Pattern Mining Model for Automatic Network Monitoring in Heterogeneous Wireless Communication Networks

Zhiguo Qu\(^1,2\), Jie Deng\(^1\), John Keeney\(^3\), Sven van der Meer\(^3\), Xiaojun Wang\(^1\), Conor McArdle\(^1\)

\(^1\) Dublin City University, The Rince Institute, Dublin, Ireland
Email: (zhiguo.qu, xiaojun.wang@dcu.ie, deng113jie@gmail.com and mcardlec@eeng.dcu.ie)

\(^2\) Jiangsu Engineering Center of Network Monitoring, Nanjing University of Information Science & Technology, Nanjing, China

\(^3\) Ericsson, Network Management Lab, Athlone, Ireland
Email: (john.keeney, sven.van.der.meer)@ericsson.com

Abstract—The rapid development of network technology and its evolution towards heterogeneous networks has increased the demand for more efficient network monitoring and management capabilities in providing high quality communication services. Current research aims to support automatic monitoring and management of heterogeneous wireless communication networks. In this work sequential pattern mining is used to extract interesting event sequence patterns from Telecom Network Monitoring data, driven by the need to support ever increasing volumes of management data. In this paper, a novel model of pattern discovery is proposed. Existing algorithms (PrefixSpan and HTPM) are extended and integrated into the model for discovering interesting sequential patterns. Evaluations show that these algorithms are suitable as part of an architecture to support automatic network monitoring.

Keywords—Automatic network monitoring, Sequential pattern mining, Episode discovery.

I INTRODUCTION
Recent advances in telecommunication network technologies and their evolution to heterogeneous networks have exacerbated the need for more efficient network monitoring and management capabilities in providing high performance communications services. In support of automatic management of heterogeneous networks, new techniques are required for analyzing the huge volumes of monitoring data being streamed into modern OSS (Operation Support System) platforms. Pattern discovery techniques and approaches for identifying anomalous events and event sequences are candidates for achieving this goal, especially in the context of ever increasing management data volume.

Considering the complex environment of heterogeneous networks, there are at least two main issues to consider: 1) Huge amounts of data with high data dimension; 2) Huge number of protocols or use cases with complex contexts. To handle high volume and high dimensionality in data, potential algorithms must achieve high time and memory efficiencies to fit off-the-shelf hardware configurations. On one hand it is next to impossible to enumerate and model all the possible contexts and use cases in heterogeneous networks, while on the other hand drive towards automatic network monitoring which aims to greatly reduce or remove the need for human input in the processing part of network monitoring. To operate in an automatic manner, and without expert domain knowledge input, there remains a requirement for general purpose algorithms for discovering potentially interesting patterns and contexts. The most appropriate techniques are based in the domain of sequential data mining and association rule discovery.

In this paper, a novel sequential pattern mining model is proposed. This generic model can efficiently achieve pattern mining for heterogeneous network monitoring data, and discover potentially interesting sequential patterns by integrating sequential pattern mining techniques and association rule discovery techniques.

II RELATED WORK
Sequential data mining attempts to extract useful knowledge from traditional static databases. The outputs of the pattern discovery can help to make decisions of recommended management actions and predict future incidents \[2, 15\]. This work aims to leverage such techniques for automated monitoring of heterogeneous telecommunications networks.
According to the classification scheme proposed by Wu and Chen [13], techniques for temporal pattern mining in sequential data can be classified into four categories: point-based techniques, interval-based techniques, hybrid-based techniques and multi-label event techniques. Among them, point-based event assumes that the events always occur at instantaneous time points, which is the common assumption adopted by most previous studies of pattern mining in sequential data. The related achievements include [1, 3-6] which are all based on Apriori Theory [1]. Research on mining patterns from point-based events attempts to find the temporal relationships between these point events. In many applications, however, events cannot be simply represented as a point. Therefore, some previous research extended pattern mining in sequential data to the situations that include interval-based events [7-10]. An interval-based event can be described by three major characteristics: event ID, event starting time, and event ending time. Research on mining temporal patterns from interval-based events attempts to find the temporal relationships between these interval events. In some applications, however, events are neither purely point-based nor purely interval-based; they are hybrid events. Correspondingly, a pattern consisting of point-based and interval-based events is referred as a hybrid temporal pattern. Research [11-13] on mining temporal patterns from hybrid events attempts to find the temporal relationships between these hybrid events. Recent research [14] has extended hybrid events to be multi-label events, because an event can have changing statuses in specific applications and contexts. When events are multi-label, an event can be described with more major characteristics: event ID, event's first label time, event's second label time, ..., event's final label time. Therefore, the research on mining temporal patterns from multi-label events attempts to find the temporal relationships between these multi-label events. 

Usually, association rule mining techniques can be viewed as an extension of sequential pattern mining whereby reliable and interesting predictive patterns are viewed as prediction rules and are augmented in terms of some mathematical methods, such as Confidence and Lift [16]. Confidence denotes whether association rules exist in discovered frequent patterns, and Lift represents whether one event or event sequence occurrence will cause or predict another event or event sequence occurrence. Correspondingly, if a pattern occurs (is supported) by at least Min-Sup percentage of sequences in a given dataset, it is said that the pattern is a frequent sequential pattern. If its Confidence and Lift are greater than their corresponding user-defined thresholds, the pattern will be regarded as a frequent reliable or interesting sequential pattern.

III CONTEXT OF THIS WORK

This paper presents work done in the context of a larger project, which aims to support automatic monitoring and management of heterogeneous telecommunications networks. To achieve this goal, a “Pattern Discovery” work-package strives to extract sequential event patterns from network monitoring data, especially in the context of ever increasing management data volumes, the subject of this paper.

a) Overall Project Architecture

The overall project architecture is shown in Figure 1. The architecture is mainly made up of five modules including a Dimension Reduction Module (DRM), an Episode Discovery Module (EDM), an Episode Classification Module (ECM), Pattern Matching & Prediction Module (PMM) and a Recommender System Module (RSM). This architecture indicates how these required modules carry out their specific tasks or take responsibilities in the workflow as a whole.

The DRM (Dimension Reduction Module) module aims to greatly reduce dimensionality of the data, and filter out most of the noisy events in the data stream for reducing the computation complexity of EDM and PMM. (Usually, management data will be recorded into trace files or transformed as trace event stream before being processed. So, for simplicity, the data stream is also called the trace-stream.)

The main goal of EDM (Episode Discovery Module) is to discover and identify episodes/sequential patterns from the dimension reduced trace-stream, while importing the episodes/sequential patterns into the PML (Pattern Model Library) and updating the episodes/sequential patterns for PMM and RSM.

The ECM (Episode Classification Module) further identifies and classifies the episodes/sequential patterns. Feedback from the ECM can help the DRM and the EDM optimize their parameter settings (such as transaction length and window size, etc.), for better performance. The identified patterns called pattern models are outputted into pattern model library, which serves PMM and RSM.

After receiving the pattern models from PML, the PMM (Pattern Matching & Prediction Module) scans the trace-stream and matches the partial pattern models for predicting impending pattern models. Its predictions will be exported to the RSM.

In response to matches of pattern prediction rules the RSM (Recommender System Module) ranks of suggestions to avert, alleviate or mitigate against the effect of predicted issues, based on previous expert judgements or historic actions and experience of network administrators.
The task of the EDM (Episode Discovery Module) is to constantly mine episodes (pattern candidates) from dimension reduced data streams. From the perspective of specific processing steps, the episode discovery module receives the output of the DRM as its input and sends its output to the PML. During this process, the EDM continuously discovers new or evolving episodes, identifies new pattern models and exports those pattern models to the PML. Simultaneously, based on ongoing feedback from the DRM and the ECM, the EDM dynamically adjusts its core parameter settings to improve its performance. Candidate parameters include: Min-Sup (the minimum frequency a pattern must occur before being considered frequent); transaction length and sliding window size (the size and characteristics of the view of the data stream from which to learn patterns). The performance of the system is based on metrics such as time efficiency, memory efficiency, and accuracy of discovered pattern models, etc.

The EDM is made up of five sub-modules: Data Analyser, Mining Controller, Data Miner, Combiner, and Episode Analyser. These five sub-modules cooperate with each other to accomplish the job of the EDM by fulfilling their functions in turn. The flowchart of the EDM is presented in Figure 2.

1. Data Analyser: performance of pattern mining algorithms is directly dependent on proper parameter settings, such as Min-Sup, transaction length and sliding window size. To determine appropriate parameter settings is challenging task, especially for high-dimension context-heavy network monitoring data. Transaction length denotes the number of events in each transaction (set of events to be considered together). In some cases, in order to improve time efficiency of the algorithm, trace sub-streams can be further divided into subgroups for parallel processing. To enable parallel processing, sliding window size is defined by the number of transactions contained in the view of each sub-stream or sub-group to be considered at one time.

2. Combiner: After obtaining configuration information from the Data Analyser, the Mining Controller splits the reduced trace-stream into sub-streams or sub-groups and then assigns computation resources according to the configuration settings, such as I/O, memory and CPU, to each sub-stream or sub-group of the data. Next, the Mining Controller recursively calls the Data Miner to process the dataset in different sliding window views over sub-streams or subgroups in order to discover frequent episodes based on parameter settings provided by the Data Analyser.

3. Data Miner: As the core sub-module of the Episode Discovery Module, the Data Miner is called recursively to discover episodes from each sliding window view following a specific approach according to its corresponding parameter settings. In this model, both PrefixSpan [5] and HTPM [13] algorithm implementations can be selected to handle this Data Miner task.

5. Episode Analyser: Finally, the Episode Analyser sub-module adopts an association rule mining approach to further identify episodes by analysing the results to filter out most of the redundant episodes that have little no context or importance for the given use case. The identified episodes will be output and stored into the Pattern Model Library (PML) as pattern models.

IV PATTERN DISCOVERY IMPLEMENTATION

In order to achieve the target of pattern discovery, one significant step is to discover frequent sequential patterns and identify interesting pattern models from a network monitoring dataset in order to build the PML.

Sequential pattern mining is a rich and varied research domain, where many of the fundamental advances were proposed from 1995 to 2007 [17]. In the literature, classic algorithms are generic with
respect to application domain type, with little requirement for a priori expert domain knowledge about the dataset (once the dataset has been adequately filtered and prepared). Two such algorithms are PrefixSpan [5] and HTPM [13]. By contrast, more recent research in sequential pattern mining has focused more on incorporating constraints or application requirements in the mining process for better performance, driven by specific a priori domain expert knowledge about the data [17]. However, as mentioned in Section 1, the management dataset in heterogeneous networks usually involves many communication protocols and consists of numerous different types of the data (e.g. netFlow data, routing data, latency, throughput, syslog data, trouble ticket, traffic traces, event-based statistics, etc.), where the goal is to find complex correlations in data from these diverse data sources. As such it is very difficult to obtain a priori knowledge covering all candidate data sources in order to accurately steer mining of sequential patterns. For this reason a generic type of sequential pattern mining algorithm is a good initial option of pattern discovery in this work. For this work PrefixSpan and HTPM were selected and implemented as our candidate algorithms in the model.

For convenience, an integration GUI (Graphical User Interface) is provided to users based on the implementations of PrefixSpan and HTPM. In this Graphical Interface shown as Figure 3 and Figure 4, two parameters are needed as input before running the algorithms over a dataset: window size of transaction (Transaction length) and Min-Sup (Minimum Support). These two parameters can significantly influence the final output. Specifically, Min-Sup has a major effect on the number of the discovered patterns and resource consumption (especially time requirements), but also directly influences the support value and maximal length of each discovered pattern in the output.

After inputting the parameters, selecting the algorithm and choosing one specific dataset, the discovered pattern set and its relevant analysis result consisting of Support, Confidence and Lift with a specific format can be determined. The final outputs of algorithms are shown in Figure 5.

Figure 3: Graphical Interface for PrefixSpan and HTPM.

Figure 4: Graphical Interface for PrefixSpan and HTPM.

V EXPERIMENTS

In this section, the performance of PrefixSpan is presented and compared with HTPM based on two different synthetic dataset. PrefixSpan and HTPM are implemented in Java. All the experiments are performed on a 2.5GHz Inter(R) Core(TM) i5-3210M PC machine with 8 GB main memory, running Microsoft Windows 7.

a) The testing dataset

The testing dataset used in our experiment is generated by a synthetic data generator developed in house, which aims to synthesize complex event-based pattern data by setting some pattern specifications as inputs.

Table 1: The configuration of “PG_Syn” dataset.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of patterns</td>
<td>3</td>
</tr>
<tr>
<td>Average pattern generation rate</td>
<td>0.2</td>
</tr>
<tr>
<td>Average number of events in patterns</td>
<td>4.3</td>
</tr>
<tr>
<td>Number of event noise types</td>
<td>500</td>
</tr>
<tr>
<td>Average event noise generation rate</td>
<td>0.05</td>
</tr>
<tr>
<td>Total number of events</td>
<td>100K</td>
</tr>
</tbody>
</table>

For any generated dataset, the pattern specification defines how events within a pattern are
correlated to each other on the timeline, by way of specifying the probability distribution of times between constituent events in each pattern type. Additionally, random noise events can be merged into the data stream along with the patterns. These inserted noisy events are not correlated to each other nor with events in the patterns. The relative proportion of noise to the rate of pattern generation can be controlled by a user specified parameter setting. For clarity, this dataset is referred to “PG_Syn” and its configuration is listed in Table 1. There are a total of 500 + 12 different types of events, incorporating 500 noise event types [100 … 599] and 12 information event types for use in patterns [0 … 11]. Three candidate sequence patterns are manually pre-defined for inclusion in the synthetically generated dataset: {0→1→2→3→4→5}, {5→6→7→8} and {9→10→11}.

b) Performance evaluation

PrefixSpan and HTPM are tested for performance comparison using the synthetically generated dataset (“PG_Syn”), where the synthetically generated events are considered as hybrid-based events. Time consumption is one of the main evaluation criteria as shown in Figure 6. For simplicity, the Min-Sup is incrementally set from 0.35 to 0.8 (35%-80%) while transaction length is fixed to 20. In the presented results it can be seen that as the Min-Sup decreases, then more candidate sequences are considered “frequent”; and so must be completely analysed, thereby increasing the resource requirements of for the system.

Figure 6: Time consumption comparison between PrefixSpan and HTPM (Min-Sup).

Figure 7 shows the transaction length increased from 20 to 50, where Min_Sup is fixed at 0.8. This increase of the resource usage as the transaction length increases is due to the situation where a longer transaction is more likely to contain more candidate patterns than a short transaction. Figure 6 and Figure 7, demonstrates that there is a big gap between the performances of PrefixSpan and HTPM, with PrefixSpan scaling more favourably that HTPM.

In addition to the resource usage comparisons of PrefixSpan and HTPM, it is noted that both algorithms effectively and accurately found all simulated patterns in the synthetically generated dataset for correctly set Min-Sup values.

c) Analysis

The experiments show that PrefixSpan and HTPM can effectively discover the patterns from datasets similar to the synthetically generated dataset (“PG_Syn”). However, compared to PrefixSpan, the time consumption of HTPM is much higher. After investigating the specific processing steps of PrefixSpan and HTPM in-depth, it can be recognized that the gap between their time requirements is mainly caused by their different internal approaches to “learn” or “grow” pattern candidates, where this learning process is the most significant and resource intensive processing step for PrefixSpan and HTPM. Typically, 70% to 80% CPU and Memory resources are assigned to perform this step.

VI CONCLUSIONS

Pattern discovery in this project requires a practical and powerful algorithm for discovering significant sequential event patterns from Telecom network management datasets. In order to achieve this goal, a novel pattern discovery model was proposed in this paper. This model is composed of five sub-modules including Data Analyser, Mining Controller, Data Miner, Combiner and Episode Analyser. These five sub-modules cooperate with each other and accomplish the job of episode discovery by fulfilling their functions in turn. After reviewing appropriate state-of-the-art data mining techniques, PrefixSpan and HTPM were selected as good candidates of the Data Miner sub-module. Either algorithm can be considered as a generic algorithm suitable for complex heterogeneous network circumstances. Their performance was compared in terms of different parameter settings by running algorithms over a synthetically generated test dataset. In this work the dataset is streamed from the Dimension Reduction Module (DRM) and
exports discovered pattern models into the Pattern Model Library (PML) database. This PML is in turn used by the Pattern Matching & Prediction Module (PMM) to drive the adaptive action recommendation module (RSM). Together these modules will form an end-to-end automated network management system.

ACKNOWLEDGEMENTS

This work was funded by Enterprise Ireland Innovation Partnership Programme with Ericsson Ireland under grant agreement IP/2011/0135.

REFERENCES


