3

INTEGRATING DATA MINING TECHNIQUES WITH INTRUSION DETECTION METHODS

Ravi Mukkamala, Jason Gagnon and Sushil Jajodia

Abstract Intrusion detection systems like NIDES depend on the ability to characterize a user's past behavior based on his/her usage patterns. The characterization is typically made in terms of statistics drawn on system parameters such as CPU, I/O and network loads, and file access patterns. For example, NIDES maintains statistics on approximately 25 such parameters for each user. The cost of data collection, statistics computation, and intrusion detection are directly proportional to the number of parameters maintained per user. If we would like to achieve real-time responses to intrusion detection, then we need to minimize the number of parameters without adversely affecting the detection capabilities. In this chapter, we propose to use some of the feature reduction and selection techniques commonly used in data mining applications to reduce the computational and storage requirements of the intrusion detection methods. Since typically several of the user behavioral parameters are correlated, applying these techniques may reduce the number of parameters needed to represent the user behavior.

Keywords: Data mining, decision tables, feature reduction, intrusion detection, logic methods

1. INTRODUCTION

With the ever increasing trend to make computer systems and databases easily accessible through the internet, there is also an increasing fear of intruders into the systems. One method of detecting intruders, who might enter the system in the guise of legitimate users is to maintain an audit of user activity [2, 3, 9]. Each audit record would summarize the user activity or a process activity in terms of a set of feature values. One of the main challenges in intrusion detection is to summarize the historical audit data so that when current activity is compared with it, one can classify it as legitimate or intrusive. Since the comparisons need to be made on-line, when the activity is taking place, it is

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35508-5_22

V. Atluri et al. (eds.), *Research Advances in Database and Information Systems Security*

necessary to minimize the processing time for such detections. This also puts a constraint on the size of the data we wish to keep in the summary.

Data mining techniques deal with very large volumes of data [8, 14]. In order for these techniques to be effective in on-line processing of queries, they attempt to reduce the data that they need to process in answering a query. These are referred to as data reduction techniques. Feature reduction is one of many such techniques [13, 14].

In this chapter, we investigate the effectiveness of integrating some data reduction techniques of data mining with the intrusion detection techniques. The work presented in this chapter is preliminary in nature. Thus the insights gained are explained more through examples rather than any formal proofs. As we continue to make progress in these efforts, we hope to obtain more concrete evidence.

In particular, we look at the significance tests on features that determine whether or not a feature contributes in correctly classifying an observed user activity as intrusive or non-intrusive. The activity itself is characterized in terms of a set of features in an audit record. We also look at measures of mutual significance. These measures are useful when all features in an audit record are not completely independent. In fact, dependencies do exist among features in real systems. When the degree of mutual dependency is high, then we could possibly eliminate one or more of the features from the audit record without affecting the intrusion capabilities. We also look at logic methods that can infer classification rules from a given set of audit records. This scheme also reduces the amount of data that need to be maintained about historical behavior of users. We find that all these techniques are quite useful in data reduction in intrusion detection schemes.

2. INTRUSION DETECTION

Detecting intruders in a computer system is vital to many organizations. This is especially relevant to commercial and military organizations since they maintain several key data on their computer systems. While preventing intruders by means of firewalls and authentication mechanisms is a necessary step, the organizations would also like to be alerted when suspicious activity is observed on the system.

NIDES, the Next-generation Intrusion Detection Expert System, is developed at SRI International for the purpose of detecting anomalous behavior in computer system [2, 3]. The system is based on statistics—the normal behavior of a user is represented by a set of parameters. Statistic are dynamically calculated based on the user's behavior. Whenever a user's behavior (expressed in terms of audit records) is found to (statistically) significantly deviate from

the normal statistic, an alarm is raised. In order to keep up with the changing behavior of a legitimate user, the system gives higher weight to the recent behavior and lower weight to the past. This is achieved by defining half-life for the statistic.

Since no single measure may characterize the behavior of a user, NIDES measures and maintains statistic on several measures. Some of the measures that were used in one of their applications, Safeguard, include user time, system time, number of open files, elapsed process time, integral of memory usage, number of page faults, etc. [2]. For each measure, a probability distribution of short-term (immediate past) and long-term behaviors was constructed.

The computational and storage requirements in NIDES is directly proportional to the number of measures (or features) being monitored. For each measure, a histogram and several statistic are maintained. This information is updated either after every audit record or after a certain period of time (or after certain number of audit records). This processing is to be done sufficiently frequently so as to avoid false alarms as well as the risk of missing detection of legitimate intrusions. Several computations are also needed to determine whether or not an intrusion has occurred.

While it is perfectly valid that we need to characterize the behavior of a user in terms of several measures, there is also a danger of keeping track of too many measures. In addition, these measures would not be completely independent. For example, large number of page faults will also result in high I/O activity. This also means larger turnaround time, more number of context switches for the process, and higher system time. In the current work, we look at the means to identify any correlations that exist among these measures (or features). When high correlations are observed between measures, some of the measures could be safely omitted. Whether or not such omissions will affect intrusion detection is also a topic of discussion in this chapter.

3. DATA MINING TECHNIQUES

Data mining is the search for valuable information in large volumes of data [1, 14]. The data itself is characterized by several features (referred to as measures in intrusion detection). In other words, the data consists of records with information related to an entity expressed in terms of characteristics or features. For example, the credit information of a credit-card holder may indicate the holder's credit limit, average monthly charges, percentage of times the payment was defaulted, the average annual earnings, etc. One of the primary applications of data mining is to predict the unknown behavior of a potential customer based on the known historical behavior of other customers.

Linear regression is one of the traditional techniques used for prediction in data mining [4]. Here, given historical data of say two variables, assuming that one variable is linearly dependent on the other, the regression techniques determine a linear relationship between them. In other words, the entire historical data is now converted to a simple linear equation.

The decision trees are also a powerful model used in data mining [4, 13, 14]. They are produced by several techniques including classification and regression trees and Chi-squared automatic induction. These are particularly useful for classification. For this reason, we propose to use these models to classify audit records into non-intrusive and intrusive classes. A decision tree is expressed as a clear sequence of decision rules. The rules themselves can be expressed as logic statements, in a language such as SQL, and hence can be applied to new audit records easily.

Techniques such as neural nets and genetic algorithms are also used for classification and prediction in data mining [4, 14]. However, in the current work we do not explore the application of these techniques for intrusion detection.

In addition, one of the main objectives of data mining techniques is to reduce the amount of data that need to be processed when an on-line query is received. Several reasons are mentioned for the reduction of data. They include the expected time for inducing a solution may be too long or the time to read the data from the database may be too large. Different parameters associated with a record of interest in the database are referred to as features. Some of the operations suggested for data reduction are: (i) reduce number of features, (ii) reduce the number of records to be processed, and (iii) reduce the number of values for a feature (also smoothing). In the proposed work, we concentrate on the first method—reduce the number of features to be examined.

4. METHODS FOR FEATURE REDUCTION

One of the main objectives of this chapter is to suggest methods to reduce data that needs to be maintained and processed for intrusion detection. One method of data reduction is reduction of the features or parameters collected in each audit record. Here, we discuss some feature reduction techniques and how they can be effective in the reduction.

Since the main focus of this chapter is intrusion detection, the role of a feature in an audit record is to help classify it as either intrusive or non-intrusive. Hence, we use this context in the following sections.

4.1 SIGNIFICANCE TEST FOR A FEATURE

One of the characteristics that a feature should satisfy is whether or not it can significantly help in distinguishing a non-intrusive audit record from an intrusive one. Several statistical tests of significance currently exist in literature [6, 10]. For example, if we would like to determine whether or not CPU time is a feature of significance, we select n_I audit records which are known to be intrusive and n_{NI} non-intrusive (or normal behavior) records. One test of statistical significance is as follows:

$$sig(CPUTime) = \frac{|\mu_I - \mu_{NI}|}{se}$$

$$se(I,NI) = \sqrt{\frac{\sigma^2(I)}{n_I} + \frac{\sigma^2(NI)}{n_{NI}}}$$

Here, se is the variance of the distribution of the sample means (the standard error of the sampling distribution), and σ^2 is the sample variance for each class. The μ's are the means of CPU times in the two classes. Since the difference in means will be approximately normally distributed for large sample sizes, independent of the distribution of the samples, we can check whether or not sig(CPUtime) is significant at a given level of significance. In cases where the significance of the feature is lower than the critical value at a given confidence level, we can say that the feature is not statistically significant in the classification of an audit record. Hence, it can probably be eliminated.

4.2 MUTUAL SIGNIFICANCE

In the above test, each feature is analyzed independently and tested for significance. However, if the features are analyzed collectively, then we may obtain more information about the features. For example, from the individual significance tests, if we were to determine that both CPU time and I/O time are significant in detecting an intrusion, then we select both the features. However, if the features are correlated to each other, then one of them will suffice for intrusion detection. One of the methods to detect mutual significance is as follows:

1. Let $\vec{\mu_I}$ and $\vec{\mu_{NI}}$ be the vector of feature means for the intrusive and non-intrusive sets of audit records, respectively. Let Cov_I and Cov_{NI} be the covariance matrices for the intrusive and non-intrusive sets of audit records, respectively.

2. Compute $Dist$=$(\vec{\mu_I}$-$\vec{\mu_{NI}})(Cov_I$+$Cov_{NI})^{-1}(\vec{\mu_I}$-$\vec{\mu_{NI}})^T$

3. Determine the smallest set of features that can result in largest value of $Dist$.

Since determining optimal feature set may be computationally expensive, heuristic procedures are suggested to achieve near optimality. In general, the best set of k independent features are the k features with the largest values of $(\mu_I - \mu_{NI})^2/(\sigma_I^2 + \sigma_{NI}^2)$ [14].

4.3 LOGIC METHODS

Given the training data, if we can convert the data into a set of decision rules or a decision tree, then we could use the resulting form in intrusion detection.

Consider the case of decision rules [12, 14]. The rules are represented as a set of IF-Then propositions. Each rule has a conditional part and a conclusion part. The conditional part is a boolean expression in a propositional form. The conclusion is an assignment of class or value.

Each decision rule is a conjunction of true-or-false terms. For example, if the system has observed that whenever CPU time is higher than 3 minutes and I/O time is greater than 4 minutes, it is an intrusion, then it is represented as:

$$CPUTime > 3.0 \wedge IOTime > 4.0 \quad \rightarrow \quad \text{Intrusion}$$

In addition, if the system observed that any time CPU time is higher than 8 minutes or I/O time greater than 15 minutes there is an intrusion, we can add two more rules to the above set:

$$CPUTime > 8.0 \rightarrow \text{Intrusion}$$

$$IOTime > 15.0 \rightarrow \text{Intrusion}$$

Similarly, the system may also infer similar rules about the non-intrusive or normal cases. For example, if the training data were to indicate that in all cases when CPU time is less than 1 minute and I/O time is less than 2 minutes there is no intrusion, then we can add another rule:

$$CPUTime < 1 \wedge IOTime < 2 \rightarrow \text{No intrusion}$$

The size of rule set depends on the number of parameters and the complexity of their relationship in the presence or absence of intrusion activity.

The second form of logic method to represent the intrusive/non-intrusive behavior is decision-tree [4, 5, 12, 14]. Decision trees are binary trees where

the nodes represent decisions. For example, since CPUTime > 8 represents a case of intrusion, the root node could be this decision. Here, the branch with the attribute "True" would lead to a leaf node of type "intrusion." The branch with the attribute "False" has several alternatives. One choice is to have a node with the decision IOTime > 15. We can thus build the decision tree.

5. APPLICATION OF DATA REDUCTION TECHNIQUES IN INTRUSION DETECTION

In this section, we illustrate the efficacy of the logic methods through examples. For easy readability, we describe the application and its results as a step-by-step procedure.

1. **Select the number of features.** We selected the following five features (F_1-F_5) to represent user behavior. These features are a subset of those chosen by the NIDES system in their experimentation [2, 3].

   Feature 1 (F_1): User CPU Time

   Feature 2 (F_2): System CPU time

   Feature 3 (F_3): Character IO during the application execution

   Feature 4 (F_4): The integral of real memory usage over time

   Feature 5 (F_5): Number of pages read in from the disk

2. **Select the rules representing the normal (i.e., non-intrusive) behavior of the system.** We choose the following for illustration.

$$F_1 < 10; F_2 < 5; F_3 < 200; F_4 < 30; F_5 < 700 \quad (1)$$

$$10 < F_1 < 20 \wedge 10 < F_2 < 15 \quad (2)$$

   In a practical situation, these rules are not available. However, since we need to generate data representing user behavior, we use these rules. The rules will also be used later to verify the efficacy of the logic methods. An audit record that satisfies any of the above rules is said to be non-intrusive or genuine. An audit record that does not satisfy any of the above six rules is said to be intrusive.

3. **Set limits for the values of the features.** We set the following limits.

$$0 < F_1 < 30; 0 < F_2 < 15; 0 < F_3 < 400; 0 < F_4 < 60; \quad (3)$$

$$0 < F_5 < 1000 \quad (4)$$

4. **Generate audit records by assigning random values to each of the five features.** The data generated follow the limits mentioned above (Eqn. (3)-(4)). We generated 20,000 audit records.

5. **Classify each record as either intrusive or non-intrusive based on the rules selected above.** Out of the 20,000 generated, 19,432 were found to be non-intrusive and the rest 568 were intrusive type.

6. **Run a program which can look at the training data of 20,000 audit records (with classification) and arrive at either a decision tree or a set of rules for classification.** We used the programs dtree and logic supplied in [14]. Table 3.1 summarizes the decision tree obtained.

Table 3.1 Rules derived from 20,000 audit records.

1:	$F_5 > 699.5$	false: 2	true: 3
2:	answer= 1		
3:	$F_3 > 199.5$	false: 4	true: 5
4:	answer= 1		
5:	$F_4 > 29.5$	false: 6	true: 7
6:	answer= 1		
7:	$F_2 > 4.5$	false: 8	true: 9
8:	answer= 1		
9:	$F_1 > 9.5$	false: 10	true: 11
10:	answer= 1		
11:	$F_2 > 10.5$	false: 12	true: 13
12:	answer= 2		
13:	$F_1 > 19.5$	false: 14	true: 15
14:	$F_1 > 10.5$	false: 16	true: 17
15:	answer= 2		
16:	answer= 2		
17:	answer= 1		

The table is to be interpreted as follows. The first column is the rule number. Rule 1 says that if feature 5 (F_5) is greater than 699.5, then go to rule 3; otherwise go to rule 2. In fact, rule 2 says that answer=1 or that the audit record that is being analyzed corresponds to class 1. In our case, class 1 corresponds to non-intrusive audit records. In other words, when $F_5 \leq 699.5$, an audit record can be automatically classified as a non-intrusive class. This matches with our original data generation rules (eqn. 7-12). When feature 5 is greater than 699.5, we go to rule 3 which further compares if $F_3 > 199.5$.

Since the rules derived from the audit data through logic methods match those of data generation, we can conclude that these methods were quite effective in intrusion behavior characterization.

In the above example, we chose all independent values for the features. Suppose we were to choose some dependencies between the feature values. In particular, if $F_1 = 2 * F_2$+random(0,5) and $F_5 = 2.5 * F_3$+random(0,50), then clearly there is a correlation between the features. (Here, $random(m,n)$ is a random number generator that generates numbers between m and n). When data was generated with these values, we obtained 17861 non-intrusive records and 2139 intrusive type. When the logic methods were run on the data, we obtained the rules for classification shown in Table 5..

Table 3.2 Decision tree derived for the correlated data.

1:	$F_5 > 699.5$	false: 2	true: 3
2:	answer= 1		
3:	$F_4 > 29.5$	false: 4	true: 5
4:	answer= 1		
5:	$F_2 > 4.5$	false: 6	true: 7
6:	answer= 1		
7:	answer= 2		

From Table 5., we notice that only features 2, 4, and 5 were considered in the decision process. So the system was able to infer the correlation between features 1 and 2 and hence eliminated feature 1. Similarly it inferred the correlation between features 3 and 5 and eliminated feature 3. Thus, we need only features 2, 4, and 5 to detect intrusion. Hence, the data size is reduced. If we do the significance tests on these features, as described in Section 4.1, we get the results of Table 5..

Table 3.3 Significance of different features.

Feature	Significance
F_1	39.2
F_2	40.2
F_3	130.4
F_4	72.5
F_5	130.6

From this table, it is clear that all features are significant. This, however, does not convey the information about correlation among the features. The mutual significance of the features is evident when we look at the correlation matrix, for example. The correlation matrix for the above data is given in Tables 5. and 5..

Table 3.4 Correlation coefficients for non-intrusive audit data.

	F_1	F_2	F_3	F_4	F_5
F_1	1.0	0.99	-0.11	-0.06	-0.11
F_2	0.99	1.0	-0.11	-0.06	-0.11
F_3	-0.11	-0.11	1.0	-0.14	0.999
F_4	-0.06	-0.06	-0.14	1.0	-0.14
F_5	-0.11	-0.11	0.999	-0.14	1.0

Table 3.5 Correlation coefficients for intrusive audit data.

	F_1	F_2	F_3	F_4	F_5
F_1	1.0	0.97	-0.03	-0.01	-0.03
F_2	0.97	1.0	-0.03	-0.02	-0.03
F_3	-0.03	-0.03	1.0	-0.01	0.99
F_4	-0.01	-0.02	-0.01	1.0	-0.01
F_5	-0.03	-0.03	0.99	-0.01	1.0

From the correlation data, not surprisingly, we can observe that features 1 and 2 are highly correlated. Similarly, features 3 and 5 are also highly correlated. Hence, we can reduce the five features of the audit record to just three.

If we were to apply the distance metric (*dist*) discussed in section 4.2, we would arrive at the same conclusion.

6. EFFECT OF DATA DEDUCTION ON INTRUSION DETECTION

While the data reduction techniques may help in reducing the size of the audit record data, the main concern is one of false alarms or missed intrusion detections. In other words, we need to address the issue of loss (if any) in intrusion detection capability of the system when we apply the data reduction techniques.

For example, consider the case with correlated features discussed above. Let us remove features 1 and 3 from the audit record. The next question we need to deal with is the set of rules for intrusion detection.

1. Consider the rule $F_3 < 200$. Since we determined that F_3 and F_5 are correlated, we need to find a relationship between them. If we were to run a linear regression for F_5 in terms of F_3, we find that $F_5 = 2.5 * F_3 + 25$. In other words, we need to translate the $F_3 < 200$ as $F_5 < 562.5$. However, since the $F_5 < 700$ rule already covers this, we can eliminate the F_3 rule completely.

2. Now consider the rule $F_1 < 10$. Once again using a linear regression, we can determine that $F_1 = 2 * F_2 + 2.5$. Hence, the rule may be rewritten in terms of F_2 as $F_2 < 3.75$. Once again, the original rule $F_2 < 5$ already covers this. So we can drop the rule.

3. Now, we need to consider the rule $10 < F_1 < 20 \wedge 10 < F_2 < 15$. Once again, using the linear regression relationship derived above ($F_1 = 2 * F_2 + 2.5$) we can rewrite it as $3.75 < F_2 < 8.75 \wedge 10 < F_2 < 15$. Since the two ranges for F_2 do not overlap, we can safely eliminate this rule also.

Thus, the new set of rules for intrusion detection are

$$F_2 < 5$$
$$F_4 < 30$$
$$F_5 < 700$$

When we ran the audit records under these rules, we found that all 17861 non-intrusive records were correctly identified as non-intrusive class. The 2139 intrusive records were also classified as intrusive. In other words, there was no loss of information due to the reduction of features from five to three.

However, we need to understand the importance of either rerunning the logic methods to determine the new rules or to modify the original rules by incorporating the relationships between the eliminated features and the existing features.

6.1 MULTI-DEPENDENCY RELATIONSHIPS

Suppose, the relationships among the features was not one-to-one as in the above example, but involve multiple features. For example, if we have the following relationships $F_1 = 1.5 * F_2 + 0.5 * F_4$+random(0,5) and $F_3 = 5 * F_4 + 3 * F_1 + F_2 + random(0, 50)$. When audit records were generated under these

conditions, with the original rules, we obtained 19665 non-intrusive type and 335 intrusive type records. If we were to eliminate two of the five features from the audit records, then the original rules also need to be changed accordingly. When we applied the logic methods on this data, we obtained the rules of Table 6.1.

Table 3.6 Rules derived for the multiply correlated data.

1:	$F_3 > 199.5$	false: 2	true: 3
2:	answer= 1		
3:	$F_5 > 699$	false: 4	true: 5
4:	answer= 1		
5:	$F_2 > 4.5$	false: 6	true: 7
6:	answer= 1		
7:	answer= 2		

When the data was validated with the new rules and features 1 and 4 removed from the audit records, we found that the system correctly identified all 19665 non-intrusive records as non-intrusive and the 335 records as intrusive. Once again, there was no loss of information due to the reduction in the number of features.

6.2 NON-LINEAR DEPENDENCIES

So far we have considered only linear relationships among the features. Suppose non-linear relationships exist among the features. For example, let $F_5 = F_4 * F_2 + F_1 + random(0, 100)$. When we generated audit data with this relation, we obtained 19845 non-intrusive records and 155 intrusive records. When logic methods were employed on the new data set, we obtained new rules.

When we checked the input records with the above rules, the system classified 19745 of the 19845 records (99.5%) non-intrusive. However the remaining 0.5% non-intrusive records were falsely classified as intrusive. This would correspond to false alarms. Out of the 155 intrusive records, 147 or 95.5% were correctly classified as intrusive. However, the remaining 4.5% were classified as non-intrusive. But the deterioration in performance is acceptable in most cases when the corresponding saving in processing and storage is taken into account.

From these examples, we can conclude that the logic methods are quite effective in reducing features and arriving at a set of rules based on audit record data. However, one needs to be cautious in applying the technique as it may

lead to a small percentage of incorrect classifications. This was especially so when non-linear correlation between features was present.

7. SUMMARY

While the area of data mining is rich with techniques to reduce time for on-line processing of records, intrusion detection in computer system requires tools to reduce its time in detecting possible intrusions. In the current work, we attempted to illustrate some of the data reduction techniques that may be effectively used for intrusion detection without compromising security. We have especially focused on statistical techniques to test individual significance and mutual significance, and logic based methods such as decision trees. The preliminary results are encouraging. In the absence of real-data, we used simulated data to show the efficacy of the techniques.

In the future work, we propose to test these methods on real audit record data. We also propose to test it on data when millions of data records are available. In addition, we propose to combine the techniques of intrusion detection techniques such as NIDES with those of the data mining to reduce the on-line processing time for intrusion detection.

References

[1] Adriaans, P. and Zantinge, D. (1998). *Data Mining*, Addison-Wesley.

[2] Anderson, D. *et al.* (1993). "SAFEGUARD Final Report: Detecting unusual program behavior using the NIDES Statistical Component. Final Report, Computer Science Laboratory, SRI International.

[3] Anderson, D. *et al.* (1995). Detecting unusual program behavior using the statistical component of the Next-generation Intrusion Detection Expert System (NIDES). Technical Report, SRI-CSL-95-06, Computer Science Laboratory, SRI International.

[4] Berry, M.J.A. and Linoff, G. (1977). *Data Mining Techniques*, John Wiley & Sons Inc.

[5] Brieman, L. and Smyth, P. (1997). Applying classification algorithms in practice. *Statistics and Computing,* **7(1)**, pp. 45–56.

[6] Henkel, R.M. (1976). *Tests of Significance*, SAGE Publications.

[7] Jain, R. (1991). *The Art of Computer Systems performance Analysis*, John Wiley & Sons Inc.

[8] James, M. (1985). *Classification Algorithms*, John Wiley & Sons Inc.

[9] Javitz, H.S. and Valdes, A. (1991). The SRI IDES Statistical Anomaly Detector. *Proceedings of the IEEE Symposium of Security and Privacy*, pp. 316–326.

[10] Kanji, G.K. (1993). *100 Statistical Tests*, SAGE Publications.

[11] Lin, T.Y., Hinke, T.H., Marks, D.G. and Thuriasingham, B. (1996). Security and data mining. *Database Security IX: Status and Prospects* (eds. D.L. Spooner, S.A. Demurjian and J.E. Dobson), Chapman & Hall, pp. 391–399.

[12] Weiss, S.M. and Indurkhya, N. (1995). Rule-based machine learning methods for functional prediction. *Journal of Artificial Intelligence Research*, **3**, pp. 383–403.

[13] Weiss, S.M. and Kulikowski, C. (1991). *Computer Systems That Learn: Classification, and Prediction Methods from Statistics, Neural Nets, Machine Learning and Expert Systems*, Morgan Kaufmann.

[14] Weiss, S.M. and Indurkhya, N. (1998). *Predictive Data Mining: A Practical Guide*, Morgan Kaufmann.