \geq 1 - \frac{2*(h+a+1)}{(Y+Z+2*h+2*a+2)}$

The third relation $d(x,y) \leq d(x,z) + d(z,y)$ is proved by just exchanging the roles of Y and Z in the proof of inequality (1).

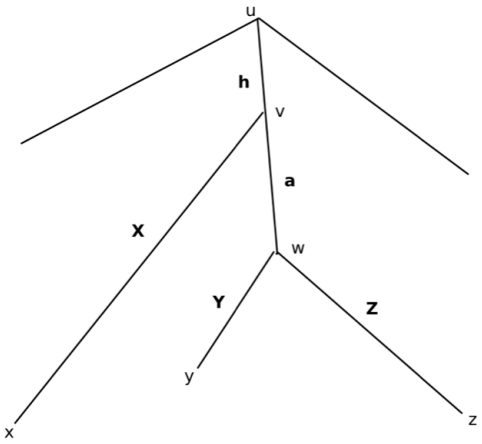
Inequality (1): We eliminate the denominators (all strictly positive), and write it out as an inequality between two polynomials $eq1, eq2$ on X, Y, Z, h, a, which must be satisfied for all their non-negative integer values:

$$eq1 : (X+Y+2*h+a+2)*(Y+Z+2*h+2*a+2)*(X+Z+2*h+a+2)$$
$$eq2 : (h+1)*(Y+Z+2*h+2*a+2)*(X+Z+2*h+a+2)$$
$$+(h+a+1)*(X+Y+2*h+a+2)*(X+Z+2*h+a+2)$$
$$-(h+1)*(X+Y+2*h+a+2)*(Y+Z+2*h+2*a+2)$$
$$eq : eq1 - 2*eq2. \quad \text{We need to check: } eq \geq 0\ ?$$

The equation eq once expanded (e.g., under *Maxima*) appears as:

$$eq : YZ^2 + XZ^2 + aZ^2 + Y^2Z + 2XYZ + 4hYZ + 2aYZ + 4YZ + X^2Z + 4hXZ + 2aXZ + 4XZ + a^2Z + XY^2 + 4hY^2 + aY^2 + 4Y^2 + X^2Y + 4hXY + 2aXY + 4XY + 8h^2Y + 8ahY + 16hY + a^2Y + 8aY + 8Y$$

The coefficients are all positive, and inequality (1) is proved.



Inequality (2): We again proceed as above: we first define the following polynomial expressions:

$eq3 : (X + Y + 2 * h + a + 2) * (X + Z + 2 * h + a + 2) * (Y + Z + 2 * h + 2 * a + 2);$

$eq4 : (h + 1) * (Y + Z + 2 * h + 2 * a + 2) * (2 * X + Y + Z + 4 * h + 2 * a + 4);$

$eq5 : (h + a + 1) * (X + Y + 2 * h + a + 2) * (X + Z + 2 * h + a + 2);$

If we set $eqn : eq3 + 2 * eq5 - 2 * eq4,$ we get

$$eqn : -2(h + 1) * (Z + Y + 2h + 2a + 2) * (Z + Y + 2X + 4h + 2a + 4) + (Y + X + 2h + a + 2) * (Z + X + 2h + a + 2)(Z + Y + 2h + 2a + 2) + 2(h + a + 1) * (Y + X + 2h + a + 2) * (Z + X + 2h + a + 2)$$

To prove inequality (2), we need to show that eqn remains non-negative for all non-negative values of X, Y, Z, h, a. Expanding eqn (with *Maxima*), we get:

$$eqn: YZ^2 + XZ^2 + aZ^2 + Y^2Z + 2XYZ + 4hYZ + 6aYZ + 4YZ + X^2Z + 4hXZ + 6aXZ + 4XZ + 8ahZ + 5a^2Z + 8aZ + XY^2 + aY^2 + X^2Y + 4hXY + 6aXY + 4XY + 8ahY + 5a^2Y + 8aY + 4hX^2 + 4aX^2 + 4X^2 + 8h^2X + 16ahX + 16hX + 8a^2X + 16aX + 8X + 8ah^2 + 12a^2h + 16ah + 4a^3 + 12a^2 + 8a$$

The coefficients are all positive, so we are done. $\square$.