Detection of Significant Sets of Episodes in Event Sequences

Mikhail Atallah *
Purdue University
Department of Computer Sciences
mja@cs.purdue.edu

Robert Gwadera †
Purdue University
Department of Computer Sciences
gwadera@cs.purdue.edu

Wojciech Szpankowski ‡
Purdue University
Department of Computer Sciences
spa@cs.purdue.edu

July 21, 2004

Abstract

We present a method for a reliable detection of "unusual" sets of episodes in the form of many pattern sequences, scanned simultaneously for an occurrence of a subsequence in a large event stream *within a window of size w*. We also investigate the important special case of all permutations of the same sequence, which models the situation where the order of events in an episode does not matter, e.g., when it is unimportant whether buying suspicious item X occurs before or after buying suspicious item Y. In order to build a reliable monitoring system we compare obtained measurements to a reference model which in our case is a probabilistic model (Bernoulli or Markov). We first present a precise analysis that leads to a construction of thresholds minimizing false positives. The difficulties of carrying out a probabilistic analysis for an arbitrary set of patterns, stems from the possible simultaneous occurrence of many members of the set as subsequences in the same window, the fact that the different patterns typically do have common symbols or common subsequences or possibly common prefixes, and that they may have different lengths. We also report on extensive experimental results, carried out on the Wal-Mart transactions database, that show a remarkable agreement with our theoretical analysis. This paper is an extension of our previous work in [8] where we laid out foundation for the problem of the reliable detection of an "unusual" episodes, but did not consider more than one episode scanned simultaneously for an occurrence.

1 Introduction

Detecting subsequence patterns in event sequences is important in many applications, including intrusion detection, monitoring for suspicious activities, and molecular biology. 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,

*Portions of this author's work were supported by Grants EIA-9903545, IIS-0219560, IIS-0312357, and IIS-0242421 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 grant R01 GM068959-01.

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

and setting thresholds for alarm is an important issue in a monitoring system that seeks to avoid false alarms.

In order to decide whether a particular sequence of events in the monitored event sequence is significant one must compare it to a reference model. In our work the reference model is a probabilistic model either generated by a memoryless (Bernoulli) source or a Markov source. The question is *when is a certain number of occurrences of a particular subsequence in a monitored even sequence unlikely to be generated by the reference model (i.e., indicative of suspicious activity or statistically significant event)?* A quantitative analysis of this question allows one to compute a *threshold* on-line while monitoring the event sequence in order to detect significant patterns (intrusions) and to avoid false alarms. By knowing the most likely number of occurrences and the probability of deviating from it, we can compute a threshold such that the probability of missing real unusual 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 observed "bad" activity as a convincing evidence against somebody if that activity is quite likely to occur under the given circumstances. Therefore it is very important to quantify such probabilities and present a universal and reliable framework for analyzing a variety of event sources.

In [11] Mannila et al. introduced the problem of finding frequent episodes in event sequences, subject to observation-window constraint, where an episode was defined as a partially ordered collection of events, that can be represented as a directed acyclic graph. The paper [11] defined the frequency $fr(\alpha, s, win)$ of an episode α as the fraction of windows of length win in which the episode occurs in an event sequence s. Given a frequency threshold min_fr, [11] considered an episode to be frequent if $fr(\alpha, s, win) \geq min\_fr$. In the framework of [11], our problem can be stated as follows. Given an episode α, what window size win and what frequency threshold min_fr should we choose to ensure that the discovered frequent episode is meaningful? Observe that for an appropriately low frequency min_fr and large window size win the episode will certainly occur in the reference model. Our problem can also be stated as follows. Given a collection of frequent episodes $\mathcal{C}(w)$, discovered using the algorithm given in [11], what is the rank of the episodes with respect to their significance?

In our paper [8] the problem of the reliable detection of unusual episodes was investigated, where we considered an episode in the form of a single sequence occurring as an ordered subsequence of a large event stream *within a window of a given fixed size.* This kind of episode is called a "serial episode" in the terminology of [11], and we henceforth adopt this terminology. In [8] we proposed a method for reliable detection of significant episodes, where as a measure of significance we used $\Omega^{\exists}(n, w, m)$ the number of windows of length w which contain at least one occurrence of serial episode S of length m as a subsequence in event sequence T after n slides of the window. We proved that appropriately normalized $\Omega^{\exists}(n, w, m)$ has the Gaussian distribution, where $\mathbf{E}[\Omega^{\exists}(n, w, m)] = nP^{\exists}(w, m)$ and $P^{\exists}(w, m)$ is the probability that a serial episode S of length m occurs at least once in a window of length w in an event sequence T over an alphabet $\mathcal{A}$. We also showed that the variance $\mathbf{Var}[\Omega^{\exists}(n, w, m)] \sim n\sigma$ for some $\sigma > 0$ and presented an exact formula for computing the variance. Given a reference model (Bernoulli or Markov), and for a given confidence level β, we presented the upper threshold for detecting significant episodes $\tau_u(w) = P^{\exists}(w, m) + \frac{b\sqrt{\mathbf{Var}[\Omega^{\exists}(n,w,m)]}}{n}$, where $P^{\exists}(w, m)$ and $\mathbf{Var}[\Omega^{\exists}(n, w, m)]$ depend on the probabilistic model and episode type. For a given probability $\beta(b) = \frac{1}{2} - \Phi(b)$ where $\Phi(b)$ is the cumulative normal probability distribution function we select b such that $P\left(\frac{\Omega^{\exists}(n,w,m)}{n} > \tau_u(w)\right) \leq \beta(b)$. That is, if one *observes* more than $\tau_u(w) \cdot n$ occurrences of windows with certain episodes, it is highly unlikely that such a number is generated by the reference source (i.e., its probability is smaller than $\beta(b)$). The quantity $\frac{\Omega^{\exists}(n,w,m)}{n}$ corresponds to the frequency $fr(S, T, w)$ [11] and is an estimator of $P^{\exists}(w, m)$ denoted $P_e^{\exists}(w, m)$.

While developing the formula for $P^{\exists}(w, m)$ we found a formula for the set of all distinct windows $\mathcal{W}^{\exists}(w, m)$ of length w containing the serial episode S of length m at least once as a subsequence. The importance of $\mathcal{W}^{\exists}(w, m)$ stems from the fact that $P^{\exists}(w, m) = \sum_{x \in \mathcal{W}^{\exists}(w,m)} P(x)$ for the Markov model of an arbitrary order including 0-th order (Bernoulli), where $P(x)$ is the probability of x as a string of symbols of length w in a given model. The advantage of the Bernoulli model versus the first order Markov or higher is that for the Bernoulli model $P^{\exists}(w, m)$ can be computed efficiently exploiting the structure of $\mathcal{W}^{\exists}(w, m)$ and the fact that the model requires only $|\mathcal{A}|$ probabilities of symbols of the alphabet $\mathcal{A}$. Therefore in [8] we focused on the Bernoulli model for which we gave an efficient dynamic programming method for computing $P^{\exists}(w, m)$. Using generating functions and complex asymptotics we presented an asymptotic approximation of $P^{\exists}(w, m)$, which is of the form $P^{\exists}(w, m) = 1 - \Theta(\rho^w)$ for large w and $0 < \rho < 1$. In 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 actual $P_e^{\exists}(w, m)$, which proved that the memoryless assumption did not limit the practical usefulness of the formula. Our paper [8] laid out foundations, but did not consider the case of detecting more than one serial episode simultaneously.

This paper builds on [8] by extending it to the case of an arbitrary number of serial episodes, monitored simultaneously for an occurrence, including the important special case of all permutations of the same serial episode, called a "parallel episode" in the terminology of [11]. The parallel episode case captures situations where the ordering of the events within the window of observation does not matter, e.g., the events correspond to basket items scanned by cashier. More formally, we analyze episodes in one of the following forms:

1. An arbitrary set of episodes $\mathbf{S} = \{S_1, S_2, \ldots, S_{|\mathbf{S}|}\}$ where every S_i of length m_i is a serial episode for $1 \leq i \leq |\mathbf{S}|$ and by an occurrence of the set $\mathbf{S}$ we mean a logical *OR* of occurrences of members of $\mathbf{S}$ within a window of size w.

2. Set of all distinct permutations of an episode $S = S[1]S[2], \ldots S[m]$ of length m (parallel episode), where by an occurrence we mean a logical *OR* of occurrences of permutations of S.

The importance of the current analysis stems from the fact that an arbitrary episode, represented by a directed acyclic graph, can be reduced to a set of serial episodes. The reason we distinguish the parallel episode case is because we will take advantage of the structure of permutations of S to design an efficient algorithm instead of representing the parallel episode as a set of serial episodes. Thus, the goal of the current paper is to quantify $\Omega^{\exists}(n, w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$ the number of windows containing at least one occurrence of $\mathbf{S} = \{S_1, S_2, \ldots, S_{|\mathbf{S}|}\}$ as a subsequence within a window of size w in a event sequence T over an alphabet $\mathcal{A}$. This analysis will lead to a formula for the threshold for detecting significant sets of episodes. We prove that $\Omega^{\exists}(n, w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$ is Gaussian with $\mathbf{E}[\Omega^{\exists}(n, w, m_1, m_2, \ldots, m_{|\mathbf{S}|})] = nP(w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$ and $\mathbf{Var}[\Omega^{\exists}(n, w, m_1, m_2, \ldots, m_{|\mathbf{S}|})] \sim n\sigma$ for some $\sigma > 0$. In order to compute $P^{\exists}(w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$ the probability that a window of length w contains an occurrence of $\mathbf{S}$ we need a formula for $\mathcal{W}^{\exists}(w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$ the set of all windows of length w containing an occurrence of $\mathbf{S}$ as a subsequence. However a considerable source of difficulty is the fact that $\mathcal{W}^{\exists}(w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$ for $|\mathbf{S}| > 1$ is not equal to the enumeration of sets of $\mathcal{W}^{\exists}(w, m_i)$ for each $S_i \in \mathbf{S}$ because in general $\mathcal{W}^{\exists}(w, m_i) \cap \mathcal{W}^{\exists}(w, m_j) \neq \emptyset$ for $i \neq j$ where $i, j \leq |\mathbf{S}|$ and therefore considering $\mathcal{W}^{\exists}(w, m_i)$ and $\mathcal{W}^{\exists}(w, m_j)$ independently would lead to a failure of the probabilistic analysis of $P^{\exists}(w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$ due to double-counting of respective probabilistic events. To appreciate the difficulty of this extension, consider the much-simplified case when there are only two pattern sequences ($\mathbf{S} = \{S_1, S_2\}$) and no symbol is common to S_1 and S_2. Even in this case the set of windows of length w containing S_1 as a subsequence and the set of windows of length w containing S_2 as a subsequence do have some elements in common, i.e., $\mathcal{W}^{\exists}(w, m_1) \cap \mathcal{W}^{\exists}(w, m_2) \neq \emptyset$ for appropriately large w. Add to this the fact that the different patterns typically do have common symbols or common subsequences or possibly common prefixes, that they may have different lengths, and the problem becomes fraught with nasty interactions that prevent any straightforward analytical solution to the case $|\mathbf{S}| > 1$.

The main contribution of the current paper is a computational formula for $P^{\exists}(w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$ for and arbitrary set of episodes including the special case of the parallel episode. We provide a recurrence system for constructing $\mathcal{W}^{\exists}(w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$. The system can be solved analytically using generating function but the final formula would include products of matrices corresponding to the interaction between symbols of members of $\mathbf{S}$ making the formula impractical algorithmically. Therefore we propose an efficient algorithmic method for enumerating $\mathcal{W}^{\exists}(w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$ using recursion graphs, which leads to a formula for $P^{\exists}(w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$ for an arbitrary order of Markov model. However, we focus on Bernoulli model in this paper because of its compactness and efficiency. Our current work builds on [8] and provides the first probabilistic analysis that quantifies $\Omega^{\exists}(n, w, m_1, m_2, \ldots, m_{|\mathbf{S}|})$ the number of windows of length w containing at least one occurrence of $\mathbf{S} = \{S_1, S_2, \ldots, S_{|\mathbf{S}|}\}$ as a subsequence. This analysis allows to compute the threshold for detecting significant sets of episodes scanned simultaneously for an occurrence. In section 2.3 we also propose a new $O(n \log(m))$ tree-based algorithm for discovering parallel episodes that requires a constant space with respect to window length w. We applied our theoretical results by running an extensive series of experiments on real data. We used a part of Wal-Mart sales data for the years 1999 and 2000. We first show that our formulas for the probability closely approximate the experimental data. Then we insert randomly some sequences, defined as "suspicious", and detect them through our threshold mechanism (cf. Fig. 11).

The paper is organized as follows. In section 2 we present our main results containing theoretical foundation. Section 3 contains experimental results demonstrating the applicability of the derived formulas.

Interested readers can visit our on-line application at `http://www-cgi.cs.purdue.edu/cgi-bin/gwadera/demo.cgi` for a demonstration of the reliable threshold computation.

2 Main Results

Given an alphabet $\mathcal{A} = \{a_1, a_2, \ldots, a_{|\mathcal{A}|}\}$ and a set of patterns $\mathbf{S} = \{S_1, S_2, \ldots S_{|\mathbf{S}|}\}$ where $S_i = S_i[1]S_i[2] \ldots S_i[m_i]$ and $S_i[j] \in \mathcal{A}$ for $1 \leq i \leq |\mathbf{S}|$, we are interested in occurrences of members of $\mathbf{S}$ as a *subsequence* within a window of size w in another sequence known as the event sequence $T = T[1]T[2] \ldots$.

We analyze the number of windows of length w containing *at least one* occurrence of $\mathbf{S}$ when sliding the window along n consecutive events in the event sequence T, where by an occurrence of $\mathbf{S}$ we mean an occurrence of one or more members of $\mathbf{S}$ (i.e, it is a logical *OR* of occurrences of $S_1, S_2, \ldots, S_{|\mathbf{S}|}$). We use $\Omega^{\exists}(n, w, m_1, m_2, \ldots, m_{|\mathbf{S}|}, \mathbf{S}, \mathcal{A})$ to denote this number, that can range from 0 to n.
Notation: Throughout the paper, whenever $\mathcal{A}$, $\mathbf{S}$ or $m_1, m_2, \ldots, m_{|\mathbf{S}|}$ are implied and we do not reference them in our notations, we simplify our notations by dropping them accordingly using $\Omega^{\exists}(n, w)$, $\mathcal{W}^{\exists}(w)$ and $P^{\exists}(w)$ instead. We also occasionally use index $m_i - k$ to mean "dropping the last k symbols of S_i", e.g., $P^{\exists}(w, m_1 - k, m_2)$ implies a pattern that is the prefix of S_1 of length $m_1 - k$ and that the second pattern is all of S_2.

2.1 Analysis of $P^{\exists}(w)$

For the sake of the presentation we focus throughout the paper on the case where either $\mathbf{S} = \{S_1, S_2\}$, or $\mathbf{S}$ is the set corresponding to a parallel episode S but our derivations will easily be seen to generalize to an arbitrary set of episodes $\mathbf{S}$.

Let $\mathcal{W}^{\exists}(w, m_1, m_2)$ be the *set* of all possible distinct windows of length w containing $\mathbf{S} = \{S_1, S_2\}$ at least once as a subsequence then the probability that they occur in a window of size w can be expressed as follows

$$P^{\exists}(w, m_1, m_2) \quad = \sum_{x \in \mathcal{W}^{\exists}(w, m_1, m_2)} P(x). \tag{1}$$

Let $\mathcal{W}^{\exists}(w, m_1, m_2)[i]$ denote the i-th lexicographically smallest element of $\mathcal{W}^{\exists}(w, m_1, m_2)$ where $1 \leq i \leq |\mathcal{W}^{\exists}(w, m_1, m_2)|$. Then the above equation can be equivalently written as:

$$P^{\exists}(w) \quad = \sum_{i=1}^{|\mathcal{W}^{\exists}(w, m_1, m_2)|} P(\mathcal{W}^{\exists}(w, m_1, m_2)[i]).$$

We now focus on deriving a formula for $\mathcal{W}^{\exists}(w, m_1, m_2)$.

In [8] we showed that the set of all distinct windows of length w, that contain a serial episode $S = S[1]S[2]\ldots S[m]$ of length m as a subsequence, can be enumerated as follows

$$\mathcal{W}^{\exists}(w,m) = \bigcup_{\sum_{k=1}^{m+1} n_k = w-m} (\mathcal{A} - S[1])^{n_1} \times \{S[1]\} \times (\mathcal{A} - S[2])^{n_2} \times \{s_2\} \times \ldots \times (\mathcal{A} - S[m])^{n_m} \times \{S[m]\} \times \mathcal{A}^{n_{m+1}}$$

and can be represented by the following graph $G(S) = (V, E)$ The graph has one start vertex (0) and

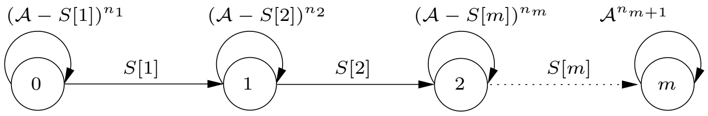

Figure 1: Graphical interpretation of the solution to $\mathcal{W}^{\exists}(w,m)$

one end vertex (m). The vertex set V of the graph, excluding vetex (0), consists of numbers $1, 2 \ldots m$ corresponding to indexes of symbols in S. The edge set E consists of edges from vertex (i) to (i+1) and self-loops from vertex (i) to itself for $0 \le i \le m$. The label for vertex from (i) to (i+1) is equal to $S[i+1]$. The label for self-loops of vertex (i) is equal to $\mathcal{A} - S[i+1]$ if $0 \le i < m$ and $\mathcal{A}$ if $i = m$. The powers n_i for $i = 1, 2, \ldots m+1$ symbolize the number of times the i-th self-loop is used on the path from vertex (0) to (m). Thus, $\mathcal{W}^{\exists}(w,m)$ is equal to all paths from (0) to (m) of length w. We also found a formula for cardinality of $\mathcal{W}^{\exists}(w,m)$, $|\mathcal{W}^{\exists}(w,m)| = \sum_{k=0}^{w-m} \binom{k+m-1}{k} (|\mathcal{A}| - 1)^k |\mathcal{A}|^{w-m-k}$.

Using the formula for $\mathcal{W}^{\exists}(w,m)$ we showed that $P^{\exists}(w,m)$ for a single serial episode S of length m has the following formula

$$P^{\exists}(w,m) = P(S) \sum_{i=0}^{w-m} \sum_{\sum_{k=1}^{m} n_k = i} \prod_{k=1}^{m} (1 - P(S[k]))^{n_k} \tag{2}$$

where P(S)= $\prod_{i=1}^{m} P(S[i])$.

We could express $\mathcal{W}^{\exists}(w,m_1,m_2)$ in terms of $\mathcal{W}^{\exists}(w,m_1)$ and $\mathcal{W}^{\exists}(w,m_2)$ using the inclusion-exclusion principle as $\mathcal{W}^{\exists}(w,m_1,m_2) = \mathcal{W}^{\exists}(w,m_1) \cup \mathcal{W}^{\exists}(w,m_2) - \mathcal{W}^{\exists}(w,m_1) \cap \mathcal{W}^{\exists}(w,m_2)$. However because $\mathcal{W}^{\exists}(w,m_1) \cap \mathcal{W}^{\exists}(w,m_2) \neq \emptyset$ for appropriate large w using the exclusion-inclusion with respect to sets $\mathcal{W}^{\exists}(w,m_1)$ and $\mathcal{W}^{\exists}(w,m_2)$ is inefficient. Below we present some example sets $\mathcal{W}^{\exists}(w,m_1)$, $W(w,m_2)$ and their intersections.

Example: Consider $\mathcal{A} = \{a, b, c, d\}$ and the following cases:

- **S** $= \{ab, cd\}$ and $w = 4$ then $\mathcal{W}^{\exists}(w,m_1) \cap \mathcal{W}^{\exists}(w,m_2) = \{abcd, acbd, acdb, cdab, cadb, cabd\}$.
- **S** $= \{ab, ac\}$ and $w = 3$ then $\mathcal{W}^{\exists}(w,m_1) \cap \mathcal{W}^{\exists}(w,m_2) = \{abc, acb\}$.
- **S** $= \{ab, a\}$ and $w = 2$ then $\mathcal{W}^{\exists}(w,m_1) \cap \mathcal{W}^{\exists}(w,m_2) = \mathcal{W}^{\exists}(w,m_1)$, i.e. this case is equivalent to **S** $= \{a\}$ because an occurrence of ab implies an occurrence of a in the window.

Clearly the example shows that contrary to $|\mathcal{W}^{\exists}(w,m)|$ the cardinality of $\mathcal{W}^{\exists}(w,m_1,m_2)$, denoted $|\mathcal{W}^{\exists}(w,m_1,m_2)|$, does depend on symbols in corresponding positions in S_1 and S_2 which determine the size of the intersection. This implies that unlike the solution to $P^{\exists}(w,m)$ we cannot find a practical analytical solution to $P^{\exists}(w,m_1,m_2)$. Therefore we adopt a computational approach to finding $P^{\exists}(w,m_1,m_2)$ with the goal of providing an efficient algorithm that uses a graph to represent the dependences (interactions) of symbols in S_1 and S_2. We start by presenting a recurrence defining $\mathcal{W}^{\exists}(w,m_1,m_2)$.

Because in the memoryless model probability of a window is equal to a product of individual probabilities of symbols, to any recursive formula for $\mathcal{W}^{\exists}(w, m_1, m_2)$ there corresponds a similar formula for $P^{\exists}(w, m_1, m_2)$ (and vice-versa). We now show that $P^{\exists}(w, m_1, m_2)$ for the set $\mathcal{A} = \{S_1, S_2\}$ satisfies the following recurrence

$$
\begin{cases}
\begin{array}{lcl}
\textbf{if} \quad S_1[m_1 - M_1 + 1] & \neq & S_2[m_2 - M_2 + 1] \quad \textbf{then} \\
P^{\exists}(W, M_1, M_2) & = & P(S_1[m_1 - M_1 + 1])P^{\exists}(W-1, M_1 - 1, M_2)+ \\
 & & P(S_2[m_2 - M_2 + 1])P^{\exists}(W-1, M_1, M_2 - 1)+ \\
 & & (1 - P(S_1[m_1 - M_1 + 1]) - P(S_2[m_2 - M_2 + 1]))P^{\exists}(W-1, M_1, M_2) \\
 & & \quad \textbf{for} \quad w > 0, M_1, M_2 > 0 \\
 & & \\
\textbf{if} \quad S_1[m_1 - M_1 + 1] & = & S_2[m_2 - M_2 + 1] \quad \textbf{then} \\
P^{\exists}(W, M_1, M_2) & = & P(S_1[m_1 - M_1 + 1])P^{\exists}(W-1, M_1 - 1, M_2 - 1)+ \\
 & & (1 - P(S_1[m_1 - M_1 + 1]))P^{\exists}(W-1, M_1, M_2) \\
 & & \quad \textbf{for} \quad W > 0, M_1, M_2 > 0 \\
 & & \\
P^{\exists}(W, 0, 0) = 1 & & \quad \textbf{for} \quad W > 0 \\
P^{\exists}(0, M_1, M_2) = 0 & & \quad \textbf{for} \quad M_1, M_2 > 0 \\
P^{\exists}(1, M_1, 0) = 1 & & \quad \textbf{for} \quad M_1 > 0 \\
P^{\exists}(1, 0, M_2) = 1 & & \quad \textbf{for} \quad M_2 > 0 \\
P^{\exists}(0, 0, 0) = 1 & &
\end{array}
\end{cases}
$$

Indeed, consider all $\binom{w}{m_1}$ and $\binom{w}{m_2}$ positions in a window in $\mathcal{W}^{\exists}(w, m_1, m_2)$ where S_1 or S_2 can occur as a subsequence. We can think of those positions as m_1-tuples and m_2-tuples respectively. Now let consider all windows in $\mathcal{W}^{\exists}(w, m_1, m_2)$ sorted according to increasing lexicographic order of those tuples then depending on whether the first symbols of S_1 and S_2 are equal or not there are two cases. If $S_1[1] \neq S_2[1]$ then there are three cases: either $S_1[1]$ is the first symbol in the window giving the term $P(S_1[1])P^{\exists}(w-1, m_1 - 1, m_2)$, or $S_2[1]$ is the first symbol in the window giving the term $P(S_2[1])P^{\exists}(w-1, m_1, m_2 - 1)$, or none of the above which leads to the term $(1 - P(S_1[1]) - P(S_2[1]))P^{\exists}(w-1, m_1, m_2)$. If $S_1[1] = S_2[1]$ then there are two cases depending on whether the first symbol of the window is equal to $S_1[1]$ or not. From the above discussion it is clear that the shape of the recursion tree is determined by interactions between symbols in S_1 and S_2, i.e., whether their symbols at pairs of positions are equal or not. Therefore in order to find a solution to $P^{\exists}(w, m_1, m_2)$ we have to enumerate all pairs of indices (M_1, M_2) such that $P^{\exists}(W, M_1, M_2)$ appears in the recursion tree (not all such pairs of indices qualify). This recursion tree in a form of a graph is now described more formally (as stated earlier, in addition to depicting the recurrence, the graph also describes all elements of $\mathcal{W}^{\exists}(w, m_1, m_2)$. Below we show a recursive description of the graph.

Let $G(\mathbf{S}) = (V, E)$ be an edge-labeled directed graph defined as follows. The vertex set V is a subset of all the pairs (i, j), $0 \leq i \leq m_1$, $0 \leq j \leq m_2$. That subset, as well as E, are defined inductively as follows.

- $(0, 0)$ is in V.
- If (i, j) is in V, $i < m_1$, and $S_1[i+1] \neq S_2[j+1]$ then $(i+1, j)$ is also in V, and an edge from from (i, j) to $(i+1, j)$ labeled $S_1[i+1]$ exists in E.
- If (i, j) is in V, $j < m_2$, and $S_1[i+1] \neq S_2[j+1]$ then $(i, j+1)$ is also in V, and an edge from (i, j) to $(i, j+1)$ labeled $S_2[j+1]$ exists in E.
- If (i, j) is in V, $i < m_1$ and $j < m_2$, and $S_1[i+1] = S_2[j+1]$ then $(i+1, j+1)$ is also in V, and an edge from (i, j) to $(i+1, j+1)$ labeled $S_1[i+1]$ $(= S_2[j+1])$ exists in E.
- A self-loop from vertex (i, j) to itself exists and has label equal to (i) $\mathcal{A}$ if $i = m_1$ or $j = m_2$, (ii) $\mathcal{A} - \{S_1[i+1]\} - \{S_2[j+1]\}$ if $i < m_1$ and $j < m_2$.

The following observations, in which we do not count self-loops towards the in-degree and out-degree of a vertex, follow from the above definition of $G(\mathbf{S})$.

- The in-degree of vertex $(0, 0)$ (start vertex) equals zero, the out-degree of vertices $(m_1, j), (i, m_2)$ for $i \le m_1, j \le m_2$ (end-vertices) equals zero.
- The in-degree and out-degree of every vertex (i, j) is at most three; if $\mathbf{S}$ consisted of $|\mathbf{S}| > 2$ serial episodes then the in-degree and out-degree of any vertex would be at most $|\mathbf{S}| + 1$.
- $|V| = O(m_1 m_2)$ and $|E| = O(|\mathbf{S}| m_1 m_2)$.

Let *Edges*(*path*) and *Vertices*(*path*) denote the sequence of consecutive edges and (respectively) vertices in any *path*, except that *Vertices*(*path*) does not include the last vertex on *path* (why this is so will become apparent below). In what follows, if vertex (i, j) is in *Vertices*(*path*), then we use $n_{i,j}(path)$ to denote the number of times the self-loop at (i, j) is used; if *path* is understood and there is no ambiguity, then we simply use $n_{i,j}$ rather than $n_{i,j}(path)$. Let $\mathcal{R}$ be the set of all distinct simple paths (i.e., without self-loops) from the start-vertex to any end-vertex. Now let $\mathcal{L}_w$ be the set of all distinct paths of length w, *including self-loops*, from the start-vertex to all end-vertices (that is, self-loops do count towards path length). Then the set of all windows containing $\mathbf{S} = \{S_1, S_2\}$ as a subsequence can be enumerated as follows

$$\mathcal{W}^{\exists}(w, m_1, m_2) \quad = \quad \{Edges(path) : path \in \mathcal{L}_w\}. \tag{3}$$

Examples of $G(\mathbf{S})$ are shown in Figure 2 and in Figure 3. To see why using the graph $G(\mathbf{S})$ we can

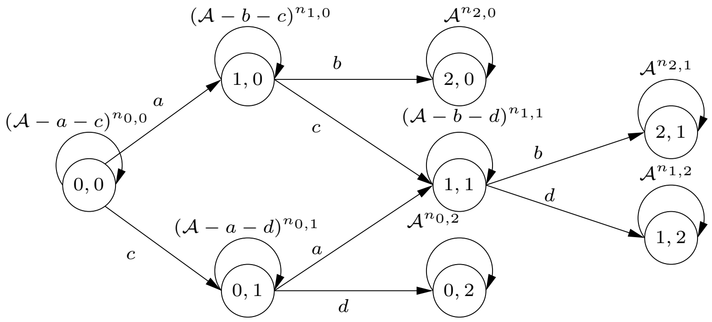

Figure 2: $G(\mathbf{S})$ for $\mathbf{S} = \{ab, cd\}$

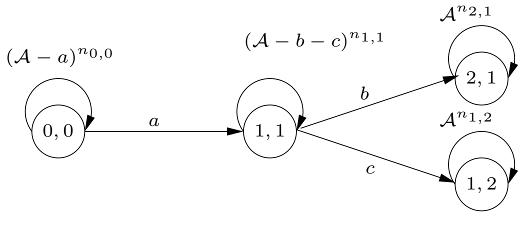

Figure 3: $G(\mathbf{S})$ for $\mathbf{S} = \{ab, ac\}$

properly enumerate $\mathcal{W}^{\exists}(w, m_1, m_2)$ we present the following example.

End vertex	$\mathcal{W}^{\exists}(3,2,2)$	$n_{0,0}$	$n_{1,1}$	$n_{2,1}$	$n_{1,2}$
(2, 1)	$ab \times A$	0	0	1	0
(2, 1)	aab	0	1	0	0
(2, 1)	adb	0	1	0	0
(2, 1)	bab	1	0	0	0
(2, 1)	cab	1	0	0	0
(2, 1)	dab	1	0	0	0
(1, 2)	$ac \times A$	0	0	1	0
(1, 2)	aac	0	1	0	0
(1, 2)	adc	0	1	0	0
(1, 2)	bac	1	0	0	0
(1, 2)	cac	1	0	0	0
(1, 2)	dac	1	0	0	0

Table 1: Enumeration of $\mathcal{W}^{\exists}(3,2,2)$ for $\mathcal{A} = \{a,b,c,d\}$ and $\mathbf{S} = \{ab, ac\}$ using $G(\mathbf{S})$ from Figure 3

Example: Consider $\mathcal{A} = \{a,b,c,d\}$, $\mathbf{S} = \{ab, ac\}$ and $w = 3$. We use $G(\mathbf{S})$ from Figure 3 and using formula (3) we enumerate elements of $\mathcal{W}^{\exists}(3,2,2)$ which are shown in Table 1. Clearly using the presented method we generate all members of $\mathcal{W}^{\exists}(3,2,2)$ without duplicates.

If we compare Figure 1 with Figures 2 and 3 it is clear that $G(\mathbf{S})$ for a set containing only one episode S reduces to the graph representing $\mathcal{W}^{\exists}(w,m)$ for a serial episode S. Formulas 1,3 and 2 lead to the following theorem.

Theorem 1 *Consider a memoryless source with $P(S_i[k])$ being the probability of generating the k-th symbol of $S_i \in \mathbf{S} = \{S_1, S_2\}$. Let also*

$$P(Edges(path)) = \prod_{edge \in Edges(path)} P(label(edge)).$$

Then for m_1, m_2 and $w \geq m_i$ we have

$$\begin{aligned} P^{\exists}(w, m_1, m_2) &= \sum_{path \in \mathcal{R}} P(Edges(path)) \sum_{g=0}^{w-|Edges(path)|} \\ &\quad \sum_{\sum n_{i,j}(path)=g} \prod_{(i,j)\in Vertices(path)} [1 - P(\{S_1[i+1]\} \cup \{S_2[j+1]\})]^{n_{i,j}(path)}. \end{aligned}$$

In our experiments, we implemented an efficient dynamic programming algorithm based on Theorem 1. In section 3.1.2 we present evidence how Theorem 1 works well on real data by comparing it to the estimate $P_e^{\exists}(w) = \frac{\Omega^{\exists}(n,w)}{n}$ given the actual $\Omega^{\exists}(n,w)$. We also solved $P^{\exists}(w)$ for an important special case when $\mathbf{S}$ consist of all permutations of one pattern S, which is the case of a parallel episode. Using Theorem 1 directly for a parallel episode S as set of all permutation of S would be inefficient because we would then need to consider a graph having a disastrous $|V| = O(m^{m!})$.

In order to simplify the graph $G(\mathbf{S})$ that would result from all permutations, and bring its number of vertices down to a manageable size (quantified below), we exploit the structure of a set of all permutations to design a different graph. Notice that, for a parallel episode, every path in $\mathcal{R}$ is a permutation of symbols in S. In addition, the out-degree of a vertex is at most m if all symbols of S are different. Furthermore a transition from from $P^{\exists}(k,i,j,\ldots)$ to $P^{\exists}(k+1,i',j',\ldots)$ takes place for any symbol of S not seen so far since the order of symbols does not matter. This observations allow us to introduce a variant of $G(\mathbf{S})$ called $G_{\|}(S)$.

Let $G_{\|}(S) = (V, E)$ be a directed edge-labeled graph defined as follows. The vertex set V consist of submultisets of size $i = 0, 1, \ldots, m$ of the multiset of sorted symbols in S denoted as $\{S[1], S[2], \ldots, S[m]\}$. We represent the submultisets equivalently as binary vectors of the form $(i_1, i_2, \ldots, i_m)$, where $i_j = 1$ if the vertex contains symbol $S[i_j]$ in its submultiset or $i_j = 0$ otherwise. V and E can be defined inductively as follows.

- $(0, 0, \ldots, 0)$ and $(1, 1, \ldots, 1)$ are in V.
- If $(i_1, i_2, \ldots, i_m)$ is V and $\sum_{j=1}^{m} i_j < m$ then $(i'_1, i'_2, \ldots, i'_m)$ is in V if $\sum_{j=1}^{m} i'_j = \sum_{j=1}^{m} i_j + 1$ and an edge with label equal to $\{i'_1 \cdot S[1], i'_2 \cdot S[2], \ldots, i'_m \cdot S[m]\} -$ $\{i_1 \cdot S[1], i_2 \cdot S[2], \ldots, i_m \cdot S[m]\}$ exists in E.
- A self-loop from vertex $(i_1, i_2, \ldots, i_m)$ to itself exists and has label equal to (i) $\mathcal{A}$ for $(1, 1, \ldots, 1)$, (ii) $\mathcal{A} - \bigcup_{i_j=0}\{S[i_j]\}$ otherwise.

The following observations, in which we do not count self-loops towards the in-degree and out-degree of a vertex, follow from the above definition of $G_{\|}(S)$.

- The in-degree of the start-vertex $(0, 0, \ldots, 0)$ equals zero, the out-degree of the end-vertex $(1, 1, \ldots, 1)$ equals zero. The in-degree and out-degree of every vertex is at most m.
- $|V| = O(2^m)$ and $|E| = O(|V|m)$.

Examples of $G_{\|}(S)$ are shown in Figure 4 and in Figure 5. The next theorem is an adoption of Theorem 1 to the case of a parallel episode.

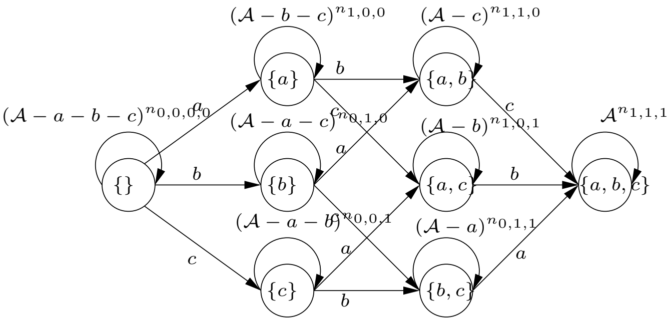

Figure 4: $G_{\|}(S)$ for $S = abc$ and $\mathcal{A} = \{a, b, c, d\}$

Theorem 2 *Consider a memoryless source with $P(S[k])$ being the probability of generating the k-th symbol of S where S is a parallel episode with*

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

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

$$P^{\exists}(w) \quad = \quad \sum_{path \in \mathcal{R}} P(S) \sum_{g=0}^{w-m} \sum_{\sum n_{i_1, i_2, \ldots, i_m}(path) = g} \prod_{(i_1, i_2, \ldots, i_m) \in Vertices(path)} \left(1 - P\left(\bigcup_{i_j=0}\{S[i_j]\}\right)\right)^{n_{i_1, i_2, \ldots, i_m}(path)} .$$

In our experiments we implemented an efficient dynamic programming algorithm based on Theorem 2 for computing $P^{\exists}(w)$. In section 3.1.1 we presented how Theorem 2 works well on real data where the formula for $P^{\exists}(w)$ agrees with $P_e^{\exists}(w)$ on the Wal-Mart transactions.

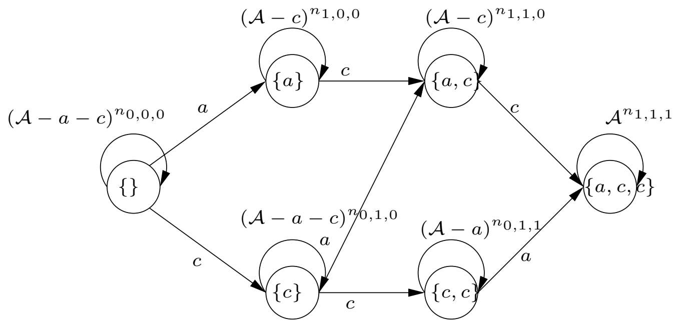

Figure 5: $G_{\|}(S)$ for $S = acc$ and $\mathcal{A} = \{a, b, c, d\}$

2.2 Computation of threshold $\tau_u(w)$

Now we derive variance and normal limiting distribution of $\Omega^{\exists}(n, w)$. Observe that

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

where

$$I_i^{\exists} = \begin{cases} 1 & \text{if any member of } \mathbf{S} \text{ occurs at least once as a} \\ & \text{subsequence in the window ending at position } i \text{ in } T; \\ 0 & \text{otherwise,} \end{cases}$$

where i is the relative position with respect to the first position ($i = 1$). Thus, we easily have

$$\begin{aligned} \mathbf{E}[I_i^{\exists}] &= P^{\exists}(w), \\ \mathbf{Var}[I_i^{\exists}] &= P^{\exists}(w) - (P^{\exists}(w))^2. \end{aligned}$$

Clearly we have $\mathbf{E}[\Omega^{\exists}(n, w)] = nP^{\exists}(w)$. In order to compute variance of $\Omega^{\exists}(n, w)$ we need $P^{\exists}_{(I_i^{\exists} \cap I_j^{\exists})}(w, k)$, defined as the probability that two overlapping windows at respective position i and j for $|i - j| < w$ have $I_i = 1$ and $I_j = 1$. The variance can be expressed as follows

$$\begin{aligned} \mathbf{Var}[\Omega^{\exists}(n, w)] &= n\mathbf{Var}[I_1^{\exists}] + 2 \sum_{1 \le i < j \le n}^{n} \mathbf{Cov}[I_i^{\exists}, I_j^{\exists}] \\ &= n(P^{\exists}(w) - (P^{\exists}(w))^2) + 2(n - w + 1) \sum_{k=1}^{w-1} \left[P^{\exists}_{(I_i^{\exists} \cap I_j^{\exists})}(w, k) - (P^{\exists}(w))^2 \right] \\ &\quad + 2 \sum_{q=2}^{w-1} \sum_{k=q}^{w-1} \left[P^{\exists}_{(I_i^{\exists} \cap I_j^{\exists})}(w, k) - (P^{\exists}(w))^2 \right]. \end{aligned}$$

The two terms involving $P^{\exists}_{(I_i^{\exists} \cap I_j^{\exists})}(w, k)$ in the above formula represent correlation between windows (with $2(w-1)$ neighborhood), where $k = w - |i - j|$ represents the length of the overlap between windows at position i and j. $P^{\exists}_{(I_i^{\exists} \cap I_j^{\exists})}(w, k)$ can be computed using the formula for $\mathcal{W}^{\exists}(w, m_1, m_2, \ldots m_{|\mathbf{S}|})$ by enumerating all possible overlaps on k symbols of two windows.

Because clearly $\mathbf{E}[I_i^{\exists}, I_j^{\exists}] \le \mathbf{E}[I_i^{\exists}]$ for $1 \le i < j \le n$ implies $P^{\exists}_{(I_i^{\exists} \cap I_j^{\exists})}(w, k) \le P^{\exists}(w)$ we can use $P^{\exists}(w)$ as an upper bound for $P^{\exists}_{(I_i^{\exists} \cap I_j^{\exists})}(w, k)$ instead of the enumerative algorithm. This leads to

$$\mathbf{Var}[\Omega^{\exists}(n, w)] \le [n + (2n - w)(w - 1)] \left[P^{\exists}(w) - (P^{\exists}(w))^2 \right].$$

We used the above formula for computing the threshold $\tau_u(w)$ in experiments. One concludes, however, that $\mathbf{Var}[\Omega^{\exists}(n,w)]$ is $O(n)$. In view of the above, and using the fact that $\Omega^{\exists}(n,w)$ is the so called $w-1$-dependent sequence (i.e., $\Omega^{\exists}(n,w)$ depends on the last $w-1$ windows), we may apply Theorem 27.5 of [3] to establish the central limit theorem for $\Omega^{\exists}(n,w)$.

Theorem 3 *The random variable $\Omega^{\exists}(n,w)$ obeys the Central Limit Theorem in the sense that its distribution is asymptotically normal. More precisely, for $a,b=O(1)$ we have*

$$\lim_{n\to\infty} P\left\{a \le \frac{\Omega^{\exists}(n,w)-\mathbf{E}[\Omega^{\exists}(n,w)]}{\sqrt{\mathbf{Var}[\Omega^{\exists}(n,w)]}} \le b\right\} = \frac{1}{\sqrt{2\pi}}\int_a^b e^{\frac{-t^2}{2}}dt$$

Observe that when b is large (say order of 10) the above probability is small enough to be qualified as a *moderate large deviations*. This captures the nature of unusual episodes, as needed. Using the above findings and the definition of $\mathbf{E}[\Omega^{\exists}(n,w)]$ we establishing the reliable threshold $\tau_u(w)$ as follows.

Given a probabilistic model of the reference source (in this paper we use Bernoulli model with probabilities $P(a_i)$ for $a_i \in \mathcal{A}, i=1,2,\ldots,|\mathcal{A}|$), a frequent episode $\mathbf{S}$, $\Omega^{\exists}(n,w)$ and probability $\beta(b)$ (e.g., $\beta(b)=10^{-5}$) we compute the upper threshold $\tau_u(w)$ subject to $P\left(\frac{\Omega^{\exists}(n,w)}{n} > \tau_u(w)\right) \le \beta(b)$, using the equation system as follows

$$\begin{cases} \tau_u(w) & = & P^{\exists}(w) + \frac{b\sqrt{\mathbf{Var}[\Omega^{\exists}(n,w)]}}{n} \\ \beta(b) & = & \frac{1}{\sqrt{2\pi}}\int_b^\infty e^{\frac{-t^2}{2}}dt \\ & = & \frac{1}{2} - \Phi(b) \end{cases}$$

where $P^{\exists}(w)$ and $\mathbf{Var}[\Omega^{\exists}(n,w)]$ depend on the episode type and the probabilistic model (Bernoulli or Markov). Once the threshold $\tau_u(w)$ is computed if $\frac{\Omega^{\exists}(n,w)}{n} > \tau_u(w)$ then the probability that the episode $\mathbf{S}$ is not significant is less than $\beta(b)$, i.e., the probability of an false alarm is less than $\beta(b)$. Given a collection of frequent episodes $\mathcal{C}(w)$ we can rank their significance using $\beta(b)$. where $\Phi(b)$ and its inverse can be computed using numerical algorithms.

2.3 Data structure for discovering parallel episodes

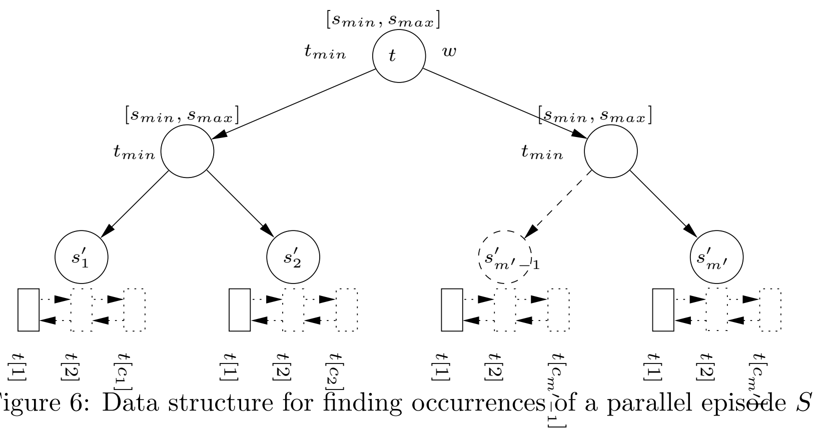

Figure 6: Data structure for finding occurrences of a parallel episode S

The main advantage of the algorithm presented in this section with comparison to the algorithm in [11] is that the space required by the data structure is independent of the window length w. Let $S=S[1]S[2]\ldots S[m]$ be the input episode for the algorithm. Let $S' = \{s'_1, s'_2, \ldots s'_{m'}\}$ be a set of cardinality m' obtained from S by eliminating duplicates and then sorting it. Let c_i for $i=0,1\ldots m'$ be the number of times an alphabet symbol s'_i occurs in S. We build a binary tree over symbols in S' as leaves. Figure 6 shows the tree. Each node in the tree contains the usual tree pointers: *parent*, *lchild*, *rchild* and a search key interval $[s_{min}, s_{max}]$ where s_{min} is the smallest key in the subtree and s_{max} is the largest key in the subtree. In addition, depending on its type a node keeps the following specific information:

- root: t an event counter, counts the number of elapsed (scanned) events, where by "time" we mean the position in the event sequence
- internal node:
  - t_{min} the minimum time of the arrival for the subtree rooted at this node
- leaf node:
  - search key value s_i
  - *dlist*: doubly linked list containing one element for each symbol a_i in the pattern S. The purpose of the list is to keep track of the most recent occurrences of the symbol a_i sorted by arrival times. So the size of the list is one unless S contains more than one symbol a_i. Let $t_{a_i}[1], t_{a_i}[2] \ldots t_{a_i}[c_i]$ for $i = 0, 1 \ldots m'$ be the times of the occurrence of symbol a_i from the left to the right in the list then they must satisfy the condition $t_{a_i}[j] < t_{a_i}[j+1]$ for $j = 1, \ldots c_i$. So the leftmost element of the list contains the oldest occurrence of a_i and the rightmost element contains the most recent occurrence of a_i.

The tree supports the following operations:

- **update(s)** : when a new symbol s arrives the time t is incremented by one and the search tree structure is used to find the proper leaf. If the search finds leaf s_i then the leftmost element of the doubly linked list with time $t_{s_i}[1]$ is removed and a new element with the current time t is attached to the right end of the list. Once the new value is attached to the list the value of the leftmost element, as the oldest one, is propagated up the tree as long as it is smaller then t_{min} of the internal node on the path to the root. This operation takes $\log(m)$ time (the height of the tree).
- **exists** : if $t - t_{min} + 1 \leq w$ at the root node then at least one permutation of S occurs as a subsequence within the window. This operation takes $O(1)$.

The time complexity for finding $\Omega^{\exists}(n, w)$ is $O(n \log m)$, because we perform n slides of the window each requiring $O(\log m)$. The space is determined by the number of nodes in the tree, which is $O(m \log(m))$. The presented tree structure can also handle a problem when instead of a window of size w we associate a time to live *ttl* with each symbol of the pattern S. In such a case each leaf stores the expiration time $expt = t + ttl$ and the each internal node stores the minimum expiration time $expt_{min}$ in its subtree. The $update(s)$ operation remains the same and *exists* returns *true* if $expt_{min} > t$ at the root node. The presented data structure can be extended to handle multiple parallel episodes (details are omitted).

3 Experiments

The purpose of our experiments was to test applicability of the analytical results for real sources. Therefore we selected Wal-Mart data available on the departmental Oracle server in the Department of Computer Sciences, Purdue University. The database contains part of Wal-Mart sales data for the years 1999 and 2000 in 135 stores. We selected one of the stores, one category of items of cardinality 35 ($|\mathcal{A}| = 35$) and extracted 9.66 million records from table *Item_Scan*, sorted by scan time. We divided our sources into training sets and testing sets. Training sets are data sets, which we consider to constitute the reference source. We used the first 9.56 million records as a training set to compute $P(a_i)$ for $a_i \in \mathcal{A}, i = 1, 2, \ldots, |\mathcal{A}|$

for the Bernoulli reference source. Once the probability model of the reference source has been built, we can start monitoring unknown data called testing data. In section 3.1 we tested how well the formulas for $P^{\exists}(w)$ worked on the Wal-Mart data by comparing $P_e^{\exists}(w) = \frac{\Omega^{\exists}(n,w)}{n}$ to the computed $P^{\exists}(w)$ for different values of w. We used the following error metric $d = \left[\frac{1}{r}\sum_{i=1}^{r} \frac{|P_e^{\exists}(w_i) - P^{\exists}(w_i)|}{P_e^{\exists}(w_i)}\right] 100\%$ where $w_1 < w_2 < \ldots w_r$ are the tested window sizes. In section 3.2 we tested the detection properties of the threshold $\tau_u(w)$ as a function of the window length w. All our algorithms have been implemented in C++ and run under Linux operating system.

3.1 Estimation of $P^{\exists}(w)$

In all experiments in this section we used the same testing source of length $n = 10^5$ events.

3.1.1 The case of a parallel episode

We set $S = \{it_0, it_4, it_5, it_6, it_9, it_{10}, it_{17}\}$ and then ran the tree based detection algorithm [**?**] for finding $\Omega(10^5, w)$ for $w \in [10, 180]$ and compared $P_e(w)$ to the analytically computed $P^{\exists}(w)$ using our algorithm based on Theorem 2. The results are shown in Figure 7, which indicate an exceptionally close fit between $P^{\exists}(w)$ and $P_e^{\exists}(w)$ with the difference d of order 2%. The results confirm our expectations that the Bernoulli model and parallel episode models well sources as the Wal-Mart item scans where the source seems to generate events independently.

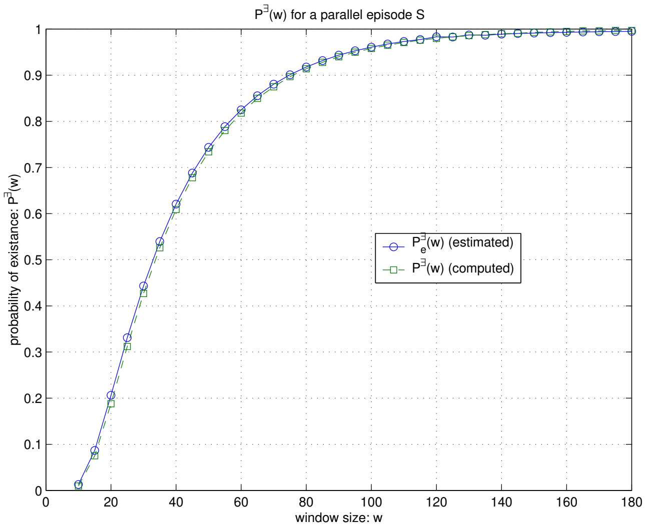

Figure 7: $P_e^{\exists}(w) = \frac{\Omega^{\exists}(n,w)}{n}$ and $P^{\exists}(w)$ for a parallel episode S, using Wal-Mart data

3.1.2 The case of a set of two serial episodes

We set $S_1 = \{it_0, it_4, it_5, it_6, it_9, it_{10}, it_{17}\}$ and $S_2 = \{it_0, it_6, it_5, it_4, it_{10}, it_9, it_{17}\}$ where S_2 is a permuted version of S_1. This case reflects a situation when a pattern of interest is only partially restricted and the serial case is too restrictive but the parallel case too relaxed. We ran an algorithm for finding $\Omega(10^5, w)$ for **S** for $w \in [10, 180]$. Then compared $P_e(w)$ to the analytically computed $P^{\exists}(w)$ using our algorithm based on Theorem 1. The results are shown in Figure 8, which indicate a very close fit between $P^{\exists}(w)$ and $P_e^{\exists}(w)$ with d of order 13% but not so close as in the parallel case ($d = 2\%$). The reason may be the fact that this set of episodes is too restricted comparing to the parallel episode given the unordered nature of the item scans.

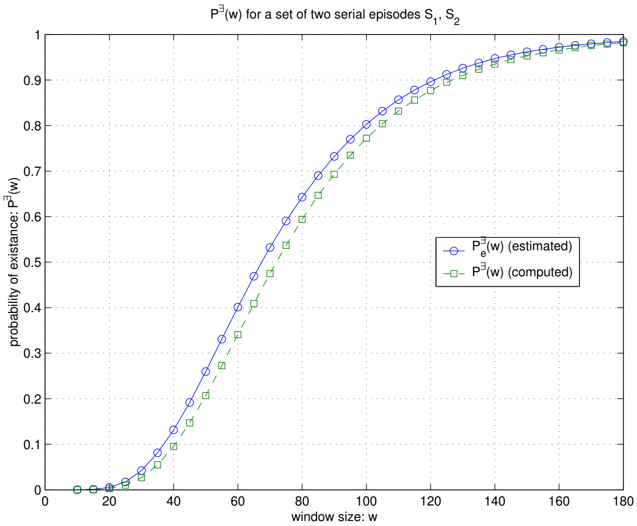

Figure 8: $P_e^{\exists}(w) = \frac{\Omega^{\exists}(n,w)}{n}$ and $P^{\exists}(w)$ for a set $\mathbf{S} = \{S_1, S_2\}$ of serial episodes, using Wal-Mart data

3.1.3 Comparison of the three cases: parallel, two serial and one serial

In this experiment we investigate the relationship between the formulas for $P(w)$ and the experimental $P_e(w)$ for the episode S in the three cases: parallel, partially ordered (S_2 is a permutation of S_1) and serial (investigated in [8]). For the first two cases we use the results obtained in the previous experiments. For the third case we ran an algorithm for discovering a serial episode in the event sequence for $w \in [10, 180]$ as in the previous experiment to create $\Omega(10^5, w)$ at the same points. The results for $P^{\exists}(w)$ are shown in Figures 9 and the corresponding estimates are shown in Figure 10. The figures clearly indicate that the serial and parallel cases of an episode S establish the lower and upper bound on the probability of existence of S in window of size w.

3.2 Threshold $\tau_u(w)$

This experiment demonstrates the application of the upper threshold $\tau_u(w)$ for a a parallel episode S. It also shows the relationship between $\tau_u(w)$ and w in detecting an unusual behavior. We set $S = \{it_8, it_{12}, it_{14}, it_{15}, it_{19}, it_{20}, it_{26}\}$ and consider the parallel case of S. We selected a part of the scans of length $n = 50000$ as the test source. We compute the threshold $\tau_u(w) = P^{\exists}(w) + b\frac{\sqrt{\mathbf{Var}[\Omega^{\exists}(n,w)]}}{n}$ subject to $P\left(\frac{\Omega^{\exists}(n,w)}{n} > \tau_u(w)\right) \leq \beta(b)$ for $\beta(b) = 10^{-6}$ for which we obtained $b = 4.26$ using an algorithm for computing the inverse of $\Phi(b)$. We repeated the threshold computation for three values of $w = 40, 30, 25$ for the same episode S. We simulated an attack by inserting the episode S as a string into the testing source until we exceeded the threshold. We normalized the number of insertion by n. Figure 11 presents results. We conclude that the if w increases then the number of attacks causing $\frac{\Omega^{\exists}(n,w)}{n}$ to rise above the $\tau_u(w)$ increases exponentially which is caused by the exponential growth of $P^{\exists}(w)$ in the formula for $\tau_u(w)$.

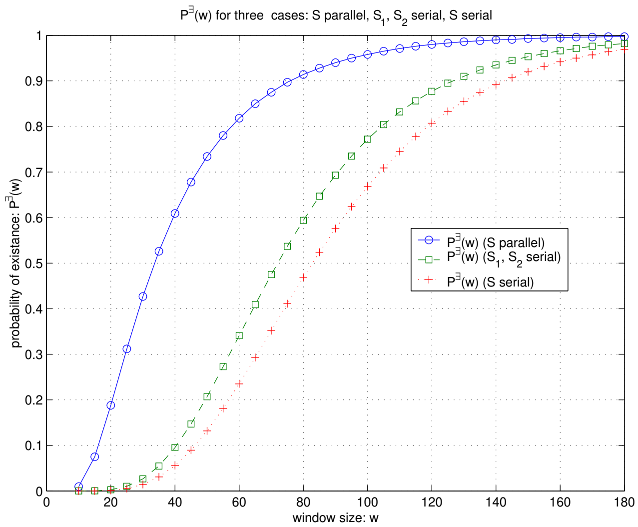

Figure 9: $P^{\exists}(w)$ for three cases: S parallel, $\{S_1, S_2\}$ serial and S serial, using Wal-Mart data

4 Conclusions

We presented the exact formulas for the probability of existence $P^{\exists}(w)$, for an arbitrary set of serial episodes $\mathbf{S}$ including the case of a parallel episode S. Using the formulas we showed how to compute the upper threshold $\tau_u(w)$ to measure significance of $\mathbf{S}$. Since we adopted a computational probability approach it is valid for the Markov model of any order. The choice of the 0-the order (Bernoulli) was dictated only by its compactness and suitability to implementation through efficient dynamic programming algorithms. However in experiments on Wal-Mart transactions we showed that formulas for the Bernoulli model very closely approximated the real life data. Realization of Markov model of order higher then 0 would require computation of $P^{\exists}(w)$ using conditional probabilities of the respective order. We will address this problem in the forthcoming paper.

References

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

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

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

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

[5] S. Brin, R. Motwani, C. Silverstein, *Beyond Market Baskets: Generalizing Association Rules to Correlations*, SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona

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

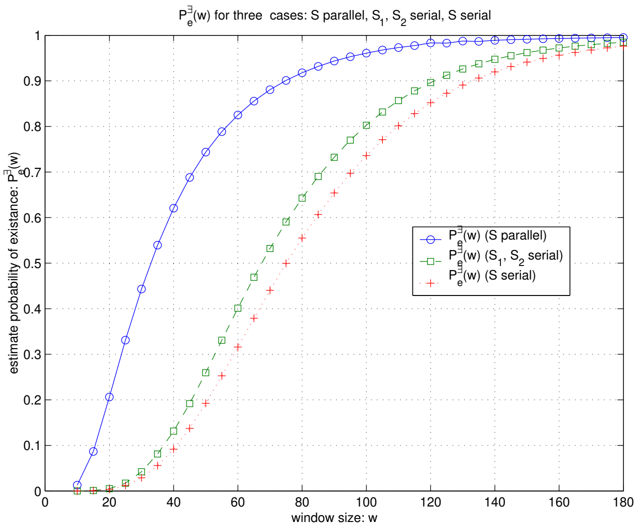

Figure 10: $P_e^{\exists}(w) = \frac{\Omega^{\exists}(n,w)}{n}$ for three cases: S parallel, $\{S_1, S_2\}$ serial and S serial, using Wal-Mart data

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

[8] R.Gwadera, M.Atallah, and W.Szpankowski, Reliable detection of episodes in event sequences, In *Third IEEE International Conference on Data Mining*, pages 67-74, Melbourne Florida.

[9] J. Han, J. Pei, Y. Yin, R. Mao, Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach, *Data Mining and Knowledge Discovery*, 8, 53-87, 2004

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

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

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

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

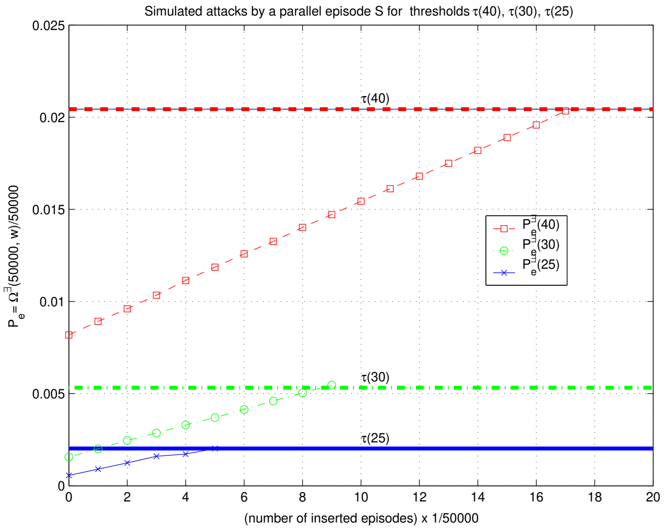

Figure 11: The upper threshold $\tau_u(w)$ as a function of w for a parallel episode S, using the Wal-Mart database