A Comprehensive View on Quantity Based Aggregation for Cadastral Databases

Firas Al Khalil
Governance, Risk, and Compliance Technology Center
University College Cork
13 South Mall, Cork, Ireland
`firas.alkhalil@ucc.ie`

Alban Gabillon *and* Patrick Capolsini
Laboratoire GePaSUD
Université de la Polynésie française,
BP 6570 Faa'a Aéroport, French Polynesia
`{alban.gabillon,patrick.capolsini}@upf.pf`

Abstract. Quantity Based Aggregation (QBA) control is a subject that is closely related to inference control in databases. The goal is to enforce k out of n disclosure control. In this paper we work on QBA problems in the context of cadastral databases: how to prevent a user from knowing 1) the owners of all parcels in a region, and 2) all parcels belonging to the same owner. This work combines and extends our previous work on the subject [AGC13; AGC14a; AGC14b]. We overview the legislative context surrounding cadastral databases. We give important definitions related to the QBA concept. We present a complete model for QBA control in cadastral databases. We show how to implement the security policy efficiently, and we present our prototype of secure cadastral databases with some performance evaluations.

Keywords: Security, Database, Access Control, Inference, Aggregation.

Preprint of the version that appeared in the *Journal of Information Security and Applications* on the 1^{st} of March, 2017.
DOI: 10.1016/j.jisa.2016.11.007

Contents

1 Introduction

The inference problem [JM95; FJ02; Dwy09; Fri+11] in databases [DH96; YL98; CW05; Sta03; CC08; TFE10], privacy preserving data publishing [Swe02; Mac+07; LLV07] and mining [Dwo06; Dua09] (and other domains [YTM07; SGZ07; AC09; DBS09]) has been heavily studied in the last couple of decades. The inference problem arises whenever (partial or complete) knowledge about some classified information can be derived (inferred, deduced) using unclassified information. In this work we address a problem that is very close to the inference problem and usually discussed with it: the aggregation problem. Actually, we are interested in a very special type of aggregation problems, called quantity-based aggregation problems (QBA henceforth).

QBA problems were, most probably, distinguished from inference and other aggregation problems for the first time in the work of Hinke [Hin88], under the name "cardinality aggregation". Lunt [Lun89] analyzed inference and aggregation problems and showed the difference between them. She coined the term quantity-based aggregation, and she gave the following example to illustrate QBA: suppose that there is a phonebook of n phone entries, where a user has the right to know k entries—at most—out of n; the goal of QBA control is to enforce this "k out of n" disclosure control.

The abovementioned example is known in the literature as the NSA (National Security Agency) or the SGA (Secretive Government Agency) phonebook problem. After reviewing the literature, it seems like the problem has not been addressed fully. The only work we are aware of, treating QBA directly, is that of Motro, Marks and Jajodia [MMJ94; MMJ96] two decades ago. In a previous paper [AGC13] we addressed two QBA problems in the context of cadastral databases. In fact, we proposed a solution to enforce the following security policy: in the cadastral database, any user has the right to know the owner's name of any given parcel. However, this permission is constrained with the following prohibitions that represent two QBA problems: the user is forbidden to know:

Pr$_1$: the list of all parcels in a region,

Pr$_2$: the list of all parcels belonging to the same family.

In [AGC13] we presented our model and an implementation based on graphs. However, this implementation was inefficient (for reasons we present in Section 9.1). In [AGC14a] we proposed an alternative implementation (using the relational model) and we showed empirically that the execution time of our QBA control algorithm grows linearly with respect to the number of users of the database; we tackled Pr$_1$ only, and presented our prototype demonstrating this prohibition on the cadastral database of the island of Maupiti, a test database provided to us by the real-estate service of French Polynesia. This work aims to synthesize and extend all of our previous work on QBA in cadastral databases. Moreover, we include results of experiments comparing the basic model [AGC13; AGC14a] to the extension [AGC14b] that was developed to improve the availability of the data while preserving their confidentiality.

This paper is organized as follows: Section 2 presents the legislative context, discussing current policies regarding online publication of cadastral data in different countries including French Polynesia. Section 3 gives some preliminary definitions that help identifying different types of aggregation problems, and distinguishing them from inference problems. Section 4 gives details about QBA problems in the cadastral application: the security policy and some aggregation properties that should be enforced to implement both Pr_1 and Pr_2. Section 5 talks about additional aspects that should be taken into account while enforcing QBA: database updates, resetting access, and inference channels that could arise from QBA enforcement itself. Section 6 talks about some recommendations that should be considered for the choice of different parameters of our model. Section 7 talks about the application of QBA control to the French Polynesian cadaster, discussing the desired workflow and authentication. Section 8 presents the prototype developed to implement Pr_1. Section 9 presents performance benchmarks for the enforcement of Pr_1 comparing the base model [AGC13; AGC14a] with the new one [AGC14b]. Section 10 gives a review of related work in both inference control and QBA control. Finally, Section 11 concludes this paper.

2 Legislative context

After investigating the state of some online cadastral applications, we will give a couple of examples from different countries reflecting the legal point of view on the publication of parcel ownership information. We will also explain the French point of view on the subject and the case of French Polynesia motivating this work.

Access to the Spanish [CV13] cadaster is provided through a mapping interface built with Google Maps. Parcel ownership information is considered sensitive and it is not available to the public[1]. Land owners form a different level of users (more privileged than the public) and they are granted access to all information related to their own properties if they provide a valid X509 certificate associated with their national electronic ID.

Similarly, the Belgian cadaster is available online for the public [2], and ownership information is considered sensitive, thus prohibited. Using their national electronic ID, authenticated users can access through another website[3] to information related to their own parcels only.

In Croatia, parcel ownership information is public. Users can access the online website[4] where they can submit a query on any parcel and get a list of information related to the parcel, including land ownership. Queries are submitted by selecting the desired department, office and parcel ID or

[1] http://www.maps.data-spain.com/cadastral/

[2] http://ccff02.minfin.fgov.be/cadgisweb/

[3] https://eservices.minfin.fgov.be/portal/fr/public/citizen/welcome

[4] http://www.katastar.hr/

deed ID (using simple rudimentary lists). The query interface is protected against repeated automatic querying/scrapping/crawling.

Similarly, the state of Montana, US, considers land ownership as public information and they provide the cadaster for online browsing through a mapping interface[5]. Access to cadastral data in the US depends on state-level legislation.

Canada publishes its cadaster freely [6]. No ownership information is present, but all parcels can be downloaded as vector data (shapefiles) from an FTP site, after agreeing on a user-license agreement.

In France, the cadaster is available through a mapping interface[7], however, only land boundaries are available to the public. This is due to the CNIL [8] recommendation [09] where it is stated that [9] *"the diffusion of any identifying information (directly or indirectly) on interactive terminals or public websites entails the risk of using this information for other purposes, including commercial, without the concerned people's consent."*

However, the Cada [10] indicates that *"punctual demands"* of cadastral excerpts are allowed [13]. Furthermore, cadastral excerpts may contain the name of land owners, but no other identifying information such as their national ID or their address. The frequency of demands and the number of parcels requested should be analyzed to ensure that these demands do not infringe the principle of free communication of cadastral documents. There is no clear definition of *"punctual demands"* and it is subject to various interpretations, therefore the Cada recommends a restrictive interpretation of the term.

French Polynesia is an overseas territory of France, where the recommendations of the CNIL and Cada are applicable. Currently, the punctuality of demands issued by citizens is ensured by employees of the real-estate service of French Polynesia when they are physically present at their desks (which is also the current situation in France). The work presented here is a requirement of the IT service of French Polynesia expressing their interpretation of the recommendations of both CNIL and Cada in order to provide the same facilities offered by the real estate service through the internet: a user should have access to the ownership information of any parcel, at random, but s/he is not allowed to exploit the service for commercial ends (or social, etc.) This interpretation is the foundation of prohibitions Pr_1 and Pr_2 presented in detail in Section 4.1.

[5] http://svc.mt.gov/msl/mtcadastral/

[6] http://clss.nrcan.gc.ca/cadastraldata-donneescadastrales-eng.php

[7] http://www.geoportail.gouv.fr/

[8] *Commission Nationale de l'Informatique et des Libertés*. An independent administrative authority whose mission is to ensure that information technology is at the service of citizens and does not undermine human identity, rights, private life, or individual and public liberties.

[9] Translated from its original language, French, by the authors.

[10] *Commission d'accès aux documents administratifs*. An independent administrative authority responsible for ensuring freedom of access to administrative documents.

3 Definitions

In this section we give a set of definitions (relying on earlier work described in Section 10) that helps identifying inference, aggregation and QBA problems:

Definition 1 *[Inference problem] The inference problem arises whenever a collection of information can be used to derive (infer, deduce) partial or complete knowledge about information stored in the database and classified higher than the classification of each subset of the collection. This collection forms an inference channel. Inference control is a mechanism used to eliminate inference channels and prevent users from performing inferences.*

To illustrate an inference problem, let us consider the phonebook example. A phonebook is represented by the relation `PHONEBOOK (NAME, TEL, DEPT)` where the classifications of `NAME` and `TEL` (say, `UNCLASSIFIED`) are lower than that of `DEPT` (say, `CONFIDENTIAL`). A user with an `UNCLASSIFIED` clearance can access both `NAME` and `TEL`, and naturally `DEPT` is prohibited. However, if we consider that `TEL` depends on `DEPT` (e.g. one telephone per department, or numbers of the same department have the same suffix, etc.), then a user can infer, using `NAME + TEL`, to which department a given employee is affiliated, or even the list of employees who work in the same department.

Notice that the definition of the inference problem does not specify the source(s) of information in the collection. They could be partially derived from the database as in the inference from external knowledge, where a user combines his *a priori* knowledge with partial knowledge acquired from objects (that s/he has the appropriate clearance to read) of the database to conduct an inference, hence deduce sensitive information.

Definition 2 *[General Aggregation problem] The general aggregation problem arises whenever the classification of a set S of k items in the database, is higher than the classification of each subset of S. Aggregation control is a mechanism used to prevent users from performing unauthorized aggregations.*

To illustrate general aggregation problems, let us consider 3 phone entries: A, B and C labeled `SECRET` (each). The aggregation of A and B is labeled `TOP SECRET` while the aggregation of A and C is labeled `SECRET`. A user with a `SECRET` clearance level should not access the aggregate $A + B$ but s/he can access $A + C$.

Definition 3 *[QBA problem] The QBA problem arises whenever the classification of more than k out of n items of a set S in the database is higher than the classification of that of k or less items. QBA control is a mechanism used to prevent users from aggregating more than k out of n items.*

Indeed, a QBA problem arises when a user has the right to query any subset of the phonebook relation (of size n), under the condition that the size of the queried subset does not exceed k (where

$k < n$). The key difference between Definitions 2 and 3 is that the former does not take into account the quantity of aggregated entries. If we apply these definitions to works found in the literature, we find that inferences from dependencies on schema and data [e.g. YL98; YTM07; CC08], or inferences from external knowledge [e.g. SGZ07] or denial of access [e.g. SJ92] fall under Definition 1. The Chinese-Wall policy [e.g. BN; Mea90] falls under Definition 2 where the role of the policy is preventing a user from aggregating data from the same conflict of interest class, while the phonebook problem as presented by Hinke [Hin88] and Lunt [Lun89], and both aggregation problems of the cadastral database—that we will define in the next section—fall under Definition 3.

4 QBA Problems in the Cadastral Database

In the following, we will define the security policy and how to enforce it. Section 4.1 introduces the prohibitions that we need to enforce.

In Section 4.2, we will talk about the enforcement of the first prohibition, Pr_1. We will introduce the notions of a zone, dominant zone, and z-region . Then we will talk about collusion resistance, which is a desired property of QBA control, intended to prevent multiple malicious users from collaborating and circumventing the security policy. We will talk about x-collusion resistance, a scheme set up to prevent x users from colluding on a dominant zone, and (x, y, z)-collusion resistance, a less restrictive scheme set up to prevent x users from colluding on y z-regions.

Finally, in Section 4.3, we will show how we can enforce the second prohibition, Pr_2, by adapting the ideas developed in the previous section, due to the similarity of Pr_1 and Pr_2.

4.1 Security Policy

A cadastral database is a geographical database used to manage parcels of a country, state, municipality, etc. Parcels are pieces of land represented in the database by geo-referenced polygons. In addition to their geometric representation, parcels are associated with information like mutation history, taxation, and most importantly ownership information. Currently, access to the cadastral database in French Polynesia is limited to employees of the real-estate service, notaries, and surveyors. The IT department of French Polynesia wishes to make this database available online and apply the following ***Security Policy***: citizens (parcel owners or not) can access ownership information of any parcel through a *“point-and-click*" mapping interface (similar to Google Maps or Bing Maps). However, this access is limited by the following prohibitions:

Pr_1: A user cannot get the list of all owners in a geographical region.

Pr_2: A user cannot get the list of all parcels belonging to the same legal entity (e.g. family).

Table 1: Table of Symbols

Symbol	Definition	
α	the estimated growth rate of a zone	
β	the estimated background knowledge of a user	
ρ_{o_1,o_2}	a social relation between the owners o_1, o_2	
d_i	the number of disclosed parcels in a zone Z_i;	$k_h < d_i < k_l$
${}_y d_p$	the number of dominant zones where a user reads more than k_l parcels in a z-region	p. 15
dist	the shortest distance between two nodes in a graph	
dist_{o_1,o_2}	the social distance between the owners o_1, o_2	
dist_{social}	the social distance between the owners of two parcels	equation 11, p. 15
$D(p)$	dominant zones of p	equation 3, p. 10
$\mathcal{D}(p)$	dominant zones containing p	equation 4, p. 10
$G(V, E)$	A graph G where $G(V)$ and $G(E)$ are the set of vertices and edges, respectively.	
k_h	the high, strict, threshold of disclosable parcels in a zone	for (x, y, z)-collusion resistance
k_i	the maximum number of disclosable parcels in a zone Z_i	
k_l	the low threshold of disclosable parcels in a zone.	for (x, y, z)-collusion resistance
m_p	the cardinality of a z-region $R(p, z)$	
n_i	the cardinality of a zone Z_i	
$N(p)$	open neighborhood of p	
$N(p, z)$	the neighbors of p of degree $\leqslant z$	equation 8, p. 13
$N_{social}(p, z)$	the neighbors of the owners of p of degree $\leqslant z$	equation 12, p. 16
o	an owner; $o \in O$	
$O(p)$	the set of owners of a parcel p	
p	a parcel; $p \in P$	
$R(p, z)$	the z-region of a parcel p	equation 9, p. 13
t	the number of parcels accessed since $T' > T$	
T	the global clock tick rate	
x	the number of colluders	
y	the number of zones in a z-region where a user is not a potential colluder	
z	the size of the z-region considered for collusion resistance	
Z_i	a zone	
$Z(p)$	zone of p; $Z(p) = N[p]$ the closed neighborhood of p	equation 1, p. 10
$\mathcal{Z}(p)$	zones containing p	equation 2, p. 10

It should be obvious by now that both Pr_1 and Pr_2 are two separate QBA problems. Indeed, we have the following analogies:

- The list of owners of a given region is analogous to a phonebook (Pr_1).
- The list of parcels of a given family is analogous to a phonebook (Pr_2).
- The association between a parcel and an owner is analogous to a phonebook entry.

In the following sections we will show how we enforce Pr_1 and Pr_2.

4.2 Enforcing Pr_1

The first challenge is to properly interpret the term *“region."* The obvious (and naive) solution is to consider that one administrative region is equivalent to a region. This is the static definition of a region, but it is inaccurate: *Which resolution is considered optimal? Is a municipality too big? Is a neighborhood too small? Should we mix resolutions?* If we choose, for the sake of argument, a neighborhood as our definition for a region, and we gave all users the right to access k out of n parcels for every neighborhood, then a malicious user can exploit the fact that regions are static, and attack a *“geographical space"* falling on the shared borders of two neighboring regions, thus knowing the owners of that *“geographical space"* that s/he considers a region.

One should understand that people’s perception of a region is dynamic itself. Moreover, every person has multiple definitions of regions, and they are all based on personal interest; it could be economical, social, or even contextual (e.g. a region that has been featured in the news). For example, Alice finds that the beach strip is interesting because she works at a construction agency seeking a spot for its new hotel; Bob finds a remote house on the hill interesting because he wants to buy it with the part of the hill facing the sea; Charlie is interested by the economical section of the city because he wants to invest in real-estate while on a tight budget; etc. In all of those cases, there is a non-negligible chance that the user’s region of interest is distributed between multiple connected static regions.

Therefore we need a definition of a *“region"* that overcomes both problems: it needs to be resolution independent and dynamic, elastic, so that it can adapt to the human’s perception of a region, regardless of the previously mentioned subjective interest.

In order to achieve such a definition, it is crucial to view the cadastral database as a planar graph. Let P be the set of parcels in the cadastral database and $\delta(.,.)$ a function that returns the minimal euclidean distance between two parcels. We create a graph $G(V, E)$ where $G(V) = P$, while $G(E)$ is defined as follows: two parcels are neighbors in the graph if they touch each other, or if they are separated by a maximal distance τ. Formally, $G(E) = \{(p, q) : p, q \in G(V), p \neq q, 0 \leqslant \delta(p, q) \leqslant \tau\}$ for a given $\tau \in \mathbb{R}_{\geqslant 0}$. We could select $\tau = 0$, i.e. only parcels touching each other, however parcels which are separated by thin boundaries, like rivers or roads, require a value of τ greater than 0 to be considered as neighbors. We consider that *"isolated"* parcels, i.e. parcels that do not have neighbors in the range τ, do not fall within the scope of Pr_1: access to these parcels is granted automatically.

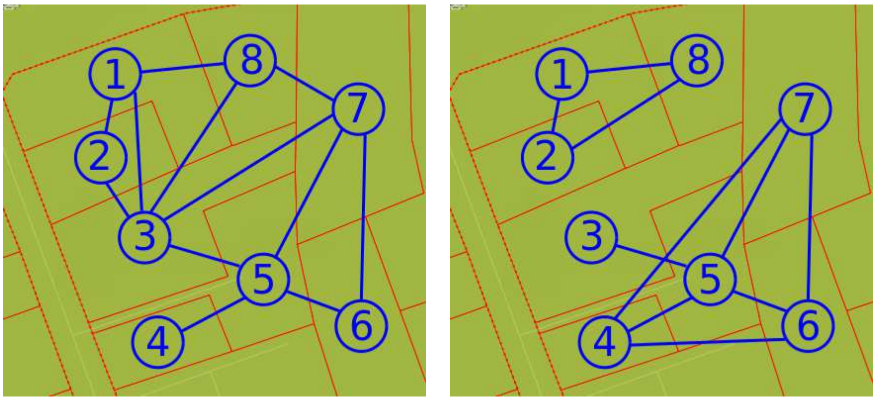
Definition 4 *[Zone] A zone of a parcel* p*,* $Z(p)$*, is the set of parcels formed by* p *itself and all of its neighbors. Formally:*

$$Z(p) = \{p\} \cup N(p) \tag{1}$$

where $N(.)$ is the *open neighborhood* of a vertex in a graph $G(V, E)$ defined as $N(v) = \{u : (u, v) \in E\}$. In graph theoretic terms, $Z(p)$ is the *closed neighborhood* of p, classically denoted by $N[p]$. A zone is the smallest region that could be modeled; every parcel belongs to its proper zone and the zone of every direct neighbor.

Figure 1a shows a graph for Pr_1 representing part of the cadastral database where parcel 1 touches $\{2, 3, 8\}$, parcel 4 touches $\{5\}$, parcel 7 touches $\{3, 5, 6, 8\}$, etc. Note that every parcel belongs to its proper zone and to every zone formed by every neighboring parcel. Formally, we denote $\mathcal{Z}(p)$ the set of zones containing a parcel p:

$$\mathcal{Z}(p) = \{Z(q) : q \in Z(p)\} \tag{2}$$



(a) A graph for Pr_1

(b) A graph for Pr_2

Figure 1: Different graphs for the same part of the cadastral database

Definition 5 *[Dominant Zone] A dominant zone of a parcel* p *is a zone containing* p *having the highest cardinality. A parcel can have multiple dominant zones. The set of dominant zones of* p*,* $D(p)$*, is defined as follows:*

$$D(p) = \{Z_i \in \mathcal{Z}(p) : |Z_i| = \max_{Z_j \in \mathcal{Z}(p)} (|Z_j|)\} \tag{3}$$

We denote $n_i = |Z_i|$ the cardinality of a dominant zone $Z_i \in D(p)$. Note that a parcel belongs to its proper dominant zones and some of the dominant zones of its neighbors. We denote $\mathcal{D}(p)$ the set of dominant zones containing p, and define it as follows:

$$\mathcal{D}(p) = \{Z_i \in D(q) : q \in Z(p), p \in Z_i\} \tag{4}$$

Let $\mathcal{D} = \{Z_i \in D(p) : p \in P\}$ be the set of all dominant zones. A user has the right to know the ownership of any parcel belonging to any dominant zone $Z_i \in \mathcal{D}$. The ***Aggregation Control Property*** is: for all dominant zones, the number of disclosed parcels for any user, k_i, should always be strictly lower than n_i. Formally:

$$\forall Z_i \in \mathcal{D}, 0 < k_i < n_i \tag{5}$$

Satisfying the *Aggregation Control Property*, namely preventing a user from accessing all parcels in a dominant zone, implies the satisfaction of the security policy and effectively preventing this user from acquiring the knowledge of all owners in any region of any size. Note that in a previous work [AGC14a] we defined the *Aggregation Control Property* on zones. Section 9.1 explains why we decided to define the *Aggregation Control Property* on dominant zones instead of zones.

Enforcing QBA control is simple: when a user requests a parcel p, the algorithm should make sure that the number of disclosed parcels k_i is strictly lower than n_i, $\forall Z_i \in \mathcal{D}(p)$. If this condition is satisfied, access is granted; otherwise, access is denied.

This is sufficient if we consider a single user accessing the cadastral database in isolation. For example, if two users are accessing a dominant zone where $k_i = n_i - 1$, none of them can get the ownership information of all parcels in that dominant zone, however, they could collaborate and combine their knowledge to bypass the limit k_i. Therefore $0 < k_i < n_i$ expressed in Equation 5 is a necessary but not sufficient condition in real-world applications where collaborating users form an actual threat to the security of the application.

This collaboration is called *“collusion."* The Merriam-Webster online dictionary defines collusion[11] as *“[a] secret agreement or cooperation especially for an illegal or deceitful purpose.”* In our context, the illegal or deceitful purpose is to access a complete dominant zone. Therefore a collusion happens when x users secretly agree or cooperate to access a given dominant zone. An important property that should be satisfied by QBA control is collusion resistance.

Definition 6 *[x-collusion] We say that x users collude to reconstruct all entries in a dominant zone if the union of accessed parcels by those x users covers the complete dominant zone.*

A QBA control mechanism is x-collusion resistant if x or fewer users cannot reconstruct a complete dominant zone. To achieve x-collusion resistance the ***Aggregation Control Property*** should be extended to:

$$\forall Z_i \in \mathcal{D}, k_i = \begin{cases} \lceil n_i/x \rceil - 1 & \text{if } n_i > x > 1 \\ 1 & \text{otherwise} \end{cases} \tag{6}$$

This way, x users are guaranteed to never collude and reconstruct a complete dominant zone even if those x users accessed disjoint subsets of Z_i.

[11] http://www.merriam-webster.com/dictionary/collusion

Now we should analyze x-collusion resistant QBA control. In fact, we should evaluate the effect of the variation in the size of dominant zones. Since we proposed a solution to achieve x-collusion resistance that relies on n_i, and n_i is variable due to the dynamic nature of the database (deletions, insertions, and divisions of existing parcels), then we should see which values would vary with respect to n_i.

We need to know if x is fixed or if it is a function of n_i. Let us consider the set of all $m = |\mathcal{D}|$ dominant zones under x-collusion resistance. If x has the same value for all dominant zones, then x or fewer users are guaranteed not to collude on all m dominant zones. If the security administrator sets for dominant zones $Z_1, Z_2 \dots Z_m$ different values $x_1, x_2 \dots x_m$ then a coalition of x_c users, where $\min(x_1, x_2 \dots x_m) < x_c < \max(x_1, x_2 \dots x_m)$ can collude to reconstruct all dominant zones with x_b-collusion resistance such as $x_b \leqslant x_c$. Therefore, we recommend setting a single value of x-collusion resistance to all zones.

Since n_i is the number of parcels in a dominant zone Z_i and x should be fixed, k_i, the number of parcels that can be disclosed, will vary with respect to n_i:

1. If k_i increases then users have access to more parcels. This means that users might collude to reconstruct the previous dominant zone Z_i before it expanded. From a security point of view this means that the previous zone has been *"declassified."* If we wish to avoid this situation then the only solution would be enforcing smaller values of k_i:

$$\forall Z_i \in \mathcal{D}, \alpha \in \mathbb{N}^+, k_i = \begin{cases} \lceil n_i/x \rceil - \alpha & \text{if } \alpha < \lceil n_i/x \rceil \\ 1 & \text{otherwise} \end{cases} \tag{7}$$

   With α carefully chosen, expanding dominant zones do not allow colluding users to reconstruct today information that was considered as sensitive yesterday.

2. If k_i decreases to a new value k_i', then users who have already accessed more than k_i' parcels might collude to reconstruct the new dominant zone Z_i'. Here also, computing smaller values of k_i (i.e. choosing a proper value for α) would eliminate this security threat.

The drawback of x-collusion resistance is that it assumes that all users are potential colluders on all dominant zones. In practice this assumption is somewhat too strong and may lead the QBA control mechanism to detect too many false positives. Another type of collusion resistance is needed where a user is assumed to be a potential colluder if his/her querying behavior is suspicious. This new type should take into account the main idea behind Pr_1, while relaxing the assumption on colluding users: recall that Pr_1 states that *"a user cannot get the list of all owners in a geographical region."* Therefore, a group of users should be considered as potential colluders if they are trying to attack a region, the more general concept of a zone.

The more general concept of a zone of a parcel p is the z-region of p, which is made of the zones of the neighbors of p of degree z. We will first define the *degree of neighborship* between two parcels, then we will define the z-region which uses the of *degree of neighborship*, and finally we will define the new collusion resistance which uses z-regions.

Definition 7 *[Degree of Neighborship] If the shortest path between two nodes p and q in a graph is $\mathrm{dist}(p, q) = z$, then it is said that p is the neighbor of degree z of q and vice versa. The set of neighbors of degree z of a node p, $N(p, z)$, is defined as follows:*

$$N(p, z) = \{q \in P : \mathrm{dist}(p, q) = z, z \in \mathbb{N}\} \tag{8}$$

Obviously, $N(p, 0) = \{p\}$, $N(p, 1) = N(p)$, and $Z(p) = N[p] = N(p, 0) \cup N(p, 1)$. Now that we have defined what we mean by *degree of neighborship*, we are ready to introduce z-regions.

Definition 8 *[z-region] The z-region of a parcel p, $R(p, z)$, is a set formed by the dominant zones of p and the dominant zones of neighbors of p of degree $\leqslant z$. Formally:*

$$R(p, z) = \{Z_i \in D(q) : q \in \bigcup_{0 \leqslant i \leqslant z} N(p, i)\} \tag{9}$$

The cardinality of a z-region $R(p, z)$ of a parcel p is $m_p = |R(p, z)|$. Now that we have defined z-regions, we can use it to define (x, y, z)-collusion resistance, a less restrictive collusion resistance than x-collusion resistance.

Definition 9 *[(x, y, z)-collusion] We say that x users collude to reconstruct y dominant zones in a z-region if the union of accessed parcels by those x users covers those y complete dominant zones.*

The idea behind (x, y, z)-collusion resistance is that as long as users cannot reconstruct more than y dominant zones in a given region $R(p, z)$ then they should not be considered as colluders. As soon as these users can reconstruct y dominant zones in $R(p, z)$ then the x-collusion resistance scheme should be applied on the remaining $m_p - y$ dominant zones.

To achieve (x, y, z)-collusion resistance, we should be able to partition the set of dominant zones of a z-region $R(p, z)$ into two distinctive sets:

1. The set of dominant zones where x-collusion resistance *is **not*** enforced, R', $|R'| = y < m_p$; i.e. for the first y dominant zones, we don't consider that a user is behaving suspiciously, thus we don't consider her/him as a potential colluder.

2. The set of dominant zones where x-collusion resistance *is* enforced according to equation 7, R'', $|R''| = m_p - y$; i.e. for the remaining $m_p - y$ dominant zones, we consider that a user is behaving suspiciously, thus we consider her/him as a potential colluder.

Formally, the ***Aggregation Control Property*** should be extended to:

$$\begin{gathered}\forall p \in P, \forall Z_i \in R(p,z), \alpha \in \mathbb{N}^+, \\ \exists \mathcal{R} = \{R', R''\} \text{ a partition of } R(p,z) \text{ where } |R'| \leqslant y, \text{ and} \\ k_i = \begin{cases} n_i - \alpha & \text{if } \alpha < n_i \text{ and } Z_i \in R' \\ \lceil n_i/x \rceil - \alpha & \text{if } \alpha < \lceil n_i/x \rceil \text{ and } Z_i \in R'' \\ 1 & \text{otherwise} \end{cases}\end{gathered} \tag{10}$$

It is useful to think about the threshold k of a dominant zone as taking two distinct forms: k_h (read `K HIGH`) and k_l (read `K LOW`), where $k_h > k_l$. The security administrator sets k_h according to the first (or third) case in equation 10, and k_l is determined by the required level of x-collusion resistance as defined in the second (or third) case in equation 10. Let d_i be the number of disclosed parcels in a dominant zone Z_i for a given user. All users have the right to access $k_l < d_i \leqslant k_h$ entries in y dominant zones, at most, in any $R(p,z)$, after which s/he is considered a potential colluder. For the remaining $m_p - y$ dominant zones s/he has the right to access k_l entries at most. The number of dominant zones where a user reads more than k_l parcels in a z-region is ${}_y d_p = |\{Z_i \in R(p,z) : k_{l_i} < d_i \leqslant k_{h_i}\}|$; ${}_y d_p$ is always less than or equal to y.

Algorithm 1 shows how to enforce QBA control with (x, y, z)-collusion resistance when a user requests a parcel p. The algorithm has 4 main steps:

1. Check if the user has already accessed p. If it is the case, then we return the requested owners and the process terminates (lines 1-2). If not, execute step 2.

2. Check if the disclosure of p would make the number of disclosed parcels of any dominant zone containing p exceed the maximal allowed limit k_h. If it is the case, then access is denied and the process terminates (lines 3-4). If not, execute step 3.

3. Check if the disclosure of p would make the number of dominant zones in all z-regions containing p, that exceed the lower limit k_l, stays less than the allowed limit y. If it is the case, access is denied and the process terminates (lines 5-13). If not, access is granted, and the final step is executed.

4. Put p in the user history, update all counters, and finally return the owners of p (lines 14-18).

Note that this algorithm performs a Breadth-First traversal (BFS) on line 7:

Algorithm 1: QBA Enforcement Algorithm

```
Input : p, y, z
Output: owners
owners = GetFromUserHistory(p)
if owners $\neq \emptyset$ then return owners
maxed = $\{Z_i \in \mathcal{D}(p) : d_i = k_{h_i}\}$
if maxed $\neq \emptyset$ then return $\emptyset$
potential = $\{q : q \in Z(p) \cap Z_i, Z_i \in \mathcal{D}(p), d_i = k_{l_i}\}$
for all q $\in$ potential do
    for all r $\in$ N(q,z) do
        if ${}_y d_r = y$ then
            Rollback All Modifications
            return $\emptyset$
        ${}_y d_r \leftarrow {}_y d_r + 1$
    end
end
PutInUserHistory(p)
for all $Z_i \in \mathcal{D}(p)$ do
    $d_i \leftarrow d_i + 1$
end
return O(p)
```

4.3 Enforcing Pr_2

The basic idea for enforcing Pr_2 is to use the scheme we developed for Pr_1 in the previous section by only modifying the definition of the graph. Let O be the set of all owners, and $O(p)$ the set of owners of a parcel p. We create a graph $G(V, E)$ where $G(V) = P$ as in Pr_1, while $G(E)$ is defined as follows: two parcels are considered neighbors in the graph if they belong to the same owner. Formally, $G(E) = \{(p, q) : p, q \in G(V), p \neq q, O(p) \cap O(q) \neq \emptyset\}$. In such a graph, vertices belonging to the same owner are all interconnected, forming a complete graph.

For instance, Figure 1b shows a graph for Pr_2 representing part of a cadastral database (the same part as in Figure 1a), where parcels $\{1, 2, 8\}$ are owned by Joe, $\{4, 5, 6, 7\}$ are owned by Elissa, and $\{3, 5\}$ are owned by Lucy. Notice that parcel 5 has two owners, namely Elissa and Lucy.

It is clear that a zone, as presented in Definition 4, depends only on the graph: for Pr_1, the zone of a parcel p is p and the set of parcels touching, or located at a given distance from p; for Pr_2, the zone of a parcel p is p and the set of parcels owned by the same owner. The dominant zone of a parcel p is the zone containing p having the highest cardinality, i.e. the owner that owns the highest number of parcels.

For x-collusion resistance to hold, the ***Aggregation Control Property*** should conform with equations 7. To achieve (x, y, z)-collusion resistance, we define a distance function dist_{social} as follows:

$$\text{dist}_{social} : V^2 \rightarrow \mathbb{N} \tag{11}$$

dist_{social} returns the smallest social distance between the owners of 2 parcels according to some social relationship (e.g. father, grand-child, etc.)

$\mathtt{dist}_{social}$ is intentionally loosely defined because it depends on the actual social relationships present in the database. For example, if we have a database associating for each tuple of owners (o_1, o_2) a social relation $\rho_{o1,o2} \in \{\mathtt{parentChild}\}$, where $\mathtt{parentChild}$ is transitive and commutative, then how do we decide on $\mathtt{dist}_{o_1,o_2}$, the distance that separates two owners? We have two options:

1. $\mathtt{dist}_{o_1,o_2} \equiv d$, the *"classical"* distance function in a graph, if the transitivity and commutativity properties of $\mathtt{parentChild}$ are not important.

2. $\mathtt{dist}_{o_1,o_2} \equiv \mathtt{parentChild}$ if the transitivity and commutativity properties of $\mathtt{parentChild}$ are important.

Notice that the first definition of $\mathtt{dist}_{o_1,o_2}$ is less restrictive that the second definition. Now that we know what are our options when it comes to measuring the social distance separating two *owners*, we need to think about $\mathtt{dist}_{social}$, the social distance that separates the owners of two *parcels*. Obviously, a parcel p can have multiple owners. So the question is how do we determine the distance separating two sets of owners O_1 and O_2, given $\mathtt{dist}_{o_i,o_j}$ for all $o_i, o_j \in O_1 \cup O_2$? We can select the smallest $\mathtt{dist}_{o_i,o_j}$. Another option would be the median distance. But of course these are not the only options.

In a more realistic scenario, there would be a bigger and more accurate set of relationships (e.g. $\{\mathtt{child}, \mathtt{parent}, \mathtt{married}, \mathtt{employer}, \mathtt{classMate}, \mathtt{aquaintance}\}$) each one of them having its own properties, in addition to a set of rules governing them (as in the case of ontologies), which makes the choice of $\mathtt{dist}_{social}$ highly dependent on the analysis of the knowledge base.

$\mathtt{dist}_{social}$ is essential to the definition of the *degree of neighborship* in Pr_2:

Definition 10 *[Degree of Social Neighborship] If the shortest social path between two nodes p and q in a graph is z, then it is said that p is the neighbor of degree z of q and vice versa. The set of neighbors of degree z of a node p, $N_{social}(p, z)$, is defined as follows:*

$$N_{social}(p, z) = \{q \in P : \mathtt{dist}_{social}(p, q) = z, z \in \mathbb{N}\} \tag{12}$$

Which means that in order to enforce (x, y, z)-collusion resistance all we need to do is to substitute $N(p, z)$ by $N_{social}(p, z)$ in the definition of a z-region of Equation 9 as follows:

$$R(p, z) = \{Z_i \in D(q) : q \in \bigcup_{0 \leqslant i \leqslant z} N_{social}(p, i)\} \tag{13}$$

Similarly to Pr_1, and in order to support (x, y, z)-collusion resistance, k should be split into 2 variables: k_h (read `K HIGH`) and k_l (read `K LOW`). Note that we consider that *"isolated"* parcels, i.e. a parcel belonging to a single owner who himself does not own other parcels, do not fall within the scope of Pr_2: access to these parcels is granted automatically.

5 Additional Aspects

In this section, we will discuss aspects of the enforcement of QBA that should be taken into consideration, namely updates, resetting access, and potential inference channels.

In Section 5.1, we will discuss 2 kinds of updates, joining and splitting, that can be performed on cadastral data, which can lead to potential security issues. We will also show how to handle these operations.

In Section 5.2, we will tackle the issue of access reset: in fact, cadastral data do not mutate that often, which means that if a user gets blocked from accessing a region, s/he will be blocked indefinitely, which can be inconvenient. We present a scheme to reset access history over time to prevent such bottlenecks.

And finally, in Section 5.3, we discuss potential inference channels that could arise from the enforcement of QBA itself.

5.1 Handling Updates

Four cadastral operations (called mutations) are performed daily on the database:

1. Buy and Sell: a parcel's ownership is transferred from its original owner to a new person, affecting the topology of the graph in Pr_2 only;
2. Merge and Split: two or more parcels are merged (split) into a single parcel (multiple parcels), affecting the topology of the graph in Pr_1 and Pr_2.

In the following, we will show how to handle mutations for Pr_1 and Pr_2 (Sections 5.1.1 and 5.1.2 respectively).

5.1.1 Mutations in Pr_1

The first operation we want to address is a Buy/Sell of a parcel p. It is an operation that changes the owner of p, therefore all users should have the opportunity of accessing this parcel and knowing its new owner. Intuitively, the solution should be the erasure of all access history to guarantee equal access to all users. But let us take a look at the options we have:

1. Erase user access history of p, in this case:
   (a) Users who have not queried p before a Buy/Sell will not be affected, whether they were blocked on any dominant zone containing p or not.
   (b) Users who queried p before a Buy/Sell have now the right to choose to re-query p or another parcel in the same zone.

i) If they were not blocked on any dominant zone containing p before the Buy/Sell, then there is nothing to worry about, however

ii) If they were blocked on any dominant zone containing p, this erasure might give them access to information that was previously *"classified."*

2. Keep user access history of p: nothing changes for any user, whether s/he queried p s/he has been blocked on a zone containing p (before the Buy/Sell).

Notice that erasing access history does not only raise a security issue (Point 1(b)ii above), but it is computationally costly too, because we need to remove access history from all $Z_i \in D(p)$ and then compute the new value of ${}_y d_p$ for the new z-region of p, for every user. Therefore the best strategy is to keep access history of a parcel that have been bought/sold.

Now we should examine merging and splitting. Merging requires all parcels involved in a merger to form a continuous geographical region: every parcel should touch at least one other parcel. Splitting does not have this requirement. However in both cases, dominant zones that contained old parcels will change, particularly in size (bigger, smaller or keep their size; we should remind you that all users should have equal right of access to the new information after merger/split) which affects k_l, k_h, d_i, and ${}_y d_p$.

We can merge/split access history, but this is problematic when there are dominant zones, post-merge and post-split, that get smaller in size: d_i and ${}_y d_p$ might exceed allowed limits (namely k_h and y, respectively), which requires special handling, per-user; in other words, not all users will have equal rights of access. Erasing access history, i.e. removing access history of merged and split parcels, is more convenient and does not induce that issue; therefore we argue for it. It is worth mentioning that merging/splitting access history and erasing it induce another security issue in one special case: when zones, post-merge and post-split, become bigger, users might gain access to information that was previously *"classified."* The security administrator can anticipate this issue by choosing a proper value for α (equation 10) by estimating the average change in the size of a dominant zone in her/his cadastral database (which obviously requires running simulations on historical records). For example, if $\alpha = 1$, the administrator is not expecting the region to grow at all; if $\alpha = 2$, the administrator is taking into account that this region might grow in size sometimes in the near future (it will get bigger by 1 parcel); if $\alpha = 3$, the administrator is anticipating even more change in the near future (2 parcels to get added to the region); etc.

Algorithm 2 shows how the update algorithm should work: the idea behind it is that we need to prune all user history, old parcels, and old dominant zones from our database, then we need to recompute the new dominant zones and their k_l and k_h counters. Once we have the new dominant zones, we can compute the counters for the z-regions that exist, namely d_i and ${}_y d_p$ for every user.

In more detail, Algorithm 2 takes 2 lists: `oldParcels` and `newParcels`. For mergers, new-

`Parcels` is a single item; for splits, `oldParcels` is a single item. We chose to write down a single algorithm for both for brevity. The algorithm works as follows: we first delete all user access history pertaining to `oldParcels`, then we remove `oldParcels` and their dominant zones (lines 2 – 3). Next we insert the `newParcels` and connect them to their neighbors (line 4). Now that we have pruned the user history and added the new parcels, we should re-compute the dominant zones. We first track the set of parcels affected by this merger/split, `affectedParcels`, which is naturally the new parcels and their neighbors (line 5). We iterate through the set of `affectedParcels`, we compute the new dominant zones of every parcel (line 8), during which we compute k_{l_i} and k_{h_i} of the dominant zones. The dominant zones affected by the merger/split are computed while computing the new dominant zones (line 10), and produce a set of `affectedDominantZones`. Now we are ready to recompute the rest of the counters which are dependent on z-regions: we iterate through the set of `affectedDominantZones`, and for every user, we perform a breadth-first traversal (with depth z) and compute d_i and ${}_y d_p$ (line 14).

Algorithm 2: Update Algorithm for Merge/Split (Pr$_1$)

```
   Input: oldParcels, newParcels
 1 neighbors = GetNeighborsOf(oldParcels)
 2 DeleteAllUserHistoryOf(oldParcels)
 3 DeleteParcelsAndDominantZonesOf(oldParcels)
 4 InsertNewParcels(newParcels)
 5 affectedParcels = newParcels ∪ neighbors
 6 affectedDominantZones = ∅
 7 for all  p ∈ affectedParcels do
 8   ComputeNewDominantZonesOf(p)
 9   dz = GetListOfDominantZones(p)
10   affectedDominantZones = affectedDominantZones ∪ dz
11 end
12 for all  z ∈ affectedDominantZones do
13   for all  u ∈ UsersOf(z) do
14     RecomputeCounters(z,u)
15   end
16 end
```

5.1.2 Mutations in Pr$_2$

Mutation operations affect Pr$_2$ very differently. In fact buying, selling, merging and splitting are all equivalent. Ownership of a parcel will be transferred from one person to itself (i.e. in the case where the resulting owners of the merger/split are the same owners of old parcels) or to other owners, and in both cases, zones affected by these mutations will get bigger, smaller or keep their sizes. And in all of those cases, the best solution is to erase parcel access history, for the same reasons presented for merge/split for Pr$_1$ in Section 5.1.1.

5.2 Resetting Access

Another important problem of QBA enforcement is the fact that after a given period of time, when users consume k entries from a zone, they become blocked on those k entries and the database itself becomes of no useful value in any future interaction [12]. Therefore, an appropriate resetting should be done so that users can still use the cadastral application. For instance, if a user was blocked in a zone Z_i on k_i out of n_i parcels and 2 or 3 years later, s/he decides to come back and query some parcels in the same zone s/he will still be blocked although a long period of time has passed and this user has a legitimate need of the requested information.

By removing previously accessed parcels from the user's history, the user gains the ability to query other parcels in zones that would normally be blocked.

The simplest resetting scheme would use a global timer that ticks every T units of time, and removes t parcel from the history of every user if they were accessed more than T units ago. More specifically, on every timer tick, for every user, we collect the parcel s/he accessed at least T units of time ago. This list is sorted from oldest to newest. We keep from that list t entries at most. Now we iterate through this list of t parcels, and on each iteration we remove the parcel p from the user's history and we decrement $d_i, \forall Z_i \in D(p)$. If the new value of d_i is less than k_{l_i}, it means that the user should no longer be considered as a potential colluder; i.e. ${}_y d_p$ should be decremented. If the new value of d_i did *not* get lower than k_{l_i}, the previous step is skipped, and the iteration continues until we are done with all t elements of the collected list.

Note that we are proposing a gradual resetting scheme where, eventually, the possibility of accessing any parcel in the database can be obtained given that the user is rarely accessing the database (i.e. resetting all access to all parcels). The choice of the value of the threshold T is of utmost importance from a security perspective. Big values (e.g. 3 years) might put in question the utility of the resetting scheme, and small values (e.g. 1 hour) put in question the utility of the whole QBA control mechanism. The security administrator should take into account how frequently the cadastral database is accessed. Big values of T lead to more data confidentiality, while small values lead to greater data availability. The same argument, although reversed, goes for t, the number of parcels to be released on every timer tick: big values of t lead to more data availability, while small values lead to more confidentiality.

Other strategies could exist. For instance, there might be a need to penalize users who insist on querying parcels that are already blocked, but the penalty should be attributed during QBA control. That is, if a user tried to access a parcel, where access is denied for x-collusion resistance or (x, y, z)-collusion resistance violation then the release of all neighboring parcels could be postponed for another timer tick.

[12] This observation was also found in inference control, e.g. [CW05]

5.3 Inference Channels

5.3.1 Potential Inference Channels from external knowledge

Most of the people using the cadastral application also have some external knowledge. Very often, they know the owner's name of some parcels from their neighborhood or their village or their families. Because of this external knowledge, users can break Pr_1 or Pr_2 without being detected by the QBA control mechanism. Dealing with external knowledge is theoretically impossible since it is simply impossible to know what a given user knows. However, the security administrator can roughly estimate the average level of users' external knowledge. This estimation is expressed in the parameter β of equation 14 which is a modification of equation 10.

$$
\begin{gathered}
\forall p \in P, \forall Z_i \in R(p,z), \alpha \in \mathbb{N}^+, \beta \in \mathbb{N} \\
\exists \mathcal{R} = \{R', R''\} \text{ a partition of } R(p,z) \text{ where } |R'| \leqslant y, \text{ and} \\
k_i = \begin{cases} n_i - (\alpha + \beta) & \text{if } \alpha + \beta > n_i \text{ and } Z_i \in R' \\ \lceil n_i/x \rceil - \alpha + \beta & \text{if } (\alpha + \beta) > \lceil n_i/x \rceil \text{ and } Z_i \in R'' \\ 1 & \text{otherwise} \end{cases}
\end{gathered}
\tag{14}
$$

$\beta = 0$ means the users are assumed to have no external knowledge whereas a $\beta > 0$ means the security administrator estimates that before querying any dominant zone Z_i, the average user of his database knows the owner(s) of β parcels.

5.3.2 Potential Inference Channels from Denial of Access in Pr_1

In the framework of multilevel database, Sandhu and Jajodia [SJ92] underlined the fact that a denial of access provides the user with the information that the data s/he is trying to access is highly classified. In the context of our application, if a user is denied an access then s/he can conclude that s/he is about to break prohibition Pr_1. If the user is trying to break Pr_1 then s/he actually does not learn much from the denial of access because Pr_1 prohibits the aggregation of parcels in a region only, and the region is public information; the security of QBA is independent of the disclosure of the regions. Therefore, from the parcels s/he has accessed before the denial of access, s/he can simply verify that s/he has queried too many parcels within a given region.

5.3.3 Potential Inference Channels from Denial of Access in Pr_2

First of all, we should note that in order to be successful, an attacker trying to break Pr_2 should already know the approximate location of all the target entity's parcels. Without this external knowledge, the attacker would need to randomly select parcels from the entire database which is of course infeasible.

Nonetheless, we consider it as a probable attack and we shall address it. Let us assume that Bob already knows the approximate location of all Alice's parcels. We also assume that after several queries, Bob has identified several parcels belonging to Alice. If Bob is denied access to an additional parcel then he can reasonably deduce that this parcel belongs to Alice. Returning *“access denied*" can even be seen as worse than returning Alice’s name since it informs Bob that he has found the last parcel in Alice’s list of parcels, if we consider $\beta = 0$.

One possible solution to prevent Bob from deducing that he has found the last parcel in Alice’s parcels list is to increase the value of β. In that case, Bob would be denied access to Alice’s parcels before finding the last parcel. However, there is no solution to prevent Bob from deducing from a denial of access that he has found a parcel belonging to Alice.

Another possible solution would be to return a cover story instead of denying access. A cover story is a lie introduced in the database in order to hide the existence of a sensitive data [CG99]. Cover stories have mainly been used in the framework of military multilevel databases [Den+88]. In our cadastral application, using a cover story would mean returning a fake owner for a given parcel. This solution is of course unacceptable for an official online public cadastral application where answers to queries have a legal value and have, therefore, to be trusted.

We propose another solution: we deny access to the remaining parcel and all its geographical neighbors. In the same example of Alice and Bob, when Bob reaches the limit k_l, we deny access to the remaining parcel, namely p, and all parcels of its geographical zone (as defined for Pr_1 in Definition 4). This can be achieved by adding a special flag associated to every parcel in the database that would be read during QBA control. When Bob reaches k_l in any dominant zone, QBA control should set this flag to true to the remaining parcel p and its geographical neighbors. Subsequently, when Bob tries to access p or any of its geographical neighbors, access should be immediately denied. This flag should be the first thing checked by QBA control.

This way, we increase the confusion for Bob, thus lowering his confidence in the inference by denial of access from $1/1$ (the case where only the remaining parcel is blocked) to $1/n$, where n is the number of parcels in the (Pr_2) zone of p. This confidence can even be lowered by increasing the number of blocked parcels by including 2nd degree neighbors of p too.

6 Choosing the Model’s Parameters

The responsibility of setting the values of the model’s parameters falls on the security administrator. We have already discussed other parameters and their significance. α is used to anticipate variations in the size of dominant zones. β is used to minimize the effect of inferences arising from QBA control itself due to users’ *a priori* knowledge. β figures twice in equation 14, for k_h and k_l, and it should hold the same value in both cases. Parameters of the resetting scheme t and T should be calibrated depending on the expected traffic.

The parameters x, y and z cannot be assigned arbitrary values. For instance, x defines the level of collusion resistance per dominant zone Z_i, therefore, x should always be strictly smaller than $|Z_i|$, [13]. In Section 4.2 we argued that x should have the same value for all dominant zones.

Let us take the example of Figure 2, and let us consider, for the sake of argument, that this is our complete cadastral database. We have two dominant zones: Z_2 of size 3, and Z_5 of size 5. If we want this database to be 4-collusion resistant, then for Z_2, $k_l = 1$ and $k_h = 2$; for Z_5, $k_l = 1$ and $k_h = 4$. Notice that this is the highest level (limit) of collusion resistance that could be attained on Z_2: we either give access to 1 parcel out of 3, or we deny access completely, which is not a desirable outcome for this specific application. Therefore, the best solution is to fix x for the whole database: $\forall p \in P, \forall Z_i \in D(p)$ where $|Z_i| \leqslant x$, $k_l = 1$, achieving $(|Z_i| - 1)$-collusion resistance; the remaining dominant zones will be x-collusion resistant.

Figure 3 shows for each point (π_1, π_2) the π_2% of parcels in the database of Maupiti (y-axis) that are attached to parcels of size $\leqslant \pi_1$ (x-axis). The value of x (on the x-axis) could be less than or equal to the average of dominant zone sizes in the database. It could even be set to the value of the mode [14] (or a value in between). If x was set to the average, then 66.41% of dominant zones will have less than x-collusion resistance. If it was equal to the mode, then 32.82% of dominant zones will have less than x-collusion resistance. Therefore, the security administrator should perform such an analysis for his cadastral database. If s/he sets x too high (e.g. 10 for the cadaster of Maupiti), then the majority of dominant zones will not be effectively x-collusion resistance as s/he desires (e.g. 91.28% for Maupiti).

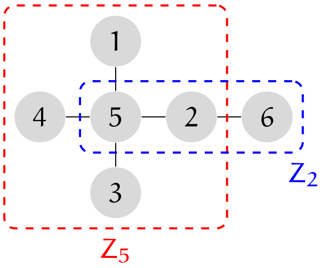


Figure 2: Example graph representing parcels

If we consider (x, y, z)-collusion resistance, then y is the number of parcels that are not under collusion resistance in a z-region, therefore for a given parcel p, $y < |R(p, z)|$. Theoretically, y could be distinct for every z-region. There is no implication on the security of the application. However, distinctive y values means that it should be calculated for every z-region, which means Breadth-First traversal should be used every time we want to know the new value of y, especially after a mutation operation, and the resulting value should be stored in the database: performance and storage hits are inevitable. Therefore, setting a global value for y is a more reasonable choice.

Practically, y can be lower than or equal to the average number of dominant zones (or mode) in a z-region. Here, dominant zones show an advantage. Figure 7 on page 32 shows that the average

[13] If $x = |Z_i|$, then any user would have access to 1 parcel in Z_i at most. A coalition of $|Z_i|$ users can recover Z_i

[14] The mode is the value that appears most in the dataset; the zone size that appears most in the database, in our case.

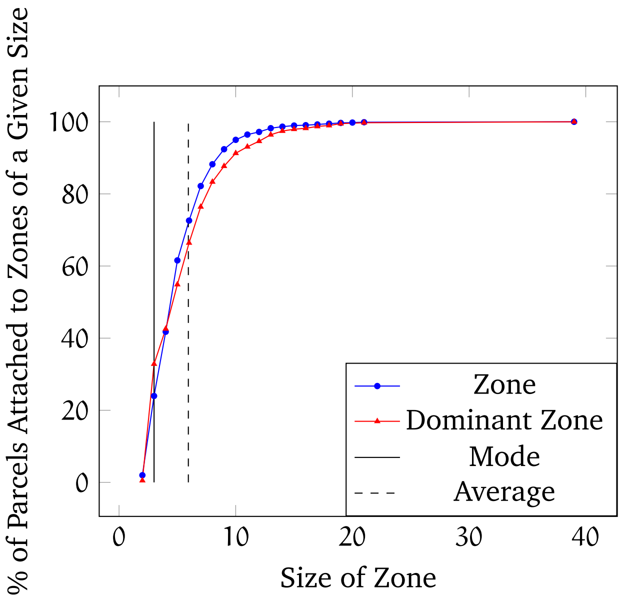


Figure 3: Each point (π_1, π_2) shows the π_2% of parcels in the database of Maupiti (y-axis) that are attached to parcels of size $\leqslant \pi_1$ (x-axis). For $x = \pi_1$, x-collusion resistance will be ensured for 100 - π_2 % parcels.

number of parcels in a z-region, for different values of z, changes less drastically for dominant zones than for zones. This advantage is even clearer if we consider the maximum number of parcels that could occur in a z-region. Indeed, the slope of maximum parcels in a dominant zones is close to that of their average, and almost as smooth; the slope of maximum parcels in zones, on the other hand, is very steep and jumps drastically especially for low values of z.

As for z, the security administrator should keep in mind that, in addition to its function in determining the balance between data availability and data confidentiality, it determines the depth in the Breadth-First traversal (to make the QBA control decision on a requested parcel p, we need to calculate the new value of ${}_y d_q$ for every dominant zone $R(q, z), \forall q \in N(p, z)$), which has a runtime complexity of $\mathcal{O}(b^z)$, where b is the branching factor (or the average number of neighbors per parcel). Therefore, z should be > 1, and an assessment of available computational resources should be taken into account to achieve the desired and most practical results. Here also, dominant zones present a significant advantage over zones, in terms of runtime, as shown in Figure 6 (page 31) and discussed in Section 9.2 (page 30).

Nevertheless, the security administrator and decision makers on this matter should test different values—while taking into account these recommendations—to see for themselves the results of different tunings and different combinations. In fact, the topology of the neighborhood graph changes from one cadastral database to another, and it is mainly related to the geography of the place in question. It could also be affected by economical or social factors. The graph of an ancient and continuously lived city like Byblos [15] differs significantly from that of a modern one like New York City, or a remote island in the pacific like Maupiti. The topology will directly affect the availability of cadastral data, which requires human intervention and judgement to get the best desired results.

7 Application to the Cadaster of French Polynesia

In this section we will describe the application of QBA control on the cadastral database of French Polynesia. Indeed, we will talked about the desired workflow in Section 7.1. We will talk about authentication, and how users should be using the system in Section 7.2. And finally, we will discuss the decisions taken regarding the enforcement of Pr_1 and Pr_2 in Section 7.3.

7.1 Desired Workflow

Currently, in order to acquire information about any given parcel, a citizen of French Polynesia needs to visit the facilities of the real-estate service of French Polynesia. There, s/he will stand in a queue waiting for her/his turn, and then s/he will meet an employee who will recieve the citizen's query. The citizen needs to provide the requested parcel's ID, or its address. Moreover, s/he can query multiple parcels at the same time. The citizen needs not to provide any identification (no driver's license, nor passport, etc).

Once provided with the parcel's ID or its address, the employee will perform a check on the query itself, the number of requested parcels and the rate at which the citizen has been issuing queries: *Is the requested information classified?* (e.g. owned by the military, the president, etc.). *Is the citizen requesting a lot of parcels?* (e.g. the owners of a complete neighborhood). *Has the citizen been asking for cadastral excerpts regularly and in a suspicious manner?*

Obviously, the employee is enforcing an internal policy constraining citizens' requests. If the employee accepts the request, the citizen must pay a fee before getting the excerpt of the requested parcel(s).

There are two main issues with this workflow:

1. Citizens must be physically present at the real-estate service. This is especially problematic in countries such as French Polynesia that are formed uniquely by archipelagos (118 islands and atolls with an Exclusive Economic Zone (EEZ) of over 5 million km^2. In comparison, Metropolitan France's EEZ is around 330 thousand km^2 only).

2. Employees enforcing the service's internal policy are themselves human, therefore error-prone. Moreover, there is not a single employee, and they do change with time.

[15] The city of Jubayl in modern-day Lebanon, first occupied between 8800 and 7000 BC.

The real-estate service wishes to make the cadastral database available online, making it easier for citizens to acquire excerpts of parcels, while adapting the original workflow as follows:

1. A user is presented with a mapping interface where s/he has the option to select a single parcel.
2. Once selected, the user has the option to *“preview*" the parcel’s ownership information, as long as this *“preview*" does not violate the service’s policies (namely Pr_1 and Pr_2).
3. If the preview was successful, the user can either cancel his order or proceed and place the order for the excerpt where s/he is required to pay a predefined fee.
4. If the preview was not successful—due to the violation of either Pr_1, Pr_2, or both—the user can still proceed and place the order for the excerpt and pay the required fee.

This *“preview*" feature acts as a guard for the user her/himself: online data can be out of date or incorrect. Therefore, s/he can profit from this feature and withhold from paying any amount of money if s/he judges that online information is not accurate. Pr_1 and Pr_2 are required to limit the abuse of this feature.

7.2 Authentication

Our model is secure with *“strong*" authentication. By *“strong*" authentication we mean a mechanism that could efficiently tie the physical identity of a user to his virtual one, so s/he could not create multiple identities on the system to circumvent the security policy. This is known in the literature as the *“Sybil Attack*" [Dou02].

However QBA control in the context of this cadastral application is only preventive as we previously showed. The service explicitly mentioned that any form of *“strong*" authentication is unnecessary and might discourage users from using the service, especially that: 1) Access to the internet on small islands is available uniquely through municipalities, and users are not necessarily tech-savvy. 2) They want to replicate the current workflow found at their offices, and they want to keep no record that identifies the user explicitly, just like the physical process. Users, for such workflows, can be authenticated with their IP addresses, which seems to be sufficient—from the service’s point of view—to enforce QBA and manage collusions. It follows that collusion resistance is also meant to prevent users from constantly changing their IP addresses (e.g. disconnecting their ADSL modem then reconnecting it) to circumvent QBA control. Collusion resistance is used to deter casual attackers, not serious ones.

Notice that the goal is not anonymous authentication. Indeed, users of the cadastral application can be traced on the online application using indirect identifiers, if needed (e.g. through browser cookies). The cadastral database holds information about people, and abusers of the application should be traceable in case tracing is needed (e.g. court order on legal action). Indeed, the real-estate service does not have the right to ask for identification when a person asks for a cadastral excerpt–at their facilities or online–but they have security cameras in their offices, and employees and other people in the building can act as eye witnesses that could possibly re-identify a person if there is a need for it.

7.3 Pr_1, Pr_2 or Both?

The security policy as defined by the real-estate service states that both Pr_1 and Pr_2 should be applied in conjunction.

Pr_1 is applicable directly to the whole database. Since French Polynesia is constituted of islands, the real-estate service has the advantage of analyzing and fine tuning QBA control for every island if it wishes to do so. Although tedious, the choice of parameters x, y and z (and α, t and T) for (x, y, z)-collusion resistance can be done independently for every island, taking into consideration the nature of every island [16], its economic and social importance [17], etc.

However, Pr_2 is problematic. We cannot know the number of parcels owned by multiple legal entities. In fact, parcels with multiple owners are registered as if they have a single owner. Ownership information in the database is not, currently, in a format that distinguishes and/or groups legal entities in a meaningful and consistant manner. For example, a married couple where each one owns a parcel outside marriage and share the ownership of a third will be identified in the database as 3 separate owners, with no links to tie them. This is not the case for the cadaster of France, for example, where every person is registered separately and relationships between people is present. If we take the same example of the married couple, in France, they would be identified as 2 separate owners—instead of 3—where everyone owns a parcel separately and they both share the ownership of a third.

Even if the real-estate service wishes to implement Pr_2 on the current database, the best level of collusion resistance that could be achieved is x-collusion resistance, because of the second reason we previously mentioned. Currently, there is no social graph in the cadastral database of French Polynesia, which is a prerequisite to (x, y, z)-collusion resistance.

8 Prototype

In this section we will describe our prototype. Please bear in mind that our prototype is only a *“proof-of-concept*" for QBA control itself, not a prototype of the production application desired by the real-estate service.

The IT department has provided us with a database to test our model and algorithms (that of the island of Maupiti). We have implemented Pr_1 only for two main reasons:

1. All owners in the cadastral database of Maupiti own a single parcel.

[16] *Is it an island? An islet? A reef islet?*

[17] Economic and social importance can be used as general indicators of expected traffic

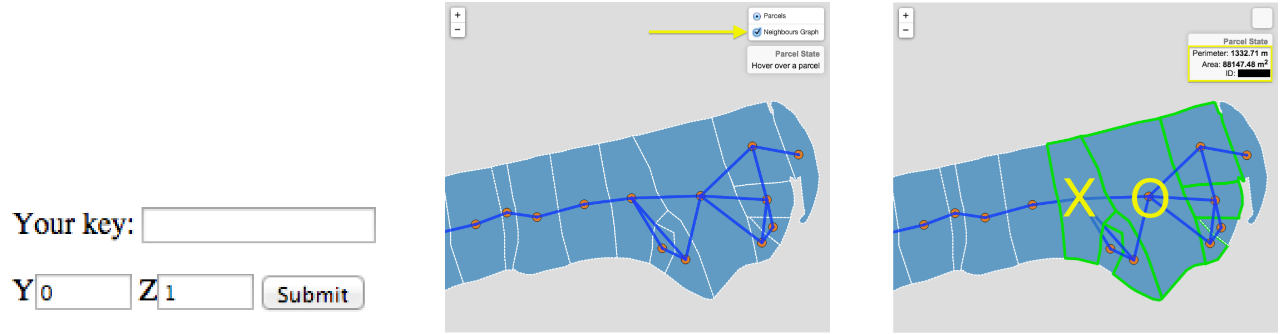
2. Information about families is not currently available in a format that allows direct analysis regarding the advantages and disadvantages of dominant zones for Pr_2.

It would be tempting to synthesize data about owners in order to simulate Pr_2, but any conclusions drawn from the results would be largely affected by the underlying assumptions: What is the average amount of parcels owned by a single person? Is it the same for rural areas as for urban ones? How likely is it to have neighboring parcels belonging to two different yet socially related persons? What are the different kinds of social relationships that we are considering? etc.

We used PostgreSQL 9 to implement QBA control: all QBA control procedures were written in PL/SQL. The polygons that represent the parcels are in GeoJSON and rendered by the client using LeafletJS. The server is written entirely in Java 1.7 . It serves two main functions: 1. serve the UI for the user, through static content (`html` and `js` files), and 2. serve the capacity to query the database, through `HTTP` APIs. .

The prototype should be normally accessible from `http://webgis.upf.pf:8080`. We only set up the parameter $x = 2$ of (x, y, z)-collusion resistance, and the user can chose his desired values for y and z from the user interface (Figure 4a). Once done, the user is presented with two instances of our map previewer.

Parcels are colored in blue and their borders are marked with a white dashed line, as shown in Figure 4b. The user can choose, from the upper right corner, to display the neighborhood graph. It is only used for display. When a user hovers over a parcel, the dominant zone is highlighted in green, as shown in Figure 4c. In this example, the user had the mouse over the parcel marked with **X**; its dominant zone is marked with **O**. Additional information about the parcel are shown in the upper right box (e.g. parcel `ID`, surface area).



Your key:

Y 0 Z 1

Submit

● Parcels

☑ Neighbours Graph

Parcel State

Hover over a parcel

Parcel State

Perimeter: **1332.71 m**

Area: **88147.48 m^2**

ID: [redacted]

(a) Entering simulation parameters

(b) Part of the cadastral database of Maupiti with the graph plotted on top

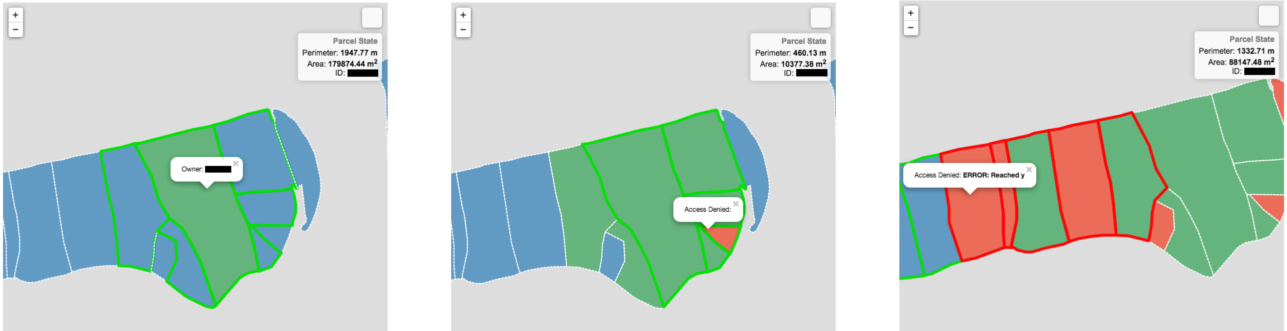
(c) Part of the cadastral database of Maupiti showing information when hovering over a parcel

Figure 4: The user interface

When a user clicks on a parcel, an asynchronous call to our APIs will be executed. In case access was granted, a popup above the parcel containing ownership information is showed, and the parcel turns green (Figure 5a).

If access was denied because the (simulated) user reached the limit k_h in a given dominant zone, a popup saying *"Access Denied*" is shown above the requested parcel. Its color turns to red (Figure 5b).

If access was denied because the disclosure of ownership information of the requested parcel might cause one or multiple regions to surpass the allowed value of y, then a message saying *"Access Denied: ERROR Reached y*" is shown in a popup above the selected parcel. Its color turns into red too. Moreover, borders of dominant zones where the value of y might surpass the limit turn into deep red too (Figure 5c).



Parcel State
Perimeter: **1947.77 m**
Area: **179874.44 m^2**
ID: [redacted]

Owner: [redacted]

Parcel State
Perimeter: **460.13 m**
Area: **10377.38 m^2**
ID: [redacted]

Access Denied:

Parcel State
Perimeter: **1332.71 m**
Area: **88147.48 m^2**
ID: [redacted]

Access Denied: **ERROR: Reached y**

(a) Access Granted

(b) Access Denied: the Limit KH is Reached

(c) Access Denied: the Limit y is Reached and Borders of Dominant Zones that Risk Surpassing y are Highlighted

Figure 5: Different Results of QBA Enforcement

It is worth mentioning that the database we received from the GIS department of French Polynesia contains original information about Maupiti's parcel owners. Due to confidentiality agreements, we eliminated every trace of information that could relate to the original owners: all names were deleted and every parcel has one fake owner.

9 Experiments

In the following, we will describe our experience in implementing QBA control for the cadastral database of Maupiti. In Section 9.1, we will discuss approaches we described in prior work, in terms of the model itself, and the algorithms used to enforce QBA. In Section 9.2, we will show, experimentally, how the use of dominant zones, as opposed to zones, provides a tangible advantage in terms of query execution time.

9.1 Previous Approaches

We tackled QBA problems in the cadastral database in an earlier work [AGC13]. Initially, we did not use dominant zones. QBA control was enforced on zones uniquely. We also defined different levels of collusion resistance (x-, (x, y)- and (x, y, z)-collusion resistance) to prevent users from colluding and bypassing Pr_1 and/or Pr_2. Our approach to implement these levels required tracking the query history of every user. This history was used to track collusions on the user level, i.e. maintaining lists of who is colluding with whom. This tracking required $\mathcal{O}(\binom{u}{x})$ space to maintain the list of colluding users, while searching for a potential collusion on a single parcel level was an exhaustive search requiring $\mathcal{O}(x^n)$ time, where u is the number of users in the system, n is the number of users who has accessed a parcel, and x is the value from x-, (x, y)- or (x, y, z)-collusion resistance. In addition, this implementation was described in terms of a graph database.

The work presented in [AGC14a] provided an alternative and more efficient implementation, using the same model, namely with zones only. In this second implementation, we were not tracking any collusion in the first place. Indeed, we were defining the number of accessible parcels in a region (Pr_1) or belonging to a given family (Pr_2) beforehand and then simply counting the number of actually accessed parcels and making sure it does not exceed a given threshold. We also dropped a level of collusion resistance, namely (x, y)-collusion resistance, and changed some definitions in order to gain performance enhancements without compromising their security properties. Moreover, our solution was described in the relational model, facilitating the integration with the existing cadastral database of French Polynesia.

For more information about the performance of the QBA enforcement algorithm, with zones only, the reader is invited to read [AGC14a]. Dominant zones were introduced later [AGC14b] to achieve more availability. The reader is invited to read [AGC14b] for more information about the experiments on availability comparing zones only and dominant zones. In this section, we will compare zones to two different implementations of dominant zones, performance-wise.

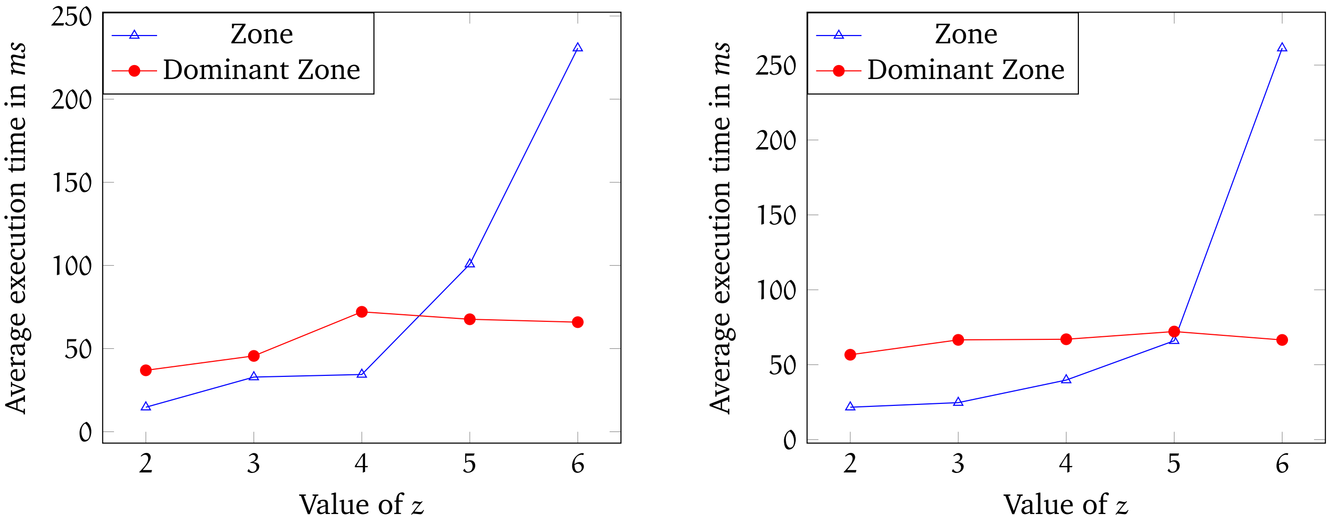
All of our experiments were run on a MacBook Air (5,2) with an Intel Core i5 1.8 GHz CPU (2 cores), 4 GB of RAM and 128 GB SSD.

9.2 Comparing Zones and Dominant Zones

In this section we will show how the enforcement of QBA control on *“dominant zones"* instead of *“zones"* only is beneficial. Figure 6 compares the execution time of (x, y, z)-collusion resistance for zones and dominant zones. Figure 6a shows the execution time of the two different implementations for $y = 3$ and for different values of z. Figure 6b does the same thing but for $y = 4$. These results are valid for other values of y, but we chose to reduce it to two examples for clarity.

Figures 6a and 6b are the results of the following experiment: we chose 300 random parcels out of

960 from the database of Maupiti. For every algorithm, for different values of y and z (3 to 4, and 2 to 6, respectively), we clear all stored history of the database. Afterwards we create 100 users, make them access selected parcels in the same order, and we calculate the average time of this traversal.



(a) Execution time for $y = 3$

(b) Execution time for $y = 4$

Figure 6: Execution time of (x, y, z)-collusion resistance for 2 values of y (3 and 4 for Figures 6a and 6b respectively)

Let us now compare the zone and dominant zone implementations. The first impression is puzzling: on one hand QBA enforcement performs better with zones, especially for low values of z, and on the other hand performance under zones accelerate exponentially, and is inferior to the one under dominant zones for higher values of z. These figures are puzzling especially that both implementations use almost the same algorithms for QBA enforcement. The explanation for both results lies in the part where we need to retrieve the z-region of the requested parcel.

Indeed, the performance hit we see for low values of z when performing QBA control on dominant zones is due to the fact that we need to extract extra-information (when compared to zones): QBA control on dominant zones need to do the extra step of fetching dominant zones of a requested parcel p. However, the number of dominant zones considered when updating user access history is far lower than the number of zones for big values of z; i.e. $D(p) \subseteq \mathcal{Z}(p)$, and the zone implementation has to update user access history for all $Z_i \in \mathcal{Z}(p)$, while the dominant zone implementation will have to update all $Z_j \in D(p)$, which guarantees that the snowball effect will make the difference in the numbers of zones to consider greater as z increases.

Figure 7 shows both average and maximum number of parcels returned by BFS for both zones and dominant zones while updating user access history. As it is clear in this figure, the average and maximum number of parcels that could be returned for dominant zones is far lower and does not experience dramatic jumps like the case for zone.

Which means that we gain on performance on the expense of storage (more storage is needed when compared to zones only). This gain in performance is not exclusive to the QBA enforcement algorithm. We also gain performance on mutation operations: every time a parcel is mutated (merge/split), all counters should be re-calculated per user per zone or dominant zone.

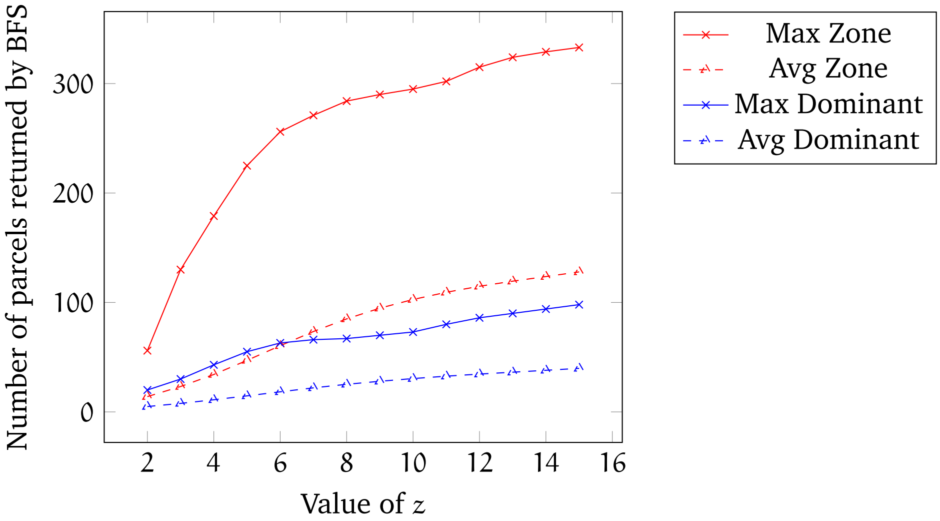


Figure 7: Comparison of the number of zones to consider while updating user access history as a function of z

10 State of The Art

The most relevant work is that of Motro, Marks and Jajodia [MMJ94] (MMJ). They developed a model to handle QBA in relational databases. The first key difference between their work and ours is in the hypothesis. They provide a model for QBA control where a user can execute “arbitrary queries”, i.e. a user can select and project tuples from a phonebook relation on any set of attributes he desires, while we only consider single queries selecting a single tuple (point and click). Their approach relies on: 1) intercepting user queries, 2) modifying the query so it would return “fresh" tuples (not previously queried by the user), and finally 3) checking if the result of that query would add up the disclosed number of tuples to more than k out of n entries. Their work was developed 20 years ago, and although they have extended their work [MMJ96] to include multi-query attacks (i.e. Join and Complementary queries), the model lacks a lot of advanced features needed in our case, most notably: 1) Collusion resistance, 2) Dynamic setting: where the complete database is subject to continuous updates, and 3) Resetting access after a period of time: in MMJ’s model, access to individual entries is not recorded which turns the issue of resetting access problematic: (a) If we want to “release access" to entries from oldest to newest, tracking access to individual records is imperative, (b) Shall we track the newest access to the phonebook only? If so, we can associate a timestamp to the phonebook instead of each entry in the phonebook and update it to the most recent date it was accessed. This way, the resetting strategy changes altogether, but the question now is “is such a resetting strategy desirable?". It was not clear for us how these features would be incorporated in their model.

Hinke [Hin88] identifies 2 types of aggregation problems: 1) Cardinality aggregation, and 2) inference aggregation; the former is what we now call QBA and the latter is the “classical" inference problem. He argues that both cardinality aggregation and inference aggregation are subclasses of the aggregation problem. He did not work on the cardinality aggregation problem because he noted that *“[the inference aggregation problem] appear to be more tractable. With cardinality aggregation, it is not always clear why ‘n’ elements of a set, such as a phonebook, are classified at one level, while ‘M’ elements are less classified, where cardinality of* $N \geqslant$ *cardinality of* M". Indeed, the phonebook example does not induce any special interest, which was apparent while surveying the literature.

The work of Lunt [Lun89] analyses inference and aggregation problems found in multilevel relational databases. She classifies some problems as inference problems and not true aggregation ones, and shows how inference problems can be remedied using proper database design. According to Lunt, the inference problem arises whenever some data x can be used to derive partial or complete information about some other data y, where y is classified higher than x. The aggregation problem arises whenever some collection of facts has a classification strictly greater than that of the individual facts forming the aggregate. To qualify as an aggregation problem, it must be the case that the aggregate class strictly dominates the class of every subset of the aggregate. Under aggregation problems, she identified quantity-based aggregations (known earlier as cardinality aggregations). A QBA problem occurs whenever a collection of up to k items of a given type is not sensitive, but a collection of greater than k items is sensitive (in the original work she used n). Lunt’s definitions are the basis of the definitions we present in Section 3.

Jajodia and Meadows [JM95] give another definition of inference problems while surveying the literature on inference control problems in multilevel secure databases. They first introduce the notion of an inference channel, which is a mean by which one can infer data classified at a high level from data classified at a low level. The inference problem is the problem of detecting and removing inference channels. At the end of their paper, they briefly talked about aggregation problems and mentioned that they are similar to inference problems but not identical. They also showed how different strategies could be adopted to control different aggregation problems. They gave the following definition of aggregation problems: The aggregation problem exists when the aggregate of two or more data items is classified at a level higher than the least upper bound of the classification of the individual items. While this definition is correct, we think that it is less accurate than than the ones given by Lunt or in this work; they do not make the distinction between -what we call- the general aggregation problem and the quantity-based aggregation problem.

Bewer and Nash [BN] presented the Chinese-Wall policy and presented a mathematical theory to implement such a policy. They might be the first to identify a real-world aggregation problem. In fact, the main motivation for the work was to prevent a user from aggregating knowledge in a domain that would help him learn sensitive information and conduct malicious behavior. However, this approach is very basic in terms of aggregation control. The policy doesn't allow controlling the limit on the number of requested datasets in a single conflict of interest class. The limit is always one dataset per class. Moreover, it doesn't say anything about a single dataset falling in several conflict of interest classes. Collusion is not treated at all, but the main ideas that could be taken from the paper are the following: 1) Their policy provide mandatory access control while always preserving free choice: (a) A user has the right to access any dataset in the same conflict of interest class, (b) User's query behavior decides the set of available datasets and the set of prohibited ones, and 2) Any system implementing such policies should track user's history.

In a different work, Meadows [Mea90] give another definition of the aggregation problem and she says that aggregation issues arise in database security when two or more data items are considered more sensitive together than they are separately. She extended the Bewer-Nash model in order to generalize it to multilevel databases. She presents a formal model that is able to handle the Chinese-Wall security policy and other types of aggregation problems. In her model, every object is assigned a security level. Aggregates are assigned a security level too. A security lattice is created from security level labels on objects. Then she defines rules of information flow: a user with a given clearance level can only have access to aggregates of the same or lower level. Her work requires storing the complete access history of every user. It is best suited for environment where MLS is required, i.e. where different objects of different security levels form an aggregate with an even higher security level. Collusion is not treated at all.

Cuppens [Cup91] studied the aggregation problem in multilevel databases and proposed a model based on modal logic. In fact, the author starts by proposing his model then shows how it could be instantiated to traditional multilevel security without aggregation. Then he shows how to express the aggregation problem, as presented by Meadows [Mea90], using this modal logic. Cuppens notes that *"[in order] to control the aggregation problem, the system must also keep track of the aggregate of all datasets that have previously been accessed by a subject*", which is an observation that we share too, for the general aggregation and QBA problems, but the work of Cuppens covers the general aggregation problem only.

We would like to mention the work of Foley [Fol91; Fol92] that addresses the aggregation problem with information flow policies. In fact, Foley described a unified framework for information flow control that takes into consideration the (general) aggregation problem as presented by Brewer and Nash, and further developed by Meadows, and subsequent works. QBA was not addressed.

Staddon [Sta03] presented in her paper a dynamic inference control scheme that does not depend (directly) on user query history, which implies fast processing time, and ensures a crowd-control property: a strong collusion resistance property that not only prevents c collaborating users (where c

is the degree of collusion-resistance) from issuing complementary queries to complete an inference channel, but also guarantees *“if a large number of users have queried all but one of the objects in an inference channel, then no one will be able to query the remaining object regardless of the level of collusion resistance provided by the scheme*". c-collusion resistance is not desirable in QBA control because it implies that at least one object out of n can never be read by any user.

Chen and Wei [CW05] extended the work of Staddon on dynamic inference control. They have described 2 schemes that prove to be more efficient than Staddon’s which is due to their key allocation scheme. Then they present a third scheme that is resilient to what they call a *“block an object*" attack where a malicious user can exhaust a channel therefore blocking access to the last object for all other database users. Their first 2 schemes can prevent an arbitrary number of collusion, unlike Staddon’s, which is c-collusion resistant. The third one guarantees a minimum collusion resistance against c users. The important thing to take from this paper is what they noticed about blocking users and how effectively a time-based key-refreshing scheme should be enforced to prevent not only “block an object" attacks, but also blocking users on a set of accessible objects, which might render the application useless after a given period of time. The problem with such schemes (Staddon and Chen-Wei), other than objects shared among multiple channels, is channel’s length itself. It is never clear how channels with varying lengths would be treated, which is very important in a real-life application such as the cadastral database that is subject to daily updates. Not to mention that the method may suffer potential inferences by denial of access. There is no clear solution for such cases. Furthermore, they do not mention external knowledge and how would a security administrator limit inferences by external knowledge; maybe the parameter t they describe in the third scheme can work as a parameter controlling additional inferences from external knowledge.

Bezzi et al. [Bez+10; Bez+12] also treated QBA. Their goal was to prevent statistical inferences. As a matter of fact, they consider that the distribution of soldier’s age in a military location can allow inferring the nature of a location itself, whether it is a headquarter or a training campus. Therefore their goal is to perform a k out of n disclosure control such that the distribution of these k records does not resemble the distribution of the sensitive information.

11 Conclusion

In this paper, we presented two distinct, yet similar, QBA problems. The goal was to publish the cadastral database of French Polynesia while enforcing two prohibitions (the QBA problems), namely Pr_1 and Pr_2. We explained the legislative point of view on the subject. Since cadastral data contain personal information, the law imposes some restrictions on its online publication. These restrictions are expressed in Pr_1 and Pr_2

Afterwards we presented our model: how to enforce Pr_1 and Pr_2. We introduce different concepts like zones, dominant zones and regions, then we tackle the subject of collusion: when multiple users collaborate to circumvent any of the prohibitions.

We also tackled additional aspects that should be handled when using QBA control: 1) mutations, which are updates in the cadaster, and how to properly handle them 2) how to reset access to regions after a period of time 3) how to anticipate inference channels that could arise from QBA enforcement itself due to users' background knowledge or a denial of access.

We showed that a successful publication of cadastral data requires serious fine-tuning by the database administrator: x, y, and z should be carefully chosen until s/he gets what s/he evaluates as the best compromise between 1) data availability and its confidentiality, and 2) computational resources and traffic.

Throughout the paper, the discussion on QBA control was general, and could be applied to any cadastral database. We dedicated a section that talked about specific aspects of the application of QBA control to the French Polynesian cadaster, namely the current physical process of "cadastral excerpt requests" and how the real-estate services intends to keep as much as possible of the workflow when developing the online application. In that context, we showed how, for the service, a basic authentication scheme (e.g. based on IP addresses) is sufficient.

Additionally, we presented our prototype that is currently accessible online. We also showed performance benchmarks for the developed algorithms: how and why dominant zones are advantageous in terms of performance, when compared to the use of zones only.

Currently, we are in the beginning stages of the implementation of a production-ready web application that will be at the disposal of the public, in partnership with the real-estate service and IT service of French Polynesia, and a third-party—a renowned company in GIS development.

Acknowledgments

The authors wish to thank the following people for fruitful discussions on some legal aspects applying to the cadastral application and for providing them access to a sample cadastral database. • Tania Berthou, Directrice des Affaires Foncières de la Polynésie Française; • Bertrand Malet, Chef de la Division du Cadastre, Direction des Affaires Foncières; • Jean-Louis Garry, Directeur du Service Informatique de la Polynésie Française; • Emmanuel Bouniot, Chef de la Cellule Systèmes d'Informations Géographique du Service Informatique de la Polynésie Française.

References

[09] *Les guides de la CNIL. Les collectivités locales.* Link. 2009.

[13] *Fiscalité locale et cadastre.* Link. accessed June 2013.

[AC09] Bechara Al Bouna and Richard Chbeir. "Detecting Inference Channels in Private Multimedia Data via Social Networks". In: *Data and Applications Security XXIII, 23rd Annual IFIP WG 11.3 Working Conference, Montreal, Canada, July 12-15, 2009. Proceedings*. 2009, pp. 208–224. DOI: 10.1007/978-3-642-03007-9_14.

[AGC13] Firas Al Khalil, Alban Gabillon, and Patrick Capolsini. "Collusion Resistant Inference Control for Cadastral Databases". In: *Foundations and Practice of Security - 6th International Symposium, FPS 2013, La Rochelle, France, October 21-22, 2013, Revised Selected Papers*. 2013, pp. 189–208. DOI: 10.1007/978-3-319-05302-8_12.

[AGC14a] Firas Al Khalil, Alban Gabillon, and Patrick Capolsini. "Implementing Quantity Based Aggregation Control for Cadastral Databases". In: *2014 IEEE World Congress on Services, Anchorage, AK, USA, June 27 - July 2, 2014*. 2014, pp. 137–144. DOI: 10.1109/SERVICES.2014.33.

[AGC14b] Firas Al Khalil, Alban Gabillon, and Patrick Capolsini. "Quantity Based Aggregation Control for Cadastral Databases". In: *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Privacy in Geographic Information Collection and Analysis, GeoPrivacy '14, Dallas/Fort Worth, Texas, USA, November 4-7, 2014*. 2014, 7:1–7:8. DOI: 10.1145/2675682.2676394.

[Bez+10] Michele Bezzi et al. "Protecting privacy of sensitive value distributions in data release". In: *Security and Trust Management - 6th International Workshop, STM 2010, Athens, Greece, September 23-24, 2010, Revised Selected Papers*. Springer, 2010, pp. 255–270.

[Bez+12] Michele Bezzi et al. "Modeling and preventing inferences from sensitive value distributions in data release". In: *Journal of Computer Security* 20.4 (2012), pp. 393–436.

[BN] David FC Brewer and Michael J Nash. "The Chinese Wall Security Policy". In: *Proceedings of the 1989 IEEE Symposium on Security and Privacy, Oakland, California, USA, May 1-3, 1989*, pp. 206–214. DOI: 10.1109/SECPRI.1989.36295.

[CC08] Yu Chen and Wesley Chu. "Protection of Database Security Via Collaborative Inference Detection". In: *Intelligence and Security Informatics, Techniques and Applications*. Springer, 2008, pp. 275–303.

[CG99] Frédéric Cuppens and Alban Gabillon. "Logical foundations of multilevel databases". In: *Data & Knowledge Engineering* 29.3 (1999), pp. 259–291. DOI: 10.1016/S0169-023X(98)00044-5.

[Cup91] Frédéric Cuppens. "A Modal Logic Framework to Solve Aggregation Problems". In: *Database Security, V: Status and Prospects, Results of the IFIP WG 11.3 Workshop on Database Security, Shepherdstown, West Virginia, USA, 4-7 November, 1991*. 1991, pp. 315–332.

[CV13] C. Conejo and A. Velasco. *Cadastral Web Services in Spain*. Link. Accessed June 2013.

[CW05] X. Chen and Ruizhong Wei. "A Dynamic Method for Handling the Inference Problem in Multilevel Secure Databases". In: *International Symposium on Information Technology: Coding and Computing (ITCC 2005), Volume 1, 4-6 April 2005, Las Vegas, Nevada, USA*. 2005, pp. 751–756. DOI: 10.1109/ITCC.2005.7.

[DBS09] Maria Luisa Damiani, Elisa Bertino, and Claudio Silvestri. "Protecting location privacy against spatial inferences: the PROBE approach". In: *Proceedings of the 2nd SIGSPATIAL ACM GIS 2009 International Workshop on Security and Privacy in GIS and LBS, SPRINGL 2009, November 3, 2009, Seattle, WA, USA*. 2009, pp. 32–41. DOI: 10.1145/1667502.1667511.

[Den+88] Dorothy E Denning et al. "The SeaView security model". In: *Proceedings of the 1988 IEEE Symposium on Security and Privacy, Oakland, California, USA, April 18-21, 1988*. IEEE. 1988, pp. 218–233. DOI: 10.1109/32.55088.

[DH96] Harry S. Delugach and Thomas H. Hinke. "Wizard: A database inference analysis and detection system". In: *IEEE Transactions on Knowledge and Data Engineering* 8.1 (1996), pp. 56–66. DOI: 10.1109/69.485629.

[Dou02] John R. Douceur. “The sybil attack”. In: *Peer-to-Peer Systems, First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7-8, 2002, Revised Papers*. 2002, pp. 251–260. DOI: 10.1007/3-540-45748-8_24.

[Dua09] Yitao Duan. “Differential privacy for sum queries without external noise”. In: *ACM Conference on Information and Knowledge Management (CIKM)*. 2009.

[Dwo06] Cynthia Dwork. “Differential privacy”. In: *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*. 2006, pp. 1–12. DOI: 10.1007/11787006_1.

[Dwy09] Catherine Dwyer. “The Inference Problem and Pervasive Computing”. In: *Proceedings of Internet Research 10.0* (2009), pp. 1–11. DOI: 10.2139/ssrn.1508513.

[FJ02] Csilla Farkas and Sushil Jajodia. “The inference problem: a survey”. In: *ACM SIGKDD Explorations Newsletter* 4.2 (2002), pp. 6–11. DOI: 10.1145/772862.772864.

[Fol91] Simon N Foley. “A taxonomy for information flow policies and models”. In: *IEEE Symposium on Security and Privacy*. 1991, pp. 98–109. DOI: 10.1109/RISP.1991.130778.

[Fol92] Simon N Foley. “Aggregation and separation as noninterference properties”. In: *Journal of Computer Security* 1.2 (1992), pp. 159–188. DOI: 10.3233/JCS-1992-1203.

[Fri+11] Gerald Friedland et al. “Sherlock Holmes’ evil twin: on the impact of global inference for online privacy”. In: *2011 New Security Paradigms Workshop, NSPW ’11, Marin County, CA, USA, September 12-15, 2011*. 2011, pp. 105–114. DOI: 10.1145/2073276.2073287.

[Hin88] Thomas H. Hinke. “Inference aggregation detection in database management systems”. In: *Proceedings of the 1988 IEEE Symposium on Security and Privacy, Oakland, California, USA, April 18-21, 1988*. 1988, pp. 96–106. DOI: 10.1109/SECPRI.1988.8101.

[JM95] Sushil Jajodia and Catherine Meadows. “Inference problems in multilevel secure database management systems”. In: *Information Security: An integrated collection of essays* 1 (1995), pp. 570–584.

[LLV07] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. “t-closeness: Privacy beyond k-anonymity and l-diversity”. In: 2007, pp. 106–115. DOI: 10.1109/ICDE.2007.367856.

[Lun89] T.F. Lunt. “Aggregation and inference: Facts and fallacies”. In: *Proceedings of the 1989 IEEE Symposium on Security and Privacy, Oakland, California, USA, May 1-3, 1989*. 1989, pp. 102–109. DOI: 10.1109/SECPRI.1989.36284.

[Mac+07] Ashwin Machanavajjhala et al. “l-diversity: Privacy beyond k-anonymity”. In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1.1 (2007), p. 3. DOI: 10.1145/1217299.1217302.

[Mea90] Catherine Meadows. “Extending the Brewer-Nash model to a multilevel context”. In: *Proceedings of the 1990 IEEE Symposium on Security and Privacy, Oakland, California, USA, May 7-9, 1990*. 1990, pp. 95–103. DOI: 10.1109/RISP.1990.63842.

[MMJ94] Amihai Motro, Donald G Marks, and Sushil Jajodia. “Aggregation in relational databases: Controlled disclosure of sensitive information”. In: *Computer Security - ESORICS 94, Third European Symposium on Research in Computer Security, Brighton, UK, November 7-9, 1994, Proceedings*. 1994, pp. 431–445. DOI: 10.1007/3-540-58618-0_77.

[MMJ96] Donald G Marks, Amihai Motro, and Sushil Jajodia. “Enhancing the controlled disclosure of sensitive information”. In: *Computer Security - ESORICS 96, 4th European Symposium on Research in Computer Security, Rome, Italy, September 25-27, 1996, Proceedings*. 1996, pp. 290–303. DOI: 10.1007/3-540-61770-1_42.

[SGZ07] Jessica Staddon, Philippe Golle, and Bryce Zimny. “Web-Based Inference Detection”. In: *Proceedings of the 16th USENIX Security Symposium, Boston, MA, USA, August 6-10, 2007*. 2007.

[SJ92] Ravi S Sandhu and Sushil Jajodia. “Polyinstantiation for cover stories”. In: *Computer Security ESORICS 92*. Springer, 1992, pp. 307–328. DOI: 10.1007/BFb0013905.

[Sta03] Jessica Staddon. “Dynamic inference control”. In: *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery, DMKD 2003, San Diego, California, USA, June 13, 2003*. 2003, pp. 94–100. DOI: 10.1145/882082.882103.

[Swe02] Latanya Sweeney. “k-anonymity: A model for protecting privacy”. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (2002), pp. 557–570. DOI: 10.1142/S0218488502001648.

[TFE10] Tyrone S Toland, Csilla Farkas, and Caroline M Eastman. “The inference problem: Maintaining maximal availability in the presence of database updates”. In: *Computers & Security* 29.1 (2010), pp. 88–103. DOI: 10.1016/j.cose.2009.07.004.

[YL98] Raymond W. Yip and E. N. Levitt. “Data level inference detection in database systems”. In: *Proceedings of the 11th IEEE Computer Security Foundations Workshop, Rockport, Massachusetts, USA, June 9-11, 1998*. 1998, pp. 179–189. DOI: 10.1109/CSFW.1998.683168.

[YTM07] Ding Yixiang, Peng Tao, and Jiang Minghua. “Secure multiple xml documents publishing without information leakage”. In: *International Conference on Convergence Information Technology*. 2007, pp. 2114–2119. DOI: 10.1109/ICCIT.2007.135.