A Probabilistic Method for Detecting Anomalous Program Behavior

Kohei TATARA[1], Toshihiro TABATA[2], and Kouichi SAKURAI[2]

[1] Graduate School of Information Science and Electrical Engineering, Kyushu University,
`tatara@itslab.csce.kyushu-u.ac.jp`,
WWW home page: `http://itslab.csce.kyushu-u.ac.jp/`
[2] Faculty of Information Science and Electrical Engineering, Kyushu University, Japan
{`tabata, sakurai`}`@csce.kyushu-u.ac.jp`

Abstract. In this paper, we, as well as Eskin, Lee, Stolfo [7] propose a method of prediction model. In their method, the program was characterized with both the order and the kind of system calls. We focus on a non-sequential feature of system calls given from a program. We apply a Bayesian network to predicting the N-th system call from the sequence of system calls of the length $N-1$. In addition, we show that a correlation between several kinds of system calls can be expressed by using our method, and can characterize a program behavior.

Keywords: Intrusion detection, Anomaly detection, System call, Bayesian network

1 Introduction

The increase of computers connected to a network with the rapid spread of the Internet is being enhanced. Therefore, many cases of an unauthorized access to the computer by the malicious user are reported, and intrusion detection system which detects such intrusions is getting more and more important.

Many intrusions exploit a vulnerability which is inherent in a program, called buffer overflow. An attacker rewrites the return address of a function by overflowing an internal buffer. Thereby, the attacker can take over the control of the program and it is possible to execute arbitrary codes [1]. On the other hand, there are researches of anomaly detection system which detects the intrusion using buffer overflow by monitoring the control flow of a program [2–11, 14].

An anomaly detection consists of a learning period which learns behavior during the normal operation of a program, and a monitoring period which supervises the action of a program by comparing with the records of normal operations. Such an anomaly detection system has an advantage of possibility to detect an novel intrusion compared with the system which only detects a known intrusion based on the signature. However, the overhead of anomaly detection which supervises the execution of a program becomes comparatively large. So it is important to chose the data which characterize the operation of a program.

Forrest et al. [2] showed that it is possible for normal operation of a program to be characterized by the history of the sequences of system calls which a program emits. If an intrusion using buffer overflow happens, since some sequences of system calls which are not seen during the normal operation will be observed, it is detectable.

Using the method based on N-gram, we can break a sequence of system calls into the sub-sequences of fixed length N, and judge whether the operation of a program is normal or not by comparing with those. In order to make only N-gram applicable to comparison, it is greatly dependent on N whether incorrect detection takes place. Forrest et al. refered to the optimal value of N, namely 6. The 6 is experimentally optimal value on the trade-off between detection capability and efficiency. That is, although efficiency improves when the N is set as less value, detection capability declines a little. They did not touch on the reason why the value of 6 was drawn as optimal value, but was treated as the "magic number" for some years ahead. In order to abolish the necessity of asking for the optimal value of N from the training data, some researchers made attempts to generate a finite state machine [6]. And, other researchers also set N from the normal operation data automatically [7]. Lee et al. [14] did research which explains the mechanism of $N = 6$, from a standpoint of information theory using conditional entropy.

In recent years, the research which clarified the reason was reported by analyzing the data which Forrest et al. exhibited [12]. Kymie et al. proved that the condition in which at least one "Minimal foreign length" exists in the sequences of system calls collected when an intrusion happened is that the value of N is six or more. Minimal foreign length is the minimum value of N in which at least one unique sub-sequence exists, when the sequences of system calls are divided into the sub-sequences of length N [12]. Clearly from their report, in order to use system call as normal data of an anomaly detection, the features from the sequences of system calls used for learning need to differ from those observed from anomalous processes. Although the more than 200 kinds of system calls exist in the current version of Linux, about dozens of kinds of the system calls are actually emitted by the program at most. The total number of possible N-grams is too small to characterize the behavior of program by the system calls in the case of $N = 1$. If Σ kinds of system calls are published in the program, then there are Σ^N possible sequences of length N. Therefore, as Σ is large, the probability under which Minimal foreign length is found increases by leaps and bounds. Kymie et al. showed the method of calculating the optimal value of N, when the stide [2] and Hofmyer et al.'s method [3] were used for an anomaly detection system. However, when these methods are applied to other data sets, we do not affirm whether $N = 6$ may become the "magic number." When the optimal value of N is 7 in a certain data set, performing anomaly detection using the value 6 contains a possibility of overlooking abnormal sequences.

On the other hand, there is a research using the rule learning program, called RIPPER, which focuses on correlation of system calls [13]. In this paper, Lee et al. input those sequences which are attached the labels, "normal" or "abnormal" and generated some If-then rules, and tried to extract the feature which can classify the normal and abnormal operations The data which they used in their experiment was the same as the data of sendmail which Forrest et al. used. And the procedure of generating the data was described in [2]. The rule set which Lee et al. generated took form, such as "if $p_2 = 104$ and $p_7 = 112$ then the sequence is *normal*" (Here, $p_i = j$ means that the i-th system call number in a certain sequence of system calls is j.) Lee et al. showed that it is possible to detect anomalous behavior using this rule set. However, in order to generate this rule set, you have to attach the label of being normal or abnormal to each sequence of training data. They also aimed to learn the correlation between each system calls. That is, they predicted the N-th system calls or the middle system calls in sequences of length N and showed as a result that detection accuracy depends on the value of N.

Eskin et al. [7] introduced a method of calculating the optimal value of N from training data [7]. They obtained it by substituting the conditional probability of transition between each system calls for the formula of the conditional entropy of Shannon and chosen sequence length with minimal entropy. This value is the most efficient value calculated information-theoretically. Then, they also proposed a "prediction model" which expects a N-th system call from the sequence of the length $N-1$ using those conditional probabilities and succeeded in increasing accuracy of anomaly detection. As shown above, we can calculate the optimal value of N. But we do not know what propety in the conventional methods based on N-gram affects detection accuracy. In order to get more efficiency, we need to clarify such a property.

In this paper, we propose an alternative method of prediction model. We apply a Bayesian network for predicting the N-th system call from the sequence of system calls of the length $N-1$ as well as Eskin et al. We show that a correlation between several kinds of system calls can be expressed by using our method, and can characterize a program behavior. Then we can decrease the number of kinds of sequences to use in anomaly detection. The composition of this paper is as follows. In Section 2, we compare our method with previous result. The difference between Eskin et al.'s and our method is specifically clarified, and our contribution is described. In Section 3, we describe the algorithm of our method. In Section 4, we also present the result of experiment. Then, in Section 5, we consider a validity of our method. We conclude in Section 6.

2 Non sequential model

Eskin et al. [7] proposed the model using sparse Markov transducers based on sparse prediction tree [7]. In this method, they replaced the overlapping symbol with the wild card in the sparse Markov tree obtained from a set of sequences of system calls, and reduced the number of branches. They showed that the probabilistic threshold outperformed mismatch threshold empirically.

The advantage of using a Bayesian network is that sequence of length $N-1$ correlating with a N-th system call can be treated as non-sequential one. For example, it is assumed that we observe the sequence of length 4 such as {open mmap stat write} and {open stat mmap write}. By using sparse Markov transducers, we can obtain the sparse Markov tree, shown as Figure 1.

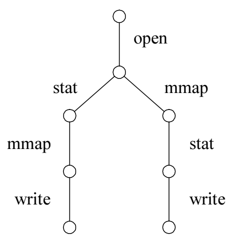

Fig. 1. The example of sparse Markov tree (The sequence length N is 4)

This tree structure has branched at the portion of open system call. On the other hand, when a Bayesian network is used for predicting the N-th system call from the preceding sequence of length $N-1$, the $(N-i)$-th system call only correlate with N-th. That is, the sequence {open mmap stat write} and {open stat mmap write} is treated as the same one. When some system calls are classified according to the kind, we consider correlation between those sets.

3 Our proposal

This section explains outline and concrete procedure of our proposal.

3.1 Outline of our proposal

In a learning period, a Bayesian network is formed from the sequences of system calls which an application program emits. A system call is a function for an application program to use the function which an operating system offers, and the number corresponding to a name is assigned, respectively. Henceforth, expression called a system call S_i designate the system call whose system call number is i.

A Bayesian network [18] can express the qualitative dependency between random variables with non-circulating directed graph, and is suitable for expressing a phenomenon including uncertainty, and causal relation. In the background of the our proposal, there is an idea that causal relation exists between each system calls in the sequence of system calls from a program. Therefore, the validity of our method is shown experimentally to the last. The relation between system call S_i and S_j is designated by the directed link $S_i \rightarrow S_j$ in a Bayesian network. S_i is called parent node, and S_j is called child node. The quantitive causal relation between S_i and S_j is expressed with the conditional probability $P(S_j|S_i)$. When there are two or more parent nodes, set of the parent nodes of S_j is designated by $\pi(S_j)$. The set of conditional probabilities, provided that all the values of parent nodes exists, is called Conditional Probability Table (CPT), and is used as normal operation data in our method.

In a monitoring period, using the learned bayesian network, the validity of the sequence of system calls which a program publishes is verified. When a program publishes a system call S_i, conditional probability $P(S_j|\pi(S_j))$ is calculated by using CPT obtained in the learning period. If the intrusion using buffer overflow occurs, some sequences of system calls which are not seen during the normal operation will be observed [2, 3]. It is expected from this character that the value of conditional probability takes a low value during anomaly operation, and a high value during normal operation continuously.

On the other hand, when sets of the parent nodes $\pi(S_i)$ are equal as for two or more system calls, it is thought that those conditional probabilities become small. When judging the height of conditional probability and the validity of issue, we are anxious about the number of incorrect detections increasing. We try to solve the problem by using the statistical technique, called Mann-Whitney U-test [19]. Mann-Whitney U-test is useful in case it tests whether there is a difference between the median values of two groups.

Hereafter, the concrete procedure in a leaning period and a monitoring period is explained.

3.2 Formation of a Bayesian network

In a learning period, the procedure which forms a Bayesian network from sequences of system calls is as follows.

1. In the sequences of system calls $X = \{X_1, \ldots, X_l, \ldots\}$, when it is assumed that a causal relation exists between the l-th system call X_l and the set of system calls $\{X_{l-1}, \ldots, X_{l-D}\}$ published for the past D times, it is set to $\pi(X_l) = \{X_{l-1}, \ldots, X_{l-D}\}$. The value of D (here, it stand for the degree of dependence) can be set up arbitrarily.
2. Suppose that the element of $\pi(X_l)$ be the node of a Bayesian network. Then all pair of two nodes if there exist a causal relation are connected by directed links.
3. The procedures (1) and (2) are repeated for all sequences of system calls.
4. $P(X_l|\pi(X_l))$ is calculated for all X_l.

Figure 2 is the example which actually complete the above-mentioned procedure from the sequences of system calls which the ftp program publishs, and form a Bayesian network. The number in parenthesis in Figure 2 expresses the system call number. In this case, a set of the parent nodes of a socketcall system

child node	parent node
...	...
stat	close, open, read
mmap	open, read, stat
close	read, stat, mmap
mprotect	stat, mmap, close
munmap	mmap, close, mprotect
socketcall	close, mprotect, munmap
gettimeofday	mprotect, munmap, socketcall
write	munmap, socketcall, gettimeofday
...	...

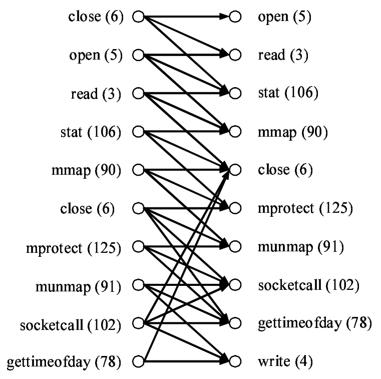

Fig. 2. The example of Bayesian network when the degree of dependency D is 3 (The number in the parenthesis expresses the system call number)

call and a gettimeofday system call is equal. Therefore, as shown above, the conditional probability of these system calls may take a low value.

3.3 The procedure of the anomaly detection in a monitoring period

The procedure of performing anomaly detection using the learned Bayesian network is as follows.

1. In the sequences of system calls $X = \{X_1, \ldots, X_i, \ldots\}$ a program emits, when it is assumed that a causal relation exists between the ith system call X_i and the set of system calls $\{X_{i-1}, \ldots, X_{i-D}\}$ published for the past D times, it is set to $\pi(X_i) = \{X_{i-1}, \ldots, X_{i-D}\}$. The value of D can be set up arbitrarily.
2. $A_i = P(X_i|\pi(X_i))$ is calculated from the CPT obtained during a learning period. Moreover, j is selected for $B_i = P(S_j|\pi(X_i))$ so as to takes the highest value.
3. Mann-Whitney U-test [19] is performed with the two groups $\{A_i, \ldots, A_{i-I+1}\}$ and $\{B_i, \ldots, B_{i-I+1}\}$. A null hypothesis presupposes 'There is no difference between the median values of the two groups'. A significant level can be set up arbitrary.
4. When the U statistics calculated in (3) takes the value below a critical value, it rejects null hypothesis. That is, since there is a difference during the median values of two groups, the sequence is flagged as anomalous. If the number of anomalous sequences exceeds a threshold, it is judged that an intrusion occurs.

Importantly, we calculate $P(X_i|\pi(X_i))$ from the CPT, as described below. First we take the set of conditional probabilies of X_i, given $\pi(X)$ from which the hamming distance to $\pi(X_i)$ take a minimum value in the CPT. Then, we select the maximum conditional probability as $P(X_i|\pi(X_i))$ in the set. Needless to say, the hamming distance between two sequences affects the detection capability.

4 Experiment

This section describes the result of experiment according to the method which was stated in Section 3. A specific purpose is to prove that our method is able to detect the intrusion using buffer overflow and to compare the accuracy and performance of our method with the other method.

4.1 Data sets

Forrest et al. [2, 3] showed that the short sequences of system calls issued by a program during its normal executions can characterize it. Those sequences are also different from the sequences of its anomalous executions as well as the executions of other programs. The many different methods are compared each other in the past research [5]. The data sets used in this research are publicly available at http://www.cs.unm.edu/˜immsec/data-sets.htm. Therefore, we also experiment using these data sets.

These data consist of "live" data which are recorded by the normal use and "synthetic" data which are recorded in the environments where the program options are chosen carefully. Although it was a somewhat rude way, in order to investigate the number of false positives and false negatives, we chose a suitable number of sequences of system calls from all these data at random, and used for training and monitoring. Table 1 describes the details of data sets.

Table 1. Details of data sets

Program	# of seq.	# of seq. for training	# of seq. for testing	# of proc.
ftp	181,663	10,000	171,663	5
xlock	9,230,067	895,924	8,334,143	2
ps	8,589	4,112	4,477	11
login	13,739	6,128	7,611	13
sendmail	143,109	73,491	69,618	34

4.2 Comparison

Many researchers have proposed the methods based on N-gram. tide, stide [5], and Hofmyer's method [3] are mentioned as an example which are successful empirically. We decided to use the Hofmyer et al. 's method [3] as comparison. Hofmyer et al. calculated the hamming distance between the sub-sequences of system calls of length N. When this hamming distance exceeds a threshold, that sequence was judged as abnormal. In their method, the program was characterized with both the order and the kind of system calls. So, we are interested in whether we obtained the same result or not, even with only non-sequential sequences of system calls.

4.3 Experimental results

We measured the accuracy and performance of anomaly detection using our method. In order to evaluate the anomaly detection system, we can observe the number of true positives and false positives. True positive rate shows the percentage of sequences which are flagged as abnormal in the set of sequences issued by an anomalous process. And false positive rate expresses the percentage of sequences accidentally judged to be anomalous in the set of sequences of system calls issued by a normal process. The result which we obtained is shown in Figure 3, 4, 5, 6, 7.

The experiment using ftp program resulted in the regrettable score shown in Figure 3. It is easy to check that the rate of true positives is remarkably small and the further analysis of the sequences of system calls in this program is required. In other programs, the accuracy of the anomaly detection of Hofmyer et al. and our proposal are dependent on some parameters. These parameters involve the threshold C in Hofmyer et al. 's method which limits the minimal hammming distance to decide whether a sequence of system calls is anomalous or not, N which is initial parameter [3], the critical value in our method and so on. They depend on the context as well as the sequence length N. That is, it means that the algorithm which finds out optimal values of them from normal data is required. Needless to say, when we select the appropriate parameter, we can achieve the results exceeding the Hofmyer et al. 's method. We selected the 6 as the sequence length N conventionally used in the past researches. The ROC curve in Figure 3, 4, 5, 6, 7 is obtained by trying the various parameters described above. The ROC curve shows that even if only the kind of system call is extracted as the feature, we can characterize the operation of program.

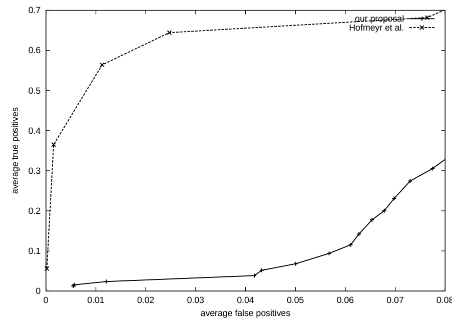

Fig. 3. ftp ($N = 6$)

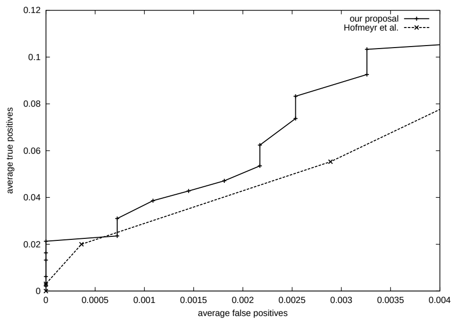

Fig. 4. login ($N = 6$)

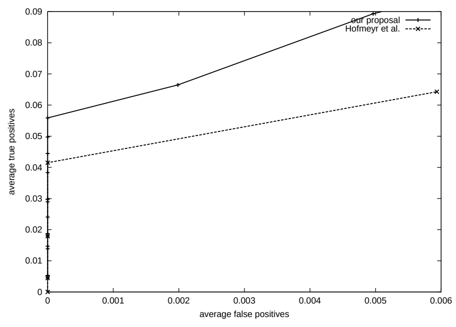

Fig. 5. ps ($N = 6$)

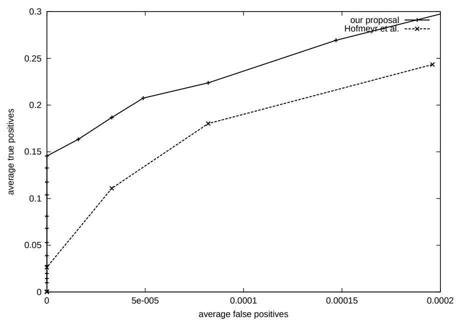

Fig. 6. sendmail ($N = 6$)

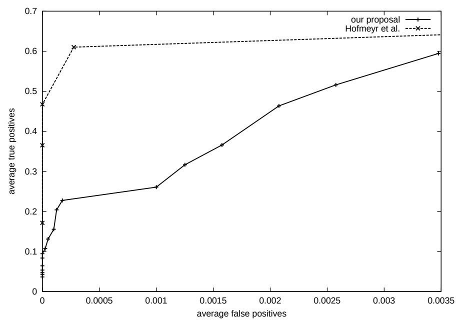

Fig. 7. xlock ($N = 6$)

5 Consideration

In this paper, we showed experimentally that by paying attention to the correlation between several kinds of system calls and the non-sequential feature of those, we can characterize the operation of program. On the other hand, there are many researches which observe sequential feature of the sequences. The reason for this trend is due to the size of space. When the Σ kinds of system calls exist in an environment, the number of possible sequences in order to catch the feature of program is $|\Sigma|^N$. But, our method has only $N \cdot \frac{(|\Sigma|+N-2)!}{(N-1)!\cdot(|\Sigma|-1)!}$ kinds of possible sequences. This means that the more varieties of control flows in the program, the less number of unique sequences of system calls we can discriminate from all the other sequences. That is, this may drop the accuracy of the anomaly detection and increase the number of false positives. Fortunately, we can prove that non-sequential feature is large enough to distinguish between normal and abnormal operation. But, because the research which shows that it is possible to avoid an anomaly detection system by camouflaging sequences of system calls skillfully is also reported [17], it is desirable that we can express many features of programs to enhance the accuracy, while holding overhead.

On the other hand, there is a research that does not only regard a sequence of system calls as a feature, but also describes the action of a program in higher dimensions with other information about it. Wagner et al. performed static analysis of a source code and created the non-deterministic pushdown automaton expressing control flow of a program [15]. In recent years, Oyama et al. developed the method, added inspection of stack information, and created a state transition diagram leading to no false positives [16]. In these anomaly detection, since there are many features which should be inspected, more amounts of calculation are needed, so we are anxious about the overhead. Our ultimate goal is an anomaly detection system as an intrusion detection system. For that purpose, it is necessary to consider not only the capability of anomaly detection in real time but the amount of calculation and usability.

6 Conclusion

In this paper, we proposed a method of anomaly detection based on N-gram. Since our proposal has put the basis on correlation being between system calls from the standpoint of information theory as well as Eskin et al. [7]. Moreover, we apply Bayesian network to concretely describe the correlation between several kinds of system calls. Thereby, we can clarified the non-sequential correlation between system calls experimentally. Future work is investigation of the validity of the proposal system in more programs.

References

1. Beyond-Security's SecuriTeam.com. Writing Buffer Overflow Exploits - a Tutorial for Beginners. http://www.securiteam.com/securityreviews/5OP0B006UQ.html (accessed 2003-09-05).
2. S. Forrest, S. A. Hofmeyr, A. Somayaji, T.A. Longstaff. A sense of self for Unix processes. In the *1996 IEEE Symposium on Computer Security and Privacy.*
3. S. Forrest, S. A. Hofmeyr, and A. Somayaji, Intrusion detection using sequences of system calls. *Journal of Computer Security*, Vol.6, pp. 151–180, 1998.
4. G. Helmer, J. Wong, V. Honavar, and L. Miller. Intelligent agents for intrusion detection. In *IEEE Information Technology Conference*, pp. 121–124, Sep. 1998.
5. C. Warrender, S. Forrest, and B. Pearlmutter. Detecting intrusions using system calls: alternative data models. In *Proceedings of the 1999 IEEE Symposium on Security and Privacy,*
6. C. Marceau. Characterizing the behavior of a program using multiple-length n-grams. In *Proceedings of the New Security Paradigms Workshop 2000*, 2000.
7. E. Eskin, W. Lee, and S. Stolfo, Modeling system call for intrusion detection using dynamic window sizes. In *Proceedings of the 2001 DARPA Information Survivability Conference & Exposition*, Anaheim, CA, June 2001.
8. S. Li, A. Jones. Temporal Signatures for Intrusion Detection. *17th Annual Computer Security Applications Conference* , Dec. 10–14, 2001

9. A. P. Kosoresow, S. A. Hofmeyr, Intrusion Detection via System Call Traces, IEEE Software, vol. 14, pp. 24–42, 1997.
10. R. Sekar, M. Bendre, P. Bollineni and D. Dhurjati, A Fast Automaton-Based Method for Detecting Anomalous Program Behaviors, In *Proceedings of the IEEE Symposium on Security and Privacy*, 2001.
11. Y. Liao, V. Rao Vemuri, Using Text Categorization Techniques for Intrusion Detection, In *Proceedings of the 11th USENIX Security Symposium.* Aug. 2002.
12. K. M. C. Tan, R. A. Maxion, "Why 6?" Defining the Operational Limits of stide, an Anomaly-Based Intrusion Detector. In *Proceedings of IEEE Symposium on Security & Privacy*, pp. 188–201, 2002.
13. W. Lee, S. Stolfo, and P. Chan, Learning Patterns from Unix Process Execution Traces for Intrusion Detection, In *Proceedings of AAAI97 Workshop on AI Methods in Fraud and Risk Management*, 50-56, 1997.
14. W. Lee and D. Xiang, Information-Theoretic Measures for Anomaly Detection, In *Proceedings of The 2001 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2001.
15. D. Wagner, D. Dean, Intrusion Detection via Static Analysis, In *Proceedings of the 2001 IEEE Symposium on Security and Privacy*, 2001.
16. M. Oka, H. Abe, Y. Oyama, K. Kato, Intrusion Detection System Based on Static Analysis and Dynamic Detection, In *Proceedings of Forum on Information Technology (FIT 2003)*, Japan, Sep. 2003.
17. D. Wagner, P. Soto, Mimicry Attacks on HostBased Intrusion Detection Systems. In *Proceedings of 9th ACM Conference on Computer and Communications Security*, Nov. 2002.
18. Y. Motomura, I. Hara, User Model Construction System using Probabilistic Networks. http://staff.aist.go.jp/y.motomura/ipa/ (accessed 2003-09-05)
19. W. J. Conover, Practical Nonparametric Statistics, John Wiley & Sons, Inc., New York, 1971.