Reliable Detection of Episodes in Event Sequences

June 10, 2003

Robert Gwadera, Mikhail J. Atallah*, Wojciech Szpankowski†
Department of Computer Science
Purdue University
W. Lafayette, IN 47907
{gwadera,mja,spa}@cs.purdue.edu

Abstract

Suppose one wants to detect "bad" or "suspicious" subsequences in event sequences. Whether an observed pattern of activity (in the form of a particular subsequence) is significant and should be a cause for alarm, depends on how likely it is to occur fortuitously. A long enough sequence of observed events will almost certainly contain any subsequence, and setting thresholds for alarm is an important issue in a monitoring system that seeks to avoid false alarms. Suppose a long sequence T of n observed events contains a suspicious subsequence patterns S within it, where the suspicious subsequence S consists of m events and spans a window of size w within T. We address the fundamental problem: is a certain number of occurrences of a particular subsequence unlikely to be fortuitous (i.e., indicative of suspicious activity)? If the probability of fortuitous occurrences is significant then an automated monitoring system flags it as suspicious anyway. Then such a system will suffer from generating too many false alarms. This paper quantifies the probability of such a S occurring in T within a window of size w, the number of distinct windows containing S as a subsequence, the expected number of such occurrences, its variance, and establishes its limiting distribution that allows to set up an alarm threshold so that the probability of false alarms is very small. We report on experiments confirming the theory and showing that we can detect bad subsequences with low false alarm rate.

Key Terms:

*This author acknowledges support of Grants EIA-9903545 and ISS-0219560 from the National Science Foundation, Contract N00014-02-1-0364 from the Office of Naval Research, by sponsors of the Center for Education and Research in Information Assurance and Security, and by Purdue Discovery Park's e-enterprise Center.

†The work of this author was supported by the NSF Grant CCR-0208709 and NIH R01 GM068959-01.

1 Introduction

Detecting subsequence patterns in event sequences is of prime importance to many applications, that include intrusion detection, monitoring for suspicious activities, and molecular biology (e.g., see [8, 12, 17, 18]). Whether an observed pattern of activity is significant or not (i.e., whether it should be a cause for alarm) depends on how likely it is to occur fortuitously. A long enough sequence of observed events will almost certainly contain any subsequence, and setting thresholds for alarm is an important issue in a monitoring system that seeks to avoid false alarms.

The basic question is then: *when is a certain number of occurrences of a particular subsequence unlikely to be fortuitous (i.e., indicative of suspicious activity)?* A quantitative analysis of this question helps one to set a *threshold* so that real "intrusions" are detected and false alarms are avoided. Setting the threshold too low will lead to too many false alarms, whereas setting the threshold too high can result in failure to detect. By knowing the most likely number of occurrences and the probability of deviating from it, we can set a threshold such that the probability of missing real suspicious activities is small. Such a quantitative analysis can also help to choose the size of the sliding window of observation. Finally even in a court case one cannot consider certain "bad" activity as a convincing evidence against somebody if that activity is quite likely to occur in given circumstances including "bad" activity type, background activities and observation window. Therefore it is very important to quantify such probabilities and present a universal and reliable framework for analyzing a variety of event sources.

Let T be an ordered sequence of events (time-ordered events in a computer system, transactions in a database, purchases made, web sites visited, phone calls made, or combinations of these). Systems designed to detect "bad things" in T usually do not look at the whole of T, they usually involve a sliding "window of observation" (of size, say, w) within which the analysis is confined. This is done for two reasons: (i) T is usually too long, and without dividing window approach it would involve having to save too much state, and (ii) T can be too long (e.g., in a continuously monitoring system) that *any* subsequence (bad or good) would likely occur within it. As an example of the need to confine the analysis to such a limited sliding window, note that three failed login attempts (with failure due to wrong password) are significant if they occur in rapid succession, but quite innocuous if they occur within a one-month interval. In this study we do not use the notion of real calendar time such as a "one month interval", instead we use the number of events as a proxy for time. This is why our interval length w is not the difference between two time stamps, but rather the size of a (contiguous) substring of T.

Formally consider an alphabet $\mathcal{A}$ of cardinality $|\mathcal{A}|$, an event sequence $T = t_1 t_2 \dots t_n$ of length n over $\mathcal{A}$ and an episode over $\mathcal{A}$ in one of the following forms: either a single pattern $S = s_1 s_2 \dots s_m$ of length m, or a set of patterns $S_1, S_2, \dots, S_d$, or the set of all distinct permutations of S; this last case captures situations where the ordering of the events within the window of observation does not matter, e.g., for the two events "bought a large amount of fertilizer" and "bought an assault rifle" it may not matter which one occurred first. We use a positive integer $w \geq m$ to represent the length of the window of observation. We assume that the pattern S is given while the event sequence is generated by a memoryless (Bernoulli) or Markov source. Our interest is in finding $\Omega^{\exists}(n, w, m)$ that represents the number of windows containing *at least one* occurrence of S when sliding the window n events in T. Based on the observed value of $\Omega^{\exists}(n, w, m)$ our task is to decide whether a suspicious activity took place or not.

We study the probabilistic behavior of $\Omega^{\exists}(n, w, m)$. We first compute the expected value of $\Omega^{\exists}(n, w, m)$, its variance, and then show that, appropriately normalized $\Omega^{\exists}(n,w,m)$ converges in distribution to the standard normal distribution. This will allow us to set up a threshold $\tau(n,m,w)$ such that for a given confidence level β we assure that $\Pr\{\Omega^{\exists}(n,w,m) > \tau(n,m,w)\} < \beta$. That is, if one observes more than $\tau(n,m,w)$ occurrences of windows with suspicious subsequences the probability that such a number is generated by randomness is highly unlikely (i.e., its probability is smaller than β). We also design the window size w so that suspicious subsequence do not occur almost surely in a window. This is necessary to reliably set up the threshold.

We verify our theoretical results by running an extensive series of experiments. One set of experiments is run on an on-line version of *War and Peace*, as an example of English text source. In another experiment we use web-logs obtained from `http://www.cs.washington.edu/ai/adaptive-data/` that contains user accesses to the music machines web site (currently at `http://machines.hyperreal.org`) from 1/01/99 through 4/30/99 . We first show that our formula for the probability approximates very well the experimental one. Then we insert randomly some sequences, defined as "suspicious", and detect them through our threshold mechanism.

Pattern matching has been used in a number of applications. In most applications, however, *substring* pattern matching was applied, which is an "occurrence" when the pattern symbols occur contiguously in the text. Kumar and Spafford [8] applied subsequence pattern matching to intrusion detection. Atallah and Apostolico [2] designed a fast algorithm to detect subsequences in a text, while Flajolet *et al.* [9] presented a precise statistical analysis of the subsequence problem. In [10] Mannila *et al.* discussed the problem of episode matching and counted the number of windows containing episodes as a subsequence. The authors in [10] also showed the effect of the window width on the number of frequent episodes. Following [10] we consider and analyze the *windowed* version of the subsequence problem. Papers [4, 7] introduced the *Window-Accumulated Subsequence Matching Problem (WASP)* and counted all w-windows within which the pattern is a subsequence of event sequence. None of the above work provides probabilistic analysis that quantifies $\Omega^{\exists}(n,w,m)$.

The paper is organized as follows. In section 2 we present our main results containing theoretical foundation. Section 3 contains experimental results demostrating applicability of the derived formulas. Derivations of theoretical results are presented in Section 4 using analytic tools of analysis of algorithms such as generating functions and complex asymptotic (cf. [15, 16]).

2 Main Results

In this section we present our main theoretical results.

2.1 Formulation of the Pattern Matching Problem

Given an alphabet $\mathcal{A} = \{a_1, a_2, \ldots, a_{|\mathcal{A}|}\}$ and a pattern $S = s_1 s_2 \ldots s_m$ of length m, we search for some occurrences of S as a subsequence within a window W of size w in another sequence known as the event sequence $T = t_1 t_2 \ldots t_n$ of length n. A valid occurrence of S in T corresponds to a set of integers $i_1, i_2, \ldots, i_m$ such that the following conditions hold:

1. $1 \leq i_1 < i_2 < \ldots < i_m \leq n$

2. $t_{i_1} = s_1, t_{i_2} = s_2, \ldots t_{i_m} = s_m$

3. $i_m - i_1 < w$

The first two conditions above state that, S is a subsequence of T, while the last condition guarantees that, S is a subsequence of T within the window of length w.

It is of interest to estimate $\Omega^{\exists}(n, w, m, S, \mathcal{A})$ that represents the number of windows of length w containing *at least one* occurrence of S when sliding the window n consecutive events in the event sequence T over the alphabet $\mathcal{A}$.
Notation: Throughout the paper we drop S and $\mathcal{A}$ in the notation $\Omega^{\exists}(n, w, m, S, \mathcal{A})$ using $\Omega^{\exists}(n, w, m)$ instead in which case S and $\mathcal{A}$ are implied. We use the same rule for all other variables depending on w, m, S and $\mathcal{A}$. We also use $m - k$ instead of m, which means "dropping the last k symbols of S", e.g., $P^{\exists}(w, m - k)$ implies a pattern that is the prefix of S of length $m - k$.

Based on the observed value of $\Omega^{\exists}(n, w, m)$ our task is to decide whether a suspicious activities took place or not. We shall find a threshold $\tau(n, m, W)$ such that, for any given β (e.g., $\beta = 10^{-5}$) the following holds

$$\Pr\{\Omega^{\exists}(n, w, m) > \tau(n, m, w)\} \quad < \quad \beta. \tag{1}$$

That is, if one observes more than $\tau(n, m, w)$ occurrences of windows with suspicious subsequences then the probability that such a number is generated fortuitously is highly unlikely (i.e., its probability is smaller than β).

Another interesting problem is the selection of parameters of a monitoring system, in particular the size of the window so one can design properly the system (e.g., to make sure that a pattern does not occur almost surely in every window).

We shall answer these questions in this paper, by assuming that the event sequence is generated by a memoryless (Bernoulli) source and/or a Markovian source. We start with the memoryless case, by assuming that the symbols of the event sequence are generated independently of each others.

2.2 Analytical results

We need to analyze $\Omega^{\exists}(n, w, m)$ in order to find the threshold as defined above. We shall prove here that appropriately normalized $\Omega^{\exists}(n, w, m)$ is normally distributed. We also find the mean and the variance of $\Omega^{\exists}(n, w, m)$.

We start with computing the mean value $\mathbf{E}[\Omega^{\exists}(n, w, m)]$. Clearly, it is equal to

$$\mathbf{E}[\Omega^{\exists}(n, w, m)] \quad = \quad nP^{\exists}(w, m)$$

where $P^{\exists}(w, m)$ is the probability that a window of size w contains at least one occurrence of the episode S of size m as a subsequence.

$P^{\exists}$ satisfies the following recurrence

$$\begin{cases} P^{\exists}(w, m) &= (1 - p_m)P^{\exists}(w - 1, m) + p_m P^{\exists}(w - 1, m - 1) & w > 0,\ m > 0 \\ P^{\exists}(w, 0) &= 1 & w > 0 \\ P^{\exists}(0, m) &= 0 & m > 0 \\ P^{\exists}(0, 0) &= 1 & \end{cases}$$

Indeed, consider a window of size $w-1$. Observe that either the last symbol of the pattern, s_m, does not occur at the w-th position of the window or it does occur. In the former situation S must occur within the window of size $w-1$ leading to the term $(1-p_m)P^{\exists}(w-1,m)$ of the above recurrence. The latter situation provides the second term of the recurrence.

In section 4 we solve the above recurrence using generating functions. Then we apply Cauchy's residue theorem to obtain an asymptotic expansion of $P^{\exists}(w,m)$ for fixed m and large w. We summarize our results in the next theorem.

Theorem 1 *Consider a memoryless source with p_i being the probability of generating symbol $a_i \in \mathcal{A}$. Let also*

$$P(S) = \prod_{i=1}^{m} p_i$$

Then for all m and $w \geq m$ we have

$$P^{\exists}(w,m) = P(S) \sum_{i=0}^{w-m} \sum_{\sum_{k=0}^{m} n_k = i} \prod_{k=1}^{m} q_k^{n_k}$$

where $q_i = 1 - p_i$.

Let now m be fixed and $i \neq j$ implies $p_i \neq p_j$. Then for any $\varepsilon > 0$

$$P^{\exists}(w,m) = 1 - P(S) \sum_{i=1}^{m} \frac{(1-p_i)^w}{p_i} \prod_{j \neq i}^{m} \frac{1}{p_j - p_i} + O(\varepsilon^w)$$

as $w \to \infty$.

When establishing the above formulas for $P^{\exists}(w,m)$ we also solved two related combinatorial problems on strings that are of interest. Namely, given $\mathcal{A}$, w and S:

1. Construct the set $\mathcal{W}^{\exists}(w,m)$ of all windows as strings of length w over $\mathcal{A}$ containing all possible occurrences of S as a subsequence. In Theorem 4 we show the enumeration formula.

2. Find the cardinality of $\mathcal{W}^{\exists}(w,m)$, that we denote $C^{\exists}(w,m)$. In Theorem 5 we prove that

$$C^{\exists}(w,m) = \sum_{k=0}^{w-m} \binom{k+m-1}{k} (|\mathcal{A}|-1)^k |\mathcal{A}|^{w-m-k}$$

We refer to section 4 for the solution for those problems.

Using the enumeration formula for $\mathcal{W}^{\exists}(w,m)$, in Theorem 8 we prove that, in the first order Markov model, $P^{\exists}(w,m)$ can be expressed as follows

$$P^{\exists}(w,m) = \sum_{i=0}^{C^{\exists}(w,m)} P(\mathcal{W}^{\exists}(w,m)[i][1]) P(\mathcal{W}^{\exists}(w,m)[i][2] | \mathcal{W}^{\exists}(w,m)[i][1]) \ldots P(\mathcal{W}^{\exists}(w,m)[i][m+1] | \mathcal{W}^{\exists}(w,m)[i][m])$$

where $\mathcal{W}^{\exists}(w,m)[i][l]$ is the l-th symbol of the i-th member of $\mathcal{W}^{\exists}(w,m)$ and $P(\mathcal{W}^{\exists}(w,m)[i][l] | \mathcal{W}^{\exists}(w,m)[i][m])$ is the conditional probability that one symbol occurs assuming that the other one has occurred at an earlier position.

Let us now come back to $\Omega^{\exists}(n,w,m)$ and establish its normal limiting distribution. Observe that

$$\Omega^{\exists}(n,w,m) = \sum_{i=1}^{n} X_i^{\exists}$$

where

$$X_i^{\exists} = \begin{cases} 1 & \text{if } S \text{ occurs at least once as a subsequence in } W \text{ ending at position } i \text{ in } T; \\ 0 & \text{otherwise.} \end{cases}$$

and easily we have $\mathbf{E}[X_i^{\exists}] = P^{\exists}(w,m)$ and $\mathbf{Var}[X_i^{\exists}] = P^{\exists}(w,m) - (P^{\exists}(w,m))^2$.

Let $P^{\exists}_{(X_i^{\exists}\cap X_j^{\exists})}(w,m,k)$ be the probability that two overlapping windows at respective position i and j for $|i-j| < w$ have $X_i = 1$ and $X_j = 1$.

The variance can be expressed as follows

$$\begin{aligned}\mathbf{Var}[\Omega^{\exists}(n,w,m)] &= n\mathbf{Var}[X_1^{\exists}] \\ &\quad +2(n-w+1)\left[\sum_{k=1}^{w-1}\left[P^{\exists}_{(X_i^{\exists}\cap X_j^{\exists})}(w,m,k) - (P^{\exists}(w,m))^2\right]\right] \\ &\quad +2\left[\sum_{k=1}^{w-2}\left[P^{\exists}_{(X_i^{\exists}\cap X_j^{\exists})}(w,m,k) - (P^{\exists}(w,m))^2\right]\right].\end{aligned}$$

The two terms involving $P^{\exists}_{(X_i^{\exists}\cap X_j^{\exists})}$ in the above formula represent correlation between windows (with $2(w-1)$ neighborhood). In section 4 we suggest how to compute $P^{\exists}_{(X_i^{\exists}\cap X_j^{\exists})}$. One concludes, however, that $\mathbf{Var}[\Omega^{\exists}(n,w,m)] \sim n\sigma$ for some $\sigma > 0$.

In view of the above, and using the fact that $\Omega^{\exists}(n,w,m)$ is the so called m-dependent sequence (i.e., $\Omega^{\exists}(n,w,m)$ depends on the last m symbols), we may apply Theorem 27.5 of [3] to establish the central limit theorem for $\Omega^{\exists}(n,w,m)$, as stated below.

Theorem 2 *Consider a memoryless source. Then for $x = O(1)$ we have*

$$\lim_{n\to\infty} \Pr\left\{\frac{\Omega^{\exists}(n,w,m) - \mathbf{E}[\Omega^{\exists}(n,w,m)]}{\sqrt{\mathbf{Var}[\Omega^{\exists}(n,w,m)]}} < x\right\} = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{\frac{-t^2}{2}}\,dt$$

for m and w fixed.

Now, we are in position to establish the threshold $\tau(n,w,m)$. Knowing that, the distribution of $\Omega^{\exists}(n,w,m)$ around the mean is normal within a *finite* number of standard deviations $\sqrt{\mathbf{Var}[\Omega^{\exists}(n,w,m)]}$, we first find a_0 such that for a given β

$$\beta = \int_{a_0}^{\infty} e^{-\frac{t^2}{2}}\,dt = \Pr\{N(0,1) > a_0\}$$

where $N(0,1)$ is a random variable with the standard normal distribution. We set the threshold $\tau(n,w,m)$ as

$$\tau(n,w,m) \quad = \quad \mathbf{E}[\Omega^{\exists}(n,w,m)] + a_0\sqrt{\mathbf{Var}[\Omega^{\exists}(n,w,m)]}$$

that is basically equivalent to our previous formulation as long as we are within the region where the central limit theorem applies, that is, $\Pr\{\Omega^{\exists}(n,w,m) > \tau(n,w,m)\} = \beta$.

3 Experimental Results

The purpose of our experiments was to test the applicability of the analytical results we derived for sources that are apparently not memoryless, i.e., they do not satisfy the assumptions under which the formulas for memoryless source were derived. We of course do not need to test the formulas for memoryless sources, because we already know the equations hold in such cases. Therefore we ran the experiments for an English text source and also for web access data.

An English text is of course not memoryless. As an example consider string "th" which occurs more frequently than "tz" or "ts". So in this example the letter "h" will occur more likely if the previous letter was "t".

The web accesses are also not memoryless not only because of hierarchical structure but also because of correlations between links. For example a person looking for a product in an on-line store will most likely visit all manufacturers of the search product.

We divided our sources into training sets and testing sets. Training sets are data sets, which we consider constitute normal behavior for the environment from which the data were drawn. Once the training data has been characterized, which in our case means our probability model has been built, we can start monitoring unknown data called testing data. During the monitoring process the testing data are compared to expectations generated by the training data.

Thus, the main focus of our experiments was to show the formula for $P^{\exists}(w,m)$ works well for apparently non memoryless sources since it is the corner stone of our theory. To accomplish this we estimated the actual probability of existence based on the actual number of windows $\Omega^{\exists}(n,w,m)$ as $P_e^{\exists}(w,m) = \frac{\Omega^{\exists}(n,w,m)}{n}$ and compared its value to the computed $P^{\exists}(w,m)$ for different values of w. We used the following error metric d

$$d = \left[\frac{1}{r}\sum_{i=1}^{r}\frac{|P_e^{\exists}(w_i,m) - P^{\exists}(w_i,m)|}{P_e^{\exists}(w_i,m)}\right] 100\%$$

where $w_1 < w_2 < \ldots w_r$ are the tested window sizes.

We used an algorithm, based on dynamic programming, for finding windowed subsequences. We also implement the dynamic programming solution to $P^{\exists}(w,m)$ given in section 4.3. We converted the sources appropriately to a special text file format that was used by the algorithm implemented in C++ and run under Linux.

3.1 English text source

The text source we use is an on-line version of *War and Peace* by Leo Tolstoy from `www.friends-partners.org/newfriends/culture/literature/war_and_peace/war-peace_intro.html`. The work consists of 15 books. Each book has over 20 chapters each of which consists of over 5000 letters. We preprocessed the chapters in order to remove all symbols but 26 letters of the English alphabet without distinguishing between upper and lower case letters.

In the first experiment we compared the analytically computed $P^{\exists}(w,m)$ (cf. Theorem 1) with its estimator $P_e^{\exists}(w,m) = \frac{\Omega^{\exists}(n,w,m)}{n}$. We used chapter 1-5 as a training set for estimation of $p_1, p_2, \ldots p_m$ based on the symbol frequencies. We set $S = gwadera$ and for each w in the range $13 - 600$ we ran the algorithm for finding $\Omega^{\exists}(n,w,m)$ in chapter 6 as the testing source. Figures 1 and 2 illustrate the results showing two main facts: $P^{\exists}(w,m)$ approaches 1 as w goes to infinity and $P^{\exists}(w,m)$ very closely reflects the real $P_e^{\exists}(w,m)$ since the difference d between them equals 12%.

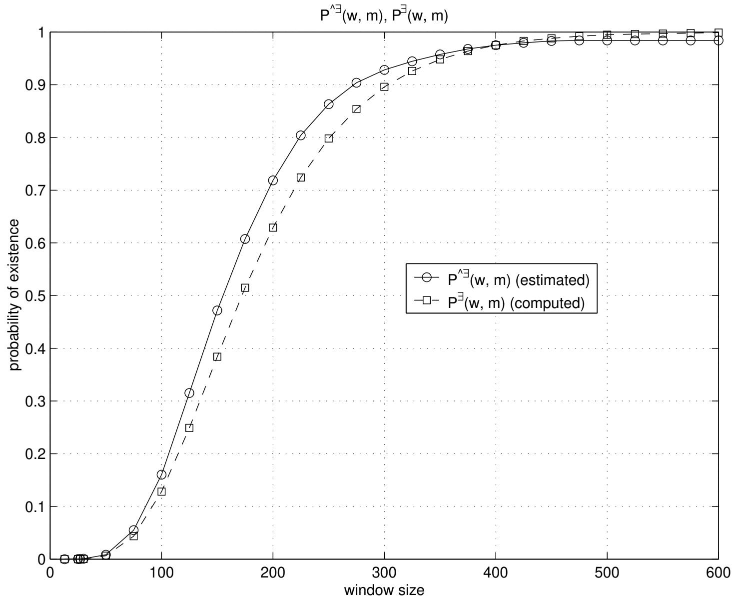

Figure 1: $P_e^{\exists}(w,m) = \frac{\Omega^{\exists}(n,w,m)}{n}$ and $P^{\exists}(w,m)$ for $S = gwadera$

In the next experiment we chose an "unusual" pattern $S = szpankowskiatallahgwadera$ and wanted to determine a *threshold of existence* i.e. the smallest w such that, for a given n, $\Omega^{\exists}(n,w,m) > 0$. The ability to analytically find such a threshold is of importance in selecting the proper window size for monitoring system. We used the same training and testing sources as in the previous experiment.

The pattern is unusual for many reasons, one of which is the fact that, letter z is very rare in English text. Results are presented in Figure 3. From the table we see that even for such "strange" patterns we can closely approximate the *threshold of existence*. The reason that $\Omega^{\exists}(n,w,m)$ and $\mathbf{E}[\Omega^{\exists}(n,w,m)]$ differ more than in the previous experiment is that $\mathbf{Var}[\Omega^{\exists}(n,w,m)]$ grows with m.

In the third experiment we estimated variance of $\Omega^{\exists}(n,w,m)$ denoted $\widehat{\mathbf{Var}}[\Omega^{\exists}(n,w,m)]$. For this purpose we randomly chose 8 chapters and shortened them to the same length 8000 letters creating 8 training sources denoted $T_1, T_2, \ldots, T_8$ and one testing source T_9. We used the following sample mean and variance estimators

$$\widehat{\mathbf{E}}[\Omega^{\exists}(n,w,m)] = \frac{\sum_{i=1}^{8} \Omega^{\exists}(n,w,m)_i}{8} \tag{2}$$

$$\sqrt{\widehat{\mathbf{Var}}[\Omega^{\exists}(n,w,m)]} = \sqrt{\frac{\sum_{i=1}^{8}(\Omega_i^{\exists}(n) - \widehat{\mathbf{E}}[\Omega^{\exists}(n,w,m)])^2}{7}}. \tag{3}$$

for $S = wojciech$ and $w = 100$. Table 4 presents the computation of these estimators. In particular, we set $\tau(n,w,m) = \sqrt{\mathbf{E}[\Omega^{\exists}(n,w,m)]} + 5\sqrt{\widehat{\mathbf{Var}}[\Omega^{\exists}(n,w,m)]}$, that is, $a_0 = 5$. Now we know what the normal behavior is in terms of $\Omega^{\exists}(n,w,m)$ given $S = wojciech$ and $w = 100$. We used such a trained model to monitor T_9 as the testing set. To verify our threshold experimentally we artificially kept injecting

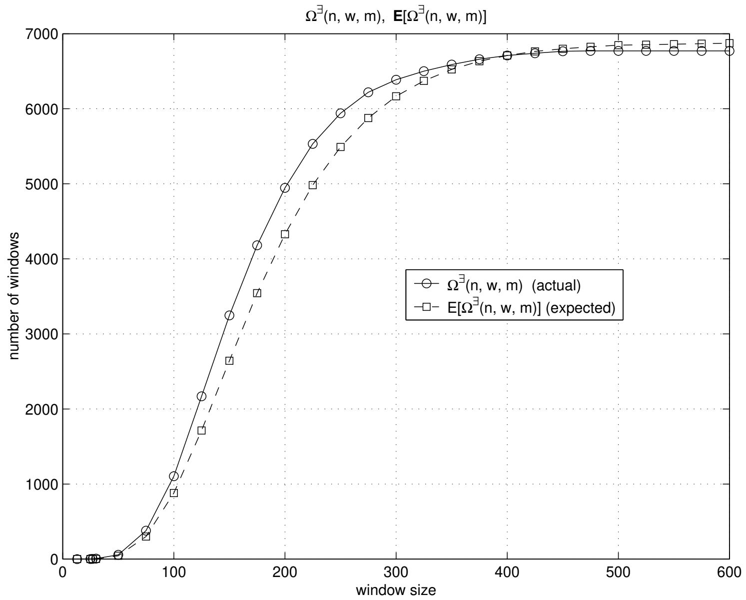

Figure 2: $\Omega^{\exists}(n, w, m)$ and $\mathbf{E}[\Omega^{\exists}(n, w, m)]$ for $S = gwadera$

$S = wojciech$ as a subsequence into different places in T_9. After each insertion we ran the algorithm for finding $\Omega^{\exists}(n, w, m)$ and checked whether we exceeded $\tau(n, w, m)$. To make it more interesting we considered two values of gaps between inserted symbols of S: $gap = 0$ and $gap = 11$. In other words we injected S as $s_1\#^{gap}s_2\#^{gap}\ldots\#^{gap}s_m$, where # means a do not care symbol. The results are shown in Figure 5. We can see that if $gap = 0$, then we need only two patterns to exceed $\tau(n, w, m)$ versus three if $gap = 11$. This makes sense if we notice that if the pattern is stretched to the window boundaries ($gap = 11$) then it is more noise like compared to the case when $gap = 0$, which suggests an intentional action (attack) and should be detected early. Furthermore notice that for the training set $\widehat{\mathbf{E}}[\Omega^{\exists}(n, w, m)] = 30$ and $\widehat{\mathbf{Var}}[\Omega^{\exists}(n, w, m)] = 20.6$ so by injecting two episodes for ($gap = 11$) we really did not go beyond the normal behavior.

3.2 Web access data

We used logs of user accesses to the music machines web site (currently at `http://machines.hyperreal.org`), which record accesses from 1/01/99 through 4/30/99 . The logs have been anonymized with respect to originating machines. That is, in each hit, the IP address of the machine generating the server request has been converted to a random looking number. All hits from one machine on a particular day are labeled with the same number. In the experiments we focused on `http://machines.hyperreal.org/manufacturers/` web page containing links to manufacturers of music instruments. Each link corresponds to an alphabet symbol and the alphabet size $|A| = 81$. The training and testing sequences were created by considering only unique accesses made by the same originating machine. If a given host made many accesses to the same manufacturer per session then we treated it as one access and consider the first access only.

w	$\Omega^{\exists}(n,w,m)$	$\mathbf{E}[\Omega^{\exists}(n,w,m)] = nP^{\exists}(w,m)$
100	0	0
200	0	0
300	0	0.05
400	0	1.46
500	0	12.48
600	3	51.67
650	53	88.54
700	103	139.66

Figure 3: Detection of pattern $S = szpankowskiatallahgwadera$

T_i	1	2	3	4	5	6	7	8	$\widehat{\mathbf{E}}[\Omega^{\exists}(n,w,m)]$	$\sqrt{\widehat{\mathbf{Var}}[\Omega^{\exists}(n,w,m)]}$	τ
$\Omega_i^{\exists}(n,w,m)$	31	30	30	47	50	0	52	0	30	20.6	133

Figure 4: Computation of τ for $a_0 = 5$ for $S = wojciech$ and $w = 100$ using 8 randomly selected chapters each of length $N = 8000$.

In the first experiment we compared the computed $P^{\exists}(w,m)$ with its estimator $P_e^{\exists}(w,m) = \frac{\Omega^{\exists}(n,w,m)}{n}$. We created three sources T_1, T_2, T_3 each of length $N = 22000$. The training set established T_1, T_2 and the testing source was T_3. We set $S = \{Akai, ARP, Korg, Moog, Yamaha, Casio, Sequential\}$ and for each w in the range $25 - 500$ we ran the algorithm for finding $\Omega^{\exists}(n,w,m)$ on T_3. Figures 6 and 7 illustrate the results. $P^{\exists}(w,m)$ still provides a good approximation of $P_e^{\exists}(w,m)$ ($d = 18\%$). The reason the value of d is bigger than for the text source is the fact that the web accesses are a more memory dependent source than the text for obvious reasons. Therefore the Markov model seems to be more suitable for the web access source.

Detection of $S = \{Peavey, Boss, Alesis, Emu, Rhodes, Linn, Ensoniq, Univox, Waldorf\}$ is shown in Figure 8. Even though we still have the same problem that there are some sequences of the same symbol in the source the results are quite good assuming also a greater variance because of the bigger alphabet size.

4 Derivations of Analytical Results

In this section we provide derivations and proofs of findings presented in the previous section.

4.1 Set of Windows Containing S as a subsequence

Let $\mathcal{W}^{\exists}(w,m)$ be the *set* (enumeration) of all distinct windows of length w containing S as a subsequence. In the memoryless model $P^{\exists}(w,m)$ is equal to the sum of probabilities of all strings in $\mathcal{W}^{\exists}(w,m)$. Therefore, given $\mathcal{A}$, S and w we need a way to enumerate all strings in the set $\mathcal{W}^{\exists}(w,m)$

Let $\mathcal{W}^{\exists}(w,m)[i]$ be the i-th string of $\mathcal{W}^{\exists}(w,m)$ since we can always order the strings lexicographically.

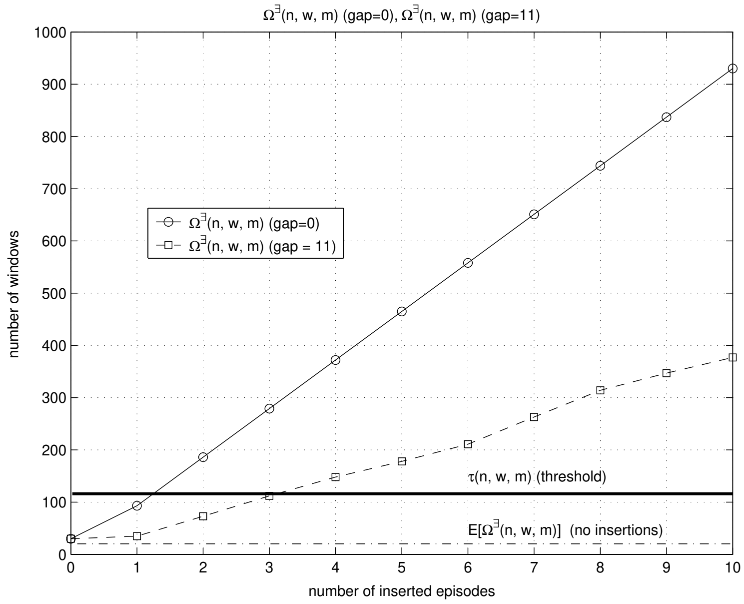

Figure 5: Detection of artificially inserted patterns *wojciech* for $gap = 0$ and $gap = 11$. The horizontal solid line represent the threshold set at $\tau(n,w,m) = \widehat{\mathbf{E}}[\Omega^{\exists}(n,w,m)] + 5\sqrt{\widehat{Var}[\Omega^{\exists}(n,w,m)]}$, where $\widehat{\mathbf{E}}[\Omega^{\exists}(n,w,m)] = 30$ and $\sqrt{\widehat{Var}[\Omega^{\exists}(n,w,m)]} = 20.66$. The horizontal dash-dot line shows $\mathbf{E}[\Omega^{\exists}(n,w,m)] = 24$ for no insertions.

Then we can express $P^{\exists}(w,m)$ as follows

$$P^{\exists}(w,m) = \sum_{i=1}^{|\mathcal{W}^{\exists}(w,m)|} P(\mathcal{W}^{\exists}(w,m)[i]) \tag{4}$$

Now we present the recurence for $\mathcal{W}^{\exists}(w,m)$.

Theorem 3 *The recursive formula for $\mathcal{W}^{\exists}(w,m)$ has the following form*

$$\begin{cases} \mathcal{W}^{\exists}(w,m) = (\mathcal{A} - \{s_m\}) \times \mathcal{W}^{\exists}(w-1,m) \cup \{s_m\} \times \mathcal{W}^{\exists}(w-1,m-1) & w > 0 \cap m > 0 \\ \mathcal{W}^{\exists}(w,0) = \mathcal{A}^w & w > 0 \\ \mathcal{W}^{\exists}(0,m) = 0 & m > 0 \\ \mathcal{W}^{\exists}(0,0) = 1 & \end{cases}$$

Proof. The proof below is combinatorial. We need to show that we enumerate all distinct elements of $\mathcal{W}^{\exists}(w,m)$. That we generate distinct elements at each level of the recursion is obvious since we divide $\mathcal{W}^{\exists}(w,m)$ into two subsets: strings having s_m as the last symbol and those that do not. Now we show we generate all strings. Consider the enumeration of all $\binom{w}{m}$ positions of S as a subsequence in a window of length w. We enumerate the positions using *reverse lex* order i.e. $(i_1, i_2, \ldots, i_m)$ is before $(i_1^{'}, i_2^{'}, \ldots, i_m^{'})$ if for some j, $1 < j \le m$, $(i_j, i_2, \ldots, i_m) = (i_j^{'}, i_2^{'}, \ldots, i_m^{'})$ and $i_{j-1} < i_{j-1}^{'}$.

Therefore $\mathcal{W}^{\exists}(w,m)$ can be split into two subsets:

1. Windows having s_m on the last position. But then if we remove the last symbol from this subset of $\mathcal{W}^{\exists}(w,m)$ then we must create $\mathcal{W}^{\exists}(w-1,m-1)$.

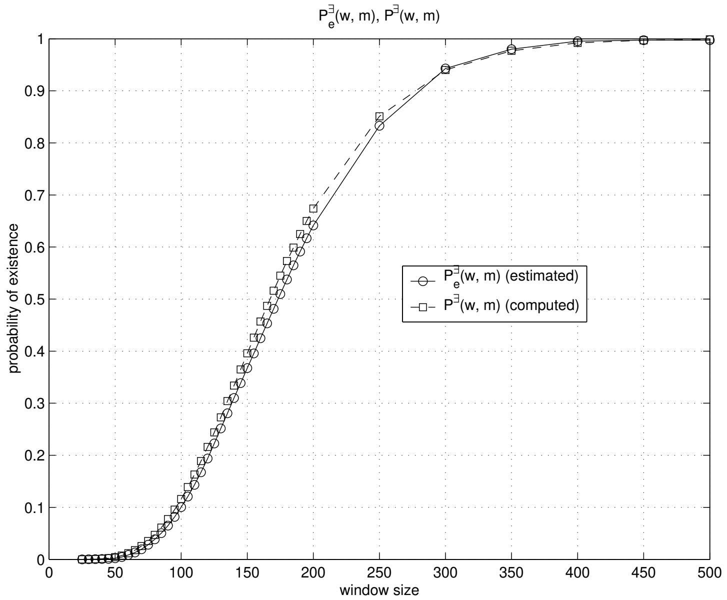

Figure 6: $\Omega^\exists(n, w, m)$ and $\mathbf{E}[\Omega^\exists(n, w, m)])$ number of occurrences for $S = \{Akai, ARP, Korg, Moog, Yamaha, Casio, Sequential\}$

2. Windows not having s_m on the last position. But then if we remove the last symbol from this subset then we must create $\mathcal{W}^\exists(w-1, m)$ $\square$.

Formally the solution to $\mathcal{W}^\exists(w, m)$ can be expressed as follows.

Theorem 4 *The set of all distinct windows of length w containing S of length m as a subsequence can be enumerated as follows.*

$$\mathcal{W}^\exists(w, m) \quad = \bigcup_{\sum_{k=1}^{m+1} n_k = w-m} (\mathcal{A} - s_1)^{n_1} \times \{s_1\} \times (\mathcal{A} - s_2)^{n_2} \times \{s_2\} \times \ldots \times (\mathcal{A} - s_m)^{n_m} \times \{s_m\} \times \mathcal{A}^{n_{m+1}}$$

We can think of the members of $\mathcal{W}^\exists(w, m)$ as strings consisting of edge symbols of all unique (at least one edge different) paths of length w from the start state, 0, to the ending state, m of the automaton given in figure 9.

Based on Theorem 4 we can divide the elements of $\mathcal{W}^\exists(w, m)$ into equivalence classes $\mathcal{V}^\exists(n_1, n_2, \ldots n_{m+1})$ with respect to the ordered sequences of $(n_1, n_2, \ldots, n_{m+1})$ such that $\sum_{i=1}^{m+1} n_i = w - m$. Since the number of such ordered partitions is trivially know to be $\binom{w-m+m+1-1}{w-m} = \binom{w}{w-m}$, it follows that $|\mathcal{V}^\exists(n_1, n_2, \ldots n_{m+1})| = \binom{w}{w-m}$.

Similarly we can define $\mathcal{W}^{\nexists}(w, m)$ as the *set* of windows over an alphabet $\mathcal{A}$, none of which has an occurrence of S as a subsequence.

$$\mathcal{W}^{\nexists}(w, m) \quad = \quad \mathcal{A}^w \setminus \mathcal{W}^\exists(w, m)$$

EXAMPLE. Let $\mathcal{A} = \{a, b\}$ $S = ba$ and $w = 3$. We generate $\mathcal{W}^\exists(3, 2)$ and compute $P^\exists(w, m)$.

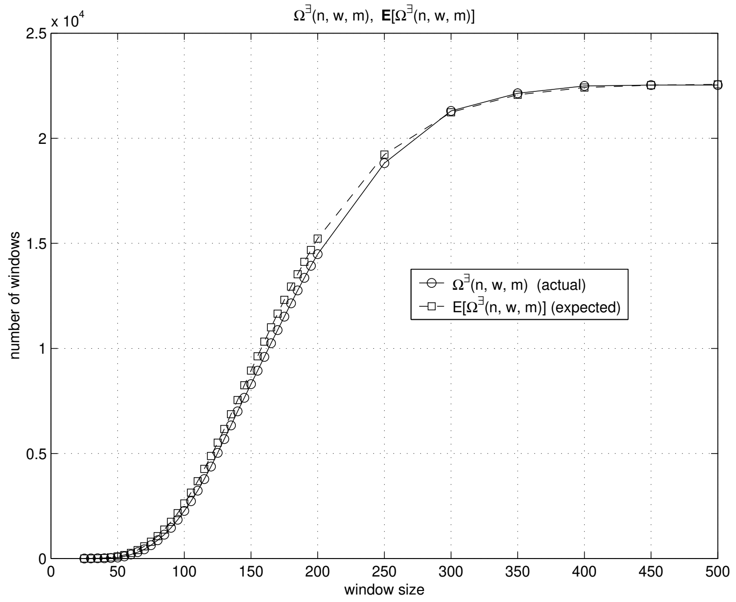

Figure 7: $P_e^{\exists}(w,m) = \frac{\Omega^{\exists}(n,w,m)}{n}$ and computed $P^{\exists}(w,m)$ for $S = \{Akai, ARP, Korg, Moog, Yamaha, Casio, Sequential\}$

i	$W(3,2)[i]$	n_1	n_2	n_3
1	$\underline{ba}a$	0	0	1
2	$\underline{ba}b$	0	0	1
3	$\underline{b}b\underline{a}$	0	1	0
4	$a\underline{ba}$	1	0	0

From (4) we obtain $P^{\exists}(3,2) = P(S)(p_a + p_b + p_b + p_a) = 2P(S)$.

4.2 Evaluation of $C^{\exists}(w,m)$

The recurrence for $C^{\exists}(w,m)$ follows directly form the one for $\mathcal{W}^{\exists}(w,m)$.

$$\begin{cases} C^{\exists}(w,m) & = & (|\mathcal{A}|-1)C^{\exists}(w-1,m) + C^{\exists}(w-1,m-1) & w > 0 \cap m > 0 \\ C^{\exists}(w,0) & = & |\mathcal{A}|^w & w > 0 \\ C^{\exists}(0,m) & = & 0 & m > 0 \\ C^{\exists}(0,0) & = & 1 & \end{cases}$$

We can also define $C^{\nexists}(w,m)$ as the number of all windows of length w *not* containing a pattern S of length m as a subsequence.

$$C^{\nexists}(w,m) \quad = \quad |\mathcal{A}|^w - C^{\exists}(w,m) \tag{5}$$

We use the method of generating functions to find the solution for $C^{\exists}(w,m)$.

Let

$$W_m(x) \quad = \quad \sum_{w=0} C^{\exists}(w,m)x^w \tag{6}$$

w	$\Omega^{\exists}(n,w,m)$	$\mathbf{E}[\Omega^{\exists}(n,w,m)] = nP^{\exists}(w,m)$
150	0	4.19
182	1	17.15
183	2	17.83
184	3	18.61
185	4	19.33
190	9	23.30
200	19	33.05
250	133	142.30
300	336	421.24

Figure 8: Detection of pattern $S = \{Peavey, Boss, Alesis, Emu, Rhodes, Linn, Ensoniq, Univox, Waldorf\}$

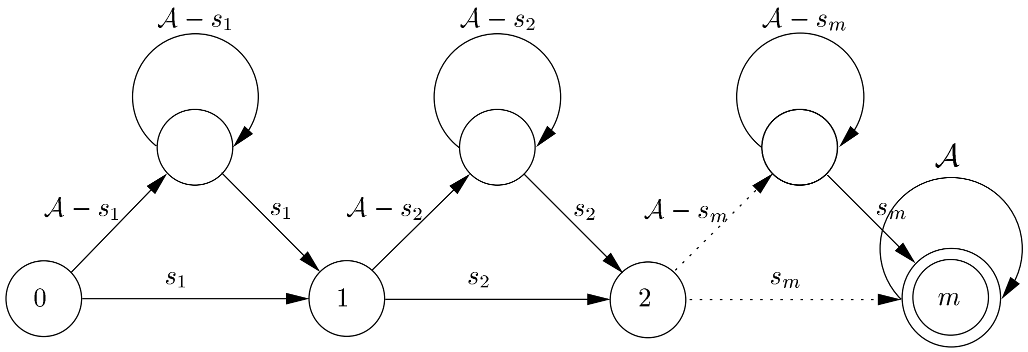

Figure 9: Graphical interpretation of the solution to $\mathcal{W}^{\exists}(w,m)$

$$\begin{cases} W_m(x) &=& (|\mathcal{A}|-1)\sum_{w=1} C^{\exists}(w-1,m)x^w + \sum_{w=1} C^{\exists}(w-1,m-1)x^w \quad m>0 \\ W_0(x) &=& \sum_{w=0} C^{\exists}(w,0)x^w \end{cases}$$

$$\begin{aligned} W_m(x) &= (|\mathcal{A}|-1)\sum_{w=1} C^{\exists}(w-1,m)x^w + \sum_{w=1} C^{\exists}(w-1,m-1)x^w && (7) \\ &= (|\mathcal{A}|-1)x\sum_{w=1} C^{\exists}(w-1,m)x^{w-1} + x\sum_{w=1} C^{\exists}(w-1,m-1)x^{w-1} && (8) \\ &= (|\mathcal{A}|-1)x\sum_{w=0} C^{\exists}(w,m)x^w + x\sum_{w=0} C^{\exists}(w,m-1)x^w && (9) \\ &= (|\mathcal{A}|-1)xW_m(x) + xW_{m-1}(x) && (10) \end{aligned}$$

$$W_m(x)(1-(|\mathcal{A}|-1)x) = xW_{m-1}(x) \tag{11}$$

$$W_m(x) = \frac{x}{(1-(|\mathcal{A}|-1)x)}W_{m-1}(x) \tag{12}$$

$$W_m(x) = x^m \frac{1}{(1-(|\mathcal{A}|-1)x)}\frac{1}{(1-|\mathcal{A}|x)} \tag{13}$$

$$\frac{1}{(1-|\mathcal{A}|x)} = \sum_{w=0} |\mathcal{A}|^w x^w \tag{14}$$

$$\frac{x^m}{(1-(|\mathcal{A}|-1)x)} = \sum_{w=0} \binom{w+m-1}{w} (|\mathcal{A}|-1)^w x^w \tag{15}$$

$$C^{\exists}(w,m) = [x^w]W_m(x) \tag{16}$$

Theorem 5 *The number of all windows of length w over an alphabet $\mathcal{A}$ which contain at least one occurrence of a pattern of length m does not depend on the symbols of the pattern and is equal to:*

$$C^{\exists}(w,m) = \sum_{k=0}^{w-m} \binom{k+m-1}{k} (|\mathcal{A}|-1)^k |\mathcal{A}|^{w-m-k} \tag{17}$$

4.3 Evaluation of $P^{\exists}(w,m)$

The recurrence for $P^{\exists}(w,m)$ follows directly form the one for $\mathcal{W}^{\exists}(w,m)$.

$$\begin{cases} P^{\exists}(w,m) &= (1-p_m)P^{\exists}(w-1,m) + p_m P^{\exists}(w-1,m-1) & w>0 \cap m>0 \\ P^{\exists}(w,0) &= 1 & w>0 \\ P^{\exists}(0,m) &= 0 & m>0 \\ P^{\exists}(0,0) &= 1 & \end{cases}$$

We can also define $P^{\nexists}(w,m)$ as the probability that a window of length w *does not* contain any occurrence of a pattern S of length m as a subsequence.

$$P^{\nexists}(w,m) = 1 - P^{\exists}(w,m) \tag{18}$$

We use the method of generating functions to find the solution for $P^{\exists}(w,m)$.

Let

$$W_m(x) = \sum_{w=0} P^{\exists}(w,m)x^w \tag{19}$$

$$\begin{cases} W_m(x) &= q_m \sum_{w=1} P^{\exists}(w-1,m)x^w + p_m \sum_{w=1} P^{\exists}(w-1,m-1)x^w & m>0 \\ W_0(x) &= \sum_{w=0} P^{\exists}(w,0)x^w & \end{cases}$$

$$\begin{aligned} W_m(x) &= q_m \sum_{w=1} P^{\exists}(w-1,m)x^w + p_m \sum_{w=1} P^{\exists}(w-1,m-1)x^w && (20) \\ &= q_m x \sum_{w=1} P^{\exists}(w-1,m)x^{w-1} + p_m x \sum_{w=1} P^{\exists}(w-1,m-1)x^{w-1} && (21) \\ &= q_m x \sum_{w=0} P^{\exists}(w,m)x^w + p_m x \sum_{w=0} P^{\exists}(w,m-1)x^w && (22) \\ &= q_m x W_m(x) + p_m x W_{m-1}(x) && (23) \end{aligned}$$

$$W_m(x)(1-q_m x) \quad = \quad p_m x W_{m-1}(x) \tag{24}$$

$$W_m(x) = \frac{p_m x}{(1-q_m x)} W_{m-1}(x) \tag{25}$$

$$W_m(x) = \prod_{i=1}^{m} p_i x^m \prod_{i=1}^{m} \frac{1}{(1-q_i x)} \frac{1}{(1-x)} \tag{26}$$

$$\frac{1}{(1-x)} \quad = \quad \sum_{w=0} x^w \tag{27}$$

$$\prod_{i=1}^{m} \frac{1}{(1-q_i x)} \quad = \quad \prod_{i=1}^{m} \sum_{w=0} q_i^w x^w \tag{28}$$

$$= \quad \bigcup_{\sum_{k=1}^{m} n_k = i}^{1 \le i \le \binom{w+m-1}{w}} q_1^{n_1} q_2^{n_2} \ldots q_m^{n_m} \tag{29}$$

$$= \quad \bigcup_{\sum_{k=1}^{m} n_k = i}^{1 \le i \le \binom{w+m-1}{w}} \prod_{k=1}^{m} q_k^{n_k} \tag{30}$$

$$P^{\exists}(w,m) \quad = \quad [x^w] W_m(x) \tag{31}$$

Theorem 6 *In the Bernoulli model the probability that a window of length w contains at least one occurrence of a pattern S of length m is equal to:*

$$P^{\exists}(w,m) \quad = \quad P(S) \sum_{i=0}^{w-m} \sum_{\sum_{k=0}^{m} n_k = i} \prod_{k=1}^{m} q_k^{n_k} \tag{32}$$

We also performed an asymptotic analysis using equation 26 and obtained an asymptotic approximation of $P^{\exists}(w,m)$.

Theorem 7 *$P^{\exists}(w,m)$ asymptotically converges to the following formula*

$$P^{\exists}(w,m) \quad \approx \quad 1 - P(S) \sum_{i=1}^{m} \frac{(1-p_i)^w}{p_i} \prod_{j \ne i}^{m} \frac{1}{p_j - p_i} \tag{33}$$

The asymptotic approximation reveals the expected fact about the behavior of $P^{\exists}(w,m)$ as $w \to \infty$.

From 4 we can compute $P^{\exists}(w,m)$ for the Markov model of any order. Let $\mathcal{W}^{\exists}{}_{[i,j]}$ be the j-th symbol in $\mathcal{W}^{\exists}(w,m)[i]$ where $j = 1, 2, \ldots, w$. Then for the first order model we obtain the following theorem

Theorem 8 *In the first order Markov model the probability that a window of length w contains at least one occurrence of a pattern S of length m is equal to:*

$$\begin{aligned} P^{\exists}(w,m) \quad = \quad & \sum_{i=0}^{C^{\exists}(w,m)} P(\mathcal{W}^{\exists}(w,m)[i][1]) P(\mathcal{W}^{\exists}(w,m)[i][2] | \mathcal{W}^{\exists}(w,m)[i][1]) \ldots \\ & P(\mathcal{W}^{\exists}(w,m)[i][m+1] | \mathcal{W}^{\exists}(w,m)[i][m]) \end{aligned}$$

where $\mathcal{W}^{\exists}(w,m)[i][l]$ is the l-the symbol of the i-th member of $\mathcal{W}^{\exists}(w,m)$ in lexicographic order and $P(\mathcal{W}^{\exists}(w,m)[i][l]|\mathcal{W}^{\exists}(w,m)[i][m])$ is the conditional probability that one symbol occurs assuming that the other one has occurred on earlier position.

Finally, we propose a dynamic programming equation for computing $P^{\exists}(w,m)$. Let $Q^n(j)$ denote the product $\prod_{k=1}^{j} q_k^{n_k}$ such that $\sum_{k=1}^{j} n_k = n$ then

$$\begin{cases} Q^n(j) & = & \sum_{k=1}^{n} Q^{n-k}(j-1)q^k(j) & 1 \le j \le m \\ Q^n(1) & = & q_1^n & \\ Q^0(j) & = & 1 & 1 \le j \le m \\ P^{\exists}(w,m) & = & P(S)\sum_{k=0}^{w-m} Q^k(m) & \end{cases}$$

The time complexity of the algorithm is $O((w-m)*m)$ and is equal to the space required to build the table for $Q^{w-m}(m)$.

4.4 Variance

We need to establish a precise formula for variance $\mathbf{Var}[\Omega^{\exists}(n,w,m)]$. First observe that

$$\mathbf{Var}[\Omega^{\exists}(n,w,m)] = \sum_{i=1}^{n} \mathbf{Var}\left[X_i^{\exists}\right] + 2\sum_{1\le i<j\le n} \mathbf{Cov}[X_i^{\exists}, X_j^{\exists}] \tag{34}$$

$$\tag{35}$$

where

$$\mathbf{E}[X_i^{\exists}, X_j^{\exists}] = \begin{cases} 0 & \text{if} \quad |i-j| \ge w \\ P^{\exists}_{(X_i^{\exists}\cap X_j^{\exists})}(w,m) & \text{if} \quad |i-j| < w \end{cases}$$

This leads to

$$\begin{aligned} \mathbf{Var}[\Omega^{\exists}(n,w,m)] &= n\left[P^{\exists}(w,m) - (P^{\exists}(w,m))^2\right] \\ &+2(n-w+1)\left[\sum_{k=1}^{w-1}\left[P^{\exists}_{(X_i^{\exists}\cap X_j^{\exists})}(w,m,k) - (P^{\exists}(w,m))^2\right]\right] \\ &+2\left[\sum_{k=1}^{w-2}\left[P^{\exists}_{(X_i^{\exists}\cap X_j^{\exists})}(w,m,k) - (P^{\exists}(w,m))^2\right]\right]. \end{aligned} \tag{36}$$

To compute $P^{\exists}_{(X_i^{\exists}\cap X_j^{\exists})}(w,m,k)$ we define $\mathcal{W}^{\exists}(w,m,k)_{(X_i^{\exists}\cap X_j^{\exists})}$ as the set of all possible pairwise overlapping windows on k symbols such that $X_i^{\exists} = 1$ and $X_j^{\exists} = 1$ where $i < j$ and $|i-j| < w$.
In other words $\mathcal{W}^{\exists}(w,m,k)_{(X_i^{\exists}\cap X_j^{\exists})}$ can be enumerated as all $2w-k$ length strings consisting of two windows $\mathcal{W}^{\exists}(w,m)[r]$ and $\mathcal{W}^{\exists}(w,m)[q]$ of length w, which have a region of overlap of size k, where the overlap is between the last k symbols of $\mathcal{W}^{\exists}(w,m)[r]$ and the first k symbols of $\mathcal{W}^{\exists}(w,m)[q]$ for $1 \le q,r \le C^{\exists}(w,m)$

Let $\mathcal{W}^{\exists}(w,m,k)_{(X_i^{\exists}\cap X_j^{\exists})}[l]$ be the l-th string of $\mathcal{W}^{\exists}(w,m,k)_{(X_i^{\exists}\cap X_j^{\exists})}$.

Then we can express $P^{\exists}_{(X_i^{\exists}\cap X_j^{\exists})}(w,m,k)$ as follows

$$P^{\exists}_{(X_i^{\exists}\cap X_j^{\exists})}(w,m,k) = \sum_{l=1}^{|\mathcal{W}^{\exists}(w,m,k)|} P\left(\mathcal{W}^{\exists}(w,m,k)_{(X_i^{\exists}\cap X_j^{\exists})}[l]\right) \tag{37}$$

Now we present an exact algorithm for computing $P^{\exists}_{(X^{\exists}_i \cap X^{\exists}_j)}(w,m,k)$.

From Theorem 4 we know we can divide all elements in $\mathcal{W}^{\exists}(w,m)$ into a set of equivalence classes $\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})$. Let $\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[i]$ be the i-th element of $\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})$. Our algorithm generates all elements in $\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})$ and finds all overlaps between them.

Let $\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[i]$ and $\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[j]$ be candidates for the overlap

$$\begin{aligned}
\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[i] &= \overline{s_1}^{n_1^i} \times s_1 \times \overline{s_2}^{n_2^i} \times s_2 \ldots \overline{s_m}^{n_m^i} \times s_m \times \mathcal{A}^{n_{m+1}^i} \\
\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[j] &= \overline{s_1}^{n_1^j} \times s_1 \times \overline{s_2}^{n_2^j} \times s_2 \ldots \overline{s_m}^{n_m^j} \times s_m \times \mathcal{A}^{n_{m+1}^j}
\end{aligned}$$

where

$$\begin{aligned}
\overline{s_i} &= \mathcal{A} - s_i \\
i,j &\leq |\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})| \\
\sum_{r=1}^{m+1} n_r^i &= w-m \\
\sum_{r=1}^{m+1} n_r^j &= w-m.
\end{aligned}$$

Let $\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[i][r]$ be the r-th symbol of $\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[i]$.
ALGORITHM

1. Generate i-th partition vector $(n_1^i, n_2^i, \ldots n_{m+1}^i)$ and $\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[i]$
2. Generate j-th partition vector $(n_1^j, n_2^j, \ldots n_{m+1}^j)$ and $\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[j]$
3. The elements overlap on k symbols if

$$\forall_{0\leq r<k}(\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[i][m-r] \cap \mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[j][1+r] \quad \neq \quad \emptyset)$$

4. If they overlap then

$$\begin{aligned}
P^{\exists}_{(X^{\exists}_i \cap X^{\exists}_j)}(w,m,k) &= P^{\exists}_{(X^{\exists}_i \cap X^{\exists}_j)}(w,m,k) + P(\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[i]) \\
&\quad + \sum_{r=k}^{w} P(\mathcal{V}^{\exists}(n_1,n_2,\ldots n_{m+1})[i][r])
\end{aligned}$$

We stop here for lack of space.

5 Conclusion

We presented an exact formula for $P^{\exists}(w,m)$, the probability that an episode S of length m occurs in a window of length w in an event sequence T over alphabet $\mathcal{A}$ for the memoryless model. In addition we gave an asymptotic approximation of $P^{\exists}(w,m)$, which shows that, for appropriately large w, $P^{\exists}(w,m)$ goes to 1 as expected. By providing an efficient dynamic programming method for computing $P^{\exists}(w,m)$ we showed its applicability for real time monitoring systems. In the experiments we chose two apparently non memoryless sources (the English alphabet and the web access data) and showed that, even for these cases $P^{\exists}(w,m)$ closely approximated the estimated $P_e^{\exists}(w,m)$. This seems to be yet another one of those intriguing situation where an equation derived under a certain set of assumptions holds in practical examples for which those assumptions are clearly violated.

Based on the formula for $P^{\exists}(w,m)$ we proposed a reliable episode detection method, where as a measure of normal behavior we used $\Omega^{\exists}(n,w,m)$ the number of windows which contain at least one occurrence of a defined "bad" episode. Reliability of $\Omega^{\exists}(n,w,m)$ as a normal behavior measure stems from the fact that, for a given S and $\mathcal{A}$, we can analytically select the window length w to minimize false alarms. We proved that, $\Omega^{\exists}(n,w,m)$ has Gaussian distribution. Knowing $\mathbf{E}[\Omega^{\exists}(n,w,m)]$, $\mathbf{Var}[\Omega^{\exists}(n,w,m)]$ or their estimates, and for a given confidence level β, we showed how to set up a threshold $\tau(n,w,m)$. Contrary to $\mathbf{E}[\Omega^{\exists}(n,w,m)]$ which can be easily computed or estimated based on the training set, computing $\mathbf{Var}[\Omega^{\exists}(n,w,m)]$ exactly for large w and $\mathcal{A}$ is a difficult problem. Because the algorithm we presented for $\mathbf{Var}[\Omega^{\exists}(n,w,m)]$ is not an efficient one we used an estimator of the sample variance $\widehat{\mathbf{Var}}[\Omega^{\exists}(n,w,m)]$ in the experiments. We tested the threshold $\tau(n,w,m)$ by artificially injecting bad episodes into the testing source and observed that, $\tau(n,w,m)$ did indeed provide a sharp detection of intentional (bad) episodes.

We also derived the formula for $P^{\exists}(w,m)$ for the first order Markov model, showing that our theory works not only for memoryless but for any order Markov model as well. This makes it applicable for a broad spectrum of event sequence sources.

References

[1] A. Aho and M. Corasick, Efficient String Matching: An Aid to Biblographic Search *Programming Techniques*, 1975.

[2] A. Apostolico and M. Atallah, Compact Recognizers of Episode Sequences, *Information and Computation*, 174, 180-192, 2002.

[3] P. Billingsley, *Probability and measure*, John Wiley, New York, 1986.

[4] L. Boasson, P. Cegielski, I. Guessarian, and Y. Matiyasevich, Window-Accumulated Subsequence Matching Problem is Linear, *Proc. PODS*, 327-336, 1999.

[5] M. Crochemore and W. Rytter, *Text Algorithms*, Oxford University Press, New York, 1994.

[6] G. Das, R. Fleischer, L. G asieniec, D. Gunopulos, and J. Kärkkäinen, Episode Matching, In *Combinatorial Pattern Matching, 8th Annual Symposium*, *Lecture Notes in Computer Science* vol. 1264, 12–27, 1997.

[7] G. Kucherov, M. Rusinowitch Matching a Set of Strings with Variable Length Don't Cares, *Theoretical Computer Science* 178, 129–154, 1997.

[8] S. Kumar and E.H. Spafford, A Pattern-Matching Model for Intrusion Detection, *Proceedings of the National Computer Security Conference*, 11–21, 1994.

[9] P. Flajolet, Y. Guivarc'h, W. Szpankowski, and B. Vallée, Hidden Pattern Statistics, ICALP 2001, Crete, Greece, LNCS 2076, 152-165, 2001.

[10] H. Mannila, H. Toivonen, and A. Verkamo Discovery of frequent episodes in event sequences *Data Mining and Knowledge Discovery*, 1(3), 241-258, 1997.

[11] P. Nicodème, B. Salvy, and P. Flajolet, Motif Statistics, *European Symposium on Algorithms*, *Lecture Notes in Computer Science*, No. 1643, 194–211, 1999.

[12] P. Pevzner, *Computational Molecular Biology: An Algorithmic Approach*, MIT Press, 2000.

[13] M. Régnier and W. Szpankowski, On pattern frequency occurrences in a Markovian sequence *Algorithmica*, 22, 631-649, 1998.

[14] I. Rigoutsos, A. Floratos, L. Parida, Y. Gao and D. Platt, The Emergence of Pattern Discovery Techniques in Computational Biology, *Metabolic Engineering*, 2, 159-177, 2000.

[15] R. Sedgewick and P. Flajolet, *An Introduction to the Analysis of Algorithms*, Addison-Wesley, Reading, MA, 1995.

[16] W. Szpankowski, *Average Case Analysis of Algorithms on Sequence*, John Wiley, New York, 2001.

[17] M. Waterman, *Introduction to Computational Biology*, Chapman and Hall, London, 1995.

[18] A. Wespi, H. Debar, M. Dacier, and M. Nassehi, Fixed vs. Variable-Length Patterns For Detecting Suspicious Process Behavior, *J. Computer Security*, 8, 159-181, 2000.

[19] S. Wu and U. Manber, Fast Text Searching Allowing Errors, *Comm. ACM*, 35:10, 83–991, 1995.