

A framework to balance privacy and data usability using data degradation

Harold van Heerde
Database management department
University of Twente
Enschede, The Netherlands
h.j.w.vanheerde@ewi.utwente.nl

Maarten Fokkinga
Database management department
University of Twente
Enschede, The Netherlands
fokkinga@ewi.utwente.nl

Nicolas AnCIAUX
SMIS Project
INRIA Rocquencourt
Le Chesnay, France
nicolas.anciaux@inria.fr

Abstract—Personal data is a valuable asset for service providers. To collect such data, free services are offered to users, for whom the risk of losing privacy by subscribing to a service is often not clear. Although the services are free in terms of money, the user does not know how much he or she *actually* pays for a given service when allowing his or her data to be collected, unaware of taking a significant privacy risk by doing so. In practice, this risk is even not taken into account when deciding how long the data will be retained; the service provider simply wants to optimize the total worth of the stored data by retaining the data as *long* as possible. In this paper, we express the privacy risk for the user in terms of such a retention period; the user wants to optimize its privacy by allowing the data to be retained as *short* as possible. Now, instead of only considering the interests of the service provider, we argue that we should optimize the *common interest* of both parties, and present a framework to reason about *worth* and *privacy* to find such optimum. Going one step further, we refine and generalize *limited retention to data degradation*, which prescribes to store data in progressively less accurate forms. Data degradation gives users and service providers a fine grained control over the price to be paid, in terms of privacy risks, and to optimize their common interest: balancing privacy and data usability.

I. INTRODUCTION

Privacy has become a popular topic, triggered by the vast amount of web services with an apparently unsatisfiable desire for their users' personal data. Acquiring personal data is big business, a new gold mine for Internet companies, boosting all kind of new web services, increasing the threat to privacy even further [9]. It works; Google can reach over half a billion unique individuals each year, collecting—among many different types of personal data—their search queries, which to an high extent encapsulate their daily lives' habits [10]. Google made in 2008 a revenue of \$22.1 billion [12], indicating the worth of that personal data for the company. Google is not alone; in their footsteps many other companies followed, and many will follow.

What do the users get in return for their personal data? Indeed, they profit from all the services which ease their lives. The web has been made accessible thanks to search engines, and communicating with friends and relatives has never been easier. However, until the Internet era, transactions between producer and consumer have been much more transparent for both parties. The consumer pays the price which has been

negotiated between producer and consumer, and the producer delivers the good or service. If the price is not satisfactory for both parties, the transaction will not take place. So, it pays off for the producer to be transparent. Today, business models are different. Services are offered for free—in terms of money—to the user, such that, at first glance, there is no reason to negotiate anymore.

Here the privacy danger becomes apparent. Transparency is one of the key foundations of privacy [19]; it must be clear for the user how his or her data is being handled, stored, and to whom it will be disclosed. In other words: the price a user has to pay for a service should be expressed in terms of privacy risks, where it was expressed in terms of money in the old days. If the service provider can argue that the data is needed to offer certain kind of services, the user may want to decide to allow the service provider to keep the data longer, paying a higher price, most probably benefiting from a better service.

So why is it a problem that companies store all these data about us? The fact is that, even if we put full trust in the service provider, this data can always be subject to data disclosure due to attacks, corrupt employees, governments demanding the data, et cetera. No access control mechanism has been proven to be both usable and fully secure; to give an example, even servers of the Pentagon [8] and FBI [23] have been hacked, credit card companies and mobile communication companies have lost personal data on several occasions [27]. Moreover, human mistakes are hardly preventable: politicians and policemen lose usb sticks or other media with sensitive information [11], obsolete personal computers sold second-hand are subject to forensic analysis with sometimes shocking results [26]. Finally, personal data is often weakly protected by obscure and loose privacy policies which are unjustly presumed to be good and acceptable for a given service.

The privacy violation will only increase with the growth of data which has been collected about us. All these data, even when “legally” obtained by the service providers themselves, foster ill-intended scrutiny and abusive usages justified by business interests, governmental pressures and inquisitiveness among people. Not only criminals and terrorists are threatened. Everyone may experience a particular event (e.g., accident, divorce, job or credit application) which suddenly makes her digital trail of interest for someone else. Moreover, identity

fraud is nowadays becoming one of the most serious crimes, with huge consequences for the victims [15]. The retention problem has become so important and the civil pressure so high that practices start changing. For instance, Google and other search engine companies announced to shorten the retention period of their query logs. Limiting the retention of personal data indeed reduces the privacy problems sketched above. *Limited retention* is a widely accepted privacy principle [2], complementary to techniques such as access control, and is included in various privacy regulations [14]. The principle prescribes that data should not be stored longer than necessary to fulfill the purpose for which the data has been collected [2]. By limiting the time that data is stored, the impact of disclosure of a store is less severe [10].

Limited retention is however difficult to put in practice, because it is difficult to determine retention periods; we need to find a mechanism to balance privacy and usability. Otherwise we either end up with a lot of worth for the data collector and zero privacy for the user, or zero worth and full privacy [10]. In this paper we propose to put service provider and user at the negotiation table again, with equal rights for both parties. The goals are clear: the service provider wants to optimize the worth of the data it can collect, and the user wants to optimize his or her privacy. We will model those goals and argue to optimize the *common interest* of both parties. As a result, both parties will agree upon a retention period for storing the users' personal data in exchange for the service offered by the provider.

However, the all-or-nothing behavior of limited retention is too rigorous: after the retention period, the data will be destroyed completely. This makes it hard to balance data usability and privacy. In this paper we also propose a new technique named data degradation, which can bridge the gap between both ends. By using well-known generalization techniques [17] (not elaborated in this paper), data degradation *degrades* the accuracy of the data after predefined retention periods, such that although the usability of that data will decrease, the privacy sensitivity will also decrease. This technique is orthogonal to other privacy enhancing techniques such as *k*-anonymity [24]; our aim is not to hide identities, but only to make the *knowledge related to those identities* less privacy sensitive. We will show that with data degradation, the common interest will be at least equal, but in many situations significantly higher, than which is possible with only limited retention.

In summary, the main contributions of our paper are these:

- a framework to reason about retention periods in order to optimize the common interest of both users and service providers, discussed in Section II;
- the technique *data degradation*, which makes it possible to achieve a common interest that is higher than possible with limited retention, discussed in Section III and analyzed, with explanation of the benefits, in Section IV.

We also indicate the challenges data degradation raises with respect to traditional databases (Section V), mainly by referring

to other work. Section VI compares data degradation to related work and indicates where data degradation is complementary to existing techniques. Finally, Section VII gives directions for further work in this new research area of limited retention techniques.

II. FINDING LIMITED RETENTION PERIODS

To let the reader get familiar with our notations and reasoning, we introduce our concepts for the limited retention principle here. In the next section we will reuse and extend those concepts when we refine limited retention to data degradation. Our goal is a qualitative framework which makes it possible to reason about retention periods. We model the interest of a service provider and, separately, the interest of the user in a way that is as simple as possible; a more elaborate model is left for future work. Then we combine these to a *common interest*, from which an optimal retention period can be derived. We make no quantitative statements about *how* exactly these interests will be expressed in practice; again, the practical implications will be left for future work.

A. Preliminaries

In the sequel, we consider only one user, which we refer to as “the user”. Considering more users is no problem but would lead to the same result if we treat them on equal footing. Also, we shall not make a distinction between different kinds of data: we treat all data on equal footing.

The history of which datum is inserted into the store at which time, is called H ; it is a set of pairs of a datum and its insertion time in the store. For example:

$$H = \{(d, b), \dots, (d', b')\}$$

We take H as a constant, we let d range over the set of data and t over the set of time points, and use letter b (birth) for ‘the insertion time of a datum into the store’. We use \mathbb{N} to measure an *age*, i.e., length of a time interval, and let a, δ range over ages. A simplifying assumption is that the insert rate of data is constant over time; that is, there exists a constant c such that during each interval of length a the amount of data inserted into the store is $c \times a$:

$$\forall t, a \bullet \#\{(d, b) : H \mid t \leq b < t + a\} = c \times a \quad (1)$$

By its definition, limited retention bounds the interval during which a datum is stored by a fixed *retention period* δ . Hence, the *store* at time t depends on δ and is expressed as follows:

$$store(\delta, t) = \{(d, b) : H \mid b \leq t < b + \delta\}$$

B. Service provider's interest

In practice, the *worth* of a datum for the service provider depends on multiple factors, such as the actual content of the datum, the context in which the datum has been acquired, the user from which the datum has been acquired, time of day, and possibly many others depending on the type of use. Our model does not limit the possibility to include those parameters; however, for the sake of simplicity we omit them. Thus, we

assume that for the service provider the worth at time t of a datum with birth b depends only on the datum's age $t-b$:

$$\text{worth}((d, b), t) = \text{wt}(t-b)$$

where $\text{wt}(a)$ is nonnegative, monotonic descending in a

The auxiliary function wt is monotonic descending since older data is assumed to be less valuable for the service provider. Figure 1(a) gives the typical shape of function $\text{wt}(a)$.

The service provider wants, at each point t in time, to maximize the *total worth* of the data in the store:

$$\begin{aligned} & \text{totworth}(\delta, t) \\ = & \quad \text{definition} \\ = & \sum_{(d,b):\text{store}(\delta,t)} \text{worth}((d, b), t) \\ = & \sum_{(d,b):\text{store}(\delta,t)} \text{wt}(t-b) \\ = & \left\{ \begin{array}{l} \text{A stored datum of age } a \text{ contributes } \text{wt}(a) \text{ to} \\ \text{the sum. Thanks to the constant insert rate (1),} \\ \text{the number of stored datums of age } a \text{ is the} \\ \text{same (namely } c \text{) for each } a = 0, \dots, \delta. \end{array} \right. \\ = & c \times (\text{wt}(0) + \text{wt}(1) + \dots + \text{wt}(\delta)) \\ = & c \times \sum_{a=0 \dots \delta} \text{wt}(a) \end{aligned}$$

It turns out that $\text{totworth}(\delta)$ does not depend on t , hence we omit parameter t and simply write $\text{totworth}(\delta)$. Figure 1(b) gives the typical shape of function $\text{totworth}(\delta)$. Without other constraints, the service provider would achieve his goal by setting δ to infinity.

C. User's interest

For the user it is risky to have data stored at the service provider: in some way or another (hackers' attacks, for instance) the data might be disclosed. We assume that the harm for the user of a disclosure of his data is proportional to the amount of data. Below, we take the risk *equal* to the amount of data since this simplifies the formulas and nevertheless gives the same results. Again, there are multiple other factors which influence the risk of storing a datum. For example, the fact that a user searched for HIV is more risky than a search for flowers. Also, we assume that disclosure of old data is as harmful as disclosure of recent data. Our model does not restrict the possibility to take those factors into account, but we leave this for future work. Thus we define and simplify *risk* as follows:

$$\begin{aligned} & \text{risk}(\delta, t) \\ = & \quad \text{definition} \\ = & \#\{(d, b) : \text{store}(\delta, t)\} \\ = & \#\{(d, b) : H \mid b \leq t < b + \delta\} \\ = & \text{constant insert rate (1)} \\ & c \times \delta \end{aligned}$$

So, $\text{risk}(\delta, t)$ doesn't depend on t and we simply write $\text{risk}(\delta)$.

The goal of the user is to minimize $\text{risk}(\delta)$. It is equivalent to *maximize* the inverse: $1/\text{risk}(\delta)$, which we call the *privacy guarantee*. It follows that the privacy guarantee is infinite when there is no data in the store, $\delta = 0$ (and goes down to zero

when retention is unlimited, $\delta = \infty$). To escape mathematical problems (division by zero) and come to a slightly more realistic model of privacy, we apply a *smoothing* technique so that privacy cannot be infinite: add a constant to the denominator of $1/\text{risk}(\delta)$. The smoothing constant s may have a reasonable interpretation; for example, the fact that there is no data in the store, might be interpreted as an indication that "the user has something to hide" and so his privacy guarantee is not infinite. Thus our definition reads:

$$\text{priv}(\delta) = \frac{1}{s + \text{risk}(\delta)} = \frac{1}{s + c \times \delta}$$

Figure 1(c) gives the typical shape of function $\text{priv}(\delta)$. The user wants to maximize $\text{priv}(\delta)$, which without further constraints is achieved by taking δ as small as possible.

D. Common interest

Above we defined the interests of both service provider and user. Those interests are conflicting; whereas the service provider benefits most when δ is large, the user aims for a δ as small as possible. We want a concept of *common interest* which both parties can agree upon. We expect that the common interest leads to a retention period which is both *limited* ($\delta < \infty$) and *non-zero* ($\delta > 0$).

To define the common interest $CI(\delta)$, we require two things. First, $CI(\delta)$ is proportional to the service provider's goal function $\text{totworth}(\delta)$ when the user's interest is viewed as constant. Second, $CI(\delta)$ is proportional to the user's goal function $\text{priv}(\delta)$ when the service provider's interest is viewed as constant. Since both goal functions are nonnegative, a suitable function $CI(\delta)$ is the *product* of these:

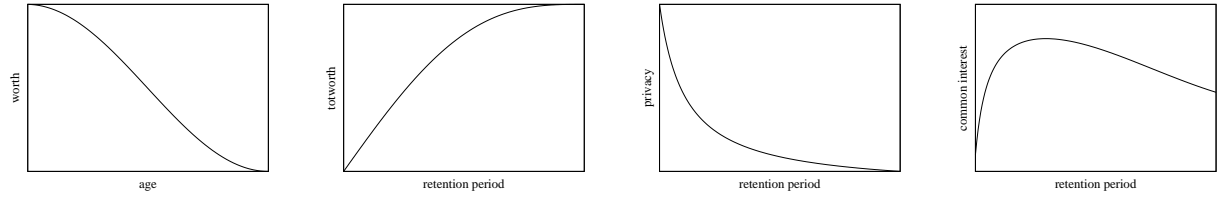
$$CI(\delta) = \text{totworth}(\delta) \times \text{priv}(\delta)$$

Figure 1(d) gives the typical shape of function $CI(\delta)$.

Since *worth* is monotonic descending and *risk* is (almost) proportional to δ , it follows that $CI(_)$ has a maximum, which it takes on argument δ_{opt} , say. The existence of a maximum can be interpreted in the following way. By setting the retention period smaller than δ_{opt} the user will gain more privacy, but the common interest will be lower because the decrease in stored data induces a greater loss of total worth for the service provider. Similarly, by setting the retention period larger than δ_{opt} the service provider will gain more total worth of the stored data, but the common interest will be lower because of a larger decrease of the user's privacy.

III. THE CONCEPT OF DATA DEGRADATION

Recall that the principle of limited retention tries to satisfy the service provider by allowing to store data for at least the retention period δ , and it tries at the same time to satisfy the privacy concern of the user by ensuring that the data is stored for at most the retention period δ (and the previous section shows how to reason about the optimal retention period). The principle is a crude all-or-nothing approach: a datum either exists completely in the store or not at all. The principle of *data degradation* overcomes the all-or-nothing approach by



(a) $worth((d, b), t)$ is monotonic descending in $age = t - b$
 (b) $totworth(\delta)$, derived from the chosen $worth$ function, is independent of t
 (c) $priv$ is inversely proportional to risk, which is proportional to δ
 (d) $CI(\delta) = totworth(\delta) \times priv(\delta)$

Fig. 1. The common interest function reaches its highest point at δ_{opt} meaning that a retention period of δ_{opt} gives the best balance between $totworth$ for the service provider and $priv$ for the user.

storing data in progressively less accurate forms so as to make it less privacy sensitive over time while still providing some worth to the service provider. A well accepted form of data degradation is data *generalization*. This technique is often used in k -anonymity research, where it is used to generalize identities in order to make the data k -anonymous [24], and applied in data mining and warehousing [17]. More about the techniques we use to generalize data—based on domain hierarchies and generalization trees—can be found elsewhere [3].

A. Life-cycle policies

To formalize the data degradation principle, we need some terminology and notation. First, we distinguish several *levels* of accuracy, say L_0, L_1, \dots, L_{n-1} in decreasing order of accuracy. Level L_0 denotes the most accurate level. Second, the degradation from L_{i-1} to L_i is denoted τ_i (τ is mnemonic for “transformation”). Third, the interval from the birth of a datum to its degradation to L_i is denoted δ_i ; it follows that $\delta_0 = 0$ because a datum is supposed to enter the store in the most accurate level, and we let δ_n be the interval from birth to removal from the store. The notation $\vec{\delta}$ abbreviates the sequence $\delta_1, \dots, \delta_n$. Almost all this information is captured in a so-called *life-cycle*, as illustrated in Figure 2.

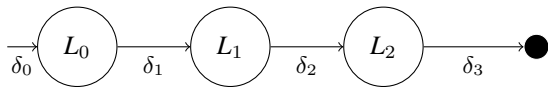


Fig. 2. Graphical representation of a simple *life-cycle*. Edges denote transitions between Levels of accuracy after a *retention period* δ .

So, the store consists of data of age at most δ_n , and degraded to the appropriate levels:

$$\begin{aligned}
 store(\vec{\delta}, t) = & \\
 & \{(d, b):H \mid b + \delta_0 \leq t < b + \delta_1 \bullet (d, b, L_0)\} \\
 \cup & \{(d, b):H \mid b + \delta_1 \leq t < b + \delta_2 \bullet (\tau_1(d), b, L_1)\} \\
 & \vdots \\
 \cup & \{(d, b):H \mid b + \delta_{n-1} \leq t < b + \delta_n \bullet (\tau_{n-1}(d), b, L_{n-1})\}
 \end{aligned}$$

The product of our framework is a *life-cycle policy*, which captures how and when data needs to be degraded, and to which the service provider has to comply with. As long as the service provider can be *trusted*, the life-cycle policy ensures

that *if* the data store is attacked, the impact of disclosure will be less severe; only a small subset of the data will be stored in an accurate form, the rest will be either degraded or destroyed. Some of the technical challenges related to the implementation and enforcement of such policies on traditional database systems are discussed in Section V.

B. Interests revised

We assume that degraded data is less worthwhile for the service provider but also less risky for the user to store, and we will revise the definitions of *worth* and *risk* accordingly. The definitions of the total worth, privacy, and common interest in terms of *worth* and *risk* remain the same except for the replacement of δ by $\vec{\delta}$.

Worth and total worth: For the service provider, the worth of a datum depends not only on the age “ $t-b$ ” but also on the level of accuracy l :

$$worth((d, b, l), t) = wt_l(t-b)$$

$$wt_l(a) \text{ is nonnegative, monotonic descending in } a \text{ and } l$$

The effect of degrading a datum to a level of lesser accuracy is that its worth for the service provider is decreased. Indeed, ‘ $wt_l(a)$ is monotonic descending in l ’ means that for all a :

$$wt_0(a) \geq wt_1(a) \geq \dots \geq wt_{n-1}(a)$$

As before, the total worth of the store at time t is the aggregation of the worth of all data in the store:

$$\begin{aligned}
 & totworth(\vec{\delta}, t) \\
 = & \text{definition} \\
 = & \sum_{(d, b, l):store(t, \vec{\delta})} worth((d, b, l), t) \\
 = & c \times (\\
 & wt_0(\delta_0) + wt_0(\delta_0+1) + \dots + wt_0(\delta_1 - 1) + \\
 & wt_1(\delta_1) + wt_1(\delta_1+1) + \dots + wt_1(\delta_2 - 1) + \\
 & \dots \\
 & wt_{n-1}(\delta_{n-1}) + wt_{n-1}(\delta_{n-1}+1) + \dots + wt_{n-1}(\delta_n - 1)) \\
 = & c \times \sum_{l=0}^{n-1} \sum_{a=\delta_l}^{\delta_{l+1}-1} wt_l(a)
 \end{aligned}$$

It follows that $totworth(\vec{\delta}, t)$ is independent of t , and we can simply write $totworth(\vec{\delta})$. Note that the contribution to $totworth(\vec{\delta})$ of the most degraded level may be negligible in

comparison to the less degraded levels: both the age and the level are higher.

Risk and privacy: The risk of having data in the store was proportional to the amount of data in the store. However, with data degradation, the assumption is that storing data in a more degraded level is less risky than for a more accurate level. To express this, we weight the risk of storing a datum in level L_l with a factor r_l such that $1 = r_0 \geq r_1 \geq \dots \geq r_{n-1}$. Hence, the definition reads:

$$\begin{aligned} & \text{risk}(\vec{\delta}, t) \\ = & \text{definition} \\ & \sum_{l=0}^{n-1} r_l \times \#\{(d, b, l') : \text{store}(\vec{\delta}, t) \mid l' = l\} \\ = & c \times \sum_{l=0}^{n-1} r_l \times (\delta_{l+1} - \delta_l) \end{aligned}$$

Again, $\text{risk}(\vec{\delta}, t)$ is independent of t and we simply write $\text{risk}(\vec{\delta})$. The definition of privacy doesn't change:

$$\text{priv}(\vec{\delta}) = 1 / (s + \text{risk}(\vec{\delta}))$$

Common interest: The goals of both service provider and user remain the same: optimizing the common interest. Except for the replacement of δ by $\vec{\delta}$ there is no change:

$$CI(\vec{\delta}) = \text{totworth}(\vec{\delta}) \times \text{priv}(\vec{\delta})$$

Note that if there is just one level of accuracy, $n = 1$, the newly defined notions coincide with the already existing notions (such as $CI(\vec{\delta})$ and $CI(\delta)$) provided we take $w_{t_0}(a) = wt(a)$ and $\delta_1 = \delta_n = \delta$.

IV. BENEFITS OF DATA DEGRADATION

The aim of data degradation is not to provide more privacy while ensuring the same amount of worth for the service provider, nor providing more worth while ensuring the same amount of privacy as what can be achieved with limited retention. Instead, we want to show that we can achieve an higher *common interest* with data degradation than with limited retention.

Since the contributions of $\delta_2, \delta_3, \dots$ to CI are nonnegative, data degradation with $n > 1$ will outperform limited retention:

$$\forall \delta_1, \delta_2, \dots \bullet CI(\delta_1) \leq CI(\delta_1, \dots, \delta_n)$$

In the following we will show, using experimental examples, that for a set of examples which all comply with our assumptions, the common interest will be higher when we use data degradation. Hence, our target is to give insights in what we can gain by choosing data degradation in comparison to limited retention.

A. Analysis

We use Matlab as the platform for our analysis. We implemented a set of worth and privacy functions which simulate the functions which have a practical shape, complying with the assumptions posed in section II. Given those functions, we let Matlab optimize the common interest considering $n = 1 \dots 4$ possible degradation steps. Hence, choosing $n = 1$ means the

same as applying limited retention, whereas $n > 1$ means we allow one or more degradation steps.

Although we can easily experiment with different types of worth functions to simulate a service provider's worth function—as long as they are monotonic descending—we choose the cosine function, as in Figure 3.

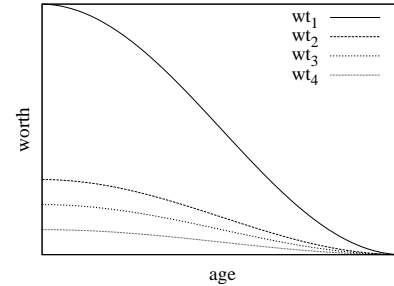


Fig. 3. $w_{t_l}(a) = w_l \times \left(1 + \cos\left(\frac{a \times \pi}{180}\right)\right)$, $0 \leq a \leq 180$

In the worth functions we use weights $w_1 \geq w_2 \geq \dots \geq w_n$. Recall that r_i are weight factors in the risk function. We choose $w_1 = r_1 = 1$ and $w_2 \dots w_n$ vary over $0 \dots 1$, similarly for r_i .

1) *Increased common interest:* To show that using data degradation indeed can result in a higher common interest, we let our script find the $\vec{\delta}$ for which common interest is maximal, with at most $n = 4$ degradation steps. We choose the following parameters:

$i =$	1	2	3	4
w_i	1	0.3	0.2	0.1
r_i	1	0.2	0.1	0.05
$s =$	18			

With those parameters, we obtain the following results (also shown in Figure 4).

$n =$	1	2	3	4
$\vec{\delta}$	[53]	[26,81]	[16,46,96]	[16,46,96,96]
$CI(\vec{\delta})$	0.7948	0.8358	0.8532	0.8532
$\text{totworth}(\vec{\delta})$	98.5582	76.8960	61.4329	61.4329
$\text{priv}(\vec{\delta})$	0.0081	0.0109	0.0139	0.0139

From this result we conclude for the chosen parameters:

- 1) Common interests is higher when progressively degrading ($n > 1$) the data than with limited retention of accurate data ($n = 1$)
- 2) With $n > 1$, δ_1 is smaller than the single δ with $n = 1$. It thus makes sense to degrade the data earlier to achieve an higher common interest.
- 3) When $n = 4$, it turns out that $\delta_3 = \delta_4$, meaning that it is not possible to achieve an higher common interest with more than three degradation steps.

2) *A closer look on weights and their effect on CI:* We already have seen that data degradation results in an higher common interest. An interesting question is how the ratios between weights $w_1 \dots w_n$ and $r_1 \dots r_n$ assigned to each level of accuracy L_i influence the gain in common interest which can be achieved with data degradation. Our expectation is the

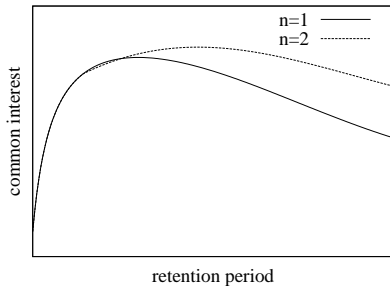


Fig. 4. Common interest function for $n = 1$ (limited retention) and $n = 2$. For $n = 2$, we only vary over δ_2 . The first retention period δ_1 is chosen such that $CI(\delta_1, \delta_2)$ is optimal for the δ_2 where this plot reaches its highest point. The graph shows that δ_1 for $n = 2$ is shorter than the optimal δ_1 when $n = 1$, and that the common interest for $n = 2$ is higher than for $n = 1$.

following: with w_i descending in i , and $w_i > r_i$, degradation results in a higher common interest compared to limited retention.

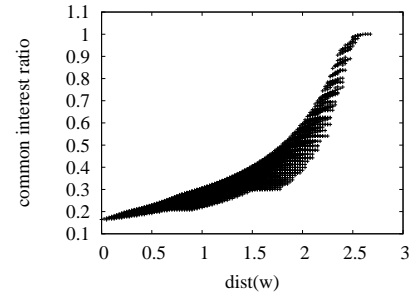
For this experiment we choose arbitrarily $r_1 = 1, r_2 = 0.2, r_3 = 0.1, r_4 = 0.05$, and vary over w_i . We use a simple distance metric: $dist(\vec{w}) = \sum_{i=2}^n w_1 - w_i$ to express the drop in *worth* when degrading the data. Note that the smaller w_i is, the less *worth* is preserved on that accuracy level.

In Figure 5 we plotted for various weights w_i the ratio between the common interest which can be achieved with only limited retention and the common interest which can be achieved by *at most* $n = 4$ degradation steps. Figure 5(b) shows only common interest points achieved with at least 3 degradation steps. The ratio indicates the fraction of common interest possible with limited retention compared to that of data degradation. Hence, this fraction cannot be higher than 1 since limited retention can never perform better than data degradation (see earlier this section).

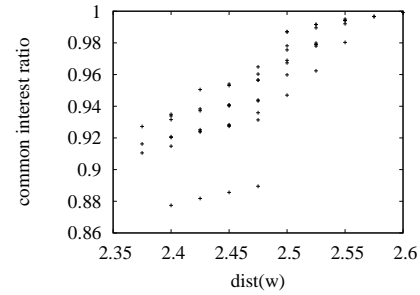
We make the following observations:

- 1) When $dist(\vec{w}) \simeq dist(\vec{r}) \simeq 2.65$, the decrease in worth is similar to the increase in privacy, so that common interest hardly increases (*CI* ratio is close to 1).
- 2) When the decrease in worth becomes higher ($dist(w)$ grows), data degradation hardly outperforms limited retention.
- 3) Most gain in common interest is achieved by applying *one-step degradation*: data is immediately degraded to a higher level, and fully degraded afterwards. This can be concluded from Figure 5(b), in which only *multi-step degradation* is allowed.

From the last observation, we conclude that for some (our current) parameter values it makes sense to generalize the data before storing it. Hence, a higher common interest can be achieved by handing over some accuracy to get much more privacy in return, especially when the privacy increase is much higher than the loss in worth. When the decrease in worth is more close to the increase in privacy, we showed again that multi-step degradation leads to an increase in common interest.



(a) one-or-multi-step degradation



(b) multi-step degradation

Fig. 5. Ratio between the common interest which can be achieved with only limited retention and the common interest which can be achieved by allowing $n = 4$ degradation steps. A ratio equal to 1 means no increase in common interest, lower than 1 indicates that limited retention performs less than data degradation. This ratio can never be higher than 1; limited retention cannot perform better than data degradation.

V. IMPACT ON TRADITIONAL DATABASES

Since data degradation will take place within the data store, the first and legitimate question which comes in mind is how complex will the technology be to support it. Traditional databases are developed in order to efficiently and durably insert data, make updates to this data and to query them. The ACID properties ensure durability and correct execution of queries and updates to keep the database in a consistent state. In our context, we have to revise the ACID properties, such that the durability requirement ends when data should be degraded, even *requiring* that data can *not* be recovered. In order to apply the degradation model correctly, we have to ensure that even operations made outside control of the DBMS cannot reverse the degradation steps. A delete statement should therefor not only be made visible on the application level, but it should also be internally irreversible. However, in current traditional database implementations this is not the case [22].

Hence, identifying the impact of making a database data-degradation aware leads to several important questions. For more details, and a comprehensive study on this topics, we refer to [3]. Here, we only give a brief overview.

Updates to the data items due to degradation, as well as final removal from the database have to be enforced. As pointed out in [22], traditional databases cannot even guarantee the non-recoverability of deleted data due to different forms of unintended retention in the data space, the indexes and the logs. With data degradation, the problem is particularly acute

considering that each data item inserted in the database undergoes multiple degradation steps. The storage of degradable attributes, indexes and logs have to be revisited in this light.

Databases have been designed to speed up queries. Some workloads induce the need of few indexes on the most selective attributes to get the best trade-off between selection performance and insertion/update/deletion cost. For other application, insertions are done off-line, queries are complex and the data set is very large. This leads to multiple indexes to speed up even low selectivity queries. Data degradation can be useful in both contexts. However, data degradation changes the workload characteristics in the sense that queries become less selective when applied to degradable attributes. This introduces the need for indexing techniques supporting efficiently degradation.

User transactions inserting data items with degradable attributes generates effects all along the lifetime of the degradation process, that is from the transaction commit up to the time where all inserted data items have reached a final state for all their degradable attributes. This significantly impacts transaction atomicity and durability and even isolation considering potential conflicts between degradation steps and reader transactions.

VI. RELATED WORK

Proposals have been made to make the donor herself responsible for protecting her own data, not just relying on the philosophy of trusting organizations to protect the privacy. In their vision paper, Aggarwal et al proposed the P4P framework [1], in which the donor keeps control about which information to release to service providers. They consider the ‘paranoid’ user who doesn’t trust the collecting organizations, in contrast to the users of policy based framework such as P3P [25]. Although such solution is robust against server attacks, the accessibility for service providers is much lower, leading to high communication costs when data needs to be queried or updated and placing constraints on how applications are developed and deployed. In contrast, data degradation doesn’t place these restrictions on applications, data can still be stored at the server side and by enforcing data degradation, donors are in control of the level of privacy risk they want to take in terms of retention periods.

Limited retention techniques have been proposed to ensure that data can no longer be subject to occasional disclosure. The limited retention principle is a key principle behind many privacy laws, and as such has been adopted in the work on Hippocratic databases [2]. Implementation frameworks behind such a system, based on *access control* mechanisms, including handling of generalized forms of the data, already exist [2], [4], [5]. However, such systems are still based on trust [1]; trust which cannot be put forever on a system. Even when secure access control techniques are used, a database administrator can be or become malicious. Even if the chosen security regime can be proven successful now, it might be not in the future [16].

In addition to access control, security measures for protecting a database server, such as data encryption, firewalls, and intrusion detection systems can be used. Those techniques make attacks more difficult without completely preventing them. Intrusion detection systems [7] are especially useful against repetitive attacks such as spying on a database, although it is still hard to find a good balance between false negative and false positive detections. However, used in addition to data degradation, intrusion detection systems would make it very hard for even a determined attacker to obtain a large consecutive history of accurate data.

Anonymization of data might be a solution to prevent disclosure of privacy sensitive-data. In fact, major companies such as Google already state they will adopt anonymization as a measure to improve privacy protection [13]. *k*-Anonymity [24] is based on the idea of masking (parts of) the (quasi) identifier of a partly privacy-sensitive tuple, such that the sensitive part of the tuple will be hidden between $k - 1$ potential identifier candidates within the same dataset. The work on *l*-diversity [20] goes a step further by taking background knowledge into account, enforcing enough diversity between the privacy sensitive attributes. Although data degradation aims not on generalizing identities, the techniques used to generalize the data are comparable.

Usually, anonymization is applied to large datasets at once, making sure that for each tuple, the tuple shares the same identifier with $k-1$ others. In practice this could result in a strictly *k*-anonymous database at the cost of losing much usability. Although Byun et al provided a technique to update anonymized databases [6], each time new data arrives, the database has to be sanitized into a *k*-anonymous state again, making it hard to obtain a clear view of the database from an application perspective, since old tuples might be sanitized at unpredictable times. Besides, correctly anonymizing the data is a hard problem [21]. To illustrate, a good example of incorrect and insufficient use of anonymization has been given when American Online decided to put a large set of search queries online [18]. AOL anonymized the IP addresses of the computers from which the queries were issued, which was not enough to prevent attackers from inferring many privacy sensitive facts. Data degradation does not suffer from such kind of vulnerabilities, because the facts themselves are degraded, not the identities.

Data degradation can be complementary to all discussed techniques. Firstly, by limiting the impact of inevitable privacy breaches, data degradation is complementary to access control, since data which has been subject to degradation either has a lower level of accuracy and thus sensitivity, or has already been removed from the system. Moreover, although only on temporary basis, accurate data can still be protected against regular attacks with the use of access control techniques. Secondly, anonymization is good practice when datasets have to be made public without revealing the identity of the users; for example, when used for disclosing datasets for research purposes, and therefore it can be a complementary technique to data degradation. Data degradation is particularly useful

when data needs to be accurate for some time, but where the details are less important when the data gets old. Moreover, within our degradation model, the identifier of the user can be kept intact; hence, user-oriented services can still exploit the information to the benefit of the user.

VII. FUTURE WORK AND CONCLUSION

In this paper, we presented a new framework to be able to reason about limited retention, and more specifically data degradation. We showed that using a function capturing the *worth* of storing data for the service provider, and a function for the *risk* involved for storing this data, we are able to find an optimal retention period such that the common interest for both service provider and user is respected.

The increase of common interest which can be achieved depends on multiple factors. When the decrease of worth is much higher than the increase in privacy, it is good to apply data degradation, even if this means that the data will never be stored accurately. In our analysis, we showed cases where it is useful to progressively degrade the data from accurate states to generalized states until final destruction of the data. We hope that our findings will lead to a break with a tradition where service providers collect everything they can, and store it as long as they can. This paper is a first attempt to achieve that goal. However, there is much future work to do:

- Firstly, we have to investigate the actual privacy guarantees of data degradation and its effect on the risk functions. How much ‘privacy’ can be provided by generalizing the data is an important question in order to correctly define the risk functions. Correctly defining privacy will always be subject of discussion, mainly because of its subjective nature. In that light, defining privacy as function based on risk, and to relate this risk to the amount of stored data is a promising first step.
- We introduced a model in which we only take retention periods of data into account. Indeed, both privacy and worth function can depend on various parameters other than retention periods. *Users* can choose different ‘risk profiles’ depending on the nature of the data items, and *service providers* can attach a lower worth to specific data based on the user, location, time of the day, *et cetera*.
- An important question is *if* and *how* service providers will be able to express their worth functions. To put our framework into practice, it is necessary to provide tools which enable service providers to give transparency about their need to collect personal data.

To conclude: our framework makes that it is indeed possible to reason about retention periods such that not only one but both parties will be satisfied. Together with data degradation, we presented a promising new approach to close the huge gap between the enormous amount of collection and storage of personal data, and the risk users have to take to be able to profit from all the new and exciting services offered to us today and in the future.

REFERENCES

- [1] G. Aggarwal, M. Bawa, P. Ganesan, H. Garcia-Molina, K. Kenthapadi, N. Mishra, R. Motwani, U. Srivastava, D. Thomas, J. Widom, and Y. Xu, “Vision Paper: Enabling Privacy for the Paranoids,” in *VLDB*, 2004, pp. 708–719.
- [2] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, “Hippocratic databases,” in *VLDB*, 2002, pp. 143–154.
- [3] N. L. G. Ancaix, L. Bouganim, H. J. W. van Heerde, P. Pucheral, and P. M. G. Apers, “The life-cycle policy model,” INRIA, Rocquencourt, France, Research report RR-6577, July 2008.
- [4] E. Bertino, J. Byun, and N. Li, “Privacy-preserving database systems,” *Lecture Notes in Computer Science*, vol. 3655, pp. 178–206, 2005.
- [5] J.-W. Byun and E. Bertino, “Micro-views, or on how to protect privacy while enhancing data usability: concepts and challenges,” *SIGMOD Rec.*, vol. 35, no. 1, pp. 9–13, 2006.
- [6] J.-W. Byun, Y. Sohn, E. Bertino, and N. Li, “Secure anonymization for incremental datasets,” in *Secure Data Management*, 2006, pp. 48–63.
- [7] H. Cavusoglu, B. Mishra, and S. Raghunathan, “The value of intrusion detection systems in information technology security architecture,” *Info. Sys. Research*, vol. 16, no. 1, pp. 28–46, 2005.
- [8] CNN.com, “Hackers stole data on pentagon’s newest fighter jet,” <http://www.cnn.com/2009/US/04/21/pentagon.hacked/>, April 2009.
- [9] G. Conti, *Googling Security: How Much Does Google Know About You?* Addison-Wesley Professional, 2008.
- [10] A. Cooper, “A survey of query log privacy-enhancing techniques from a policy perspective,” *ACM Trans. Web*, vol. 2, no. 4, pp. 1–27, 2008.
- [11] EDRI, “Uk government loses personal data on 25 million citizens,” <http://www.edri.org/edrigram/number5.22/personal-data-lost-uk>, November 2007.
- [12] Y. Finance, “Key statistics for google inc.” <http://finance.yahoo.com/qs?s=GOOG>, April 2009.
- [13] P. Fleischer and N. Wong, “Taking steps to further improve our privacy practices,” <http://googleblog.blogspot.com/2007/03/taking-steps-to-further-improve-our.html>, March 2007.
- [14] E. Gemeenschap, “Richtlijn 95/46/eg van het europees parlement en de raad,” http://ec.europa.eu/justice_home/fsj/privacy/docs/95-46-ce/dir1995-46_part1_nl.pdf, 1995.
- [15] G. Gordon and N. Willox, *Identity Fraud: a Critical National and Global Threat*. Economic Crime Institute, 2003.
- [16] L. Gordon, M. Loeb, W. Lucyshyn, and R. Richardson, *2005 CSI/FBI computer crime and security survey*. Computer Security Institute, 2005.
- [17] R. J. Hilderman, H. J. Hamilton, and N. Cercone, “Data mining in large databases using domain generalization graphs,” *J. Intell. Inf. Syst.*, vol. 13, no. 3, pp. 195–234, 1999.
- [18] D. Hillyard and M. Gauen, “Issues around the protection or revelation of personal information,” *Knowledge, Technology, and Policy*, vol. 20, no. 2, pp. 121–124, 2007.
- [19] X. Jiang, J. I. Hong, and J. A. Landay, “Approximate information flows: Socially-based modeling of privacy in ubiquitous computing,” in *UbiComp '02*. London, UK: Springer-Verlag, 2002, pp. 176–193.
- [20] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkatasubramanian, “L-diversity: Privacy beyond k-anonymity,” in *ICDE*, Washington, DC, USA, 2006, p. 24.
- [21] A. Meyerson and R. Williams, “On the complexity of optimal k-anonymity,” in *PODS '04*. New York, NY, USA: ACM Press, 2004, pp. 223–228.
- [22] G. Miklau, B. N. Levine, and P. Stahlberg, “Securing history: Privacy and accountability in database systems,” in *CIDR*, 2007, pp. 387–396.
- [23] T. W. Post, “Consultant breached fbi’s computers,” http://www.washingtonpost.com/wpdyn/content/article/2006/07/05/AR2006070501489_pf.html, July 2007.
- [24] L. Sweeney, “k-anonymity: A model for protecting privacy,” *International Journal on Uncertainty Fuzziness and Knowledge-based Systems*, pp. 557–570, 2002.
- [25] W3C, “Platform for privacy preferences (P3P) project,” <http://www.w3.org/P3P/>, June 2005.
- [26] C. Valli and A. Woodward, “Oops they did it again: The 2007 australian study of remnant data contained on 2nd hand hard disks,” in *Proceedings of the 5th Australian Digital Forensics Conference*, 2007.
- [27] VNUNet.com, “T-mobile loses 17 million customer details,” <http://www.vnunet.com/vnunet/news/2227691/loss-headaches-mobile>, October 2008.