Discovering Flow Anomalies: A SWEET Approach

James M. Kang[1] Shashi Shekhar[1] Christine Wennen[2] Paige Novak[3]

[1]Department of Computer Science,
[2]Graduate Program in Water Resources Science, [3]Department of Civil Engineering,
University of Minnesota, MN, USA
[1]{jkang, shekhar}@cs.umn.edu, [2,3]{wenne052,novak010}@umn.edu

Abstract

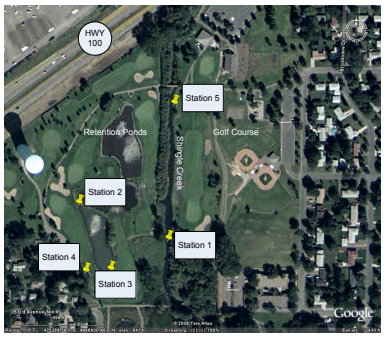
Given a percentage-threshold and readings from a pair of consecutive upstream and downstream sensors, flow anomaly discovery identifies dominant time intervals where the fraction of time instants of significantly mis-matched sensor readings exceed the given percentage-threshold. Discovering flow anomalies (FA) is an important problem in environmental flow monitoring networks and early warning detection systems for water quality problems. However, mining FAs is computationally expensive because of the large (potentially infinite) number of time instants of measurement and potentially long delays due to stagnant (e.g. lakes) or slow moving (e.g. wetland) water bodies between consecutive sensors. Traditional outlier detection methods (e.g. t-test) are suited for detecting transient FAs (i.e., time instants of significant mis-matches across consecutive sensors) and cannot detect persistent FAs (i.e., long variable time-windows with a high fraction of time instant transient FAs) due to a lack of a pre-defined window size. In contrast, we propose a Smart Window Enumeration and Evaluation of persistence-Thresholds (SWEET) method to efficiently explore the search space of all possible window lengths. Computation overhead is brought down significantly by restricting the start and end points of a window to coincide with transient FAs, using a smart counter and efficient pruning techniques. Experimental evaluation using a real dataset shows our proposed approach outperforms Naïve alternatives.

1. Introduction

Motivation. Mining flow anomalies (FA) is important in several spatio-temporal application domains such as environmental monitoring networks for detection of potential flood conditions, chemical spills, and/or pollutants entering river networks. Maintaining sufficient water of high quality for the world population is one of our greatest global challenges [14]. Several recent articles from the popular press report that many dangerous contaminants are entering our water from unknown sources and at unknown times, resulting in expensive experimental investigations (e.g. [11]).

Currently, hydrologists and environmental engineers are placing advanced sensors in water bodies around the United States to understand the behavior of river networks and lakes [10]. Figure 1a shows a satellite image of Shingle Creek, MN where there are five sensors monitoring various water quality parameters [6] (e.g., turbidity, dissolved oxygen, specific conductivity, nitrate levels, etc.). Two sensors (water flow from 5 to 1) are placed in the creek and three others are placed in neighboring ponds. Figure 1a illustrates that the water from neighboring ponds can flow into the creek between sensors 5 and 1. Because of the large amount of data collected continuously, a pollutant entering the river between two monitoring sensors could easily go unnoticed. A robust method for identifying such a contamination event and pin-pointing the time intervals at which it occurred while minimizing false alarms could vastly improve not only warning systems but the ability of scientists and engineers to understand the cause of the event.

Problem Statement. Given pairs of time-series of measured variables from consecutive upstream and downstream sensors, flow anomaly discovery flags pollution events occurring between the sensor pair by finding time-periods with a (user-defined) high fraction of time-instants having significantly different readings across consecutive upstream and downstream sensors. In the absence of contamination between sensors, the observations are similar across both upstream and downstream sensors. If a phenomenon is seen only at one sensor and not the other, a transient flow anomaly has occurred. A single persistent flow anomaly may consist of several transient flow anomalies with a few time-instants without flow-anomalies. A persistent flow anomaly is dominant (e.g., entire oil spill event) if its time-interval is not a subset of time intervals of any other persistent flow anomaly. For example, Figure 1b shows one of the largest oil spills in San Francisco, CA history in November of



(a) Shingle Creek, MN (Source: Google Maps)

(b) Oil Spill in San Francisco (Courtesy: [7])

Figure 1. Study Site and Motivational Example (Best Viewed in Color)

2007. As can be seen, the spill does not flow as a single unit; rather there are several gaps within the spill [7].

Challenges. Mining FA patterns is computationally challenging for several reasons. First, a single persistent FA pattern may consist of subsets that may or may not be anomalies. Second, the size of the time-interval for the longest dominant persistent FAs may not be known in advance. Third, the temporal length of each persistent FA pattern may be different. Fourth, the match between the observations at the upstream and downstream sensors may be one-to-one, one-to-many, or many-to-many. Finally, the datasets of time instants may be potentially infinite in size.

Related Work. Studies related to the discovery of flow anomalies may be categorized into two groups: string matching and data stream correlations. In string matching, relationships are created based on the similarity or dissimilarly between two strings. Amir et al. proposed an inverse string matching technique that finds a pattern between two strings that maximizes or minimizes the number of mismatches [1]. Lee et al. proposed a similar method to inverse pattern matching that included wild cards [9]. There are several main differences between string matching and the discovery of FA patterns. First, several string matching techniques use an exact matching technique between multiple strings, whereas the FA problem needs a statistical measure because an exact match may not exist between data streams. Second, string matching uses a discrete alphabet whereas multiple time series analysis uses a continuous alphabet.

In data streams, relationships are made using a fixed sliding window and a correlation measure. Chan et al. found local correlations between multiple data streams using a sliding window [3]. Sayal found global relationships between data streams and a single siding window to summarize a single data stream [12]. Balut introduced an incremental approach to find correlations of a query for multiple predefined time windows [2]. Datar and Muthukrishnan identified rarity and similarity between data streams using a fixed window size [4]. These approaches focus on finding correlations using a fixed size sliding window and some correlation measure. Using a fixed window size implies that the domain specialist knows the duration of the anomalous events which may not be known in environmental systems (e.g. rain events). Correlation measures are used to find associations between data streams whereas we propose a statistical interest measure to find interesting relationships.

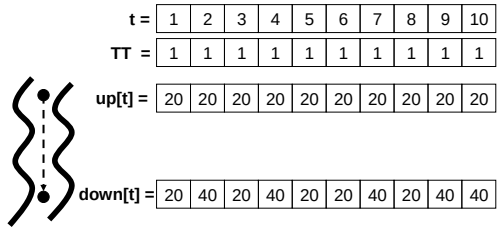
To some extent, basic outlier detection techniques (e.g. t-test [5]) detecting transient FAs may discover transient and some persistent FAs (with a 100% mismatched time-interval) (e.g. [8, 13]). These methods are not designed to detect persistent FAs with less than 100% mismatched time-instants and may miss many persistent flow anomalies.

Contributions. In this paper, we propose a Smart Window Enumeration and Evaluation of persistent-Thresholds (SWEET) approach that utilizes several key insights into this problem to efficiently identify persistent FAs between upstream and downstream sensors. First, to reduce the the search space for different sized windows, the space of interesting persistent FAs is constrained to start and end at transient FAs, i.e., time-instants with significantly different measurements. Second, a smart counter is used to reduce the computation required to evaluate a candidate time-interval for the interest measure, from a linear function of time-interval-size to a constant function. Finally, a pruning strategy is used to reduce the number of persistent FAs (pFAs) to discover the dominant pFAs (dpFAs). We experimentally evaluate our proposed method using a real dataset.

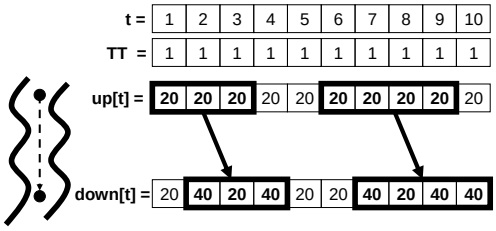
The rest of the paper is organized as follows. Section 2 presents the basic concepts to provide a formal model of FAs and the problem statement of discovering FAs. Section 3 presents our proposed SWEET method. Section 4 gives the experimental evaluation and Section 5 concludes the paper and discusses future work.

2. Basic Concepts and Problem Statement

In this section, we first introduce several key concepts to model Flow Anomalies (FAs), and then we give a formal problem statement. Figure 2, referenced throughout this section, illustrates an input and output example of discovering FAs. The input consists of time series of 10 time instants for the upstream and downstream sensors with a constant travel time (TT) of 1 for simplicity. The output contains two dominant persistent FAs; using the time instants at the upstream sensor, the periods are 1-3 and 6-9.



(a) FA Example Input



(b) FA Example Output

Figure 2. Flow Anomaly Discovery Example

2.1 Basic Concepts

This section presents several relevant definitions to our problem statement and proposed approaches.

Definition 1 *An observation o is a measured reading of a variable at every time instant t at the upstream up and a downstream $down$ sensor.*

Figure 2a gives an up sensor (e.g., Station 5 in Figure 1a) having an observation of 20 at time-instants 1,2,. . .,etc. and a $down$ (e.g., Station 1 in Figure 1a) sensor having an observation of 40 at time instant 2.

Definition 2 *Travel Time, TT, is the duration between the time (t) when an observation o is made at the up sensor and the time (t+TT[t]) of the corresponding observation at the down sensor.*

Figure 2a gives an example where the travel time (TT) is 1. For time instant 2, up[2] and down[2+1] have corresponding observations of 20 and 20 respectively.

Definition 3 *An Instant Pair, IP, is two observations of a common phenomenon made at the up and $down$ sensors having a temporal length of TT between them.*

Definition 3 can be formally expressed in Equation 1.

$$IP[t] = (up[t], down[t + TT]) \tag{1}$$

Figure 2a gives an example where the instant pair (IP) at time instant of 1 is IP[1]=(20,40) where the TT=1. We note that this definition assumes a very small temporal footprint for the phenomenon of interest at each sensor. This assumption is made to simplify the discussion in this paper. It is revisited in the last section on future work.

Definition 4 *An upstream contiguous set of instant pairs are IPs that occur one right after the other for some temporal length k.*

Definition 4 can be formally expressed in Equation 2.

$$S = \{< up[t+i], down[(t+i)+TT[t+i]] > | i \in 0, \ldots, k\} \tag{2}$$

Figure 2a gives an example of an upstream contiguous set of instant pairs for period 1-3 containing $\{< 1, 2 >, < 2, 3 >, < 3, 4 >\}$ where the travel time, TT, is 1.

Definition 5 *A transient Flow Anomaly, tFA, is when the difference between corresponding observations of each sensor in the IP is larger than the given error threshold, Θ_e.*

The term "flow" refers to anomalies found between the flow of two sensors (e.g. sensors 5 and 1). Definition 5 can be formally expressed in Equation 3.

$$tFA[t] \iff (|up[t] - down[t + TT]| > \Theta_e) \tag{3}$$

Suppose the error threshold is zero (or 10 or 15); then Figure 2a gives an example of a flow anomaly at time interval 1 where the IP[1]=(20,40) because the difference in observations is greater than zero.

Definition 6 *A persistent Flow Anomaly, pFA, is an upstream contiguous set of IPs where a high fraction ($\geq$ persistent threshold Θ_p) are transient flow anomalies. Also, the start s and end e IP in a pFA must be a transient flow anomaly, tFA.*

Definition 6 can be formally expressed in Equation 3.

$$pFA[s \ldots e] \iff (tFA[s]) \;\&\; (tFA[e]) \;\&\; \left(\frac{\sum_{i=s}^{e} tFA[i]}{(e-s)+1} \geq \Theta_p\right) \tag{3}$$

Figure 2b gives an example of a pFA pattern when the persistent threshold is 0.6 and the error threshold is 0. The period range of 1 to 3 time intervals of the up sensor is a pFA pattern where the first and ending IPs are transient flow anomalies and the interest measure has a value of 2/3, which satisfies the persistent threshold. The reason that the up time intervals from 1 to 9 is not a pFA is that it does not satisfy the persistence threshold of 0.6 (i.e., five transient flow anomalies by the temporal length of 9 is less than 0.6).

Definition 7 *A dominant persistent Flow Anomaly, $dpFA$, is a pFA that is not a subset of any other $dpFA$.*

Figure 2b gives an example of a $dpFA$ pattern as shown by the up sensor time intervals from 1 to 3 and 6 to 9. 6 to 9 is dominant because there are smaller $pFAs$ such as 8 to 9 and it is not a subset of the other $dpFA$ from 1 to 3.

2.2 Problem Statement

The FA discovery problem can be expressed as follows:
Given: (1) An upstream, up, and a downstream, $down$, sensor, (2) Direction of flow between the up and $down$ sensor, (3) An upstream contiguous set of Instant Pairs, IP at time intervals $t = 1...n$, where n is the length of the time series for the up sensor, (4) The travel time, $TT[t]$, between the up and $down$ sensors at every t, (5) An error threshold, Θ_e, and (6) A persistent threshold, Θ_p.
Find: All dominant persistent Flow Anomalies (dpFAs).
Objective: Minimize the computational costs.
Constraints: A single directional flow between sensors and a single one-to-one match between observation upstream and downstream.

Example. Figure 2a gives an example of an input time series from the up and $down$ sensors where the travel time is the temporal length when an observation is expected to be seen once at each sensor. Figure 2b gives an example output of the types of FA patterns when the error threshold is zero and the persistence threshold is 0.6. The dpFAs are the time instants at the up sensor of 1 to 3 and 6 to 9. The dpFA of 1 to 3 satisfies the pFA condition as follows: (1) the ratio of two anomalies and the period length of three is greater than 0.6 and (2) the first and last IP is a tFA. Likewise, the dpFA of 6 to 9 also satisfies the pFA condition as follows: (1) the ratio of three anomalies and the period length of 4 is greater than the persistence threshold of 0.6 and (2) the first and last IP is a tFA. Also, periods 1-3 and 6-9 are not subsets of each other satisfying Definition 7.

3. Mining Flow Anomalies

In this section, we first discuss the Naïve method and then propose a novel Smart Window Enumeration and Evaluation of persistence-Thresholds (SWEET) method to mine dominant persistent flow anomalies.

3.1 Naïve Method

This section presents the Naïve method to find all the dominant persistent Flow Anomalies (dpFAs). The Naïve method has two main phases, namely, the *identification of the persistent FAs* and *identification of the dominant pFAs*. The main idea of the first phase is to exhaustively search through the entire dataset for every possible size window and determine if each window satisfies the persistence threshold, Θ_p, based on the number of transient Flow Anomalies (tFAs) found within the window using the error threshold, Θ_e. In the second phase, the pFAs having the largest temporal length are kept as the dpFAs, while any other pFA that is a subset of the answers is removed and the final result is returned.

Algorithm 1 Pseudo code for the Naïve Method

Inputs:
- up and $down$ time series of o at $t = 1 \ldots n$
- Travel Time, TT, of o at $t = 1 \ldots n$
- Error Threshold, Θ_e
- Persistence Threshold, Θ_p

Outputs:
- Dominant Persistent Flow Anomalies (dpFA)

Algorithm

{**Phase I**: Identity pFAs}
1: *tFA_count* = 0
2: *pFA* $\leftarrow \emptyset$
3: **for** each window size $i = 1 \ldots n$ **do**
4: **for** each shift of window $j = 1 \ldots n$ **do**
5: **for** each element k in window of size i **do**
6: **if** (window exists in both up and $down$) AND (diff(up[k],down[k+TT[k]]) $> \Theta_e$) **then**
7: *tFA_count*++
8: **end if**
9: **end for**
10: **if** (first and last IP is a tFA) AND ((tFA_count/i) $\geq \Theta_p$) **then**
11: *pFA* $\leftarrow$ period j to $j + i$
12: **end if**
13: *tFA_count* = 0
14: **end for**
15: **end for**
{**Phase II**: Identify Dominant pFAs}
16: dpFA $\leftarrow \emptyset$
17: **while** pFA set size $\neq \emptyset$ **do**
18: Identify pFA c with largest temporal length
19: Remove c from pFA
20: dpFA $\leftarrow c$
21: Remove any pFA that is a subset of c
22: **end while**
23: **return** *dpFA*

Algorithm 1 presents the pseudo code for the Naïve method. The inputs of the Naïve method include the following: (1) up and $down$ time series of a single observation o at $t = 1 \ldots n$, where n is the number of observations in each time series, (2) the Travel Time, TT, of o at $t = 1 \ldots n$, (3) the error threshold, Θ_e, and (4) the persistent threshold, Θ_p. The output of the Naïve method consists of the dpFAs in terms of the start and end time intervals for the up sensor.

Phase I: Identify Persistent FAs. This phase is concerned with identifying all the possible windows between the up and $down$ time series that satisfy the *pFA* definition (Definition 6). Initially, the counter for the number of tFAs in a window is set to zero and the set pFA is set empty (Lines 1-2 of Algorithm 1). An exhaustive search is completed by examining each window size to the length of the

Algorithm 2 Pseudo code for the SWEET Method

(Inputs and Outputs are the same as Naïve)
{**Phase 1**: Identity pFAs}
1: *tFA_count* = 0
2: *tFA* $\leftarrow \emptyset$
3: *pFA* $\leftarrow \emptyset$
4: **for** each o at $i = 1 \ldots n$ **do**
5: **if** (diff(up[i],down[i+TT[i]]) $> \Theta_e$) **then**
6: *tFA* $\leftarrow$ i
7: **for** each *tFA*, j = 1. . . tFA.size() **do**
8: **for** each element k in window $tFA[j]$ to i **do**
9: **if** diff(up[k],down[k+TT[k]]) $> \Theta_e$ **then**
10: *tFA_count*++
11: **end if**
12: **end for**
13: **if** (*tFA_count*/i) $\geq \Theta_p$ **then**
14: *pFA* $\leftarrow$ period j to $j + i$
15: **end if**
16: **end for**
17: *tFA_count* = 0
18: **end if**
19: **end for**
{**Phase II**: Identify Dominant pFAs (Identical to Naïve's Phase II}

time series and then shifting each window size through the entire series (Lines 3-4 of Algorithm 1). As each window size slides throughout the time series, each instant pair is checked against Θ_e (Lines 5-6 of Algorithm 1). If a tFA is found, then the counter (tFA_counter) in this window is incremented (Line 7 of Algorithm 1). Once each instant pair in the window of size i is checked for a transient flow anomaly, then the method determines whether the window satisfies the *pFA* definition and if it does, the period is added to the *pFA* set (Lines 10-12 of Algorithm 1). This process continues until all possible windows have been analyzed.

Phase II: Identify Dominant pFAs. This phase is concerned with identifying all dominant pFAs from the *pFA* set found in Phase 1 with the largest temporal length such that it is not a subset of any other *dpFA* (Definition 7). Initially, the *dpFA* is set to empty (Line 16 of Algorithm 1). The *pFA* with the largest temporal length is found, removed from the *pFA* set, and added to the *dpFA* set (Lines 18-20 of Algorithm 1). All *pFAs* that are a subset of this *dpFA* answer are removed from the *pFA* set (Line 21 of Algorithm 1). This process continues until there are no more *pFAs* available. Finally, the dpFAs are returned (Line 23 of Algorithm 1).

3.2 SWEET Method

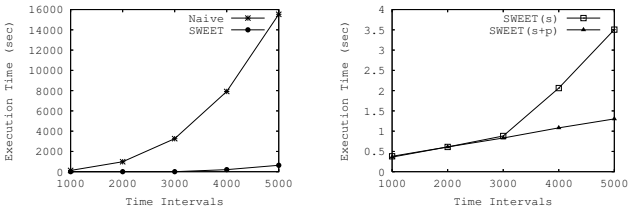
This section presents the SWEET (Smart Window Enumeration and Evaluation of persistence-Thresholds) method to find all the dominant persistent Flow Anomalies (dpFA). The SWEET method has two main phases, namely, *identification of the persistent FAs* and *identification of the dominant pFAs*. Unlike the Naïve approach, the main idea in the first phase of the SWEET method is to initially perform a single scan and examine potential pFAs starting and ending with a transient Flow Anomaly (tFA). The second phase is the same as that of the Naïve approach to find the dpFA patterns. Algorithm 2 presents the pseudo code for the SWEET method. The inputs and outputs of the SWEET method are the same as those of the Naïve approach.

Phase I: Identify pFAs. This phase is concerned with identifying all the possible windows between the up and $down$ time series that satisfy the *pFA* definition (Definition 6). Initially, the number of tFAs in a window (tFA_count) for a period is set to zero, the set *tFA* that contains the time interval at the up sensor when a transient flow anomaly has occurred is set to empty, and the *pFA* is set to empty. Unlike for the Naïve method, only a single scan of the entire time series is performed (Line 4 of Algorithm 2). The time series is scanned until a tFA is found for an Instant Pair (IP) and then the time interval at the up sensor is added to the *tFA* set (Lines 5-6 of Algorithm 2). Once the tFA is found, a scan from the first tFA in the *tFA* set to the current tFA is checked for any additional anomalies (Lines 7-12 of Algorithm 2). Then, this period is added to the pFA if it satisfies the persistence threshold, Θ_p (Lines 13-15 of Algorithm 2). If it does not satisfy, any remaining tFAs are checked with the current tFA for any pFAs. This process continues until no more tFAs are found.

Phase II: Identify Dominant pFAs. This phase is the same as for the Naïve method, that is, it identifies all the dominant FAs (dFAs) from the *pFA* set found in Phase 1 with the longest temporal length such that the dpFA is not a subset of any other dFAs (Definition 7). Initially, *dFA* is set to empty. The *pFA* with the largest temporal length is found, removed from the *pFA* set, and added to the *dpFA* set. All pFAs that are a subset of this *dpFA* answer are removed from the *pFA* set. This process continues until there are no more pFA patterns available. Finally, the dpFAs are returned.

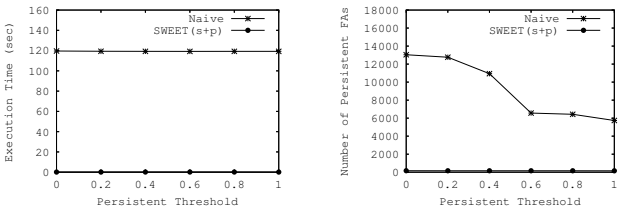
Design Decisions. Several additional design decisions can be applied to the SWEET approach to improve the overall execution time. First, a **smart counter** can be added after a tFA has been found to keep track of the total number of transient flow anomalies in the time series (Line 6 of Algorithm 2). This smart counter will identify the number of transient flow anomalies between the first tFA in the *tFA* set and the current tFA without scanning the window and determine if the window satisfies the persistence threshold. This smart counter will remove the re-scan of the window between the two tFAs (Line 8 of Algorithm 2).

The second design decision is a **pruning** strategy that can be applied as soon as a pFA candidate has been found (Line 14 of Algorithm 2). If a larger period has been found to satisfy the pFA definition, then any other periods that are a subset of the larger period do not need to be checked. As will be shown in the experimental evaluation in Section 4, this pruning strategy results, on average, in far fewer *pFA* patterns than the Naïve method.



(a) Naïve vs. SWEET (b) SWEET(s) vs. SWEET(s+p)

Figure 3. Execution Time



(a) Execution Time (b) Number of pFAs

Figure 4. Varying Θ_p

4. Experimental Evaluation

In this section, we present our experimental evaluations of several design decisions and workload parameters for the proposed SWEET method. We evaluated Naïve, SWEET, SWEET using the smart counter ("s"), and SWEET using both smart counter and pruning technique ("s+p") by varying the number of time intervals and the persistent threshold using a real dataset. The real dataset was obtained from the study site shown in Figure 1b between sensors 5 and 1 using dissolved oxygen having approximately 5k time intervals. Travel time was obtained by using discharge and depth measured at sensor 5, the width of the creek, and the length between 5 and 1. All approaches were compared in terms of execution time and the number of pFAs found. All experiments were performed on Intel P4 2 GHz 1.2 GB RAM.

Effect on Size of Time Intervals. Figure 3 gives the execution time for the real dataset using dissolved oxygen as the number of Time Intervals increase. In Figure 3a, SWEET performs better than Naïve due to the single scan and checking for periods when a tFA has occurred. Figure 3b shows that the execution time for both methods is similar up to the 3000th time interval because the number of pFAs for both approaches is close to zero. Thus, the pruning strategy is not effective due to very few number of pFAs. As the number of temporal intervals increases from 3k to 5k, a decrease in execution time occurs for the pruning technique (Figure 3b) due to the reduction of the number of pFAs.

Effect on Size of the Persistent Threshold. Figure 4 gives the comparison between Naïve and SWEET(s+p) as the persistent threshold is varied for dissolved oxygen. Figure 4a gives the execution time of Naïve, which performs worse than SWEET(s+p) due to its exhaustive search. Figure 4b shows that the number of pFAs for SWEET(s+p) is less than Naïve due to its pruning strategy.

5. Conclusion and Future Work

In this paper, we have introduced a novel problem of finding all dominant persistent Flow Anomalies. Several new concepts and interest measures are introduced. Finally, an experimental evaluation was performed on a real dataset. For future work, we plan to explore discovering FAs over more than two sensors and many to many matchings.

6. Acknowledgements

This work has been supported by NSF IGERT, NSF IIS-CXT, NSF grant EAR-0607138, USDOD, CUASHI grant EAR-0326064, and the University of Minnesota. We would like to thank Kim Koffolt for her comments.

References

[1] A. Amir, A. Apostolico, and M. Lewenstein. Inverse pattern matching. *Journal of Algorithms*, 24(2):325–339, 1997.

[2] A. Bulut and A. K. Singh. A unified framework for monitoring data streams in real time. In *ICDE*, pages 44–75, 2005.

[3] A. Chen, C. Tang, C. an Yuan, J. Peng, and J. Hu. Mining Correlations Between Multi-streams Based on Haar Wavelet. In *ASIAN*, pages 270–271, 2005.

[4] M. Datar and S. Muthukrishnan. Estimating Rarity and Similarity over Data Stream Windows. In *ESA*, pages 323–334, 2002.

[5] M. DeGroot and M. J. Scheverish. *Probability and Statistics, 3rd. Edition*. Addison Wesley, 2002.

[6] EPA. Drinking water contaminents, 2008, http://www.epa.gov/safewater/contaminants/index.html.

[7] R. Galbraith. Bay spill, san francisco chronicle, 2007, http://www.sfgate.com/cgi-bin/news/oilspill/busan.

[8] E. Knorr and R. Ng. A Unified Notion of Outliers: Properties & Computation. In *KDD*, 1997.

[9] H. Lee, R. T. Ng, and K. Shim. Estimating Rarity and Similarity over Data Stream Windows. In *VLDB*, pages 195–206, 2007.

[10] D. A. Matthews, S. W. Effler, C. T. Driscoll, S. M. O'Donnell, and C. M. Matthews. Electron budgets for the hypolimnion of a recovering urban lake, 1989-2004. *Limnology and Oceanography*, 53(2):743–759, 2008.

[11] S. Saulny. Fish-killing virus spreading in the great lakes, new york times, 2007.

[12] M. Sayal. Detecting time correlations in time-series data streams. *Hewlett-Packard Company*, 2004.

[13] S. Shekhar, C. T. Lu, and P. Zhang. A unified approach to spatial outliers detection. *GeoInformatica*, 7(2):139–166, 2003.

[14] WFUNA. Millenium project: Global challenges facing humanity, 2007.