Pattern Detector: Fast Detection of Suspicious Stream Patterns for Immediate Reaction

Ira Assent
Department of Computer Science
Aalborg University, Denmark
ira@cs.aau.dk

Hardy Kremer Stephan Günnemann Thomas Seidl
Data management and data exploration group
RWTH Aachen University, Germany
{kremer, guennemann, seidl}@cs.rwth-aachen.de

ABSTRACT

Detecting emerging problems in information and manufacturing systems is the goal of monitoring tools. Good and timely detection of problematic conditions from measured indicators requires efficient and effective detection of critical patterns in a stream of incoming observations.

We present Pattern Detector, an interactive system which is capable of immediate detection and signaling of such patterns. Using user-defined query patterns which indicate e.g. low rate denial-of-service attacks in network traffic, this system signals problems fast and transparently.

The underlying detection algorithm is based on matching patterns using the Dynamic Time Warping (DTW). Fast query processing is achieved by reliably filtering out candidates via a highly efficient multistep filter-and-refine framework, anticipatory DTW (ADTW). This framework is capable of processing continuous streams such that appropriate action can be taken as soon as suspicious patterns occur.

While our pattern detector system is developed specifically for network traffic by incorporating recent patterns from computer networking, it easily generalizes to many online stream monitoring tasks.

Categories and Subject Descriptors

H.2.8 [**Information Systems**]: Database Management Database applications

General Terms

Pattern detection, time series, efficient query processing

1. INTRODUCTION

In many application domains, production and information systems have to be routinely monitored. For example, temperature and humidity of a manufacturing laboratory, or incoming and outgoing network traffic in a computing environment are important for surveillance of vital company assets. Automatically detecting critical situations as they arise is crucial for appropriate counter-action in time. As networked information systems are increasingly vulnerable to security attacks from outside, detecting e.g. denial-of-service attacks before they bring down the system greatly reduces the cost of restoring the system and bringing it back up to regular operation. Similarly, monitoring production systems and observing critical patterns in the development of production conditions allows corrective measures before damages or quality degradation occur.

Successful monitoring should be capable of effectively identifying patterns in streaming data which might indicate critical events. For most systems, but especially for highly dynamic systems, or those systems which require following a large number of measurements, and / or several stream sources, efficiency of the detection is crucial. Especially if several patterns for different problematic scenarios are monitored, they have to be compared against the stream in a highly efficient manner. Moreover, patterns may be of arbitrary length. And finally, from a user's point of view, the faster patterns are detected, the easier counter measures can be taken.

In short, our pattern detector system has to meet the following requirements:

- **effective** pattern detection
- **efficient** online processing
- easy interactive **configuration** of patterns and sources

In this work, we propose a system for monitoring continuous streams of data, and for immediate signaling of critical situations. The system visualizes the incoming information stream(s), the critical pattern(s) that need to be detected, and highlights the state of the system using an expressive color code and sound signals for alerts. Users may interactively choose streams to monitor, set up sensitivity thresholds, as well as define and modify critical patterns.

The **effectivity** of the pattern detection is based on the widely used Dynamic Time Warping Distance (DTW) measure, which is known to successfully identify patterns in time series in a variety of application domains [4, 1, 5, 9, 6, 2]. As DTW computation is costly, we achieve **efficient** pattern detection by employing a fast multistep filter-and-refine algorithm, anticipatory DTW (ADTW) [3]. ADTW reduces the number of necessary computations by faster rejection of false candidates.

In our demonstration scenario, we exemplarily show how network traffic can be monitored to identify threats such as low rate TCP attacks, for which recent research result in the

EDBT 2010, March 22–26, 2010, Lausanne, Switzerland.

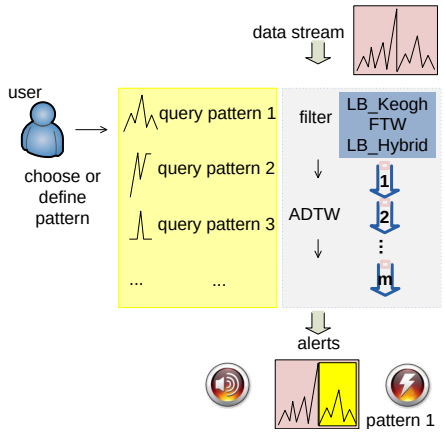

Figure 1: Pattern detector workflow: the user specifies query patterns which are continuously checked against the data stream using DTW lower bounds and ADTW for efficient processing. Detected patterns results in visual and audio alerts and highlighted areas in the stream.

network community use DTW-based monitoring [12]. As the data continuously arrives during the demonstration, efficient detection and signaling of problematic traffic patterns can be observed. Detailed information on monitoring, continuous query processing, parameterization and evaluation will be available for interactive **configuration** and usage.

Our pattern detector easily generalizes to other online monitoring applications, e.g. monitoring of production processes where detection of critical patterns can be used for corrective measures before problems occur.

2. SYSTEM ARCHITECTURE

Monitoring network traffic requires an online algorithm for efficient and effective detection of the relevant patterns and their immediate signaling to the user. Our pattern detector mechanism builds upon two main building blocks: an effective pattern detection technique for low rate denial-of-service (DoS) attacks as in [12], and an efficient algorithm for the computation of DTW (Dynamic Time Warping) [3].

The overall workflow is illustrated in Figure 1. The pattern detector can be configured to fit the needs of the user by loading predefined patterns (e.g. for low rate DoS attacks as defined in [12]), or by manually defining a pattern that describes potentially suspicious time series. For each pattern, sensitivity thresholds are adjustable. These sensitivity thresholds corresponds to the similarity between the query pattern and the traffic stream. By adjusting them, users can opt for earlier alerts, which might induce more false alarms, or for a more tolerant processing where only very similar patterns lead to an alert. Likewise, for Dynamic Time Warping, the degree of stretching and scaling of the pattern along the time axis, can be set by the user (see also technical discussion in the query processing section 3). This allows direct parametrization of how close the matching between query and stream should be.

The continuous data stream is monitored for the pattern(s) and their parameterization(s) that the user has specified. We apply an efficient filtering scheme [3] to substantially reduce the runtime.

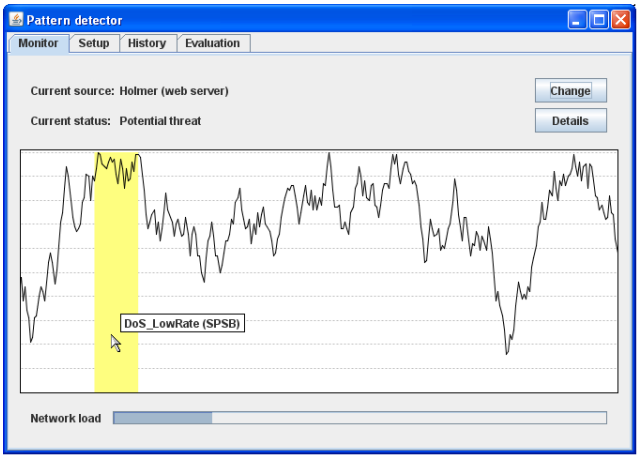

Figure 2: Monitoring screen with alert information: displays the currently observed data stream source, its threat status, as well as the life stream situation.

If a suspicious pattern is detected, an audio alarm is played, and the corresponding area in the stream is highlighted. Figure 2 shows the corresponding screen. A mouse click on the highlighted area shows the patterns that have caused the alert. The currently monitored source and the threat level are indicated. As the stream continues in the main monitoring window, the history tab allows users to go back to any suspicious pattern by navigation back and forth along the stream to inspect the entire stream, or just the highlighted areas. Additional evaluation information that comprises statistical information on past detection is provided for further in-depth analysis.

We provide several tools for statistical analysis of detected patterns, both for the ongoing streamed data, and for historical data. Figure 3 shows the overview screen. Users may choose which time periods they would like to compare. Statistical information on the detected patterns and their threat level is made available. Moreover, aggregated views on the stream are available at different levels of granularity such that users can gain both an overview and take a look at details of any monitored screen.

User may additionally choose to open several monitoring windows for different sources. This allows displaying the information for a number of observed entities side-by-side. This feature is also helpful to compare the effect of different parameterization or query pattern setups as the stream continues.

The setup screen is displayed in Figure 5: a pattern can be added to the current monitoring routine by loading it from a file, or by creating a new pattern. To avoid a large number of detected patterns, relevant thresholds can be selected. The degree of stretching and scaling in DTW can be adjusted from the default values to user defined parameter values. For advanced users, we offer the possibility of choosing different DTW processing algorithms (see Section 3 below for details).

3. QUERY PROCESSING

Dynamic Time Warping (DTW) is a distance measure that was originally developed in speech recognition to overcome the shortcomings of the Euclidean distance. Instead of simply comparing values in two time series one value at

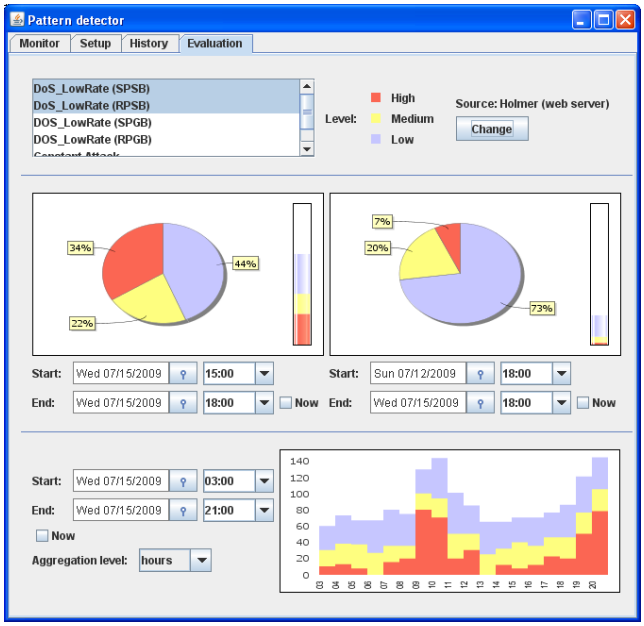

Figure 3: Statistics screen for (historical) analysis: allows for in-depth analysis of a specified data stream source for different patterns and periods of time, aggregated to the desired time granularity level.

a time (as in Euclidean Distance), DTW allows for stretching and scaling along the time axis to find the best possible match between the two patterns (cf. Fig. 4). To avoid degenerate matchings, the amount of stretching is usually restricted to a band of k values. The best match DTW distance is defined recursively over time series of shorter length:

DEFINITION 1. ***k-band DTW.***
The Dynamic Time Warping distance between two time series s,t of length n,m (w.l.o.g. $n \leq m$) with respect to a bandwidth k is defined as:

$$DTW([s_1,...,s_n],\ [t_1,...,t_m]) =$$

$$dist_{band}(s_n,\ t_m) + \min \begin{cases} DTW([s_1,...,s_{n-1}],\ [t_1,...,t_{m-1}]) \\ DTW([s_1,...,s_n],\ [t_1,...,t_{m-1}]) \\ DTW([s_1,...,s_{n-1}],\ [t_1,...,t_m]) \end{cases}$$

with

$$dist_{band}(s_i,\ t_j) = \begin{cases} dist(s_i,t_j) & |i - \lceil \frac{j \cdot n}{m} \rceil| \leq k \\ \infty & else \end{cases}$$

$$DTW(\emptyset,\emptyset) = 0, \quad DTW(x,\emptyset) = \infty, \quad DTW(\emptyset,y) = \infty$$

Thus, DTW between two time series s and t is recursively defined as the minimum over time series shorter by one element. The values at any given time point ($dist_{band}$) are taken into account, as long as their values are within the band constraint ($|i-j| \leq k$ for time series of equal length).

Dynamic time warping (DTW) can be computed using dynamic programming. Its computational complexity, however, is too high for many applications. As a consequence, speeding up DTW is an active research area. A number of approaches are based on the multistep filter-and-refine architecture [8, 13, 10]. Instead of naively comparing the

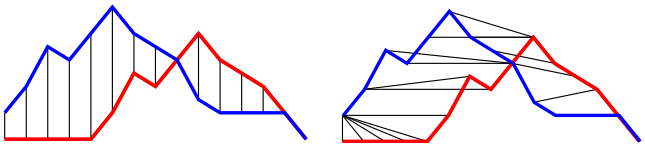

Figure 4: Euclid (left) vs. DTW (right): black vertical lines indicated how Euclide compares the top blue pattern to its shifted counterpart at the bottom (red), and how DTW matches corresponding parts in the patterns through stretching.

query against all possible time series, the idea is to define a filter distance for efficiently filtering out possible candidates. Only these - ideally few - candidates are then refined using the costly DTW distance. If the filter is lower bounding, i.e. it never overestimates the true DTW distance value, then multistep computation is lossless [7, 11]. Lossless query processing means that the final results corresponds to exactly the same as if the query had been naively compared to all time series, but runtime is reduced.

For online algorithms, lower bounding filters may still not be fast enough. They prune many time series, but many still have to be evaluated using the costly exact DTW computation. In our recent prior work, we have proposed a novel approach, called anticipatory pruning for the DTW distance (ADTW for short) [3]. It is based on the observation, that once a candidate has passed the lower bounding filter step, the DTW computation starts from scratch. ADTW re-uses the filter information to devise a novel pruning scheme during DTW computation.

DTW is cumulative, i.e. in each step of its computation, a matrix column with partial DTW matching results is filled. The minimum of the values per column is known to be monotonously non-decreasing. As a result, the column minima serve as a pruning step in what is known as early stopping or early abandon [10, 9].

We have shown that many existing lower bounding filters have a similar property, called piecewise, which complements the cumulative nature of DTW. ADTW tightens the DTW estimate by using a combination of column minima of DTW plus an estimate of the remaining columns that is derived directly from the previously computed filter step.

As a consequence, ADTW requires surprisingly little overhead. It needs to simply keep track of the piecewise filter information that is available anyhow, and combine it with column minima of DTW. As the cumulative DTW matrix is filled as in any standard DTW computation but for reversed (query) time series, the computational effort for ADTW is low, yet its pruning power has been shown to be greatly improved [3].

In this system, employing ADTW leads to the necessary efficiency gains that enable online monitoring via DTW. In our system, we provide implementations of the ADTW combined with state-of-the-art lower bounding filters (LB_Keogh, LB_Hybrid, and FTW) [8, 13, 10]. For those interested in the inner workings of the algorithms, we also provide standard implementations of these filters and a sequential DTW processing. In our real world scenario, the difference in runtimes that stems from the different query processing algorithms will be directly perceivable.

4. DEMONSTRATION SCENARIO

In the demonstration scenario, we will provide users with the opportunity to try out the monitoring tool themselves. With a number of predefined patterns and parameterizations that have been empirically set for both the sensitivity threshold and the tolerance in the DTW matching, we provide settings where the conference participants will be able to immediately get results.

Additionally, all of the settings, the patterns, and data streams can be modified by the attendees. In this way, it will be possible to observe the performance of the network monitoring and the detection of potential threats. For advanced users, it will be possible to compare different parameter settings, or different algorithmic approaches for DTW computation.

Based on the approach presented in [12], we define query patterns that correspond to potential low rate denial-of-service attacks. These attacks are visible in regular network traffic as occasional bursts that are periodically observable. Bursts in traffic that exceed the victim's capacity lead to a loss of regular network traffic. This traffic cannot be serviced as the attacker's data floods the victim's system for a short period of time.

The authors provide a formalization of a family of such low rate attacks. Based on samples of regular network traffic, query patterns are determined. To account for the natural differences in the query patterns of periodic attack signatures and the actual network traffic, the authors use DTW-based matching. This is followed by techniques for narrowing down the source of the attack.

Our demonstration system not only provides the means for monitoring, but additionally contains analytical tools for evaluating the history of detected patterns. We also include extensive information on the DTW query processing. Different algorithmic approaches can be seen in direct comparison. Moreover, while we provide empirically tested default parameterization settings, we allow users to set their own values to study their effect on the current data stream.

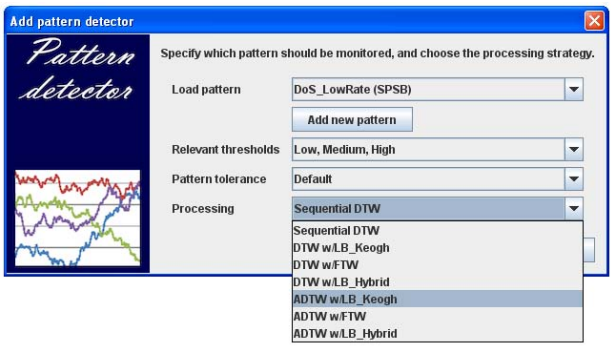

Figure 5: Setup dialog for patterns: allows loading or specifying of patterns, and setting the desired level of sensitivity via the alert threshold and its tolerance via the DTW band. Additionally, the underlying query processing algorithm can be chosen.

5. CONCLUSION

The pattern detector is an online monitoring tool for continuous stream data as arises e.g. in early detection of network attacks in regular network traffic. We provide query patterns which correspond to DTW-based detection of low rate denial-of-service attacks as recently proposed in the network community. We achieve efficient query processing by using our anticipatory pruning scheme for fast DTW computation. In the demonstration scenario, the setup is fully configurable by the user, and we provide tools for thorough analysis.

Acknowledgment

This work has been supported in part by the UMIC Research Centre, RWTH Aachen University, Germany.

6. REFERENCES

[1] J. Aach and G. M. Church. Aligning gene expression time series with time warping algorithms. *Bioinformatics*, 17(6):495–508, 2001.

[2] I. Assent and H. Kremer. Robust adaptable video copy detection. In *Symposium on Spatial and Temporal Databases (SSTD)*, pages 380–385. Springer, 2009.

[3] I. Assent, M. Wichterich, R. Krieger, H. Kremer, and T. Seidl. Anticipatory dtw for efficient similarity search in time series databases. *Int. Conf. on Very Large Databases (PVLDB)*, 2(1):826–837, 2009.

[4] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *AAAI Workshop on KDD*, pages 229–248, 1994.

[5] A.-P. Chen, S.-F. Lin, and Y.-C. Cheng. Time registration of two image sequences by dynamic time warping. In *Int. Conf. on Networking, Sensing and Control (ICNSC)*, pages 418–423, 2004.

[6] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh. Querying and mining of time series data: Experimental comparison of representations and distance measures. *Int. Conf. on Very Large Databases (PVLDB)*, 1(2):1542–1552, 2008.

[7] C. Faloutsos. *Searching Multimedia Databases by Content*. Kluwer Academic Publishers, 1996.

[8] E. J. Keogh. Exact indexing of dynamic time warping. In *Int. Conf. on Very Large Databases (VLDB)*, pages 406–417, 2002.

[9] E. J. Keogh, L. Wei, X. Xi, S. Lee, and M. Vlachos. LB_Keogh supports exact indexing of shapes under rotation invariance with arbitrary representations and distance measures. In *Int. Conf. on Very Large Databases (VLDB)*, pages 882–893, 2006.

[10] Y. Sakurai, M. Yoshikawa, and C. Faloutsos. FTW: fast similarity search under the time warping distance. In *Symposium on Principles of Database Systems (PODS)*, pages 326–337, 2005.

[11] T. Seidl and H.-P. Kriegel. Optimal multi-step k-nearest neighbor search. In *Special Interest Group on Management of Data (SIGMOD)*, pages 154–165, 1998.

[12] H. Sun, J. Lui, and D. Yau. Defending against low-rate TCP attacks: Dynamic detection and protection. In *Int. Conf. on Network Protocols (ICNP)*, pages 196–205, 2004.

[13] M. Zhou and M. H. Wong. Boundary-based lower-bound functions for dynamic time warping and their indexing. In *Int. Conf. on Data Engineering (ICDE)*, pages 1307–1311, 2007.