Security Deciding in Publishing Views Based on Entropy

Zhang Kun, Pan De-fen
Hebei University of Science and Technology, Shijiazhuang, Hebei, China
euphkun@163.com

Abstract: Publishing views have been a convenient tool for data exchange. But as the increasing of data exchange technology, the security in publishing views has been paid more attention to. Inspired by the knowledge of probabilistic database, this paper erects a probability model of security deciding in publishing views, and it gives the formularized definition of publishing views, private information, prior knowledge and so on. Then a new deciding theorem based on entropy is provided, and introduce techniques for measuring the magnitude of information disclosures in publishing view. At last it is proved by the experiment.

Key words: Publishing Views; Probabilistic Database; Information Dependency；Entropy

1. INTRODUCTION

With the development of database technology, data exchange frequency and quantity increase continually, the problem of information disclosure is outstanding day by day in the view publishing process, so guaranteeing security of published view becomes a new subject of database security. As enterprises collect and maintain increasing amounts of personal data, individuals are exposed to greater risks of privacy breaches and identity theft. Many recent reports of personal data theft and misappropriation highlight these risks. As a result, many countries have enacted data protection laws requiring enterprises to account for the disclosure of personal data they manage [1, 2, 3, 4]. Hence, modern information systems must be able to track who has disclosed sensitive data and the circumstances of disclosure.

The following scenario illustrates a practical problem of the information disclosure.

The threaten of data exchange Consider a manufacturing company that needs to exchange XML messages with several partners. Each message type is a dynamic view that computes on request some information about the company's manufacturing data: V_1 contains detailed information about parts for a specific product, to be exchanged with suppliers; V_2 contains detailed information about products' features, options, and selling prices, to be exchanged with retailers and customers; while V_3 provides labor cost information to be sent to a tax consulting firm. The company wants to keep secret the internal manufacturing cost for its products, which can be expressed in terms of a query S. Conventional protection mechanisms can ensure that each message is received and read only by authorized users, but are powerless beyond that point. At a minimum, the company would like to audit each of the three views and ensure that it doesn't disclose the secret information to its authorized recipient. But in addition the company would like to understand if any information is disclosed when views are combined (colluded), for example when V3 is accidentally or intentionally sent to a supplier, or when the tax consulting firm merges with one of the customers. None of these tasks are performed today, manually or automatically, because there is no clear understanding of when and how much information is disclosed at the logical level[5].

The threaten of different edition view publishing Alice has already published a view U. This already leaked some information about a secret query S, but was considered an acceptable risk by Alice. Now she wishes to publish an additional view V. Does V disclose any additional information about S over what was already disclosed by U?

2. Related Work

Traditional security mechanisms protect

QShine 2008, July 28-31, 2008, Hong Kong, Hong Kong.
 ISBN 978-963-9799-26-4
DOI 10.4108/ICST.QSHINE2008.4262

data at the physical level. For example, firewalls and other perimeter mechanisms prevent the release of raw data, as do conventional access controls for file systems and databases. In data exchange, however, such mechanisms are limited since they can only protect the data up to the first authorized recipient. When data is exchanged with multiple partners, information may be unintentionally disclosed, even when all physical protection mechanisms work as intended. As an extreme example, Sweeney proved this [6] when she retrieved the privileged medical data of William Weld, former governor of the state of Massachusetts, by linking information from two publicly available databases, each of which was considered secure in isolation.

The problem of auditing a log of past queries and updates by means of an audit query that represents the leaked data has been studied in [7]. The suspicious queries are identified by finding past queries in the log whose results depend on the same "indispensable" data tuples as the audit query; a tuple is considered indispensable for a query if its omission makes the result of the query different. The indispensability of a tuple bears resemblance to the notion of a critical tuple in [5]. However, given some sensitive data, it is often difficult to formulate a concise audit query with near-perfect recall and precision. Moreover, the tuples in the sensitive table may have undergone a certain amount of arbitrary perturbation. Finally, the number of suspicious queries produced can be very large, necessitating an ordering based on relevance for an auditor's investigation. In addition Rakesh Agrawal presented an auditing methodology that ranks potential disclosure sources according to their proximity to the leaked records. Given a sensitive table that contains the disclosed data, methodology prioritizes by relevance the past queries to the database that could have potentially been used to produce the sensitive table [8].

Database watermarking [9] has also been proposed to track the disclosure of information. Database fingerprinting [10] can additionally identify the source of a leak by injecting different marks in different released copies of the data. Both the techniques require data to be modified to introduce a pattern and then recover the pattern in the sensitive data to establish disclosure. These techniques depend on the availability of a set of attributes that can withstand alteration without significantly degrading their value. They also require that a large portion of the pattern is carried over in the sensitive data.

Oracle [11] offers a "finegrained auditing" function where the administrator can specify that read queries should be logged if they access specified tables. This function logs various user context data along with the query issued, the time it was issued, and other system parameters such as the "system change number". Oracle also supports "flashback queries" whereby the state of the database can be reverted to the state implied by a given system change number. A logged query can then be rerun as if the database was in that state to determine what data was revealed when the query was originally run. However, there does not appear to be any automated facility to determine which queries should be audited.

3. The Security-Deciding Model

We assume a standard relational schema consisting of several relation names R1, R2, . . ., each with a set of attribute names. Our research is based on the standard relational schema.

In Probabilistic database, a database instance is regarded as a real world. In the schema, a database instance is denoted by I, and the probabilistic of the database instance is P(I). Let define all database instance as INST=$\{I_1, I_2, I_3 \ldots, I_n\}$. Any tuple t_i in the INST can be seen an incident in probabilistics. In the security –deciding model, if these tuples are independent on others, that is $P(t_1 \wedge t_2)=P(t_1)\times P(t_2)$.

Let's define $V=\{v_1, v_2, v_3\ldots,v_n\}$ as publishing view. In the publishing view, v_i (i=(1,2.,.n)) is a tuple of V, $A_1,A_2,..,A_m$ is a set of attribute. Let $VD=\{VD_1, VD_2, \ldots VD_m\}$be the finite domain, which includes all values that can occur in any attributes in any of the relations on the publishing view. For example VD may be the set of decimal numbers up to an upper bound, and all strings up to a given length. In a particular setting we may consider further restricting VD, e.g. to include only valid disease names, valid people names, or valid phone numbers.

Private information: Use boolean query Q to define private information, boolean query would get two result that is 'true' and 'false'. $S=\{S_1,S_2,S_3\ldots\}$ is the private information which is a 2-dimensional table. S_i represent the tuples in I_i which make the result of query Q be 'true'. We also can consider it as a relational schema, and see A1, A2, ..,A_m as each set of attribute names of private information S; $SD=\{SD_1, SD_2, \ldots SD_m\}$is the finite domain of INST. Here, we call Q private query.

Prior knowledge: The prior knowledge K is the information that you can't get it by publishing view V. However, the K is relevant to view V, that is, anyone can get some information of V by analyzing prior knowledge K.

In the security-deciding model, prior knowledge test which we call K test will be considered. Firstly, The test will check a single tuple. For example, if attacker want to steal the information of person in one college. In the past, the attacker knew Wang, and Wang is a student, so when he steals the information t=(Wang department of financial affairs …), the attacker can quickly infer that t is not in the database instance. Secondly, it will check different tuples. There are two relationship in different tuples, positive relationship and negative relationship. Positive relationship: a couple of tuples $\{t_i, t_j\}$ must be in the same database instance, negative relationship: the tuples t_i and t_j must not be in the same database instance.

Database instance set: Use Tup(D)=Tup(VD) ∪ Tup(SD) to represent all tuples which pass K test, and we can use all tuples composition in Tup(D) to generate a database instance set INST. If the INST passes K test, we can get the sub(INST) which is the sub set of INST. In the security-deciding model, sub(INST) is the database instance set. In the situations that attackers are different, that is the values of K are different, there will be different database instance sets (sub(INST)).

Definition 1. In a limited domain D, if P(t) represents the probability of any tuple in Tup(D) occurs in database instance; sub(INST) is the database instance set which pass the K test, the number of the sub(INST) is k_n; the Probabilistic of database instance in sub(INST) is presented by P(I); $\overline{sub(INST)}$ is the database instance set which don't pass the K test, it's database instance probability is presented by P(I'); Therefore, we have a probability:

$$P(I)=\prod_{t\in sub(INST)}P(t)\prod_{t\notin sub(INST)}(1-P(t))+\frac{\sum_{I\in sub(INST)}P(I')}{k_n}$$

Definition 2. For every pair of tuples $s_i \in S$ and $v_i \in V$, define a d-dimensional comparison vector $\gamma = \gamma(s_i, v_i)$ such that $\gamma^j = 1$ if the tuples match on the j^{th} attribute and 0 otherwise. If the j^{th} attribute is missing in one of the tuples, let $\gamma^j = *$:

$\gamma(s_i,v_i)=<\gamma^1, \gamma^2, \ldots, \gamma^d>$:

$$\forall\ j=1\ldots d:\ \gamma^j=\begin{cases}1 & S_i^j = V_i^j\\ 0 & S_i^j \neq V_i^j\\ * & \text{missing } S_i^j \text{ or } V_i^j\end{cases}$$

Overall, we have $|S|\times|V|$ vectors $\gamma(s_i, v_i)$, one for each pair of tuples. Let Γ denote the $|S|\ |V|\times d$ matrix of all comparison vectors. We shall define a probabilistic model that describes the distribution of these vectors. The model is centered on the notion of true matching between

two tuples. We assume that there is an unknown function Match:

$$S \times V \rightarrow \{M, U\}$$

Here "M" means "tuples match" and "U" means "tuples do not match".

4. Security Deciding in Publishing Views

The theory of security deciding in publishing views is enlightened by "query-view security". The attacker knows prior knowledge K, the domain D, and the probabilistic of database instance P(I) and publishing view V. Attacker's aim is to infer the private information S by these factors. To realize the security deciding in publishing views, we introduce the concept of entropy. In the information system, information entropy is always used to judge the indefinity among variables.

Next, the definition of private information entropy, publishing view entropy, and union entroty will be provided.

Definition 3. If private query is denoted by Q, private information is $S=\{S_1,S_2,S_3\ldots\}$, S_i represent the tuples in I_i which make the result of query Q be 'true', we can get private information entropy :

$$H(Q)=-\sum_{Q(I)=true} p(I)\log p(I)=-\sum_{Si\in S\wedge Si=\phi} P(S_i)\log P(S_i)$$

Definition 4. If publishing view is denoted by $V=\{v_1, v_2, v_3\ldots,v_n\}$, v_i (i=(1,2.,.n)) is a tuple of V, we can get publishing view entropy :

$$H(V)=-\sum_{V\in I} p(I)\log p(I)=-\sum_{Vi\in V\wedge Vi=\phi} P(V_i)\log P(V_i)$$

Definition 5. If private query is denoted by Q, private information is $S=\{S_1,S_2,S_3\ldots\}$, publishing view is $V=\{v_1, v_2, v_3\ldots,v_n\}$, we can get the union entropy of private information and publishing view:

$$H(Q,V)=-\sum_{Si\in S}\sum_{Vi\in V} p(S_i,V_i)\log p(S_i,V_i)$$

4.1 The Decision of Absolute Secure View

The absolute secure view is that the attacker can't infer any information in private information S, even if he knows publishing view V and prior knowledge K.

Theorem 1. If private query is denoted by Q, publishing view is V, The prior knowledge is K (K is indicated by a 2- dimensional table U).The union entropy of Q, V and U equals the sum of the union entropy Q, U and the union entropy U, V. That is:

$$H(Q, U, V)+H(U)=H(Q, U)+H(U, V) \qquad (4\text{-}1)$$

If attacker gets prior knowledge K, V is an absolute secure view for private query Q.

Proof. According to definition 3, we get private information entropy:

$$H(Q)=-\sum_{Q(I)=true} p(I)\log p(I)=-\sum_{Si\in S\wedge Si=\phi} P(S_i)\log P(S_i)$$

If the equation H(Q, U, V)+H(U)=H(Q, U)+H(U, V) is correct, then:

$$-\sum_{Ui\in U}\sum_{Vi\in V}\sum_{S_i\in S} p(S_i,U_i,V_i)\log p(S_i,U_i,V_i)$$

$$+(-\sum_{Ui\in U} p(U_i)\log p(U_i))$$

$$=-\sum_{Si\in S}\sum_{Ui\in U} p(S_i,U_i)\log p(S_i,U_i)$$

$$+(-\sum_{Ui\in U}\sum_{Vi\in V} p(U_i,V_i)\log p(U_i,V_i))$$

If S and U, V is independent, that is: P (S_i, U, V)=P(S_i)P(U, V)

The left of the equation (4-1) is:

$$-\sum_{Ui\in U}\sum_{Vi\in V}\sum_{S_i\in S} p(U_i,V_i)(\log p(U_i,V_i)+\log p(S_i))$$

$$+\log p(U_i,)p(V_i)$$

$$=-\sum_{Ui\in U}\sum_{Vi\in V}\sum_{S_i\in S} p(U_i,V_i)\log p(U_i,V_i)+\log p(S_i)$$

$$+\log p(U_i,)p(V_i) \qquad (4\text{-}2)$$

If S and U is independent, that is: P (S_i, U)=P(S_i)P(U)

The right of the equation (4-1) is:

$$-\sum_{Ui\in U}\sum_{Vi\in V}\sum_{S_i\in S} p(S_i,U_i)\log p(S_i,U_i)p(V_i)$$

$$+p(U_i,V_i)\log p(U_i,V_i)p(S_i)$$

$$=-\sum_{Ui\in U}\sum_{Vi\in V}\sum_{S_i\in S}\log p(S_i)+p(V_i)\log p(U_i)$$

$$+p(U_i,V_i)\log p(U_i,V_i) \qquad (4\text{-}3)$$

Because of formula 4-2 equal to formula 4-3

That is: H (Q, U, V)+H(U)=H(Q, U)+H(U, V).

We can get P (S_i, U, V)P(U)=P(Q, U)P(U, V), this can show S and U, V is independent. So if attacker knew prior knowledge K, he can't get any information of S. Then V is an absolute secure view for private query Q.

When a view is published first time, attacker does not have any prior knowledge K, which is 2- dimensional table U is empty. Then we can simplify Theorem 1，and use Theorem 2 to decide secure of publishing view V.

Theorem 2. If private query is denoted by Q, publishing view is V. The union entropy of two variables equals the sum of entropy Q and V. That is:

$$H(Q, V)=H(Q)+H(V)$$

Proof. According to definition 3, we get private information entropy:

$$H(Q)=-\sum_{Q(I)=true} p(I)\log p(I)=-\sum_{Si\in S\wedge Si=\phi} P(S_i)\log P(S_i)$$

According to definition 5, we get private information entropy:

$$H(Q,V)=-\sum_{Si\in S}\sum_{Vi\in V} p(S_i,V_i)\log p(S_i,V_i)$$

If $H(S_i, V)=H(S_i)+H(V)$, that is:

$$-\sum_{Si\in S}\sum_{Vi\in V} p(S_i,V_i)\log p(S_i,V_i)=$$

$$-\sum_{Si\in S} p(S_i)\log p(S_i)-\sum_{Vi\in V} p(v)\log p(v)$$

$$=-\sum_{Si\in S}\sum_{Vi\in V} p(V_i)p(S_i)(\log p(S_i)+\log p(V_i))$$

$$=-\sum_{Si\in S}\sum_{Vi\in V} p(S_i,V_i)\log p(S_i,V_i)$$

We can get $P(S_i,V)=P(S_i)P(V)$, incident Q is independent on V. That is publishing view V is safe.

4.2 MEASURING DISCLOSURES

In chapter 3, we knew "M" means "tuples match to S and V" and "U" means "tuples do not match to S and V".

The record linkage process attempts to classify each tuple pair s_i, v_i as either M or U, by observing comparison vectors γ (s_i, v_i). This clarification is possible because the distribution of γ (s_i, v_i) for M-labeled tuple pairs is very different from its distribution for U-labeled pairs. Let us define two sets of conditional probabilities:

$$m(\gamma) = P[\ \gamma(s_i, v_i)\ |\ s_i, v_i \in M\ ];$$

$$u(\gamma) = P[\ \gamma(s_i, v_i)\ |\ s_i, v_i \in U\ ].$$

Assume that the M-label and U-label are themselves independently assigned to each pair, with probability $p\in[0, 1]$ to assign an M-label and probability $1-p$ to assign a U-label. Then, the probability that some unlabeled pair s, v has a comparison vector γ equals:

$$P[\gamma(s, v) =\gamma] = p\ P[\gamma\ |M] + (1 - p)P[\gamma\ |U]$$

$$= pm(\gamma) + (1 - p)\ u(\gamma)$$

Relevance computation of S and V. If private query is denoted by Q, publishing view is V, The prior knowledge is K(K is indicated by a 2- dimensional table U). The value of Relevance computation of S and V is H_{sec1}, $H_{sec1}=H(Q, U, V)+H(U)-H(Q, U)-H(U, V)$。

Matching computation of S and V. We shall assume that, all pairs and their comparison vectors $\gamma\in\Gamma$ with index $k = 1. . . K_B$ are left unlabeled, whereas all γ_k with index $k = K_{B+1}. . . |S|\ |V|$ are labeled with U. Now one can use maximum likelihood estimation to search for m(γ) and u(γ) that maximize the probability. This estimation is carried out through the EM algorithm [12, 13]. Before we turn to EM, let us denote by $z_k \in \{0, 1\}$ a random variable such that $z_k = 1 <==> $ Match $<s_i, v_i>$. Computation of the expectations z_k for non-blocked pairs is the "E-step" of the EM algorithm, and the subsequent recomputation of next-iteration parameters p, m (γ_k), u(γ_k) to maximize is the "M-step." Denote the n^{th} iteration parameters by p_n, $m_n(\gamma_k)$, $u_n(\gamma_k)$; So, we follow [8, 14, 15] and assume that individual attribute matchings are conditionally independent given the "true matching" label M or U. For $\gamma \in \{0, 1\}^d$ we get:

$$m(\gamma)=\prod_{j=1}^{d}(m^j)^{\gamma^j}(1-m^j)^{1-\gamma^j}$$

$$m^j=P[\gamma^j=1\ |\ M]$$

$$u(\gamma)=\prod_{j=1}^{d}(u^j)^{\gamma^j}(1-u^j)^{1-\gamma^j}$$

$$u^j = P[\gamma^j = 1 \mid U]$$

The value of matching computation of S and V is H_{sec2}, $H_{sec2} = \log \frac{m(\gamma)}{u(\gamma)}$.

Our standard security of publishing view is very strong. In many applications we can tolerate deviations from this strong standard, as long as the deviations are not too large. We discuss here a measure of information disclosure that attempts to quantify the amount. Our definition of leakage is the following:

$$Leak = xH_{sec1} + yH_{sec2} \quad (4\text{-}4)$$

Here, We call H_{sec1} security coefficient 1 and H_{sec2} security coefficient 2. The variable x and y are determined by expert.

5. EXPERIMENTAL RESULTS

Now we have done some experiments to prove the theory of security deciding in publishing views, and certify the validity of the above theory.

We implemented the experiment as C++ applications and performed experiments on a Windows XP Professional Version 2002 SP 2 workstation with 2.4GHz Intel Xeon dual processors, 2 GB of memory.

The data set that we used is the database of a faculty's information in a college. We set P(t)=0.5, the probabilities of all database instances are equal. We set x=1, y=1.

Now we analyze the experiment. The inputs of the experiment are prior knowledge K, publishing view V, and private query Q. According to domain D, the program compute database instance set sub (INST). According to Definition 3～Definition 5, we can get H(Q, U, V), H(U), H(Q, U) and H(U, V). As Theorem 1 demonstrates, we can get the security of publishing view V. In order to measure the disclosure, we can get "leak" by formula (4-3).

A faculty's information in a college is descript with language Datalog: College (name, age, profession) , publishing view V(profession):-College (name, age, profession); private query: Q←("Wang", "20", "tutor"),then we can got S =("Wang", "20", "tutor").

N_{name}、N_{age}、$N_{profession}$ indicate the size of attribute of name, age, profession.

$n = N_{name} + N_{age} + N_{profession}$

The experimental results are as Table 5-1:

Table 5-1 Result of experiment

No.	N_{Name}	N_{Age}	$N_{Profession}$	N	leak
1	1	1	1	3	0.5
2	3	2	5	10	0.457
3	3	2	7	12	0.357
4	5	3	8	16	0.356
5	5	3	10	18	0.326
6	5	4	12	21	0.319
7	6	5	14	25	0.271
8	5	8	15	28	0.255
9	8	6	16	30	0.246

From the table, we can see the leak value varies with quantity of experiment data N, figure 5-1 describes this variety and figure 5-2 describes the leak value varies with $N_{profession}$.

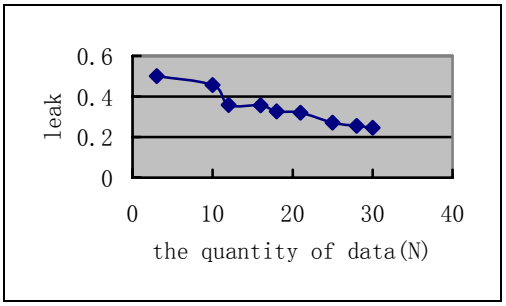

Figure 5-1 Variational curve of leak value with quantity of experiment data

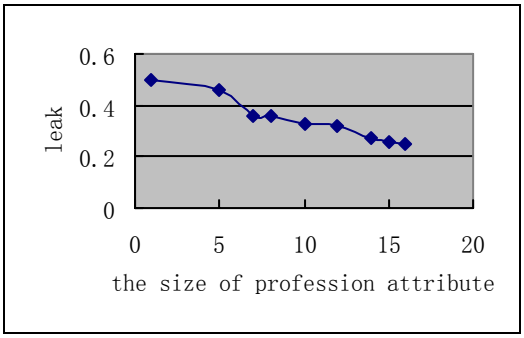

Figure 5-2 Variational curve of leak value with quantity of $N_{profession}$

Analysis of the experiment: with the increase of

variable's amount, the relevance of publishing view and private information will decrease. So the disclosure of information will also decrease. From table 5-1, we can see that with the increase of the size of publishing attribute, the leak will decline quickly. The reason is that attacker always infers the private information through publishing view, so when the size of $N_{profession}$ increases, the probalility that attacker can get a single tuple will decrease. For example, if the publishing view V only has one tuple ("tutor"), and $N_{profession}$=1; Q ← Employee("Wang" , "20" , "tutor"). So you can get sub(INST) ={ Φ ,{("Wang","20","tutor")}},and H(V)=0.5, H(Q)=0.5, H(Q,V)=0.5, leak=0.5. According to theorem 1, V is not a safe publishing view. But with the increase of $N_{profession}$, H (V) will decrease. H_{sec1} and H_{sec2} will be close to zero. That is the value of leak will be close to zero, the insecurity of the publishing view will decrease. In the real world, when S is private information, we will still publish the data of profession attribute.

6. CONCLUSION

We have presented a novel definition of security for analyzing. The information disclosure of publishing views and shown several important results. We argue that it is indispensable for developing practical tools for monitoring information disclosure. We have already shown the theory of security deciding in publishing views (Theorem1 and Theorem2) .We provide a method of measure leakage (formula (4-4)). We believe these results may be a basis for logically security of publishing views in the future.

7. REFERENCES

[1] Personal information protection and electronic documents act, second session, thirty-sixth parliament, 48-49 elizabeth ii, 1999-2000, statutes of canada, 2000.

[2] Australian privacy act of 1998, 1998. http://www.privacy.gov.au/ACT/privacyact/

[3] European Union Directive on Data Protection, Official Journal of the European Communities, 1995.

[4] Health insurance portability and accountability act of 1996, united states public law 104-191, 1996.

[5]G.Miklau and D.Suciu. A formal analysis of information disclosure in data exchange. University of Washington Technical Report,Dec 2003.

[6] L. Sweeney. k-Anonymity: a model for protecting privacy. Int. J. on Uncertainty, Fuzziness and Knowledge- based Systems, 10(5), 2002.

[7] N. R. Adam and J. C. Wortman. Security-control methods for statistical databases. ACM Computing Surveys, 21(4):515–556, Dec. 1989.

[8] Rakesh Agrawal, Alexandre Evfimievski and Jerry Kiernan. Auditing Disclosure by Relevance Ranking SIGMOD, June 11–14, 2007

[9] R. Agrawal, P. J. Haas, and J. Kiernan. Watermarking relational databases. VLDB Journal, 12(2):157–169, August 2003.

[10] Y. Li, V. Swarup, and S. Jajodia. Fingerprinting relational databases: Schemes and specialties. IEEE

Trans. Dependable Sec. Computing (TDSC), 2(1):34–45, 2005.

[11] A. Nanda and D. K. Burleson. Oracle Privacy Security Auditing. Rampant, 2003.

[12] H. O. Hartley. Maximum likelihood estimation from incomplete data. Biometrics, 14:174–194, 1958.

[13] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, 39(1):1–38, 1977.

[14] L. A. Goodman. Exploratory latent structure analysis using both identifiable and unide-ntifiable models. Biometrika, 61:215–231, 1974.

[15] M. A. Jaro. Advances in record linkage methodology as applied to matching the 1985 census of Tampa, Florida. Journal of the American Statistical Association, 84:414–420, 1989.