

# Privatezza e Protezione dei Dati - Papers

Francesco Fontana, Alessandro Marchetti, Alfredo Santarcangelo

2024

## 0.1 Introduction

# Chapter 1

## $k$ -anonymity

Contesto è il rilascio di *microdata*. De-identificazione non garantisce anonimità.

### 1.1 $k$ -anonymity e Table $k$ -anonime

Il concetto di  $k$ -anonymity cerca di catturare sulla Private Table (PT) il vincolo che i dati rilasciati dovrebbero essere associabili in maniera indistinguibile a non meno di un certo numero di respondent.

Il set di attributi disponibili esternamente e quindi sfruttabili per fare linking è chiamato *quasi-identifier*.

#### **Definizione 1 ( $k$ -anonymity requirement)**

*Ogni rilascio di data deve essere tale che ogni combinazione di valori del Quasi-Identifier può essere matchata in maniera indistinguibile con almeno  $k$  respondent.*

$k$ -anonymity richiede che ogni valore del *Quasi-Identifier* abbia almeno  $k$  occorrenze nella table rilasciata, come da def 1.1 che segue:

#### **Definizione 2 ( $k$ -anonymity)**

*Date una table  $T(A_1, A_2, \dots, A_m)$  e un insieme di attributi  $QI$ , Quasi-Identifier sulla table  $T$ :*

*$T$  soddisfa  $k$ -anonymity rispetto a  $QI$  se e solo se ogni sequenza di valori in  $T[QI]$  appare almeno con  $k$  occorrenze in  $T[QI]$ <sup>1</sup>*

Definizione 1.1 è sufficiente per  $k$ -anonymity. Applicazione di  $k$ -anonymity richiede una preliminare identificazione del *Quasi-Identifier*.

Il *Quasi-Identifier* dipende dalle informazioni esterne disponibili al recipiente poichè determina le capacità di linking dello stesso. Diversi *Quasi-Identifier* possono potenzialmente esistere per una data table.

Per semplicità a seguire nel paper si assume che:

---

<sup>1</sup> $T[QI]$  denota la proiezione con tuple duplicate degli attributi  $QI$  in  $T$

- PT ha unico *Quasi-Identifier*.
- *Quasi-Identifier* è composto da tutti gli attributi nella PT disponibili esternamente.
- PT contiene al massimo una sola tupla per ogni respondent.

$k$ -anonymity si concentra su due tecniche di protezione: *Generalization* e *Suppression*, le quali preservano la veridicità dei dati (diversamente da swapping e scrambling).

### 1.1.1 *Generalization*

Sostituzione dei valori di un attributo con valori più generali. Consideriamo:

- *Domain*: set di valori che un attributo può assumere.
- *Generalized domains*: contiene valori generalizzati e relativo mapping tra ogni domain e ogni sua generalizzazione.
- *Dom*: set di domini originali con le loro generalizzazioni.
- *Generalization relationship*  $\leq_D$ : dati  $D_i, D_j \in \text{Dom}$ ,  $D_i \leq D_j$  significa che i valori in  $D_j$  sono generalizzazioni dei valori in  $D_i$ .

$\leq_D$  definisce ordinamento parziale su *Dom* ed è richiesto nelle seguenti condizioni:

#### Condizione 1 (C1 - Determinismo nel processo di generalizzazione)

$\forall D_i, D_j, D_z \in \text{Dom}$ :

$$D_i \leq_D D_j, D_i \leq_D D_z \implies D_j \leq_D D_z \vee D_z \leq_D D_j^2.$$

#### Condizione 2 (C2 - )

Tutti gli elementi massimali di *Dom* sono singoletti (*singleton*)<sup>3</sup>.

- **DGH<sub>D</sub>** - *Domain Generalization Hierarchy*: gerarchia di ordinamento totale per ogni dominio  $D \in \text{Dom}$ .

Per quanto riguarda i valori nei domini consideriamo:

- *Value generalization relationship*  $\leq_V$ : associa ogni valore in  $D_i$  ad un unico valore in  $D_j$ , sua generalizzazione.
- **VGH<sub>D</sub>** - *Value Generalization Hierarchy*: albero dove
  - Foglie sono valori in  $D$ .
  - Radice è il valore, singolo, nell'elemento massimale di DGH<sub>D</sub>

---

<sup>2</sup>Questo comporta che ogni dominio  $D_i$  ha al massimo un solo dominio di generalizzazione diretta  $D_j$

<sup>3</sup>La condizione assicura che tutti i valori in ogni dominio possano essere generalizzati ad un singolo valore

### 1.1.2 Suppression

Consideriamo Suppressione di Tupla. "Modera" la *Generalization* quando un numero limitato di *outlier*<sup>4</sup> forzerebbe una generalizzazione elevata.

## 1.2 Generalizzazione $k$ -Minima

### Definizione 3 (Table Generalizzata con Suppressione)

Consideriamo  $T_i$  e  $T_j$  due table sugli stessi attributi.

$T_j$  è generalizzazione (con soppressione di tupla) di  $T_i$ , riportata come  $T_i \preceq T_j$ , se:

1.  $|T_j| \leq |T_i|$
2. Dominio  $\text{dom}(A, T_j)$  è uguale o una generalizzazione di  $\text{dom}(A, T_i)$ , dove  $A$  indica ogni attributo in  $T_{i,j}$
3. E' possibile definire funzione iniettiva che associa ogni tupla  $t_j \in T_j$  con una tupla  $t_i \in T_i$ , per la quale ogni attributo in  $t_j$  è uguale o generalizzazione del corrispondente in  $t_i$ .

### Definizione 4 (Distance Vector)

Siano  $T_i(A_1, \dots, A_n)$  e  $T_j(A_1, \dots, A_n)$  tali che  $T_i \preceq T_j$ .

il distance vector di  $T_j$  da  $T_i$  è il vettore

$$DV_{i,j} = [d_1, \dots, d_n]$$

dove ogni  $d_z, z = 1, \dots, n$  è la lunghezza dell'unico percorso tra  $\text{dom}(A_z, T_i)$  e  $\text{dom}(A_z, T_j)$  nella  $DGH_{D_z}$

### Corollario 1 (Ordine Parziale tra DV)

$DV = [d_1, \dots, d_n] \leq DV' = [d'_1, \dots, d'_n]$  se e solo se  $d_i \leq d'_i$  per  $i = 1, \dots, n$ .

Si costruisce una gerarchia di distance vectors come lattice (diagramma) corrispondente alla  $DGH_D$  come in fig. 1.1

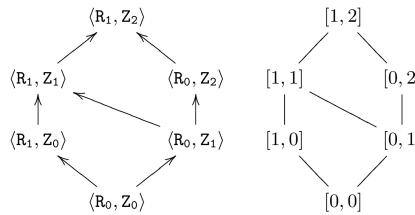


Figure 1.1

---

<sup>4</sup>TODO outlier

Per bilanciare tra perdita di precisione dovuta a *Generalization* e perdita di completezza dovuta a *Suppression* si suppone che data holder determini la soglia **MaxSup**, che indica il numero di tuple che possono essere soppresse.

**Definizione 5 (Generalizzazione *k*-minima con Soppressione)**

Siano  $T_i$  e  $T_j$  due table tali che  $T_i \preceq T_j$ , e sia **MaxSup** la soglia di soppressione accettabile scelt.  $T_j$  è una generalizzazione *k*-minima di  $T_i$  se e solo se:

1.  $T_j$  soddisfa *k*-anonymity applicando soppressione minima, ossia  $T_j$  soddisfa *k*-anonymity e:  $\forall T_z : T_i \preceq T_z, DV_{i,z} = DV_{i,j}, T_z$  soddisfa *k*-anonymity  $\implies |T_j| \geq |T_z|$ .
2.  $|T_i| - |T_j| \leq \text{MaxSup}$ .
3.  $\forall T_z : T_i \preceq T_z$  e  $T_z$  soddisfa le condizioni 1 e 2  $\implies \neg(DV_{i,z} < DV_{i,j})$ .

Ultima espressione rende meglio come  $DV_{i,z} \geq DV_{i,j}$ . Il concetto che esprime è che "non esiste un'altra *Generalization*  $T_z$  che soddisfi 1 e 2 con un *DV* minore di quello di  $T_j$ "

Diversi **preference criteria** possono essere applicati nella scelta della generalizzazione minimale preferita:

- **Distanza assoluta minima:** minor numero totale di passi di generalizzazione (indipendentemente dalle gerarchie di *Generalization* considerate).
- **Distanza relativa minima:** minimizza il numero relativo di passi di generalizzazione (passo relativo ottenuto dividendo per l'altezza del dominio della gerarchia a cui si riferisce).
- **Massima distribuzione:** maggior numero di tuple distinte.
- **Minima soppressione:** minor tuple soppresse (maggior cardinalità).

### 1.3 Classificazione tecniche di $k$ -anonymity

Classificazione in fig. 1.2.

Generalization	Suppression			
	<i>Tuple</i>	<i>Attribute</i>	<i>Cell</i>	<i>None</i>
<i>Attribute</i>	<b>AG_TS</b>	<b>AG_AS</b> $\equiv$ AG_	<b>AG_CS</b>	<b>AG_</b> $\equiv$ AG_AS
<i>Cell</i>	<b>CG_TS</b> not applicable	<b>CG_AS</b> not applicable	<b>CG_CS</b> $\equiv$ CG_	<b>CG_</b> $\equiv$ CG_CS
<i>None</i>	<b>_TS</b>	<b>_AS</b>	<b>_CS</b>	- not interesting

Fig. 8. Classification of  $k$ -anonymity techniques

Figure 1.2

Casi *not applicable* (**CG\_TS** e **CG\_AS**): supportare *Generalization* a grana fine (cella) implica poter applicare soppressione allo stesso livello.

Algorithm	Model	Algorithm's type	Time complexity
Samarati [26]	AG_TS	Exact	exponential in $ QI $
Sweeney [29]	AG_TS	Exact	exponential in $ QI $
Bayardo-Agrawal [5]	AG_TS	Exact	exponential in $ QI $
LeFevre-et-al. [20]	AG_TS	Exact	exponential in $ QI $
Aggarwal-et-al. [2]	_CS	$O(k)$ -Approximation	$O(kn^2)$
Meyerson-Williams [24] <sup>2</sup>	_CS	$O(k \log k)$ -Approximation	$O(n^{2k})$
Aggarwal-et-al. [3]	CG_	$O(k)$ -Approximation	$O(kn^2)$
Iyengar [18]	AG_TS	Heuristic	limited number of iterations
Winkler [33]	AG_TS	Heuristic	limited number of iterations
Fung-Wang-Yu [12]	AG_	Heuristic	limited number of iterations

Figure 1.3: Alcuni approcci a  $k$ -anonymity ( $n$  è numero di tuple in PT).

### 1.4 Algoritmo Samarati (AG\_TS)

Il primo algoritmo per garantire  $k$ -anonymity è stato proposto insieme alla definizione di  $k$ -anonymity. La definizione di  $k$ -anonymity è basata sul QI quindi l'algoritmo lavora solo su questo set di attributi e su table con più di  $k$  tuple.

Data una *DGH* ci sono diversi percorsi dall'elemento in fondo alla gerarchia alla radice. Ogni percorso è una differente *strategia* di generalizzazione. Su ogni percorso c'è esattamente una *Generalization* minima localmente (nodo più basso che garantisce  $k$ -anonymity).

In maniera naif si può cercare su ogni percorso il minimo locale per poi trovare il minimo globale tra questi ma non è praticabile per l'elevato numero di percorsi.

Per ottimizzare la ricerca si sfrutta la proprietà che salendo nella gerarchia la soppressione richiesta per avere  $k$ -anonymity diminuisce:

- Ogni nodo in  $DGH$  viene associato ad un numero, **height**, corrispondente alla somma degli elementi nel Distance Vector associato.
- Altezza di ogni DV nel diagramma (*distance vector lattice*  $VL$ ) si scrive come  $height(DV, VL)$ .

Se non c'è soluzione che soddisfi  $k$ -anonymity sopprimendo meno di  $MaxSup$  ad altezza  $h$  non può esistere soluzione che soddisfi ad una altezza minore.

L'algoritmo usa binary search cercando la minore altezza in cui esiste un  $DV$  che soddisfa  $k$ -anonymity rispettando  $MaxSup$  e ha come primo passo:

$$\text{Cerco ad altezza } \lfloor \frac{h}{2} \rfloor : \begin{cases} \text{trovo vettore che soddisfa } k\text{-anonymity} & \implies \lfloor \frac{h}{4} \rfloor \\ \text{altrimenti} & \implies \text{cerco in } \lfloor \frac{3h}{4} \rfloor \end{cases} \quad (1.1)$$

La ricerca prosegue fino a trovare l'altezza minore in cui esiste vettore che soddisfa  $k$ -anonymity con  $MaxSup$ .

#### 1.4.1 Evitare il calcolo delle table generalizzate

Algoritmo richiederebbe il calcolo di tutte le table generalizzate. Per evitarlo introduciamo il concetto di  $DV$  tra tuple.

##### Definizione 6 (Distance Vector tra tuple - Antenato Comune)

Sia  $T$  una table.

Siano  $x, y \in T$  due tuple tali che  $x = \langle v'_1, \dots, v'_n \rangle$  e  $y = \langle v''_1, \dots, v''_n \rangle$  con  $v'_i$  e  $v''_i$  valori in  $D_i$  con  $i = 1, \dots, n$ .

Il **distance vector** tra  $x$  e  $y$  è  $V_{x,y} = [d_1, \dots, d_n]$ . dove  $d_i$  è la lunghezza (uguale) dei due percorsi da  $v'_i$  e  $v''_i$  al loro comune antenato comune più prossimo  $v_i$  sulla  $VGH_{D_i}$ .

In altri termini ogni distanza in  $V_{x,y}$  è una distanza uguale dal dominio di  $v'_i$  e  $v''_i$  al dominio in cui sono generalizzati allo stesso valore  $v_i$ .

Allo stesso modo  $V_{x,y}$  per  $x, y \in T_i$  equivale a  $DV_{i,j}$  per  $T_i \preceq T_j$  per cui  $x$  e  $y$  vengono generalizzate alla stessa tupla  $t$ .

Per il momento il resto è delirio



## 1.5 Bayardo-Agrawal: *k-Optimize* (AG\_TS)

Approccio considera che la generalizzazione di attributo  $A$  su dominio **ordinato**  $D$  corrisponde ad un partizionamento del dominio dell'attributo in intervalli. Ogni valore del dominio deve comparire in un intervallo e ogni valore in un intervallo precede ogni valore degli intervalli che lo seguono.

Si assume quindi un ordine tra gli attributi del *Quasi-Identifier*. Inoltre associa un valore intero chiamato *index* ad ogni intervallo di ogni dominio degli attributi del *Quasi-Identifier*.

Una *Generalization* è quindi rappresentata come l'unione dei valori di indice per ogni attributo (il valore più piccolo in un dominio viene omesso perché comparirà sicuramente nella generalizzazione per quel dominio).

*k-Optimize* costruisce un *set enumeration tree* sul set  $I$  di valori di indice, con radice vuota. Ogni nodo figlio è costruito dal padre inserendo in coda un index di  $I$  maggiore degli altri index già presenti nel padre (per ordinamento totale).

La visita dell'albero permette di valutare con Depth First Search ogni nodo, prunando ogni nodo e tutti i suoi figli se questi non possono corrispondere a soluzioni ottimali. Nello specifico *k-Optimize* pruna un nodo  $n$  quando determina che nessuno dei suoi discendenti può essere ottimale. Data una funzione di costo l'algoritmo calcola un limite inferiore sul costo che può essere ottenuto sul subtree del nodo  $n$ , prunando se nodo a costo minore è già stato trovato.

Se un nodo viene prunato allora anche altri nodi, non per forza del sottoalbero, possono essere prunati: supponendo di prunare il nodo  $\{1,3\}$  in figura 1.4 allora posso prunare qualunque altro nodo contenente 1 E 3, come ad esempio  $\{1,2,3\}$ .

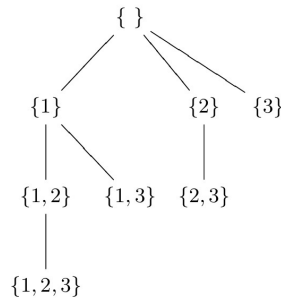


Figure 1.4: Set enumeration tree sull'insieme di indici  $I=\{1,2,3\}$

## 1.6 LeFevre-DeWitt et al.: Incognito (AG\_TS)

Idea di base è riassunta nella seguente definizione:

**Definizione 7 ( $k$ -anonymity dei subset di  $QI$ )**

*Se una table  $T$  con Quasi-Identifier  $QI$  composto da  $m = |QI|$  attributi soddisfa  $k$ -anonymity,  $T$  soddisfa  $k$ -anonymity anch per qualunque Quasi-Identifier  $QI'$  talce che  $QI' \subset QI$ .*

*Pertanto  $k$ -anonymity su un subset di  $QI$  è condizione necessaria (e non sufficiente) per  $k$ -anonymity di  $T$  sull'intero  $QI$ .*

Algoritmo esclude in anticipo alcune generalizzazioni della gerarchia con un calcolo a priori.

Strategia: bottom-up BFS sulla  $DGH$ . Incognito genera tutte le possibili table  $k$ -anonime minime secondo i seguenti passaggi:

1. (Iterazione 1): verifica  $k$ -anonymity per ogni singolo attributo nel *Quasi-Identifier*, scartando quelle che non soddisfano.
2. (Iterazione 2): combina a coppie le *Generalization* non scartate al passo 1 verificando  $k$ -anonymity.
3. (...)
4. (Iterazione  $m$ ): arriva a considerare l'intero set di attributi di  $QI$

Utilizzando approccio bottom-up, per la condizione citata prima, se una *Generalization* soddisfa  $k$ -anonymity allora anche sue ulteriori dirette generalizzazioni soddisfano  $k$ -anonymity e pertanto non sono considerate ulteriormente.

### 1.6.1 esempio Incognito

## Chapter 2

# Protezione Microdati

### 2.1 Introduzione

### 2.2 Macrodata vs Microdata

### 2.3 Classificazione delle tecniche di protezione riguardo il Disclosure dei microdati

Il controllo del disclosure dei microdati è un'importante problema pratico sia nel privato che nel pubblico.

La protezione dei microdati ha apparentemente due obiettivi tra di loro contrastanti. Da una parte si vuole evitare la re-identificazione, la quale accade ogni qualvolta l'informazione del respondent che appare nella table di microdati è identificata; il quale, è associata con le identità dei respondent corrispondenti.

Dall'altra invece le applicazioni di tecniche dovrebbero preservare le \*chiavi delle proprietà statistiche\* dei dati originali i quali sono state segnati come importanti dai destinatari dei dati (ossia chi riceverà i dati finali)

Precisamente data una table di microdati  $T$ , una tecnica di protezione dei dati dovrebbe trasformare questa table  $T$  in una table  $T_1$  in modo che:

1. il rischio che un utente malevolo possa usare  $T_1$  per determinare informazioni confidenziali o identificare un respondent, sia basso
2. l'analisi statistica su  $T$  e  $T_1$  possa produrre risultati simili

Generalmente i seguenti fattori contribuiscono al rischio di disclosure:

- L'esistenza di tuple altamente visibili (ossia tutte quelle tuple con caratteristiche che hanno caratteristiche uniche)

- Possibilità di matching tra le table di microdati e le informazioni esterne.
- L'esistenza di un alto numero degli attributi comuni che possono aumentare la possibilità di linking.

Mentre i fattori che minimizzano il rischio di disclosure sono:

- Una table di microdati contenenti un subset della popolazione intera. Questo implica che le informazioni di un respondent specifico, il quale potrebbe essere un utente malevolo potrebbe voler sapere, potrebbe non essere incluso nella table di microdati.
- Le informazioni specificate nelle table rilasciate, quindi pubbliche non sono aggiornate. Potrebbero cambiare nel tempo, indi per cui il linking con le informazioni esterne potrebbero non essere accurati
- Una table di microdati e le informazioni esterne contengono rumore, riducendo la probabilità di linking delle informazioni
- Una table di microdati e le risorse esterne possono contenere dati espressi in forme diverse, riducendo l'abilità di linking

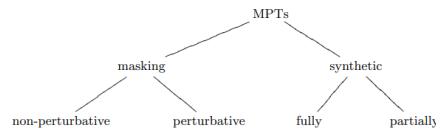


Fig. 3. Classification of microdata protection techniques (MPTs)

Figure 2.1: Classificazione delle tecniche di protezione dei microdati

Generalmente, per limitare il rischio di disclosure di una table di microdati è necessario sopprimere gli identifiers impliciti ed espliciti come step iniziale (e.g. SSN, Name)

Questo processo è noto come de-identificazione, quest'ultima non necessariamente rende le tuple anonime, siccome è possibile fare re-identificazione usando informazioni esterne.

Tipicamente le tecniche sono basate sul principio che la re-identificazione possa essere contrastata riducendo la quantità delle informazioni rilasciate, mascherando i dati (e.g., non rilasciando o perturbando i valori), o rilasciando valori plausibili modificati invece di quelli reali. Seguendo questo principio le tecniche si suddividono in due macro categorie:

- Tecniche di mascheramento: I dati originali sono trasformati producendo nuovi dati che non sono validi per le analisi statistiche in modo da preservare la confidenzialità dei respondent. Le tecniche di mascheramento si suddividono in:

- non perturbative: il dato originale non viene modificato, ma alcuni dati vengono soppressi o vengono rimossi dettagli.
- perturbativi: i dati originali sono modificati.
- Generazione dei dati sintetici: I set di tuple originale nelle table di microdati sono sostituiti da un nuovo set di tuple generati in modo di preservare le proprietà statistiche chiave del dato originale. Dal momento che le table dei microdati rilasciati contengono dati sintetici, il rischio di re-identificazione viene ridotto. Questi possono essere interamente sintetici (fully synthetic) o misti con i dati originali (partially synthetic).

Un'altra caratteristica importante delle tecniche di protezione è che possono essere usati su tipo di dati differenti. In particolare i tipi di dati possono essere categorizzati come

- *Continui*: Un attributo è detto continuo se le operazioni numeriche e aritmetiche sono definite sull'attributo. E.g. Data di nascita e temperatura sono attributi continui.
- *Categorici*: Un attributo è definito categorico se le operazioni aritmetiche non hanno alcun senso se fatto sull'attributo. E.g. Razza, su cui non si possono fare operazioni aritmetiche

Di seguito, sono descritti le tecniche di protezione dei microdati principali, e se sono applicabili a dati continui, categorici o entrambi.

Technique	Continuous	Categorical
Sampling	yes	yes
Local suppression	yes	yes
Global recoding	yes	yes
Top-coding	yes	yes
Bottom-coding	yes	yes
Generalization	yes	yes

Figure 2.2: Applicazione delle tecniche non perturbative su dati differenti

Technique	Continuous	Categorical
Resampling	yes	no
Lossy compression	yes	no
Rounding	yes	no
PRAM	no	yes
MASSC	no	yes
Random noise	yes	yes
Swapping	yes	yes
Rank swapping	yes	yes
Micro-aggregation	yes	yes

Figure 2.3: Applicazione delle tecniche perturbative su dati differenti

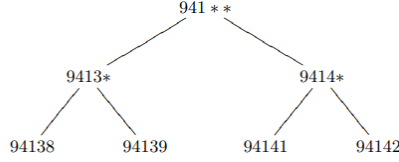
## 2.4 Tecniche di Mascheramento

Alcune tecniche di mascheramento

### 2.4.1 Tecniche non perturbative

Le tecniche non perturbative producono microdati eliminando dettagli dai dati originali. Di seguito alcune di esse:

- *Sampling*: I table dei microdati protetti includono solo tuple formate da sample (campioni) della popolazione totale. Siccome c'è dell'incertezza se uno specifico respondent sia presente nei sample, il rischio di re-identificazione viene ridotto. Per esempio possiamo decidere se pubblicare solo le tuple pari della table originale. Questa tecnica può operare solo su dati categorici
- *Local suppression*: Sopprime i valori degli attributi in modo da limitare la possibilità di analisi. Questa tecnica lascia uno spazio vuoto su alcuni attributi (sensitive cells) che contribuiscono in modo significativo al rischio di disclosure delle tuple coinvolte.
- *Global Recoding*: Il dominio degli attributi, ossia tutti i possibili valori, sono divisi intervalli disgiunti di grandezza uguale, e ogni intervallo viene associato a una label. La table dei microdati protetti sono ottenuti sostituendo i valori degli attributi con le label associate agli intervalli corrispondenti. Questa tecnica riduce i dettagli della table e quindi riduce il rischio di re-identificazione. Sono presenti due tecniche particolari di recoding, ossia *Top-Coding* e *Bottom-Coding*, qui sotto descritte:
  - *Top-Coding*: E' basata sulla definizione di limite superiore (upper limit), chiamato top-code, per ogni attributo che deve essere protetto. Ogni valore maggiore di questo valore è sostituito dal top-code. Questa tecnica può essere applicata agli attributi categorici che possono essere ordinati linearmente come gli attributi continui.
  - *Bottom-Coding*: Simile al top-coding ma definisce il limite inferiore (lower limit). Quindi, ogni valore più basso di questo non verrà ripubblicato e verrà sostituito con il bottom-code.
- *Generalization*: Consiste nel rappresentare i valori di un dato attributo usando valori più generali. Questa tecnica è basata sulla definizione di generalizzazione gerarchica, dove i valori più generali sono alla radice della gerarchia e le foglie corrispondono ai valori specifici. Un processo di generalizzazione perciò procede con la sostituzione dei valori rappresentati dai nodi foglia con dei valori al nodo superiore.



**Fig. 6.** Generalization hierarchy for attribute ZIP

Figure 2.4: Generalizzazione gerarchica

### 2.4.2 Tecniche perturbative

Con le tecniche perturbative le table dei microdati vengono modificate per la pubblicazione. Le modifiche possono portare a combinazioni dei valori originali uniche le quali svaniscono non appena vengono introdotte nuove combinazioni.

- *Resampling*: Questa tecnica sostituisce i valori degli attributi sensibili continui con un valore medio computato sul numero di samples della popolazione originaria. Precisamente, assumiamo che  $N$  sia il numero di tuple presenti in una table di microdati, e  $S_1, \dots, S_t$  con  $t$  i sample di dimensione  $N$ . Ogni sample viene rankato indipendentemente e successivamente viene calcolata la media del  $j_{esimo}$  valore ranked. I valori medi ottenuti sono riordinati prendendo in considerazione l'ordine dei valori originali, seguendo quindi l'ordinamento della tabella iniziale.

Microdata Protection									
$S_1$	$S_2$	$S_3$	$S_4$		$S_1$	$S_2$	$S_3$	$S_4$	Average
260	220	170	210		170	150	170	170	165
170	280	290	190		170	180	185	185	180
200	210	220	230		185	190	190	190	188.75
280	310	270	200		190	210	200	200	200
190	290	185	185		200	220	220	210	212.5
185	180	300	260		200	265	250	220	233.75
200	285	250	220		200	270	260	230	240
290	265	260	290		260	280	270	230	260
170	150	190	230		280	285	270	260	273.75
300	270	270	310		290	290	290	290	290
200	298	200	170		300	310	300	310	305
(a) Initial samples					(b) Ordered samples				
Original value ( $S_1$ )					Released value				
260					260				
170					165				
200					212.5				
280					273.75				
190					200				
185					188.75				
200					233.75				
290					290				
170					180				
300					305				
200					240				
(c) Released data									

**Fig. 8.** An example of resampling over attribute Chol

Figure 2.5: Esempio di resampling

- *Rounding*: Simile alla tecnica usata anche per la protezione dei macrodati ed è applicabile solo agli attributi continui. Sostituisce i valori originali con dei valori approssimati, quest'ultimi sono scelti da un set di *rounding points*  $p_i$  ognuno il quale definisce il *rounding set*. Per esempio:

- ***Rounding Points*** possono essere scelti come multipli di un valore base  $b$ , con  $b = p_{i+1} - p_i$ .
- ***Rounding Sets*** definiti come

$$* \begin{cases} [p_i - \frac{b}{2}, p_i + \frac{b}{2}) \text{ con } i = 2, \dots, r-1 \\ [0, p_1 + \frac{b}{2}) \\ [p_r - \frac{b}{2}, X_{max}], \text{ con } X_{max} \text{ il valore più grande possibile} \end{cases}$$

\* rispettivamente per  $p_1$  e  $p_r$

\* Un valore  $v$  di  $X$  viene sostituito dal round point corrispondente al rounding set dove è contenuto  $v$ .

- *Random Noise*: Perturba gli attributi sensibili aggiungendo o moltiplicando una variabile casuale con una distribuzione. Il rumore additivo (additive noise) è preferito al rumore moltiplicativo (multiplicative noise) e viene formalmente definito così:

Poniamo  $X_j$  come la  $j$ -esima colonna della table dei microdati originaria corrispondente a un attributo sensibile e supponiamo che ci siano  $N$  tuple. Ogni valore  $X_{ij}$ , con  $i = 1, \dots, N$ , viene sostituito con  $x_{ij} + \epsilon_{ij}$ , dove  $\epsilon_j$  è il vettore degli errori normalmente distribuiti ricavati da una variabile casuale con la media uguale a 0, in generale, con una varianza che sia proporzionale agli attributi originari

*Forse da approfondire*

1

- *Swapping*: Consiste nel modificare il subset di una table di microdati facendo lo swapping dei valori di un set formato degli attributi sensibili, chiamato *swapped attributes*, tra coppie di tuple selezionate (le coppie sono selezionate secondo un criterio ben definito). Intuitivamente, questa tecnica riduce il rischio di re-identificazione perché introduce dell'incertezza sul valore reale legato al dato del respondent. Anche se questa tecnica è facile da applicare, generalmente ha lo svantaggio di non preservare le proprietà dei sottodomini. Inizialmente la tecnica originale fu presentata solamente per gli attributi categorici.

---

<sup>1</sup>N.B  $\epsilon$  viene usato per perturbare i dati ma lasciando intatta la correlazione con la tabella iniziale



- *Micro-Aggregation (o Blurring)*: Consiste nel raggruppare le tuple individuali in piccole aggregazioni di una dimensione fissata  $k$ : verrà pubblicata la media di ogni aggregato invece dei valori individuali. Anche se esistono diverse funzioni per calcolare la similarità, può essere difficile trovare una soluzione di raggruppamento ottimale. Ci sono diverse varianti di *micro-aggregation*, per esempio la media può sostituire il valore originale anche solo per una singola tupla nel gruppo o per tutte. Molti attributi possono essere protetti grazie a *micro-aggregation* usando lo stesso o un diverso raggruppamento e, la dimensione del gruppo può essere variabile o fissata ad un minimo.

## 2.5 Tecniche per la Generazione di Dati Sintetici

La generazione dei dati sintetici è una valida alternativa per la protezione dei microdati. Il principio base su cui queste tecniche sono basate riguarda la non correlazione tra i contenuti statistici dei dati e l'informazione provvista da ogni respondent, quindi un modello ben rappresentante il dato potrebbe sostituire il dato stesso. Un requisito molto importante per la generazione dei dati sintetici, che rende il processo di generazione un problema importante, è che i dati sintetici e il dato originale potrebbero presentare la stessa qualità delle analisi statistiche. Il vantaggio maggiore di questi tipi di tecniche è che i dati sintetici rilasciati non sono collegati a nessun respondent, indi per cui il rilascio non può essere soggetto al fenomeno di re-identificazione; questo, inoltre, lascia ai proprietari dei dati di porre maggiore attenzione sulla qualità del dato rilasciato, invece che al problema di re-identificazione.

## 2.6 Misure per valutare la confidenzialità e l'utilità dei Microdati

Come discusso precedentemente, ci sono svariate tecniche per la protezione dei microdati. Le performance di queste tecniche di protezione sono valutate in termini di *information loss* e *disclosure risk*.

- *Information loss* è la quantità di informazioni che esistono nei microdati originali e che, a causa delle tecniche di protezione, questo non si verifica nei microdati protetti.

## Chapter 3

# Cloud Security: Issues and Concerns

### **3.1 Summary**

Il presente paper si pone l'obiettivo di spiegare come il garantire la sicurezza significa anche assicurare la *confidenzialità*, *integrità* dei dati e anche la loro *disponibilità*, CIA in una parola.

### **3.2 Introduzione**

Nella prima parte

### **3.3 CIA nel Cloud**

### **3.4 Problemi e Sfide**