Investigative Profiling with Computer Forensic Log Data and Association Rules

Tamas Abraham and Olivier de Vel
Information Networks Division,
Defence Science and Technology Organisation,
PO Box 1500, Edinburgh SA 5111, Australia
{tamas.abraham,olivier.devel}@dsto.defence.gov.au

Abstract

Investigative profiling is an important activity in computer forensics that can narrow the search for one or more computer perpetrators. Data mining is a technique that has produced good results in providing insight into large volumes of data. This paper describes how the association rule data mining technique may be employed to generate profiles from log data and the methodology used for the interpretation of the resulting rule sets. The process relies on background knowledge in the form of concept hierarchies and beliefs, commonly available from, or attainable by, the computer forensic investigative team. Results obtained with the profiling system has identified irregularities in computer logs.

1 Introduction

Computer Forensics undertakes the *post-mortem*, or "after-the-event" analysis of computer crime. Of particular importance is the requirement to successfully narrow the potentially large search space often presented to investigators of such crimes. This usually involves some form(s) of guided processing of the data collected as evidence in order to produce a shortlist of suspicious activities. Investigators can subsequently use this shortlist to examine related evidence in more detail [6].

Investigative profiling is an important activity in computer forensics that can significantly narrow the search for the perpetrator and reason about the perpetrator's behaviour. This is analogous to criminal profiling which focuses on establishing personality characteristics of an offender in order to identify the type of person involved in the crime under investigation (e.g., arson). Profiling can also aid in identifying the type of activity the perpetrator is engaged in e.g., e-mail authorship analysis may identify the educational level or gender of the offender and may, consequently, be able to establish if an e-mail has been masqueraded [8].

Data Mining is employed to analyse large data sets, as might occur in a typical computer forensics investigation, in order to discover potentially useful, previously unknown regularities within data. In contrast to other, more conventional technologies, it has been able to produce good results on large data sets where both incompleteness and noise may be present, e.g., [9].

Data mining for the more specific purpose of constructing personal profiles has been used in the context of customer personalisation. Here, marketing content and services are tailored to an individual on the basis of knowledge about their preferences and behaviour. Applications include content-based and collaborative filtering-based recommendation systems, customer profiling [2, 1, 13], fraud detection [10], web browsing activities [7, 18, 23, 17]. Content-based recommendation systems model the link between data content and a person's preferences for that content whereas collaborative recommendation systems model the link between a person's preferences and other persons' preferences for the given data content [15, 19]. Customer profiling is growing in importance in e-commerce. Both factual and individual behavioural information are derived from the customer's e-transactional history. The personalisation of web browsing activities for the purpose of improving the user's access to the web has also attracted interest recently. Techniques for the modeling of the user's web access behaviour are varied including; the use of a page content interestingness metric (N-grams) for capturing a user's interests [7], web page navigation dependencies for page predictive pre-fetching [18], web page clustering for deriving aggregate user profiles [17], sequential web page patterns for discovering negatively-correlated components within a web site structure [23]. However, most web personalisation applications deal with aggregate or classes of user profiles rather than individual user profiles.

In this paper, we describe techniques to profile and analyse computer forensic data. We use a combination of existing techniques not yet employed in this application domain, modified where necessary to accommodate the particular environment. In Section 2, we introduce the elements

Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)
0-7695-1754-4/02 $17.00 © 2002 IEEE

used in our approach to computer forensic profiling. Further details are given in Section 3 about data preparation and the need for guiding the investigative process. Section 4 describes the algorithms we use and how they have been adopted for our needs. Tests performed on actual computer log data are detailed in Section 5, followed by our conclusions in Section 6.

2 Background to Investigative Profiling

An offender profile consists of two components, namely the factual component and behavioural component. The factual profile ($\boldsymbol{FP}$) consists of factual background knowledge about the offender such as their name, employee status, computer user name(s), relationships with other employees and organisations etc. The behavioural profile ($\boldsymbol{BP}$) incorporates knowledge about an offender's crime scene-related behaviour. Behaviour profile knowledge is derived from a variety of sources namely, log file transactions, header and body of e-mails, telecommunications call-record data patterns and so on.

The behavioural profile, $\boldsymbol{BP}$, can be modeled in different ways. For example, a $\boldsymbol{BP}$ can be represented as a union of sub-profile hierarchies (PH_j) such as, authorship profile, software application usage profile, log-in profile etc, or

$$\boldsymbol{BP} \leftarrow \bigcup_j^M PH_j$$

A profile hierarchy is a knowledge representation scheme using a hierarchy of multi-slot frames, similar to a concept hierarchy (described in Section 3.1), that characterises a behavioural profile.

Alternatively, $\boldsymbol{BP}$ can also be modeled as a set of association rules:

$$\boldsymbol{BP} \leftarrow \{R_i | i = 1, 2, \ldots N\}$$

Here, the rule attributes can be obtained from the raw data and/or selected from the profile hierarchy nodes. For example, the rule "If user X is a system administrator, then the application $Y = nmap$ (a stealth port scanner) executed" may be a valid rule in a system administrator profile (assuming that port scanning, as used in his/her current job context, is employed for system hardening), but probably not in a finance contractor profile.

In this paper we study user behavioural profiles derived from event data in log files. These profiles are conveniently represented by a set of implications or association rules $\{R_i | i = 1, 2, \ldots N\}$ of the form

$$R_i : antecedent \Rightarrow consequent$$

These rules provide an intuitive and declarative way to describe user behaviour [10]. For example, the rule

$R_0 : (\mathbf{StaffType} = admin) \wedge (\mathbf{DayOfWeek} = tuesday) \wedge (\mathbf{Application} = database) \Rightarrow (\mathbf{Access} = valid)$

states that "Administration staff that work on Tuesday have a valid access to a database application". Note that an association rule indicates the presence of some correlation between the rule's antecedent and consequent, but does not necessarily imply any causality.

2.1 Profiling with Association Rules

Association rule generation has been one of the most successful techniques in data mining. It originated as a tool for discovering sets of items that occur together in supermarket basket data [4]. Since then, it has evolved to address a multitude of other types of problems, to a point where it can even be used for purposes such as multi-dimensional inter-transaction mining [16]. Supposing $\mathcal{I} = \{i_1, i_2, \ldots, i_m\}$ is a set of items occurring in a data set, an association rule can be expressed by the formula $A \Rightarrow B : (s, c)$, where $A, B \subseteq \mathcal{I}$ are groups of items of size k_A and k_B, respectively, where $k_A + k_B \leq m$ and $A \cap B = \emptyset$. We refer to the combined collection of items in A and B as an *itemset* of length $k = k_A + k_B$. The variables s and c express support and confidence percentages for the rule, where support s indicates how frequently the items in A and B occur together in the data, while confidence c is the conditional probability $P(B|A)$ where the probability $P(x)$ is estimated using the support percentage of the set x. For example, the rule $(bread \wedge butter) \rightarrow (milk) : (15\%, 70\%)$ produced from a supermarket transactional database states that customers that buy bread and butter together are also 70% likely to purchase milk, with 15% of the total number of records supporting this claim.

One of the potential uses for associations is the building of rule sets that describe behavioural data [2, 1, 3]. This may be data collected about people or the operation of some systems. Often in computer forensic investigations, this information could be found in log files on a computer system. The rule sets generated from this data can be considered to describe a profile contained within the data set. Profiles produced this way, however, are usually not complete. The support percentage parameter used in association mining introduces loss into the rule set. This is because only data that occurs frequently enough (that is, satisfies a pre-defined minimum support) is used in the rule generation process. In the forensic sense, however, this is not necessarily a disadvantage. Regularities that are not picked up in the profile due to not satisfying support may be looked at as non-habitual and can be investigated as contrary to regular behaviour, if necessary.

Another important aspect of forensic profiling is that a user profile is generated using available evidence and does not change once produced. Additional evidence may be added later, but this should be regarded as the incompleteness of initial evidence rather than the evolution of an existing profile. In this case, the profile should be re-generated.

Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)
0-7695-1754-4/02 $17.00 © 2002 IEEE

Recognising temporal segments, or evolution *within* a profile, however, is quite important and analysis of such phenomena can be a major part of the profile evaluation process.

2.2 Deviation in Association Rule Profiles

One of the first steps in a computer forensic investigation is to look for unusual events. For example, if an attacker gains super-user privileges on a computer system, he may use them to perform actions not normally instigated by the real super-user(s). This would clearly be a deviation from the super-user profile as supported by data up to the time of intrusion. There are two ways this may be evidenced in the data and the profile generated on the full data set:

1. As data entries with not enough support to be represented as association rules: In order to find such entries, there must exist a mechanism for the investigator to query the data set for entries not fitting the profile.

2. As association rules making up part of the profile: It would hence be important to identify this section of the profile as being anomalous, or at least different from other parts of the profile. Assigning a temporal scope to rules making up the profile could help an investigator recognise that something potentially unusual may have occurred at a certain time in the life of the system being investigated.

There are, of course, caveats to the above. The attacker may have covered his/her tracks, for example, by removing entries from the log files. If he/she was thorough, he/she may have only removed entries corresponding to his/her own actions, or, alternatively, may have removed all records, or every record stored during the period of the criminal activity. In this case, the lack of evidence may warrant further investigation.

3 Building Profiles

The data obtained for computer forensic investigations are usually information stored on computers and networks. They range from system log files to databases, personal user files and other items that may be located on a computer. To build a profile for a particular user, many of these items may need to be examined both individually and as a collection of interrelated items with potential relationships existing between recorded activities. Profiling algorithms are therefore expected to be of varying complexity. A simple algorithm may produce rules based on the sporadic occurrences of data observed in a single file, another may be required to recognise temporal dependencies or causal relationships in user activity recorded across several files.

Since computer forensics undertakes the post-mortem analysis of computer crime, much of the analysis is done off-line. Therefore, emphasis is more often on effectiveness than efficiency in order to produce a smaller set of targeted conclusions, and reduce overall human investigation time. For example, it is preferable to achieve a low rate of false negatives at the expense of increased computational time and number of false positives.

Much of the information found on a computer is expected to be in a format not suited for immediate analysis. An investigator must facilitate this by providing details on the subsets of data intended for analysis, their format and conversion requirements, and available background knowledge. Some of this can be achieved through automated means. Filtering, the removal of unwanted information and the aggregation of separate data items are some of the more important activities during this stage of the analytical process.

3.1 Concept Hierarchies and Beliefs

Background knowledge in data mining is popularly expressed in the form of concept hierarchies and is often used in the rule generation process [21, 11]. The hierarchies convey a generalisation of concepts from node to root (which is usually the concept any) and can be represented as a set of parent-child relationships in a data file. An investigator may be prompted with a set of node level concepts found in the data and asked to abstract it to higher level ones according to his/her liking. This hierarchy, which is generally domain-dependent in forensics investigations, can then be used in the mining process to produce a profile that contains rules with elements at an arbitrary level of abstraction. There should be no requirement for a concept hierarchy to be complete for profiling to operate correctly. A set of hierarchy fragments is often more desirable as it helps to avoid over-generalisation by not including very high level concepts in the search process.

Concept hierarchies may be employed in two different ways during profile generation. In a *drill-down* approach, rules are initially generated for high concept levels. Interesting high-level rules can be further investigated by descending the concept hierarchies for some attributes. In a *drill-up* approach, a larger number of rules are produced with a potentially low support level requirement using the attribute values present in the data. By ascending concept hierarchies, higher level rules with increased support levels may be obtained.

Evolution within a profile is an important indicator of potentially irregular activities. A profiling algorithm is expected to be able to attach temporal tags to rules indicating intervals of validity if so required. Concept hierarchies, therefore, must accommodate such functionality. This hap-

Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)
0-7695-1754-4/02 $17.00 © 2002 IEEE

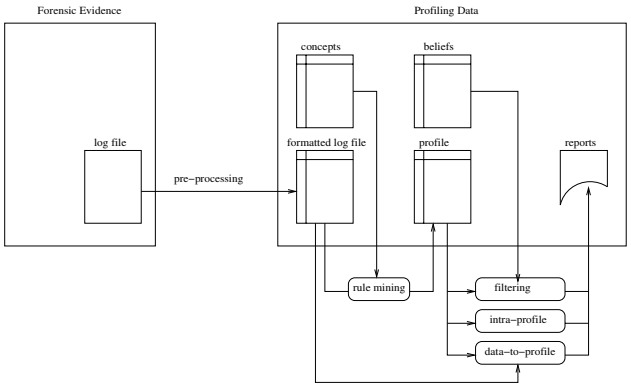

Figure 1. Data flow diagram of the profiling process.

pens at two levels – changes in the structure of concept hierarchy over time, and changes in the position of a leaf node value over time.

In addition to using hierarchical abstraction of attribute values, investigators may have pre-conceived beliefs about a case being investigated. A separate collection of rules can be used to describe a set of such beliefs. These can be used to focus the investigation by searching for specific regularities in profiles, or may also be used to reduce the profile by discarding rules that are defined as trivial [1]. Furthermore, the use of these rules may allow a post-processing algorithm to identify rules that contradict existing beliefs.

4 The Profiling Process

The data flow diagram in Figure 1 describes the data, rules and processes used for profiling purposes. It incorporates references to background knowledge such as concept hierarchies and beliefs, and the final conclusions resulting from the forensic analysis (detailed in Section 4.2).

4.1 Basic Profile Generation Algorithm

The association mining algorithm implemented for forensic analysis is designed to generate a profile using a single input file. Depending on the desired level of background knowledge to be employed, three approaches can be distinguished:

- Generating rules with no concept hierarchies and beliefs. This method is likely to produce a large set of rules that may require extensive user analysis [1].
- Generating rules with concept hierarchies but no beliefs. This solution allows for production of high-level rules and/or generalisation of lower level rules permitting both *drill-down* and *drill-up*.
- Generating rules with concept hierarchies and beliefs. This permits the same possibilities as above, as well as filtering made possible by the availability of existing beliefs.

The usual steps of *data filtering*, *data conversion*, and, when background knowledge is used, the creation of *concept hierarchies* and *beliefs* precedes profiling.

The association mining algorithm M2IS-c (*Matrix to Itemsets using concepts*) we employ, shown in Algorithm 1, is a version of the classic *Apriori* association mining algorithm [5]. Note that the algorithm is not a new, improved implementation, and was mainly selected because it suits our analytical environment. Its novelty lies in the fact that it performs binary mining in memory in conjunction with concept hierarchy ascension. Details of this process are outlined below.

Let $\mathbb{A} = \{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_l\}$ be a set of l attributes. Each attribute $\mathbf{A}_i$, $i = 1, \ldots, l$, can take on a discrete set of mutually exclusive values. Let a record r be a conjunction of values taken from each available attribute. Let $\mathbb{R}$ be a collection of n records. In this finite collection, each attribute $\mathbf{A}_i$ may take on a finite number of discrete values. Let the number of these values be denoted by m_i for attribute $\mathbf{A}_i$. The total number of distinct attribute values that appear in $\mathbb{R}$ is then $\sum_{i=1}^{l} m_i$. Let $\mathbb{H} = \{H_1, H_2, \ldots, H_N\}$ be a set of domain-dependent *concept hierarchies* or *attribute taxonomies*. Each concept hierarchy $H_j, j \in \{1, 2, \ldots, N\}$ in the concept forest is formulated as a direct acyclic graph (DAG), with none, one or more hierarchies assigned to each attribute $\mathbf{A}_i$, $i = 1, \ldots, l$. Concept hierarchies are structurally similar to profile hierarchies discussed in Section 2. The main difference is that each pair of adjacent nodes in a DAG H_j represents an "*is-a*" or generalisation-specialisation relationship, rather than multi-slot frame profile content relationship. Examples of concept hierarchies are the IP (Internet Protocol) domain name hierarchy, the functional directory of an organisation, etc. Leaf nodes in concept hierarchies belonging to attribute $\mathbf{A}_i$ therefore generally (but not necessarily) represent values occurring in $\mathbb{R}$. Non-leaf-nodes in concept hierarchies represent higher level abstractions of leaf-nodes and can not be values that occur in the original data. Denote the collection of concept hierarchies with these leaf-nodes removed by $\hat{\mathbb{H}} = \{\hat{H}_1, \hat{H}_2, \ldots, \hat{H}_N\}$. Let $\hat{m}_i$ denote the number of non-leaf concept nodes defined in all hierarchies for attribute $\mathbf{A}_i$ in $\mathbb{H}$, or equivalently, the number of nodes in $\hat{\mathbb{H}}$. Let $m = \sum_{i=1}^{l} m_i + \hat{m}_i$ be the length of a binary vector $v = \{b_1, \ldots, b_m\}$ where bit $b_i \in \{0, 1\}$ uniquely corresponds to an attribute value or concept occurring in $\mathbb{R} \cup \hat{\mathbb{H}}$. Require that bits corresponding to values and concepts of a given attribute be consecutive, with concepts having higher indexes than attribute values. That is,

$$v = \{\underbrace{b_1, \ldots, b_{m_1}, b_{m_1+1}, \ldots, b_{m_1+\hat{m}_1}}_{\mathbf{A}_1}, \ldots, \underbrace{b_{m-m_l-\hat{m}_l+1}, \ldots, b_m}_{\mathbf{A}_l}\}$$

Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)
0-7695-1754-4/02 $17.00 © 2002 IEEE

Let $M : r \rightarrow v$ be a mapping of an actual record r in $\mathbb{R}$ to a binary vector v such that each attribute value and its higher level concepts in corresponding hierarchies are represented by 1 in v with all other values set to 0. Let the function $attr(b_i)$ for bit $b_i \in v$ return the attribute $\mathbf{A}_j$, $j \in \{1, \ldots, l\}$, to which b_i was mapped to. According to the consecutiveness requirement above this means that if $attr(b_i) \neq attr(b_j)$ for some pair $(i, j) \in \{1, \ldots, m\}$, and $i < j$, then the indexes for all bit-pairs for the two attributes involved will exhibit the same *less than* relationship. This property is utilised in the algorithm below.

Algorithm 1: M2IS-c

Inputs: An $(n \times m)$ bit-matrix $\mathcal{M}$; an $(m \times m)$ concept relationship bit-matrix $\mathcal{C}$; $minsup \in [0, 1]$

Outputs: A collection of itemsets $\mathcal{I}$ satisfying $minsup$, $\mathcal{I} = \bigcup_k I^k$

1. Initialise $k{:=}1$
2. For each column $i{=}1,\ldots,m$ of $\mathcal{M}$,
   2.1. Initialise support $sup_i{:=}\sum_{j=0}^{n} b_{ji}, b_{ij} \in \{0,1\}$
3. Add 1-itemsets I_1^i to $\mathcal{I}$ where $sup_i/n \geq minsup$
4. Increment k. Stop if $k{>}l$, otherwise for the current k:
   4.1. Initialise k-itemset count $count_k{:=}0$
   4.2. Generate potential k-itemset from existing $(k-1)$-itemsets by finding next pair $\{I_i^{k-1}{=}\{i_i^1,\ldots,i_i^{k-1}\}, I_j^{k-1}{=}\{i_j^1,\ldots,i_j^{k-1}\}\}$ so that $i_i^o{=}i_j^o$, $o{=}1,\ldots,k{-}2$, $i_i^{k-1}{<}i_j^{k-1}$, and i_i^{k-1} is not in a concept relationship with i_j^{k-1}
   4.3. For potential itemset $I_{ij}^k{=}\{i_i^1,\ldots,i_i^{k-1},i_j^{k-1}\}$ calculate support sup in $\mathcal{M}$ by counting the rows where all bits appearing in I_{ij}^k are set.
   4.4. If $sup/n \geq minsup$, add I_{ij}^k into $\mathcal{I}$ as a k-itemset and increment $count_k$
   4.5. Go to Step 4.1 until all potential k-itemsets are found.
5. Stop if $count_k{=}0$, otherwise go to Step 4.

An extension to the *Apriori* algorithm in M2IS-c is the incorporation of concept hierarchy values into the mining process by including them in the binary mapping[1]. This is desirable in cases where individual values may not have enough support to be represented in a profile, but their higher level equivalents have. A consequence of this approach is the introduction of potential itemsets with both child and parent concepts present. Itemsets containing such pairs express trivial relationships and need to be pruned as they dilute the final rule set. The removal of itemsets containing child-parent pairs is an additional feature of the modified algorithm used in the profiling process. This ensures that the maximum length of any itemset produced is limited to the number of attributes l in the original data set. Child-parent relationships can be represented by a bit-matrix with ones indicating relationship and zeroes not[2]. As this lookup can be achieved in a single step, we refer the reader to the original article for discussion of the complexity of the algorithm [5].

[1] Note that the use of memory for storage of the main bit-matrix may be problematic for large data sets on non-specialised systems.

[2] A single matrix would suffice for this purpose. Individual matrices for each attribute could be preferable for memory efficiency as ones can only appear along the diagonal in $(m_i + \hat{m}_i) \times (m_i + \hat{m}_i)$ subsets for attributes $\mathbf{A}_i$, $i = 1, \ldots, l$.

4.2 Profile Analysis Algorithms

The generation of profiles is only the first step in an investigation. Algorithms for analysing the profiles need to be provided and utilised either interactively or by automated means. Some of the functionality required can be described by the following list:

- *Filtering* profiles. This process allows investigators to reduce the profile set to concentrate on subsets that may be of higher interest. It can be guided by a previously defined set of beliefs about the expected behaviour of the profile. Rules complying with beliefs may automatically be dropped, while rules in contradiction with beliefs may be assigned higher priority in the investigative process.
- Contrasting raw *data to profiles*. This produces a list or summary of data entries that deviate from the profile. It is generated for data that did not have enough support to be part of the profile, but convey potentially unexpected information different from the profile.
- Generating *intra-profile* contrasts. This means finding rules in a profile that are in contradiction with other rules in the same profile. These rules may indicate a shift in behaviour, whose causes may need to be investigated. To measure difference between rules, a distance metric will be required.

We propose a simple profile analysis algorithm to measure the degree of anomalies in the profile elements.

4.3 A Metric for Profile Element Distance

One of the more interesting and complex issues in the analysis of a profile is the discovery of contradicting elements within the profile. These contradictions may be identified both at the itemset level and in the final ruleset. In this paper, we concentrate on contradictions in itemsets, using a Manhattan distance based metric that makes differences easy to detect. For example, some of the characteristics of a particular person may be repeated in several itemsets with only a single attribute value being different. This difference can be attributed to:

- *Repetition.* In this case, the attribute represents a value (for example, day of the week), that indicates that the same set of characteristics is valid for multiple occasions.

Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)
0-7695-1754-4/02 $17.00 © 2002 IEEE

- *Generalisation.* The attribute value has been replaced by the higher level concept that retains the same set of characteristics, but possibly with larger support.
- *Contradiction.* The attribute value is in contradiction with another, potentially pre-defined as a belief. For example, a user may be allocated a particular computer but the profile indicates the use of a different one.

The recognition of the occurrences of these differences may be automated. Some may be combined (repetition) or discarded as unimportant (generalisation or trivial beliefs). Others may require inspection by investigators to decide if they are worth following up. Algorithm 2 (*Itemset to itemset distance*) describes the calculation of a metric that indicates the closeness or similarity of two k-itemsets by comparing their elements. It employs the $attr()$ function defined prior to presenting Algorithm 1 and assumes that the itemsets to be compared are represented by bits from the bit-vector v format defined there. Because of the consecutiveness requirement, it follows that the bits at position o in a k-itemset be in three distinct relationships:

1. They may belong to different attributes $\mathbf{A}_i \neq \mathbf{A}_j$.
2. They may belong to the same attribute $\mathbf{A}_i$ and be the same attribute value or concept, or have a child-parent relationship.
3. They may belong to the same attribute $\mathbf{A}_i$ but be different values/concepts with no relationship.

Algorithm 2: IS2IS-dist

Inputs: K-itemsets $I_i = \{b_1^i, \ldots, b_k^i\}$ and $I_j = \{b_1^j, \ldots, b_k^j\}$; an $(m \times m)$ concept relationship bit-matrix $\mathcal{C}$; attribute function $attr(b)$
Outputs: Distance $d \in [0, \ldots, k+1]$

1. Initialise $d := 0$
2. For $o := 1$ to k
   2.1. If $attr(b_i^o) \neq attr(b_j^o)$, set $d := k+1$ and stop
   2.2. If $(b_i^o \neq b_j^o) \wedge (b_i^o$ not in child-parent relationship with $b_j^o)$, increment d

It can be seen that distance d of Algorithm 2 can be less than the length of the itemset k only if the same attribute value/concept is found duplicated (i.e. equals or is in a concept relationship with) at least once in the two itemsets being compared. It also follows that the total number of such duplicates found and d equals k, with $d = 0$ only if the respective elements of the two itemsets are the same or are in a concept relationship. Thus, the metric is a non-negative integer $d \in [0, \ldots, k+1]$, from which only values $0 < d < k$ are of interest.

A similar algorithm can be devised to calculate the distance between a k-itemset and a data record (IS2DAT-dist)[3]. This can be achieved by expanding the itemset to hold *nil* values for attributes not originally in the itemset, then comparing this itemset with the data record the same way as comparing two l-itemsets. The difference between Algorithm 2 and this modified version is that d is not incremented for attributes where the itemset holds a *nil* value. This limits d to a maximum value of k.

[3]This algorithm is not presented due to its similarity to Algorithm 2.

User	*Console*	*Origin*	*Weekday*	*Date*	*Time*	*Duration*
samson	ftp	cpe-203-21-225-3	Mon	Apr 16	21:24 - 21:31	(00:07)
ftp	ftp	c81010.upc-c.che	Mon	Apr 16	20:58 - 20:59	(00:00)
everett	ttyp1	host-216-252-150	Mon	Apr 16	20:24 - 22:25	(02:01)
max	ftp	max2.cs.xyu.edu	Mon	Apr 16	15:59 - 16:00	(00:00)
evelyn	ttyp1	marlin.xyu.edu.a	Mon	Apr 16	15:07 - 15:12	(00:05)
joseph	ftp	188.191.47.170	Mon	Apr 16	14:23 - 14:23	(00:00)
joseph	ttyp1	188.191.47.170	Mon	Apr 16	13:05 - 14:45	(01:40)
pedro	ftp	188.191.47.157	Mon	Apr 16	12:49 - 13:05	(00:15)
pedro	ftp	188.191.47.157	Mon	Apr 16	12:15 - 12:38	(00:23)
robert	ttyp1	hamilton.cs.xyu.	Mon	Apr 16	11:03 - 11:03	(00:00)
pedro	ftp	188.191.47.157	Mon	Apr 16	10:58 - 11:09	(00:10)
robert	ttyp3	hamilton.cs.xyu.	Mon	Apr 16	10:45 - 10:58	(00:13)

Figure 2. Example time slice of past and current user login information as obtained by executing the UNIX last command.

5 Data, Experiments and Results

To evaluate the profiling methodology proposed in this paper, a number of experiments have been performed. Both Algorithms 1 and 2 have been implemented as well as IS2DAT-dist. As input, log files captured by executing the UNIX **last** command were used, which searches the *wtmp* system log file and lists past and current user login information for a computer. An example output from executing the **last** command is shown in Figure 2.

Note that the data used in our experiments are actual log data recorded by a UNIX-based computer set up as a server with remote login access. However, in order to preserve anonymity, the data attribute name instances have been modified. Furthermore, there was no implication of inappropriate behaviour in the data set.

Of several columns of information generated, six attributes were copied or composed into a table containing formatted input. Some filtering was performed at this stage to remove incomplete (current) and non-user (e.g. *shutdown*) logins. The table, using additional higher level concepts from attribute hierarchies, was then mined to produce a profile containing association rules. Intra-profile and data-to-profile contrasting was then performed.

The distance metric of IS2IS-dist and IS2DAT-dist was employed to produce reports for both contrasting methods.

5.1 Intra-Profile Experiments

Intra-profile contrasts were calculated only for itemsets of the same length. For example, in one test, from about 2000 original data records, approximately 2200 itemsets with more than 1 element were produced. Intra-profile contrasting produced roughly 43000 distances that were less than the lengths of the itemsets being compared. Although

Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)
0-7695-1754-4/02 $17.00 © 2002 IEEE

this is a far smaller number than what it potentially could have been, it is still more than what can be perused manually. To reduce this set further, additional strategies need to be devised. One option is to prioritise attributes. That is, if difference is measured only in a particular attribute that may not be carrying important information (such as day of the week), then pairs exhibiting distance only in such attributes may be dropped. Similarly, a strategy may be employed to drop contrasts that are too "high". That is, the distance metric for a particular itemset length may be regarded as high, even though it satisfies the initial constraint of being less than the length. This may, for example, render all distances produced for 2-itemsets unnecessary. Finally, focusing techniques may be provided to filter the distances for certain attributes or attribute values. One of the more interesting contrasts produced by IS2IS-dist during testing was the 1-distance pair

$I_0 : (\mathbf{User} = pedro) \wedge (\mathbf{Origin} = miami)$
$I_1 : (\mathbf{User} = pedro) \wedge (\mathbf{Origin} = adelaide)$

which indicates that the same user has been logging in from two very different geographic locations. Further inspection of this contrast revealed that the user in question left his place of work in Adelaide for another in Miami while still regularly accessing his old Adelaide account.

5.2 Data-to-Profile Experiments

The filtering requirement to reduce the set of distances to manageable proportions becomes even more evident with data-to-profile contrasts. Without pre-processing, each itemset needs to be compared to every data record, potentially producing a much larger result set than for intra-profile contrasts. This is partly due to the fact that a number of records are not included in the profile due to unsatisfactory support. Each of these records could produce small distances to itemsets similar to it that made it into the profile. As in the case of intra-profile contrasts, measures can be taken to reduce the final result set. In addition to the strategies outlined in Section 5.1, duplicate records may be removed by post-processing the results. Also, data-to-profile distances may be zero, if a particular data record was one of those used to generate the itemset it is being compared to. These distances should also be pruned from the results.

Figure 3 shows some of the distances from a test calculated for a particular itemset of length 5 (top row). Non-zero distances up to a maximum value of 2 were allowed in order to list contrasts where difference is present in not too many attributes. Duplicates were removed and as mentioned, some attributes were sanitised to remove confidential information from the data. The itemset contains generalised concepts for both the *User* and *Origin* attributes, while *Duration* is represented by concepts categorising a

Dist	*Duration*	*User*	*Console*	*Origin*	*Weekday*
-	UpToHalfHour	lecturer	ftp	cs.xyu.edu.au	Mon
1	UpToHalfHour	clyde	ftp	188.191.47.160	Mon
2	FewMinutes	clyde	ftp	188.191.47.160	Mon
2	UpToHalfHour	ftp	ftp	lizard.cs.xyu.ed	Fri
2	FewMinutes	everett	ftp	host-216-252-150	Mon
2	UpToHalfHour	john	ftp	188.191.47.161	Thu
2	NoMinutes	joseph	ftp	188.191.47.170	Mon
1	NoMinutes	max	ftp	max2.cs.xyu.edu	Mon
1	UpToHalfHour	max	ftp	max2.cs.xyu.edu	Tue
1	UpToHalfHour	max	ftp	max2.cs.xyu.edu	Thu
1	UpToHalfHour	max	ftp	max2.cs.xyu.edu	Fri
2	FewMinutes	max	ftp	max2.cs.xyu.edu	Thu
2	FewMinutes	milo	ftp	bet.cs.xyu.edu.a	Fri
2	NoMinutes	milo	ftp	bet.cs.xyu.edu.a	Sun
2	UpToHalfHour	milo	ttyp1	bet.cs.xyu.edu.a	Fri
1	UpToHalfHour	pedro	ftp	188.191.47.157	Thu
1	UpToHalfHour	pedro	ftp	188.191.47.157	Fri
2	FewMinutes	pedro	ftp	188.191.47.157	Fri
2	NoMinutes	pedro	ftp	188.191.47.157	Thu
2	30-60Minutes	pedro	ftp	188.191.47.157	Wed
2	UpToHalfHour	pedro	ttyp1	cay.cs.xyu.edu.a	Wed
2	NoMinutes	pedro	ftp	188.191.47.157	Wed
2	NoMinutes	stuart	ftp	sloth.cs.xyu.edu	Tue
2	UpToHalfHour	stuart	ttyp2	1cust81.tnt2.tow	Mon
2	FewHours	stuart	ftp	1cust81.tnt2.tow	Mon
2	UpToHalfHour	vernon	ttyp2	barn.cs.xyu.edu.	Mon

Figure 3. Example data-to-profile distances from IS2DAT-dist for a sample profile element and a collection of data records, ordered by *User* for readability

potentially large number of discrete values. From the definition of the metric, values/concepts in the same hierarchy have a distance of zero, which explains the diversity of rows of (non-generalised) values in the data having similar distances. For readability, we give here some of the concept relationships from the otherwise rather large hierarchies that exist for *User* and *Origin*:

$$\{max, milo, pedro, stuart\} \subset lecturer,$$
$$\{*.cs.xyu.edu.au, 188.191.47.*\} \subset cs.xyu.edu.au \subset$$
$$xyu.edu.au \subset adelaide,$$
$$\{*.tnt2.tow.net.au, 198.twn0103.twn.net.au\} \subset$$
$$ISP.adelaide \subset adelaide.$$

Using this information, the first data row with $d = 1$ shows that user $clyde$ is not a $lecturer$, whilst for $lecturers$ max and $pedro$, who log onto university computers, we can observe that the same $wtmp$ login information is valid for several weekdays other than Monday.

Figure 3, as is, contains superfluous information. Depending on the support used in mining the profile, some or most of the data records contribute to itemsets generated by the algorithm. Comparing a k-itemset to data that contributes to another k-itemset is a repetition of comparing an itemset with another. Crosschecking a data record against every other k-itemset prior to calculating a distance would, however, be even less cost-effective. Instead, a strategy of producing distances in a matrix form for k-itemsets, $k = 2, \ldots, l$, then discarding rows with at least one zero in it, would be a better solution. Alternatively, a separate algorithm may parse the data set to locate individual occurrences of records that do not contribute to any itemset of a given length, and then run the contrasting algorithm against this filtered data set only. This is indeed the requirement proposed in Section 4.2 for data-to-profile contrasting.

Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)
0-7695-1754-4/02 $17.00 © 2002 IEEE

6 Conclusions and Future Directions

The initial implementation of the profiling analysis process described in this paper has resulted in promising results capable of identifying irregularities in computer logs that can serve as useful evidence in computer crime investigations. Profile analysis, however, forms only a part of the investigative process and relies heavily on expert knowledge. It is therefore best perceived as a component in a larger collection of tools designed to aid the forensic investigator.

The profiling tool presented in this paper presents further opportunities for enhancement. One such area is the handling of multiple log information in a single process. Multi-dimensional mining may offer a solution for this problem, with some interesting work already found in the literature [16, 20]. Alternatively, it may be possible to "flatten" several logs into a sequence of "events", for which more traditional sequential mining techniques can be applied. Further improvements may be achieved by replacing the mining algorithm used in profiling. One obvious candidate is the attribute-oriented induction technique [14]. This technique compacts a collection of records into a generalised relation, or a conjunction of generalised records where individual attribute values are replaced by higher level concepts by ascending concept hierarchies. One of the advantages of this technique is that the final rule set incorporates information about every record in the original data set. Further work is also to be carried out in the intelligent presentation of results, notably in the provision of appropriate visual interpretation of the profiles and its potential contrasts. Contrast measures currently used are itemset-specific. Deriving distance measures for rules, such as the value distance metric (VDM) [22] may yield better results in identifying discrepancies. Some of the better known data mining interestingness measures [12], or variations of, may also be adopted for this purpose.

References

[1] G. Adomavicius and A. Tuzhilin. Expert-driven validation of rule-based user models in personalization applications. *Data Mining and Knowledge Discovery*, 5(1/2):33–58, 2001.

[2] G. Adomavicius and A. Tuzhilin. Using data mining methods to build customer profiles. *Computer*, 34(2):74–82, 2001.

[3] C. Aggarwal, Z. Sun, and P. Yu. Online algorithms for finding profile association rules. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM-98)*, Bethesda, MD, USA, 1998.

[4] R. Agrawal, T. Imielinski, and A. Swami. Mining associations between sets of items in massive databases. In *Proceedings of the ACM SIGMOD Int. Conference on Management of Data*, Washington, DC, USA, May 1993.

[5] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. Verkamo. *Advances in Knowledge Discovery and Data Mining*, chapter Fast discovery of association rules. AAAI Press, 1996.

[6] E. Casey. *Digital Evidence and Computer Crime*. Academic Press, 2000.

[7] P. K. Chan. A non-invasive learning approach to building web user profiles. In *Proceedings of the Workshop on Web Usage Analysis and User Profiling (WEBKDD'99)*, 1999.

[8] O. de Vel, A. Anderson, M. Corney, and G. Mohay. Mining e-mail content for author identification forensics. *SIGMOD Record*, 30(4), 2001.

[9] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second Int. Conference on Knowledge Discovery and Data Mining*, 1996.

[10] T. Fawcett and F. Provost. Adaptive fraud detection. *Data Mining and Knowledge Discovery*, 1(3):291–316, 1997.

[11] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *Proceedings of 21st VLDB Conference*, September 1995.

[12] R. J. Hilderman and H. J. Hamilton. Knowledge discovery and interestingness measures: A survey. Technical Report CS-99-04, Dept of Computer Science, University of Regina, 1999.

[13] M. Hirsh, C. Basu, and B. Davidson. Learning to personalize. *Communications of the ACM*, 43(8):102–106, 2000.

[14] H. J, Y. Cai, and N. Cercone. Knowledge discovery in databases: an attribute-oriented approach. In *Proceedings of 18th Int. Conference on Very Large Databases*, 1992.

[15] J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordon, and J. Riedl. Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, 1997.

[16] H. Lu, L. Feng, and J. Han. Beyond intra-transaction association analysis: mining multi-dimensional inter-transaction rules. *ACM Transactions on Information Systems*, 18(4):423–454, 2000.

[17] B. Mobasher, H. Dai, T. Luo, Y. Sun, and J. Wiltshire. Discovery of aggregate usage profiles for web personalization. In *Proceedings of the Workshop on Web Mining for E-Commerce (WEBKDD'00)*, August 2000.

[18] A. Nanopoulos, D. Katsaros, and Y. Manolopoulos. Effective prediction of web-user accesses: a data mining approach. In *Proceedings of the Workshop on Mining Logdata Accross All Customer Touchpoints (WEBKDD'01)*, 2001.

[19] S. Nesbitt and O. de Vel. A collaborative filtering agent system for dynamic virtual communities on the web. In *Proceedings of the Conference on Learning and Discovery (CONALD98)*, June 1998.

[20] T. Oates and P. R. Cohen. Searching for structure in multiple streams of data. In *Proceedings of the Thirteenth International Conference on Machine Learning*, 1996.

[21] R. Srikant and R. Agrawal. Mining generalized association rules. In *Proceedings of 21st VLDB Conference*, 1995.

[22] C. Stanfill and D. Waltz. Toward memory-based reasoning. *Communications of the ACM*, 29(12):1213–1228, 1986.

[23] P. N. Tan and V. Kumar. Mining indirect associations in web data. In *Proceedings of the Workshop on Mining Logdata Accross All Customer Touchpoints (WEBKDD'01)*, 2001.

Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)
0-7695-1754-4/02 $17.00 © 2002 IEEE