

STAMP: On Discovery of Statistically Important Pattern Repeats in Long Sequential Data

Jiong Yang
UIUC
jioyang@cs.uiuc.edu

Wei Wang
UNC Chapel Hill
weiwang@cs.unc.edu

Philip S. Yu
IBM T. J. Watson Research Center
psyu@us.ibm.com

Abstract

In this paper, we focus on mining periodic patterns allowing some degree of imperfection in the form of random replacement from a perfect periodic pattern. In InfoMiner+, we proposed a new metric, namely *generalized information gain*, to identify patterns with events of vastly different occurrence frequencies and to adjust for the deviation from a pattern. In particular, a penalty is allowed to be associated with gaps between pattern occurrences. This is particularly useful in locating repeats in DNA sequences. In this paper, we present an effective mining algorithm, STAMP, to simultaneously mine significant patterns and the associated subsequences under the model of generalized information gain.

1 Introduction

Periodic pattern discovery is an important problem in mining time series data and has wide applications. A periodic pattern is a list of ordered events, which repeats itself in the event sequence. It is useful in characterizing the cyclic behavior of the time series. In practice, not every portion in the time series may contribute to the periodicity. For example, a company's stock may often gain a couple of points at the beginning of each trading session but it may not have much regularity at a later time. This kind of looser periodicity is often referred to as *partial periodicity* [10]. Moreover, due to some random noise, a pattern may not always be repeated perfectly. In turn, the event sequence can be viewed as a series of perfect pattern repetitions with a few *random replacements*¹. If the amount of "replacement" is below some reasonable threshold, we may regard that the pattern still exhibits in the event sequence.

As a newly developed research area, most previous work on mining time series data addresses the issue by creating a mapping to the association rule mining technique [9, 10] and therefore uses the support and confidence as the metrics to identify the significant patterns from the rest. Most association rule mining algorithms favor frequently occurred event(s) due to the nature of the problem. However, patterns involving infrequent events may also be as significant as (or even more significant than) frequent events in an event sequence. This issue becomes more critical when different events occur at divergent frequencies.

Information gain is introduced in [27] to measure the importance/significance of the occurrence of a pattern. The information gain of an occurrence of a rare event is high while the information gain of the occurrence of a frequent event is low. Thus, we are able to find the statistically significant patterns with the information gain threshold. However, the ma-

ior limitation of this model is that it does not take into account where the occurrences are in the sequence. Let's take a look at two sequences: $S_1 = a_1, a_2, a_3, a_3, a_2, a_3, a_1, a_2, a_4, a_5, a_1, a_2$ and $S_2 = a_1, a_2, a_1, a_2, a_1, a_2, a_3, a_3, a_2, a_3, a_4, a_5$. The elements in the two sequences are identical. The only difference is the order of the events. The pattern (a_1, a_2) repeats perfectly in the first half of S_2 while it scatters in S_1 . The two patterns have the same information gain in the two sequences. In some applications (e.g., repeats discovery in bio-informatics domain), a series of consecutive repeats are considered more significant than the scattered ones. That is, there should be some "penalty" associated with the gap between pattern repeats. As a result, the measure of generalized information gain (GIG) was introduced [28] to capture the significance of a pattern in a sequence/subsequence. The occurrence of a pattern will be given a positive GIG while a mis-occurrence (or a gap) will generate a negative GIG. The overall generalized information gain will be the aggregate GIG of all occurrences and mis-occurrences of the pattern in a sequence/subsequence.

Since the characteristics of a sequence may change over time, many patterns may only be valid for a period of time. The degree of significance (i.e., generalized information gain) of a pattern may be diluted if we only consider the entire event sequence. In addition, a user may be interested not only in a significant pattern, but also where/when the pattern is significant as well. The identification of significant pattern in a subsequence is of great importance in many applications. In our proposed scheme, a user can specify the minimum GIG that a significant pattern must carry over a subsequence of data. Upon satisfying this requirement, the subsequence(s) that maximizes the GIG of a pattern will be identified. In the previous example, the pattern (a_1, a_2) is very significant in the first half of S_2 , but may not be significant over the entire sequence.

Although the generalized information gain is a more meaningful metric for the problems addressed previously, it does not preserve the *downward closure* property (as the *support* does). For example, the pattern $(a_1, a_2, *)$ may have sufficient GIG while both $(a_1, *, *)$ and $(*, a_2, *)$ do not². We can not take advantages of the standard pruning technique developed for mining association rules. The observation that the *triangle inequality*³ is still preserved by the generalized information gain motivates us to devise a threefold algorithm as the core of our pattern discovery tool, STAMP.

1. First, the significant patterns involving one event are discovered. Two novel strategies, **optional information surplus pruning** and **maximum information gain counting**,

²We will explain it in more detail later in this paper.

³For example, the GIG of $(a_1, a_2, *)$ can not exceed the summation of that of $(a_1, *, *)$ and $(*, a_2, *)$.

¹Or equivalently, the event sequence would become a series of perfect repetitions of some pattern after a few replacements (of events).

are proposed to dramatically reduce the search space.

2. Next, candidate patterns involving multiple events are generated based on the *triangle inequality*.
3. All candidate patterns are validated and for each pattern which is significant, the corresponding subsequence containing the pattern is also identified.

The remainder of this paper is organized as follows. The model of generalized information gain and its properties are presented in Section 2 while the algorithm of STAMP is presented in Section 3. Section 4 presents some experiment results. We discuss some related work in Section 5. Finally, the conclusion is drawn in Section 6.

2 Generalized Information Gain

In this section, we provide a brief overview of the model of generalized information gain [28] and discuss its properties. Let $\mathfrak{S} = \{a_1, a_2, \dots\}$ be a set of events and D be a sequence of events in \mathfrak{S} .

DEFINITION 2.1. A **pattern** with **period** l is an array of l events (p_1, p_2, \dots, p_l) , each of which is either an event in \mathfrak{S} or $*$, i.e., $p_j \in \mathfrak{S} \cup *$ ($1 \leq j \leq l$). We say that the j th position is **instantiated** if $p_j \in \mathfrak{S}$. For any two patterns $P = (p_1, p_2, \dots, p_l)$ and $P' = (p'_1, p'_2, \dots, p'_l)$ of the same period l , P is a **superpattern** of P' (P' is a **subpattern** of P) if $p'_j = *$ or $p_j = p'_j$, for all $1 \leq j \leq l$.

Note that an event can appear at multiple positions in a pattern. For example, $(a_1, a_4, *, *, a_4)$ is a pattern of period 5 and its first, second and fifth positions are instantiated. It is also a superpattern of $(a_1, *, *, *, a_4)$.

DEFINITION 2.2. For an event $a_j \in \mathfrak{S}$ and a sequence D of N events, let $Prob(a_j)$ be the expected probability that a_j occurs at any given position in D ⁴. Then the **information** of a_j with respect to D is defined as $I(a_j) = \log \frac{1}{Prob(a_j)} = -\log Prob(a_j)$. The information of the “eternal” event $*$ is always 0⁵.

In practice, the probability of each event can be determined in many ways including, but not limited to

- $Prob(a_1) = Prob(a_2) = \dots = Prob(a_j) = \dots = \frac{1}{|\mathfrak{S}|}$;
- $Prob(a_j) = \frac{Num_D(a_j)}{N}$ for all $a_j \in \mathfrak{S}$ where $Num_D(a_j)$ and N are the number of occurrences of the event a_j in D and the length of the event sequence D , respectively;
- $Prob(a_j)$ is determined according to some domain knowledge.

In this paper, we adopt the second option and will not assume the same probability for every event, i.e. an occurrence of frequent event carries less information than a rare event. Note that this also coincides with the original intention of *information* in the data communication community.

⁴For the sake of simplicity of exploration, we assume that, without additional qualification, a_j occurs equally likely at any position with probability $Prob(a_j)$. All results presented in this paper can be modified to apply to a more general scenario.

⁵Another way of looking at it is that $Prob(*) = 1$ at any time.

DEFINITION 2.3. The **information** of a pattern $P = (p_1, p_2, \dots, p_l)$ is the summation of the information carried by each individual position, i.e., $I(P) = \sum_{1 \leq j \leq l} I(p_j)$.

Figure 1 shows the set of events corresponding to different workload states of a web server. There are total 1024 different events. Their probabilities of occurrence are arbitrarily assigned and the corresponding information is calculated accordingly. We use 1024⁶ as the base in the calculation. The information of pattern $(a_1, a_4, *, *, a_4)$ is $I(a_1) + I(a_4) + I(*) + I(*) + I(a_4) = 0.997 + 1.33 + 0 + 0 + 1.33 = 3.657$. After defining the information of a pattern, now we begin to formulate the definition of information gain of a pattern in a subsequence of events.

Given a pattern $P = (p_1, p_2, \dots, p_l)$ with period l and a sequence of l events $D' = d_1, d_2, \dots, d_l$, we say that D' is **in compliance with** P at position j ($1 \leq j \leq l$) iff either $p_j = *$ or $p_j = d_j$ holds. For example, the sequence a_1, a_1, a_2, a_3 is in compliance with the pattern $(a_1, a_2, a_3, *)$ at positions 1 and 4.

Given a pattern $P = (p_1, p_2, \dots, p_l)$ with period l and a sequence of l events $D' = d_1, d_2, \dots, d_l$, we say that P **matches** D' (or D' **supports** P), iff D' is in compliance with P at every position j ($1 \leq j \leq l$). For instance, the sequence a_1, a_4, a_2, a_3, a_4 supports the pattern $(a_1, a_4, *, *, a_4)$ while the sequence a_1, a_4, a_2, a_3, a_6 does not support it since the sequence is not in compliance with the pattern on the last position.

DEFINITION 2.4. Given a pattern P with period l and a sequence D of N ($N \geq l$) events: d_1, d_2, \dots, d_N , the **support** of P within D is the number of subsequences $d_{l \times j + 1}, d_{l \times j + 2}, \dots, d_{l \times j + l}$ that match P .

Intuitively, the event sequence can be viewed as a list of segments, each of which consists of l contiguous events. There would be $\lfloor N/l \rfloor$ full segments, among which the segment that P matches will count for the support of P . The “*” symbol is a wild card which matches any symbol.

Consider two subsequences $D_1 = a_1 a_2 a_1 a_2 a_1 a_2$ and $D_2 = a_1 a_2 a_1 a_1 a_1 a_2 a_1 a_2$ for the pattern $P = (a_1, a_2)$. The support of P in D_1 is the same as that in D_2 , which is 3. However, the generalized information gain of P with respect to D_1 should be higher than that of D_2 because there is no noise in D_1 but some noise in D_2 . Therefore, D_2 should “pay some penalty” for its noise, i.e., taking away some generalized information gain from D_2 . The amount of generalized information gain taken away depends on how D_2 can be repaired to perfection. In this case, if we replace an event a_1 with a_2 , then D_2 would be perfect for P . Thus, we decide to take away the information of a_2 (i.e., the information loss of P for the mismatched period in D_2) from D_2 .

DEFINITION 2.5. Given a pattern $P = (p_1, p_2, \dots, p_l)$ with period l and a sequence of l events $D' = d_1, d_2, \dots, d_l$, the **information loss** of D' on position j with respect to P is the information of the event p_j iff D' is not in compliance with P at position j and there is no information loss otherwise. The overall information loss of D' with respect to P is the summation of the information loss of each position.

⁶Here we choose the number of distinct events in the sequence as the base for calculating the information. It is inconsequential what is the base as long as the generalized information gain threshold specified by the user is consistent with the base.

Event	Machine 1	Machine 2	Machine 3	Machine 4	Machine 5	Probability	Information
a1	low	low	low	low	low	0.0010	0.997
a2	low	low	low	low	relatively low	0.0024	0.87
a3	low	low	low	low	relatively high	0.0013	0.959
a4	low	low	low	low	high	0.0001	1.33
a5	low	low	low	relatively low	low	0.0018	0.912
a6	low	low	low	relatively low	relatively low	0.0008	1.03
a7	low	low	low	relatively low	relatively high	0.0005	1.1
a8	low	low	low	relatively low	high	0.0023	0.876
a9	low	low	relatively low	low	low	0.0007	1.05
.
.
.
a1024	high	high	high	high	high	0.0003	1.17

Figure 1: The set of possible events

For example, the information loss of $a_1 a_4 a_5 a_2 a_3$ on position 2 with respect to $(a_1, a_4, *, *, a_4)$ is 0 while the information loss on position 5 is $I(a_4) = 1.33$.

DEFINITION 2.6. Given a pattern P with period l and a sequence D of N ($N \geq l$) events: d_1, d_2, \dots, d_N , the **information loss** of D with respect to P is the summation of the information loss of each segment $d_{l \times j + 1}, d_{l \times j + 2}, \dots, d_{l \times j + l}$ with respect to P . The **generalized information gain** of D with respect to P is defined as $I(P) \times (S_D(P) - 1) - L_D(P)$ where $I(P)$, $S_D(P)$, and $L_D(P)$ are the information of P , the support of P within D , and the information loss of D with respect to P , respectively.

In a subsequence, the first match of a pattern is viewed as an example, and only subsequent matches contribute to the generalized information gain⁷. With the generalized information gain concept, let's consider the event sequence $a_1 a_1 a_1 a_1 a_1 a_2 a_1 a_1 a_2 a_1 a_3 a_2 a_3 a_1 a_3 a_3 a_1 a_3 a_1 a_1$. In this sequence, the information of the occurrence of a_1 is $-\log_3(12/20) = 0.46$ while the information of the occurrence of a_2 is $-\log_3(3/20) = 1.73$. The generalized information gain of $(a_2, *, *)$ in subsequence $a_2 a_1 a_1 a_2 a_1 a_3 a_2 a_3 a_1$ is 3.46 while the generalized information gain of $(a_1, *, *)$ in subsequence $a_1 a_1 a_1 a_1 a_1 a_2 a_1 a_1 a_2 a_1 a_3 a_2$ is 1.38. This is due to the fact that event a_1 occurs much more frequent than a_2 , and hence it is likely that $(a_1, *, *)$ also occurs frequently in the event sequence. On the other hand, event a_2 occurs relatively scarce, thus the occurrence of $(a_2, *, *)$ carries more information. Therefore, if $g = 3$ is specified as the generalized information gain threshold, then pattern $(a_2, *, *)$ would satisfy the threshold, but not $(a_1, *, *)$. However, with the traditional support confidence thresholds, $(a_1, *, *)$ always has higher support value than $(a_2, *, *)$.

DEFINITION 2.7. Given a pattern P , a sequence D and a generalized information gain threshold g , if there exists a subsequence D' of D so that the generalized information gain of D' with respect to P is at least g , then P is a **valid pattern**.

Theoretically, the period of a valid pattern could be arbitrary, i.e., as long as the event sequence. In reality, a user can specify an upperbound of period length according to his/her

⁷Since we aim at mining periodic patterns, only repeated occurrences of a pattern are used to accumulate the generalized information gain.

domain knowledge. As a result, we use L_{max} to denote the maximum period allowed for a pattern. However, L_{max} can be arbitrarily large, e.g., ranging to several thousands. Now we can rephrase our problem model by employing the generalized information gain metric. For a sequence of events D , a generalized information gain threshold g , and a period bound L_{max} , we want to discover all valid patterns P whose period is less than L_{max} .

For each valid pattern P , we want to find the subsequence which maximizes the generalized information gain of P . In the remainder of this section, we give some more definitions which enable us to present our approach STAMP and communicate to readers more effectively.

DEFINITION 2.8. For any two patterns $P = (p_1, p_2, \dots, p_l)$ and $P' = (p'_1, p'_2, \dots, p'_l)$ of the same period l , P and P' are **complementary** if either $p_j = *$ or $p'_j = *$ for all $1 \leq j \leq l$.

A set of patterns of the same period are said to be **complementary** if every pair of patterns in the set are complementary.

DEFINITION 2.9. Given a set Π of complementary patterns of the same period l , the **minimum common superpattern (MCSP)** of Π is the pattern P of period l , which satisfies the following two conditions.

- Each pattern in Π is a subpattern of P .
- There does not exist a subpattern P' of P ($P' \neq P$) such that each pattern in Π is also a subpattern of P' .

It follows from the definition that the information of the MCSP of a set, Π , of complementary patterns is the summation of the information of each pattern in Π . For example, $(a_1, a_3, *, *, *)$, $(*, *, *, a_2, *)$, and $(*, *, *, *, a_4)$ are complementary and their MCSP is $(a_1, a_3, *, a_2, a_4)$. The information of $(a_1, a_3, *, a_2, a_4)$ is $I(a_1) + I(a_3) + I(a_2) + I(a_4)$ which is exactly the summation of the informations of $(a_1, a_3, *, *, *)$, $(*, *, *, a_2, *)$, and $(*, *, *, *, a_4)$. For a given event segment $D' = d_1, d_2, \dots, d_l$ and a set, Π , of complementary patterns, the information loss of D' with respect to the MCSP of Π satisfies the following equality

$$L_{D'}(MCSP(\Pi)) = \sum_{P \in \Pi} L_{D'}(P)$$

where $L_D(P)$ is the information loss of D' with respect to P . The rationale is that if D' is not in compliance with a pattern P

in Π on position j , then the j th position must be instantiated and D' must not be in compliance with the MCSP of Π on position j either. For instance, the information loss of the segment a_1, a_1, a_2, a_1, a_4 with respect to $(a_1, a_3, *, *, a_2, a_4)$ is $I(a_3) + I(a_2)$, which is equal to the aggregate of information loss of that segment with respect to $(a_1, a_3, *, *, *)$, $(*, *, *, a_2, *)$, and $(*, *, *, *, a_4)$. In general, for any event sequence D , the overall information loss of D with respect to the MCSP of a set of complementary patterns Π is equal to the summation of the information loss of D with respect to each pattern in Π .

PROPOSITION 2.1. (Triangle Inequality) *Given an event sequence D and two complementary patterns P and P' of the same period, let Q be the minimum common super pattern of P and P' . Then the generalized information gain of D with respect to Q is at most the summation of that of P and P' .*

Proof. Since P and P' are complementary, the information of Q is $I(Q) = I(P) + I(P')$ and $L_D(Q) = L_D(P) + L_D(P')$ for any event sequence D . Then for any sequence D , the generalized information gain of D with respect to Q is

$$\begin{aligned} & I(Q) \times (S_D(Q) - 1) - L_D(Q) \\ = & (I(P) + I(P')) \times (S_D(Q) - 1) - L_D(P) - L_D(P') \\ = & I(P) \times (S_D(Q) - 1) - L_D(P) + I(P') \times (S_D(Q) - 1) \\ & - L_D(P') \\ \leq & I(P) \times (S_D(P) - 1) - L_D(P) + I(P') \times (S_D(P') - 1) \\ & - L_D(P') \end{aligned}$$

because the support of Q in D (i.e., $S_D(Q)$) is at most the support of P in D (i.e., $S_D(P)$). Thus this proposition holds.

Proposition 2.1 can be easily generalized to a set of complementary patterns, which is stated as follows.

PROPOSITION 2.2. *Given an event sequence D and a set of complementary patterns Π , let Q be the minimum common super pattern of Π , then the generalized information gain of D with respect to Q is at most the summation of that of each pattern in Π .*

3 STAMP

In this section, we outline the general strategy we use to mine patterns that meet certain generalized information gain threshold g . There exist three challenges for mining patterns with information gain: (1) The number of different patterns is

$$\sum_{0 < l \leq L_{max}} (|\mathfrak{S}|^l - 1) = O(|\mathfrak{S}|^{L_{max}})$$

where $|\mathfrak{S}|$ and L_{max} are the overall number of distinct events and the maximum period length, respectively. Since L_{max} can be quite large, e.g., in the thousands, it is infeasible to verify each pattern against the data directly. Some pruning mechanism has to be developed to circumscribe the search space. (2) By definition, the generalized information gain measure does not have the property of downward closure as the traditional support measure does. For the sequence shown in Figure 2, the generalized information gains⁸ of the pattern $(a_4, *, *)$ and $(*, a_9, *)$ are $(5 - 1) \times 1.33 - 1.33 - 1.33 = 2.66$ and $(6 - 1) \times 1.05 - 1.05 = 4.2$, respectively; while the generalized

information gain of $(a_4, a_9, *)$ is $(5 - 1) \times 2.38 - 1.33 - 1.05 - 1.33 = 5.81$, which is greater than that of $(a_4, *, *)$ and $(*, a_9, *)$. If the generalized information gain threshold is set to $g = 5$, then only $(a_4, a_9, *)$ qualifies while the other two do not. This prevents us from borrowing existing algorithms developed for association rule problems to mine the qualified patterns. (3) The subsequence concept introduced in this paper poses a difficult challenge to determine when a subsequence should start and end. If a pattern misses some ‘‘matches’’, it is hard to tell whether this signals the end of a subsequence or this merely means some noise within a subsequence.

Fortunately, the *triangle inequality* holds for the generalized information gain. In other word, for a set of complementary patterns Π , the generalized information gain of the minimum common super pattern (MCSP) of Π is always less than or equal to the sum of that of each individual pattern in Π over the same subsequence of events. For example, the generalized information gain of $(a_4, a_9, *)$ is less than the sum of that of $(a_4, *, *)$ and $(*, a_9, *)$ over the same subsequence as demonstrated in Figure 2. Inspired by this observation, we can first collect the generalized information gain of all singular patterns, and then generate candidate patterns by combining these singular patterns. Figure 3 outlines our approach, STAMP. In the first phase, the valid singular patterns are discovered. The second phase generates the candidates of valid complex pattern based on the candidates of valid singular patterns via triangle inequality. Finally, STAMP verifies all candidates, and finds the corresponding subsequence for each valid pattern so as to maximize its generalized information gain. The **maximum information gain (MIG)** counting is designed to determine whether an event a may participate in a pattern P of period l and can discover all valid singular patterns. However, the overall number of MIG counters could be quite large. As a result, it is beneficial if the number of MIG counters can be reduced to limit the number of scans through the event sequence. We, thus, propose a pruning technique, **optimal information surplus (OIS)**, to prune out disqualified periods of each event before the MIG counting. The OIS pruning and MIG counting constitute the first phase of STAMP. After MIG counting, the candidate complex patterns are generated, and then verified. We will explain each component in detail in the following sections.

3.1 MIG Counting We first consider the issue of how to generate the MIG for a singular pattern, $(*, \dots, a, *, \dots, *)$, where the MIG serves as an estimate of the maximum achievable generalized information gain based on the maximum repetition of the singular pattern in a given sequence. We find that the problem of evaluating the MIG for a singular pattern on an event sequence d_1, d_2, \dots, d_N , is equivalent to discover the maximum summation of any subsequence of a sequence of real numbers derived from the event sequence based on the singular pattern. The generalized information gain of an event sequence D with respect to a pattern P is $(S_D(P) - 1) \times I(P) - L_D(P)$. If P is a singular pattern $(*, \dots, a, *, \dots, *)$ of period l , then $I(P) = I(a)$. We can partition D into segments of length l . The information loss on a segment that does not support P is $I(a)$. Let m be the number of segments that do not support P . The generalized information gain of D with respect to P can be rewritten as $(S(P) - 1 - m) \times I(a)$. More specifically, each segment is associated with $I(a)$ if it supports P and $-I(a)$ otherwise. Therefore, we can map the event sequence D (with N events) into a sequence of $\lfloor \frac{N}{l} \rfloor$ real numbers $x_1, x_2, \dots, x_{\lfloor \frac{N}{l} \rfloor}$. As shown in Figure 2, the sequence of real numbers with respect to $(a_4, *, *)$ is 1.33, -1.33, 1.33, 1.33, -1.33, 1.33, and

⁸We will use the information shown in Figure 1 constantly in all subsequent examples in this paper.

	a4	a9	a2	a6	a2	a2	a4	a9	a7	a4	a9	a2	a6	a9	a2	a4	a9	a1	a4	a9	a7	GIG
(a4, *, *)	1.33			-1.33			1.33			1.33			-1.33			1.33			1.33			2.66
(*, a9, *)		1.05			-1.05			1.05			1.05			1.05			1.05			1.05		4.2
(a4, a9, *)		2.38			-1.33	-1.05		2.38			2.38			-1.33			2.38			2.38		5.81

Figure 2: Violation of the Downward Closure

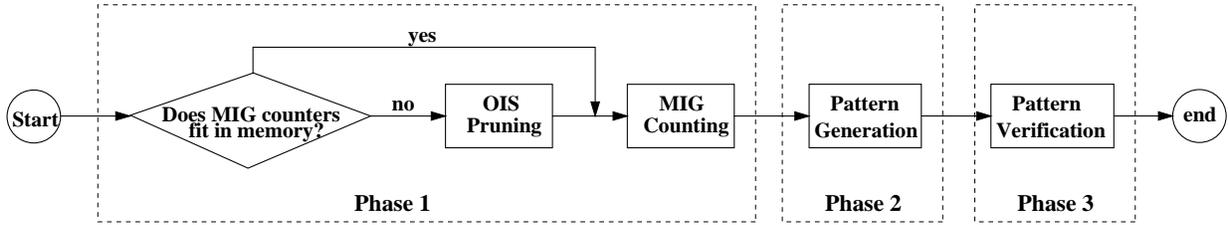


Figure 3: Outline of STAMP Approach

1.33. Now the problem becomes finding the maximum summation of any subsequence of $x_1, x_2, \dots, x_{\lfloor \frac{N}{T} \rfloor}$. The maximum information gain problem can be then formulated as follows [2].

$$b(j+1) = \max\{0, b(j)\} + x_{j+1}$$

$$c(j+1) = \max\{b(j+1), c(j)\}$$

where $b(j)$ and $c(j)$ are the maximum value of the summation of any subsequence ending exactly at x_j and the maximum value of the summation of any subsequence ending either before or at x_j , respectively. By choosing $b(0) = 0$ and $c(0) = 0$, the MIG of $(*, \dots, a, *, \dots, *)$ is equal to $\max\{c(\lfloor \frac{N}{T} \rfloor) - I(a), 0\}$. This forward recurrence can be easily solved by making one scan through the event sequence and maintaining two counters for each singular pattern to capture $b(\cdot)$ and $c(\cdot)$ at the current scanning position. The starting position and ending position of the corresponding subsequences can also be maintained simultaneously [2]. Since the number of MIG counters is $O(|\mathcal{S}| \times L_{max}^2)$, and the main memory of a common computer is limited to a few hundred MBytes which could be far less than $O(|\mathcal{S}| \times L_{max}^2)$, we need a mechanism to limit the number of MIG counters considered before the MIG counting procedure.

3.2 Segment-based OIS Pruning To reduce the number of singular patterns for collecting the MIG counts, we introduce a pruning step OIS based on a concept to score consecutive periodic appearances of an event based on generalized information gain. OIS determines for each event, what are the likely periods (or equivalently for each period, what are the likely events that will have the period). For each likely period l of event a , there are l possible singular patterns, e.g., for $l = 3$, the three singular patterns are $(a, *, *)$, $(*, a, *)$, and $(*, *, a)$. We then use the MIG counters to screen these singular patterns. As we shall see the OIS storage requirement is $O(|\mathcal{S}| \times L_{max})$ which is substantially lower than that of MIG counters and the OIS step can greatly reduce the candidates for MIG counters. Consider the occurrence of event a_k in an event sequence $D = d_1, d_2, \dots$. Intuitively, a_k would generate the most information gain if it repeats itself perfectly at a certain period in a subsequence of D .

If a_k participates in some pattern of period l , each occurrence of a_k could contribute to the generalized information gain of the pattern by an amount $I(a_k)$ in the optimistic scenario. However, in most case, the *net* information gain that a_k can contribute to a pattern is much smaller because (1) some information loss would incur due to the absence of a_k at some position; (2) some scattered/random occurrence of a_k has to be matched with the eternal event $*$. For example, the circled position in Figure 4 is considered as an absence of a_2 for pattern $(*, a_2, *)$. On the other hand, the third occurrences of a_2 have to be matched with the eternal event for pattern $(*, a_2, *)$. It is obvious that all of these suboptimal scenarios would weaken the net information gain that a_2 may contribute to any pattern. Since it is very complicated to track all of them, we choose to consider only one suboptimal scenario for estimating the information loss at this moment: the distance between two consecutive occurrences is greater than the length of the period. Let $Y(a_k, l)$ be the set of patterns of period l which contains a_k . We employ the following disciplines to estimate the OIS that an event a_k might contribute to any pattern in $Y(a_k, l)$.

1. Each occurrence of a_k is considered to be a positive contribution by amount $I(a_k)$.
2. When the distance between any two consecutive occurrences is greater than l , information loss must incur due to the absence of a_k . In fact, there are at least $\lceil \frac{\text{distance} - l}{T} \rceil$ absences, one for each segment following the previous occurrence of a_k . Each absence would cause an information loss of $I(a_k)$.

Figure 5 shows the process to estimate the OIS that event a_2 could contribute to any pattern of period 3. Each occurrence introduces a generalized information gain. There are three places where the distances between two consecutive occurrences are 4, 5, and 7, respectively. Information loss of $I(a_2)$ incurs for each period in these subsequence(s) deficient in a_2 .

These information losses and gains are essentially two lists of real numbers, namely $x(a_k, l, j)$ and $y(a_k, l, j)$ in Figure 5(b). At the j th occurrence of a_k , we can easily compute the the optimal net information surplus a_k could contribute

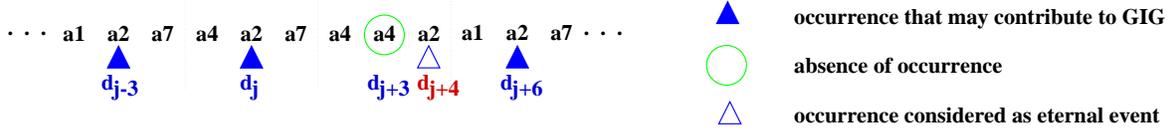


Figure 4: Occurrences of a_2

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
event	a1	a2	a7	a4	a9	a2	a4	a2	a2	a4	a9	a7	a4	a2	a2	a6	a9	a2	a4	a2	a1	a4	a9	a7	a6	a6	a2
		a2			a2			a2	a2			a2	a2					a2		a2							a2
info loss						-0.87								-0.87													-1.74
info gain		0.87				0.87		0.87	0.87					0.87	0.87			0.87		0.87							0.87

(a)

position	2	6	8	9	14	15	18	20	27
j	1	2	3	4	5	6	7	8	9
$x(a_2, 3, j)$		-0.87			-0.87				-1.74
$y(a_2, 3, j)$	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87
$f(a_2, 3, j)$	0.87	0.87	1.74	2.61	2.61	3.48	4.35	5.22	4.35
$OIS(a_2, 3, j)$	0	0	0.87	1.74	1.74	2.61	3.48	4.35	3.48

(b)

Figure 5: Optimal Information Surplus of Event a_2 for Period $l = 3$

in any event subsequence ending at that position, denoted by $OIS(a_k, l, j)$. Let $f(a_k, l, j)$ be the maximum net information aggregation of any subsequences ending exactly at the j th occurrence of a_k . We have

$$f(a_k, l, 0) = 0$$

$$f(a_k, l, j+1) = \max\{0, f(a_k, l, j) + x(a_k, l, j+1)\} + y(a_k, l, j+1)$$

$$OIS(a_k, l, j+1) = \max\{0, f(a_k, l, j+1) - I(a_k)\}$$

In the formula for $f(a_k, l, j+1)$, $\max\{0, f(a_k, l, j) + x(a_k, l, j+1)\}$ represents the contribution from subsequence ending at j and $y(a_k, l, j+1)$ represents the contribution from $(j+1)$ th position. Note that since $x(a_k, l, j+1)$ captures the potential information loss in the portion of subsequence prior to $(j+1)$ th position and should not affect the subsequence starting at $(j+1)$ th position, $x(a_k, l, j+1)$ and $y(a_k, l, j+1)$ are treated differently in the above formula. A linear algorithm would compute all OIS values as illustrated in Figure 5(b). Note that the OIS is an optimistic estimation and only gives an upperbound of the generalized information gain that an event a_k would contribute to any pattern. It is obvious that event a_2 does not exhibit strong pattern in Figure 5(a). However, the above OIS pruning method overestimates the generalized information gain of a_2 in Figure 5(b). Therefore, we propose another more sophisticated, but more effective OIS pruning method at the end of this section.

MIG Counter Generation after OIS Pruning After obtaining OIS values, for each period length l , we want to find which event is likely to appear in a valid pattern of period l . Let E_l denote the set of such events. By definition, any pattern of period l may contain at most l different events. The problem can be converted to testing whether the combined OIS of a set of l events may exceed the generalized information gain

threshold g at some position in the sequence. Even though there are totally $\binom{|S|}{l}$ different event combinations, it is not necessary to examine all of them. Conceptually, each event in a valid pattern must play a supporting role in accumulating generalized information gain. Therefore, we only need to consider the set of events with positive OIS at any time. (Note that this set may vary over time.) As we mentioned before, the event sequence can be treated as a list of segments of length l as shown in Figure 6. Each segment might serve as the last segment of a valid subsequence for some pattern. E_l need to be updated at the end of each segment. Let $T(l, s)$ be the set of events with positive OIS at the end of the s th segment, i.e., $T(l, s) = \{a_k \mid OIS(a_k, l, j_k) > 0, j_k \text{ is the last occurrence of } a_k \text{ before the end of the } s\text{th segment}\}$. For example, a_2 and a_4 are the only events with positive OIS value in segment 3, i.e., $T(3, 3) = \{a_2, a_4\}$. Since the OIS value of an event a_k is updated for each occurrence of a_k , it might not be updated in every segment and might also be updated multiple times within a segment. In any case, we always use the most recent value for the computation. In Figure 6, the OIS value of a_2 is not updated in segment 4 and is updated twice in segment 5. Then the OIS value that we use for these two segments are 1.74 and 2.61, respectively.

For each segment s , let $E(l, s)$ be the set of events that may appear in a pattern whose valid subsequence ends at the s th segment. $E(l, s)$ is essentially a subset of $T(l, s)$ and can be computed easily⁹. After we calculate $E(l, s)$ for all segments,

⁹One way to compute it is to examine the events in $T(l, s)$ in descending order of their OIS values. $E(l, s)$ is \emptyset if the combined OIS of the l largest ones is below g . Otherwise, all of these l events (with largest OIS values) are put into $E(l, s)$; and each remaining event $a_k \in T(l, s)$ is also added to $E(l, s)$ until the combined OIS of a_k and the $(l-1)$ largest ones is below g .

		a1	a2	a7	a4	a9	a2	a4	a2	a2	a4	a9	a7	a4	a2	a2	a6	a9	a2	a4	a2	a1	a4	a9	a7	a6	a6	a2
s		1	2		3		4		5		6		7		8		9											
OIS	a1	0																			0							
	a2	0	0		0.87	1.74					1.74	2.61	3.48		4.35					3.48								
	a4	0		1.33		2.66		3.99						3.99		5.32												
	a6																			0	0		1.03					
	a7	0												0														
	a9	0												0														
T(3, s)		{}	{}	{a2, a4}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a6}										
E(3,s)		{}	{}	{}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a6}										
E ₃		{}	{}	{}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a4}		{a2, a4, a6}										

Figure 6: Candidate Event(s) Generation for Period $l = 3$ and Minimum Generalized Information Gain $g = 3.5$

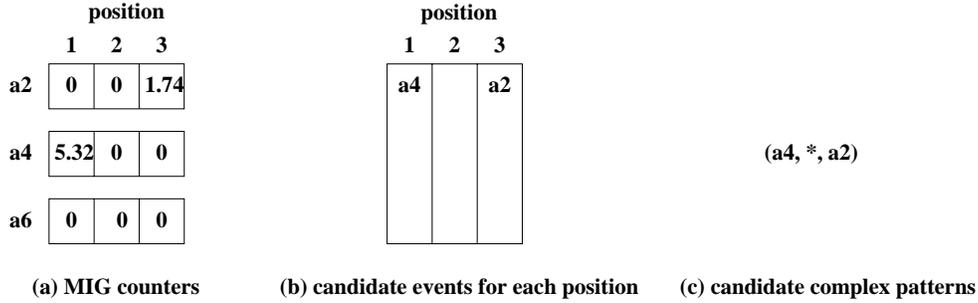


Figure 7: MIG Counters and Candidate Patterns for Period $l = 3$ and Minimum Generalized Information Gain $g = 3.5$

the set E_l can be trivially obtained by taking the union, i.e., $E_l = \bigcup_{\forall s} E(l, s)$. Figure 6 shows the process to compute the candidate events for period l . Note that a single scan of the event sequence is sufficient to compute the candidate events for all possible periods. For any given period l , if the candidate event set E_l is empty, then no pattern of period l would be able to meet the required generalized information gain. Once E_l is determined, for each event in E_l , we proceed to determine the singular pattern candidates with period l using the MIG counters. A counter is initialized for each event $a_k \in E_l$ at each position in the period. There are totally $l \times |E_l|$ counters where $|E_l|$ is the cardinality of E_l . For example, $E_3 = \{a_2, a_4, a_6\}$ for period 3. There are 3 different positions an event may occupy in a period. In turn, 3 MIG counters are initialized for each event, one for each position as illustrated in Figure 7(a). The procedure presented in the previous subsection is then carried out to collect all these MIG counters.

3.3 Candidate Pattern Generation After all these MIG counters have been gathered, for each position of a period (of length l), we keep all events with positive MIG as the candidate events for this position as shown in Figure 7(b). The candidate complex patterns¹⁰ of period l can be generated iteratively. A

pattern (p_1, p_2, \dots, p_l) is constructed each time by assigning each position p_j either an event in the corresponding candidate event set or eternal event. Let $MIG(p_j)$ is the MIG count for the singular pattern $(*, \dots, p_j, *, \dots)$ with p_j at the j th position. If $\sum_{1 \leq j \leq l} MIG(p_j) \geq g$, this pattern will be taken as a candidate pattern to the verification process presented later in this section. Otherwise, this generated pattern is simply discarded since it is impossible for this pattern to meet the generalized information gain threshold g . For example, Figure 7(c) shows the set of candidate patterns generated from the candidate events in Figure 7(b) with the threshold $g = 3.5$.

3.4 Pattern Verification The verification process of a candidate pattern $P = (p_1, p_2, \dots, p_l)$ of period l is similar to that to compute the MIG counts. The event sequence d_1, d_2, \dots is first mapped to a list of real numbers as follows. Each segment of l events $d_{l \times j + 1}, d_{l \times j + 2}, \dots, d_{l \times j + l}$ is examined at a time. It is mapped to a positive number $I(P)$ if it supports P . Otherwise, a negative number $-I(p_k)$ is mapped from each violated position of p_k . Then, a similar procedure to MIG computing can be applied to locate the subsequence that maximize the generalized information gain of the pattern. Figure 8 shows the validation of pattern $(a_4, *, a_2)$. The bracketed subsequence is the one that provides the maximum generalized information gain 3.53. Note that multiple subsequences may have the same generalized information gain. If that is the case, our algorithm will output the

¹⁰All singular patterns have already been considered in the MIG counting procedure.

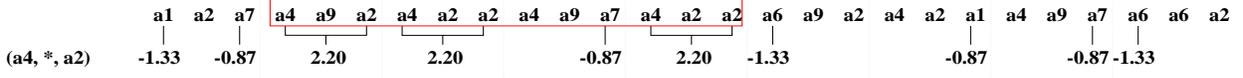


Figure 8: Verification of pattern $(a_4, *, a_2)$

first one¹¹.

3.5 Discussion: Sophisticated OIS Pruning with Superfluous Gain Elimination In this section, we discuss some techniques that can provide a better estimation of the value OIS. In section 3.2, we only consider the case that the gap between two consecutive occurrences of an event a_k exceeds the period when the information loss is calculated. We now examine the case that the gap is less than the period. Let's first consider the scenario that a_k repeats itself perfectly at certain distance l in a subsequence D' . For every occurrence of a_k (except the first and last one) at position j within D' , we can observe two companion occurrences of a_k at position $j - l$ and $j + l$, referred to as the **backward** and **forward** companion, respectively. Figure 9(a) shows a series of 4 occurrences of event a_2 with period 3, each of which (e.g., at position d_j) has two companions at the third preceding position (e.g., d_{j-3}) and the third subsequent position (e.g., d_{j+3}) of the current one. The total generalized information gain of the pattern $(*, a_2, *)$ generated from the subsequence is $(4 - 1) \times I(a_2)$. The above claim also holds in the case where an event occurs multiple times within a period. Figure 9(b), shows two overlapping series of repetitions of a_2 which may bring $(4 - 1) \times 2 \times I(a_2)$ generalized information gain to pattern $(*, a_2, a_2)$.

As we mentioned before, the *net* information gain a_k can contribute to a pattern is much confined if one of the following is true.

1. Some information loss would incur due to the absence of a_k at some position. This absence can be viewed as a companion absence of the neighboring occurrence(s). For example, the circled position in Figure 9(c) is considered as an absence of a_2 for pattern $(*, a_2, *)$. This absence makes both position d_{j-3} and position d_{j+3} lose one companion.
2. Some misaligned occurrence of a_k has to be treated as an eternal event or even as an absence of some other event and hence no generalized information gain can be collected from it. These occurrences usually are associated with companion absences. The third occurrences of a_2 in Figure 9(c) have to be treated as an eternal event for pattern $(*, a_2, *)$ and treated as "absence of a_7 " for pattern $(*, a_2, a_7)$. We also observe that in Figure 9(c) the event a_2 at position d_{j+4} has no companion at both position d_{j+1} and position d_{j+7} .

It is obvious that all of these suboptimal scenarios would weaken the net generalized information gain that a_2 may contribute to any pattern. By carefully identifying *companion absence* and assessing its impact to the generalized information gain, we can obtain a tighter estimation of the net contribution to the generalized information gain that a_k could make to any pattern in $Y(a_k, l)$. Since the companionship is essentially a mutual relationship, we choose to examine the backward companion to avoid looking forward in the event sequence. This is

achieved by employing an additional discipline to estimate the OIS that an event a_k might contribute to any pattern in $Y(a_k, l)$.

- 3. For each occurrence of a_k at position d_j , if the distance to the previous occurrence of a_k is less than l , then the previous occurrence of a_k is not the backward companion of a_k at position d_j . So an information adjustment of $-I(a_k)$ is needed.

The new calculation of OIS is as follows.

$$\begin{aligned}
 f(a_k, l, 0) &= 0 \\
 f(a_k, l, j + 1) &= \max\{e(a_k, l, j), \\
 &\quad f(a_k, l, j) + x(a_k, l, j + 1) \\
 &\quad + y(a_k, l, j + 1) + z(a_k, l, j + 1)\} \\
 OIS(a_k, l, j + 1) &= f(a_k, l, j + 1) - I(a_k)
 \end{aligned}$$

where $e(a_k, l, j)$ is the product of $I(a_k)$ and the number of occurrences of a_k in the previous l events prior to the j th occurrence of a_k . Here $z(a_k, l, j)$ is the adjustment according to the third discipline stated above. The revised computation of OIS is shown in Figure 10(a).

For each period length l , the addition storage requirement to perform the generalized information gain adjustment is an array of l elements (to store the previous l events). The computation complexity remains the same.

Finally, we give some rationale for the third discipline. Intuitively, when the companion absence is present, the generalized information gain would not be as high as expected. Some adjustment needs to be taken to provide a tighter estimation. Let's reconsider the example shown in Figure 5(a), which is also described in Figure 10(b). $(*, a_2, *)$, $(*, *, a_2)$, and $(*, a_2, a_2)$ are the three possible patterns that involve a_2 because a_2 only appears at the second and third position on each segment.

- The adjustment in position 8 comes from the following reasons. The solid ovals indicate the adjustments that are taken according to the third discipline. For $(*, a_2, *)$, comparing the actual information gain/loss with the previous estimation in section 3.2, the generalized information gain on position 6 is superfluous. For $(*, *, a_2)$, the generalized information gain on position 8 is superfluous, while the information on position 6 for pattern $(*, a_2, a_2)$ is superfluous. Therefore, by position 8, one generalized information gain of 0.87 is superfluous for all patterns, thus, we adjust the generalized information gain by -0.87.
- The adjustment in position 15 is necessary because of the following reasons. Generalized information gain on position 14 and 15 is superfluous for pattern $(*, *, a_2)$ and $(*, a_2, *)$, respectively, due to the similar reasons described above. For $(*, a_2, a_2)$, on position 11 and 12, we need to deduct generalized information gain by 1.74; however, we only deduct generalized information gain by 0.87 on position 14. As a result, an additional 0.87 needs to be deducted from the net generalized information gain. Thus, we add an adjustment of -0.87 on position 15.

¹¹With minor modification of the algorithm, all such subsequences can be output.

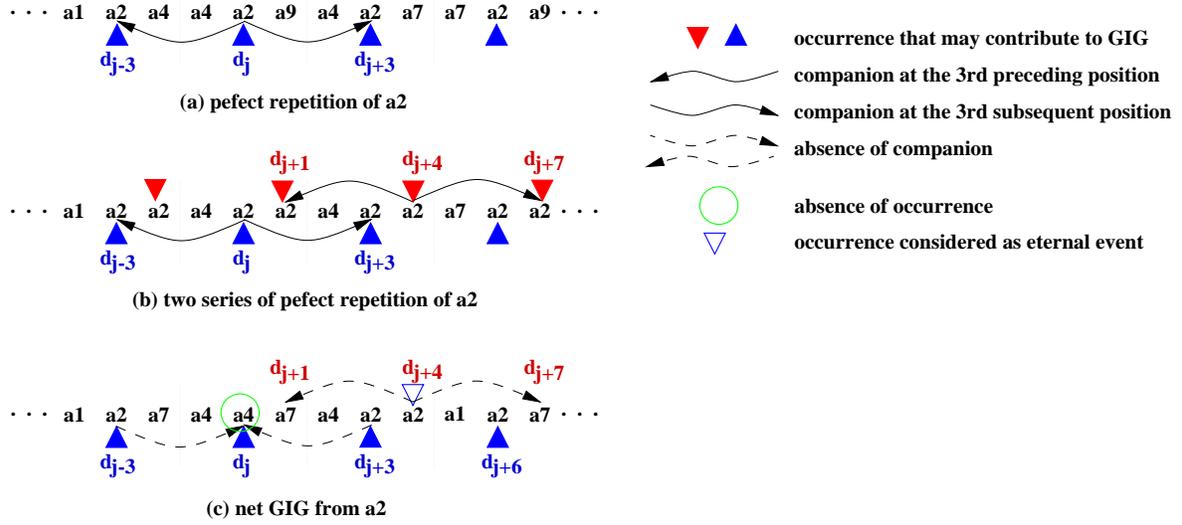


Figure 9: (Revisit) Occurrences of a_2

- On position 20, another adjustment is created. There are two superfluous information gain of 0.87 on position 18 and 20 for $(*, a_2, a_2)$. Also, there is one superfluous information gain of 0.87 by position for both $(*, a_2, *)$ and $(*, *, a_2)$ as indicated in Figure 10(b). Therefore, one generalized information gain adjustment of -0.87 is created on position 20.

As a rule of thumb, for an event a_k , the adjustment is postponed until the time it would apply to all possible patterns involving a_k . Therefore, the new estimation could be used as a tighter bound on the generalized information gain.

4 Experimental Results

We implemented the STAMP in C programming language on an IBM RS-6000 (300 MHz CPU) with 128MB running AIX operating system. In the following experiments, we set $L_{max} = 1000$.

4.1 Synthetic Sequence Generation For the purpose of evaluation of the performance of STAMP, we use four synthetically generated sequences. Each sequence consists of 1024 distinct events and 20M occurrences of events. The synthetic sequence is generated as follows. First, at the beginning of the sequence, the period length l of the next pattern is determined, which is geometrical distributed with mean μ_l . The number of events involved in a pattern is randomly chosen between 1 and l . The number of repetitions m of this pattern is geometrical distributed with mean μ_m . The events that are involved in the pattern are chosen according to a normal distribution with mean $\frac{1}{1024}$ (there are total 1024 distinct events) and standard deviation 2. However, the pattern may not perfectly repeat itself for m times. To simulate the imperfectness of the subsequence, we employ a parameter δ to control the noise. δ is uniformly distributed between 0.5 and 1. With probability δ , the next l events match the pattern. Otherwise, the next l events do not support the pattern. The replacement events are chosen from the event set with the same normal distribution (mean and standard deviation equal to $\frac{1}{1024}$ and 2, respectively). This subsequence ends when there are m matches, and a new subsequence for a new pattern starts. This process repeats until it reaches the end of the

sequence. Four sequences are generated based on values of μ_l and μ_m in Table 1.

Data Set	μ_l	μ_m	Distinct events	Total Events
$l3m20$	3	20	1024	20M
$l100m20$	100	20	1024	20M
$l3m1000$	3	1000	1024	20M
$l100m1000$	100	1000	1024	20M

Table 1: Parameters of Synthetic Data Sets

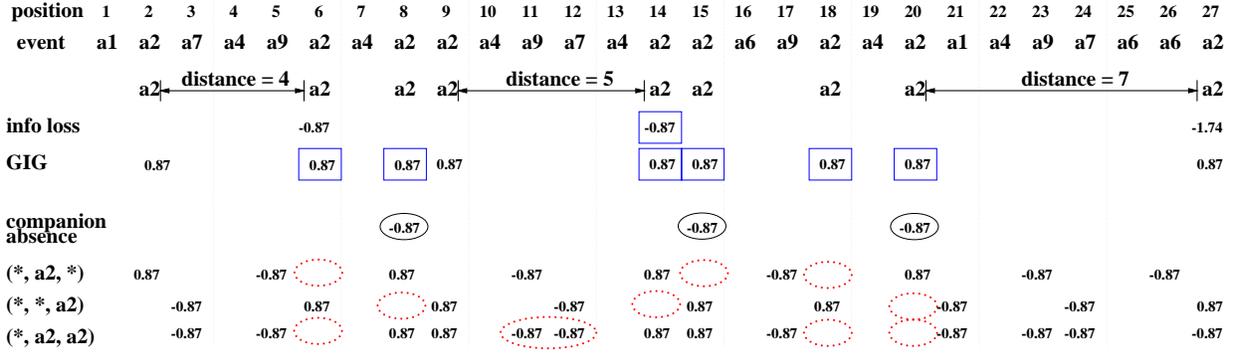
4.2 Effects of OIS Pruning Figure 11 (a) shows the difference of the pruning power of the sophisticated OIS pruning with superfluous information gain elimination and the segment-based OIS pruning. Since the behaviors are similar with different event sequences, we only show the pruning results for sequence $l3m20$. It is evident that the sophisticated one is much more effective. Although the more sophisticated OIS pruning requires a little bit more space and time, the result is improved dramatically. Therefore, we decide that in the remainder of this section, we use the implementation of the more sophisticated OIS pruning technique in STAMP.

Figure 11 (b) shows the effectiveness of the more sophisticated OIS pruning. The y-axis shows the fraction of the MIG counters that would not be needed. It is evident that when the generalized information gain threshold g increases, the OIS is more effective because less events at each period may qualify for MIG counting. However, even with $g = 2$, the OIS pruning can filter out more than 50% of the MIG counters.

4.3 Effects of MIG Counting The number of candidate patterns depends on the average number of events (with positive MIG values) in each position of each period. Figure 12(a) shows the average number of events (α) in each position for period (l) between 5 and 100 with the generalized information gain threshold $g = 5$. (Note that the Y-axis is in log scale.) The α value is similar for all four data sets and α decreases with l . In Figure 12(a), when $l > 30$, $\alpha < 1$ for all four sequences. In other words, many positions of a pattern with period larger than 30 are null. The total number of candidate patterns (β) for each

position	2	6	8	9	14	15	18	20	27
j	1	2	3	4	5	6	7	8	9
$x(a_2, 3, j)$		-0.87			-0.87				-1.74
$y(a_2, 3, j)$	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87
$z(a_2, 3, j)$			-0.87			-0.87		-0.87	
$e(a_2, 3, j)$	0.87	0.87	1.74	1.74	0.87	1.74	0.87	1.74	0.87
$f(a_2, 3, j)$	0.87	0.87	1.74	2.61	2.61	2.61	3.48	3.48	2.61
OIS($a_2, 3, j$)	0	0	0.87	1.74	1.74	1.74	2.61	2.61	1.74

(a)



□ GIG that need to be adjusted ○ information adjustment ○ original position for adjustment

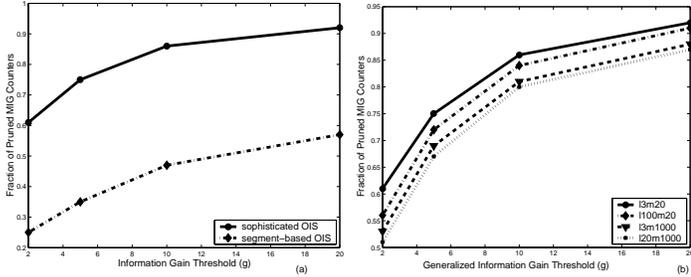
Figure 10: Sophisticated OIS Estimation of Event a_2 with respect to Period $l = 3$ 

Figure 11: Effects of OIS Pruning

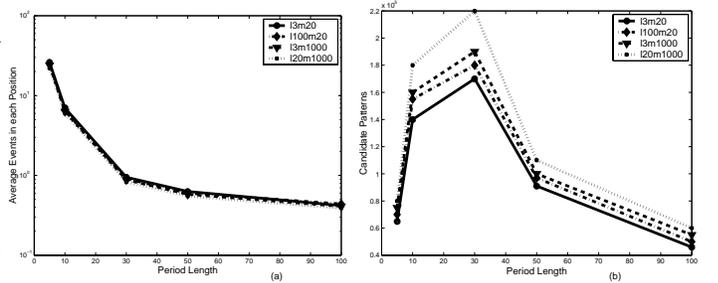


Figure 12: Candidates Pattern

period between 5 and 100 is illustrated in Figure 12(b). β increases with l when $l < 30$ due to the longer periods. On the other hand, β decreases with l when $l > 30$ due to the smaller number of possible events in each position.

4.4 Overall Performance The overall performance of STAMP depends largely on the number of MIG counters and candidate patterns. If MIG counters and candidate patterns can not fit into main memory, then multiple scans of the event sequence is needed to generate the counters or verify the candidate patterns. For all four sequences, after the OIS counting, the MIG counters can fit into the main memory. However, the candidate patterns can not fit into main memory at once. One third of the candidate patterns of $l100m1000$ sequence can be loaded into memory each time while half of the candidate patterns of the other three sequences can be loaded into memory each time for $g = 5$. Figure 13(a) shows

the overall response time of STAMP for four event sequences with respect to the generalized information gain threshold. The average performance of $l2m20$, $l100m20$, and $l3m100$ is similar because of the similar number of disk I/Os whereas the performance of $l100m1000$ is significantly higher.

To analyze the benefits of using the generalized information gain as a threshold versus using support and confidence as thresholds, we also implemented a data mining tool which finds the subsequences with maximum support while maintaining the confidence and support above certain thresholds. We call this subsequence discovery tool *s-c Miner*. In *s-c Miner*, an approach similar to [10] is used to generate all patterns that satisfy the support threshold followed by an algorithm adapted from [7] to find the desired subsequence. Figure 13 (b) shows the performance difference between STAMP and *s-c Miner*. (Note the y-axis in Figure 13 (b) is in log scale.) Since the performance on all four sequences is similar, thus, we only

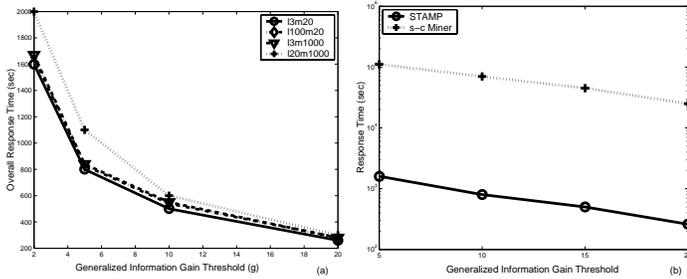


Figure 13: Performance of STAMP

show the performance of sequence $l3m20$. The support and confidence thresholds in s-c Miner is set in such a way that all subsequences found by STAMP can also be found by s-c Miner. Since a large number of patterns are generated by s-c Miner which are not deemed as valid by STAMP, the performance of s-c Miner is severely impact. However, readers should keep in mind that in some applications, if the support and confidence are the proper measurement to qualify a valid pattern, then the s-c Miner should be preferred.

5 Related Work

In this section, we provide a brief overview of recent advances that is closely related to our work presented in this paper.

5.1 Mining Sequence Data Most previous work on mining sequence data fell into two categories: discovering sequential patterns [1, 3, 4, 6, 15, 24, 29] and mining periodic patterns [9, 10, 16, 26]. The primary difference between them is that the models of sequential pattern purely take into account the number of occurrences of the pattern while the frameworks for periodic patterns focus on characterizing cyclic behaviors.

5.1.1 Sequential Patterns Discovering frequent sequential patterns was first introduced in [1]. The input data is a set of sequences, called data-sequences. Each data-sequence is a list of transactions and each transaction consists of a set of items. A sequential pattern also consists of a (fully ordered) list of transactions. The problem is to find all frequent sequential patterns with a user-specified minimum support, where the support of a sequential pattern is the percentage of data-sequences that contain the pattern. Apriori-based algorithms, such as AprioriALL [1] and GSP [24], were proposed to mine patterns with some minimum supports in a level-wise manner. To further improve the performance, a projection-based algorithm called FreeSpan [11] and its successor PrefixSpan [21] were introduced to reduce the candidate patterns generated and hence reduce the number of scans through the data. Additional useful constraints (such as time constraint and regular expression constraint) and/or taxonomies were also studied extensively in [8, 24, 29] to enable more powerful models of sequential patterns.

As a more generative model, the problem of discovering frequent episodes from a sequence of events was presented in [15]. An episode is defined to be a collection of events that occur relatively close to each other in a given partial order. A time window is moved across the input sequence and all episodes that occur in some user-specified percentage of windows are reported. The model was further generalized by Padmanabhan et al. [17] to suit temporal logic patterns.

5.1.2 Periodic Patterns Full cyclic pattern was first studied in [16]. The input data to [16] is a set of transactions, each of which consists a set of items. In addition, each transaction is tagged with an execution time. The goal is to find association rules that repeat themselves throughout the input data. In [9, 10], Han et. al. presented algorithms for efficiently mining partial periodic patterns. In practice, not every portion in the time series may contribute to the periodicity. For example, a company's stock may often gain a couple of points at the beginning of each trading session but it may not have much regularity at later time. This type of looser periodicity is often referred to as *partial periodicity*. The difference between our model and [9, 10] is that we aim at mining statistically important periodic patterns while Han et al. focused on frequent periodic patterns.

The most related work is the InfoMiner proposed in [27]. The InfoMiner uses the information gain as the measure of the interestingness of a pattern. However, there is no penalty for a gap between the occurrences of the pattern. In the STAMP model, we allow penalty to be associated with the gap between pattern occurrences (which is more suitable for many applications such as bioinformatics) and focus on devising efficient algorithms to mining patterns under this new model.

5.2 Models of Interestingness Despite the difference in problem formulation, most work surveyed in the previous subsection adopted the support as the measure of interestingness/significance and aimed at discovering frequent patterns. Recently, many efforts have been carried out to address the potential disadvantages associated with the support model and to propose alternative solutions.

5.2.1 Refining Mined Results As a well-known fact, the number of patterns/rules discovered under the support model can be very large. Many post-processing techniques have been developed to reduce the number of discovered patterns into a manageable size while preserving the discovered knowledge as much as possible. Human interaction is involved in [12, 22, 23] to specify the interestingness or beliefs to guide the process while others [13, 14] focused on reducing redundant information possessed by the discovered rules. It is clear that these post-processing techniques are typically used as an additional pruning step after the normal mining procedure (which produces a large rule set). In contrast, our proposed scheme successfully avoids the generation of large number of insignificant/uninteresting patterns from the beginning and enables a much more efficient solution.

Another approach to reduce redundancy is to return only closed frequent itemset [19, 20, 30]. Intuitively, an itemset is a closed itemset if all of its supersets have smaller support. While the set of frequent closed itemsets is typically much smaller, it has been proved that all frequent itemsets can be uniquely derived from the set of frequent closed itemsets. Again, this approach still focuses on mining frequent itemsets and fails to address the problem we mentioned previously.

5.2.2 Multiple Supports Scheme Multiple supports scheme was introduced by Liu et. al. [13] and later extended by Wang et al. [25] to find itemsets which do not occur frequently overall, but have high correlation to occur with some other items. The support threshold to qualify a frequent itemset can be specified as a fraction of the minimum support of all items [13] or subsets of items [25] in the itemset. This variable support has similar effect as the generalized information gain introduced in

this paper. However, there exists some fundamental difference between these two concepts. For example, if the support of item A, B, and C is 0.01, 0.02, 0.8, respectively, then the support threshold to qualify itemset AB and AC is the same. Nevertheless, the itemset AC is expected to occur more frequently than AB because the support of C is much larger than that of B. This aspect was not fully taken into account by the multiple support model¹². In contrast, the generalized information gain metric would capture the difference of occurrences between B and C.

5.2.3 Statistically Significant Patterns There are much work in discovering statistically significant patterns [5, 18, 27]. All those work only takes into account the occurrence of a pattern in a sequence or a transaction. However, it does not assign any penalty if a pattern fails to be present when it is supposed to. In addition, all those work only discovers the significant patterns for the entire data set, and does not identify the subsequence (subset) where a pattern is significant.

6 Conclusions

In this paper, we focus on mining partial periodic patterns with random replacement. The generalized information gain is used to seamlessly accommodate the different frequency of event occurrences as well as gaps in an event sequence. The triangle inequality preserved by the generalized information gain enables us to devise a linear algorithm, STAMP, to mine the significant pattern in any subsequence combinations. The OIS pruning and MIG counting strategies also provide additional performance improvement. The efficiency is demonstrated by the experimental results.

References

- [1] R. Agrawal and R. Srikant. Mining Sequential Patterns. *Proc. Int. Conf. on Data Engineering (ICDE)*, Taipei, Taiwan, 3-14, March 1995.
- [2] J. Bentley. Programming pearls. *Communications of ACM*, vol. 27, no. 2, 865-871, 1984.
- [3] D. Berndt and J. Clifford. Finding patterns in time series: a dynamic programming approach. *Advances in Knowledge Discovery and Data Mining*, 229-248, 1996.
- [4] Claudio Bettini, Xiaoyang Sean Wang, Sushil Jajodia, and Jia-Ling Lin. Discovering frequent event patterns with multiple granularities in time sequences. *IEEE Transaction on Knowledge and Data Engineering*, 10(2), 222-237, 1998.
- [5] S. Brin, R. Motwani, C. Silverstein. Beyond market baskets: generalizing association rules to correlations. *Proc. ACM SIGMOD Conf. on Management of Data*, 265-276, 1997.
- [6] Gautam Das, King-Ip Lin, Heikki Mannila, Gopal Renganathan, and Padhraic Smyth. Rule discovery from time series. *Proc. Int. Conf. on Knowledge Discovery and Data Mining*, 16-22, 1998.
- [7] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Mining optimized association rules for numeric attributes. *Proc. 15th ACM Symposium on Principles of Database Systems*, 182-191, 1996.
- [8] M. Garofalakis, R. Rastogi, and K. Shim. SPIRIT: sequential pattern mining with regular expression constraints. *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, 223-234, 1999.
- [9] J. Han, W. Gong, and Y. Yin. Mining segment-wise periodic patterns in time-related databases. *Proc. Int. Conf. on Knowledge Discovery and Data Mining*, 214-218, 1998.
- [10] J. Han, G. Dong, and Y. Yin. Efficient mining partial periodic patterns in time series database. *Proc. Int. Conf. on Data Engineering*, 106-115, 1999.
- [11] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M. Hsu. FreeSpan: frequent pattern-projected sequential pattern mining. *Proc. Int. Conf. on Knowledge Discovery and Data Mining*, 2000.
- [12] M. Klemetinen, H. Mannila, P. Ronkainen, H. Toivonen, and A. Verkamo. Finding interesting rules from large sets of discovered association rules. *Proc. CIKM*, 1994.
- [13] B. Liu, W. Hsu, and Y. Ma. Mining association Rules with multiple minimum supports. *Proc. ACM SIGKDD*, 337-341, 1999.
- [14] B. Liu, M. Hu, and W. Hsu. Multi-level organization and summarization of the discovered rules. *Proc. ACM SIGKDD*, 208-217, 2000.
- [15] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, vol. 1, no. 3, 259-289, 1997.
- [16] B. Ozden, S. Ramaswamy, and A. Silberschatz. Cyclic association rules. *Proc. 14th Int. Conf. on Data Engineering*, 412-421, 1998.
- [17] B. Padmanabhan and A. Tuzhilin. Pattern discovery in temporal databases: a temporal logic approach. *Proc. ACM KDD*, 351-354, 1996.
- [18] B. Padmanabhan and A. Tuzhilin. Small is beautiful: discovering the minimal set of unexpected patterns. *Proc. ACM SIGKDD*, 54-63, 2000.
- [19] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. *Proc. Int. Conf. on Database Theory*, 398-416, 1999.
- [20] J. Pei, J. Han, and R. Mao. CLOSET: an efficient algorithm for mining frequent closed itemsets. *Proc. ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 21-30, 2000.
- [21] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu. PrefixSpan: mining sequential patterns by prefix-projected growth. *Proc. 17th Int. Conf. on Data Engineering*, 215-224, 2001.
- [22] S. Sahar. Interestingness via what is not interesting. *Proc. 5th ACM Int. Conf. on Knowledge Discovery and Data Mining (SIGKDD)*, pp. 332-336, 1999.
- [23] Myra Spiliopoulou. Managing interesting rules in sequence mining. *Proc. European Conf. on Principles and Practice of Knowledge Discovery in Databases*, 554-560, 1999.
- [24] R. Srikant and R. Agrawal. Mining sequential patterns: generalizations and performance improvements. *Proc. 5th Int. Conf. on Extending Database Technology (EDBT)*, 3-17, 1996.
- [25] K. Wang, Y. He, and J. Han. Mining frequent itemsets using support constraints. *Proc. 26th Intl. Conf. on Very Large Data Bases (VLDB)*, 2000.
- [26] J. Yang, W. Wang, and P. Yu. Mining asynchronous periodic patterns in time series data. *Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (SIGKDD)*, pp. 275-279, 2000.
- [27] J. Yang, W. Wang, and P. Yu. InfoMiner: mining surprising periodic patterns. *Proc. ACM Knowledge Discovery and Data Mining*, 395-400, 2001.
- [28] J. Yang, W. Wang, and P. Yu. InfoMiner+: mining partial periodic patterns with gap penalties. *Proc. 2nd IEEE Int. Conf. on Data Mining*, 2002.
- [29] M. Zaki. Sequence mining in categorical domains: incorporating constraints. *Proc. 9th Int. Conf. on Information and Knowledge Management*, 422-429, 2000.
- [30] M. Zaki. Generating non-redundant association rules. *Proc. ACM Knowledge Discovery and Data Mining*, 34-43, 2000.

¹²Theoretically, the model in [25] is capable of addressing this problem by explicitly enumerating all itemsets, which is unfortunately impractical in general.