
DATA STREAMS:
MODELS AND ALGORITHMS

DATA STREAMS: MODELS AND ALGORITHMS

Edited by
CHARU C. AGGARWAL
IBM T. J. Watson Research Center, Yorktown Heights, NY 10598

Kluwer Academic Publishers
Boston/Dordrecht/London

Contents

List of Figures	xi
List of Tables	xv
Preface	xvii
1	
An Introduction to Data Streams	1
<i>Charu C. Aggarwal</i>	
1. Introduction	1
2. Stream Mining Algorithms	2
3. Conclusions and Summary	6
References	7
2	
On Clustering Massive Data Streams: A Summarization Paradigm	9
<i>Charu C. Aggarwal, Jiawei Han, Jianyong Wang and Philip S. Yu</i>	
1. Introduction	10
2. The Micro-clustering Based Stream Mining Framework	12
3. Clustering Evolving Data Streams: A Micro-clustering Approach	17
3.1 Micro-clustering Challenges	18
3.2 Online Micro-cluster Maintenance: The CluStream Algorithm	19
3.3 High Dimensional Projected Stream Clustering	22
4. Classification of Data Streams: A Micro-clustering Approach	23
4.1 On-Demand Stream Classification	24
5. Other Applications of Micro-clustering and Research Directions	26
6. Performance Study and Experimental Results	27
7. Discussion	36
References	36
3	
A Survey of Classification Methods in Data Streams	39
<i>Mohamed Medhat Gaber, Arkady Zaslavsky and Shonali Krishnaswamy</i>	
1. Introduction	39
2. Research Issues	41
3. Solution Approaches	43
4. Classification Techniques	44
4.1 Ensemble Based Classification	45
4.2 Very Fast Decision Trees (VFDT)	46

4.3	On Demand Classification	48
4.4	Online Information Network (OLIN)	48
4.5	LWClass Algorithm	49
4.6	ANNCAD Algorithm	51
4.7	SCALLOP Algorithm	51
5.	Summary	52
	References	53
4	Frequent Pattern Mining in Data Streams	61
	<i>Ruoming Jin and Gagan Agrawal</i>	
1.	Introduction	61
2.	Overview	62
3.	New Algorithm	67
4.	Work on Other Related Problems	79
5.	Conclusions and Future Directions	80
	References	81
5	A Survey of Change Diagnosis Algorithms in Evolving Data Streams	85
	<i>Charu C. Aggarwal</i>	
1.	Introduction	86
2.	The Velocity Density Method	88
2.1	Spatial Velocity Profiles	93
2.2	Evolution Computations in High Dimensional Case	95
2.3	On the use of clustering for characterizing stream evolution	96
3.	On the Effect of Evolution in Data Mining Algorithms	97
4.	Conclusions	100
	References	101
6	Multi-Dimensional Analysis of Data Streams Using Stream Cubes	103
	<i>Jiawei Han, Y. Dora Cai, Yixin Chen, Guozhu Dong, Jian Pei, Benjamin W. Wah, and Jianyong Wang</i>	
1.	Introduction	104
2.	Problem Definition	106
3.	Architecture for On-line Analysis of Data Streams	108
3.1	Tilted time frame	108
3.2	Critical layers	110
3.3	Partial materialization of stream cube	111
4.	Stream Data Cube Computation	112
4.1	Algorithms for cube computation	115
5.	Performance Study	117
6.	Related Work	120
7.	Possible Extensions	121
8.	Conclusions	122
	References	123

<i>Contents</i>	vii
7	
Load Shedding in Data Stream Systems	127
<i>Brian Babcock, Mayur Datar and Rajeev Motwani</i>	
1. Load Shedding for Aggregation Queries	128
1.1 Problem Formulation	129
1.2 Load Shedding Algorithm	133
1.3 Extensions	141
2. Load Shedding in Aurora	142
3. Load Shedding for Sliding Window Joins	144
4. Load Shedding for Classification Queries	145
5. Summary	146
References	146
8	
The Sliding-Window Computation Model and Results	149
<i>Mayur Datar and Rajeev Motwani</i>	
0.1 Motivation and Road Map	150
1. A Solution to the BASICCOUNTING Problem	152
1.1 The Approximation Scheme	154
2. Space Lower Bound for BASICCOUNTING Problem	157
3. Beyond 0's and 1's	158
4. References and Related Work	163
5. Conclusion	164
References	166
9	
A Survey of Synopsis Construction in Data Streams	169
<i>Charu C. Aggarwal, Philip S. Yu</i>	
1. Introduction	169
2. Sampling Methods	172
2.1 Random Sampling with a Reservoir	174
2.2 Concise Sampling	176
3. Wavelets	177
3.1 Recent Research on Wavelet Decomposition in Data Streams	182
4. Sketches	184
4.1 Fixed Window Sketches for Massive Time Series	185
4.2 Variable Window Sketches of Massive Time Series	185
4.3 Sketches and their applications in Data Streams	186
4.4 Sketches with p -stable distributions	190
4.5 The Count-Min Sketch	191
4.6 Related Counting Methods: Hash Functions for Determining Distinct Elements	193
4.7 Advantages and Limitations of Sketch Based Methods	194
5. Histograms	196
5.1 One Pass Construction of Equi-depth Histograms	198
5.2 Constructing V-Optimal Histograms	198
5.3 Wavelet Based Histograms for Query Answering	199
5.4 Sketch Based Methods for Multi-dimensional Histograms	200
6. Discussion and Challenges	200

References	202
10	
A Survey of Join Processing in Data Streams	209
<i>Junyi Xie and Jun Yang</i>	
1. Introduction	209
2. Model and Semantics	210
3. State Management for Stream Joins	213
3.1 Exploiting Constraints	214
3.2 Exploiting Statistical Properties	216
4. Fundamental Algorithms for Stream Join Processing	225
5. Optimizing Stream Joins	227
6. Conclusion	230
Acknowledgments	232
References	232
11	
Indexing and Querying Data Streams	237
<i>Ahmet Bulut, Ambuj K. Singh</i>	
1. Introduction	238
2. Indexing Streams	239
2.1 Preliminaries and definitions	239
2.2 Feature extraction	240
2.3 Index maintenance	244
2.4 Discrete Wavelet Transform	246
3. Querying Streams	248
3.1 Monitoring an aggregate query	248
3.2 Monitoring a pattern query	251
3.3 Monitoring a correlation query	252
4. Related Work	254
5. Future Directions	255
5.1 Distributed monitoring systems	255
5.2 Probabilistic modeling of sensor networks	256
5.3 Content distribution networks	256
6. Chapter Summary	257
References	257
12	
Dimensionality Reduction and Forecasting on Streams	261
<i>Spiros Papadimitriou, Jimeng Sun, and Christos Faloutsos</i>	
1. Related work	264
2. Principal component analysis (PCA)	265
3. Auto-regressive models and recursive least squares	267
4. MUSCLES	269
5. Tracking correlations and hidden variables: SPIRIT	271
6. Putting SPIRIT to work	276
7. Experimental case studies	278

<i>Contents</i>	ix
8. Performance and accuracy	283
9. Conclusion	286
Acknowledgments	286
References	287
13	
A Survey of Distributed Mining of Data Streams	289
<i>Srinivasan Parthasarathy, Amol Ghoting and Matthew Eric Otey</i>	
1. Introduction	289
2. Outlier and Anomaly Detection	291
3. Clustering	295
4. Frequent itemset mining	296
5. Classification	297
6. Summarization	298
7. Mining Distributed Data Streams in Resource Constrained Environ- ments	299
8. Systems Support	300
References	304
14	
Algorithms for Distributed Data Stream Mining	309
<i>Kanishka Bhaduri, Kamalika Das, Krishnamoorthy Sivakumar, Hillol Kargupta, Ran Wolff and Rong Chen</i>	
1. Introduction	310
2. Motivation: Why Distributed Data Stream Mining?	311
3. Existing Distributed Data Stream Mining Algorithms	312
4. A <i>local</i> algorithm for distributed data stream mining	315
4.1 Local Algorithms : definition	315
4.2 Algorithm details	316
4.3 Experimental results	318
4.4 Modifications and extensions	320
5. Bayesian Network Learning from Distributed Data Streams	321
5.1 Distributed Bayesian Network Learning Algorithm	322
5.2 Selection of samples for transmission to global site	323
5.3 Online Distributed Bayesian Network Learning	324
5.4 Experimental Results	326
6. Conclusion	326
References	329
15	
A Survey of Stream Processing Problems and Techniques in Sensor Networks	333
<i>Sharmila Subramaniam, Dimitrios Gunopulos</i>	
1. Challenges	334

2.	The Data Collection Model	335
3.	Data Communication	335
4.	Query Processing	337
4.1	Aggregate Queries	338
4.2	Join Queries	340
4.3	Top- k Monitoring	341
4.4	Continuous Queries	341
5.	Compression and Modeling	342
5.1	Data Distribution Modeling	343
5.2	Outlier Detection	344
6.	Application: Tracking of Objects using Sensor Networks	345
7.	Summary	347
	References	348
	Index	353

List of Figures

2.1	Micro-clustering Examples	11
2.2	Some Simple Time Windows	11
2.3	Varying Horizons for the classification process	23
2.4	Quality comparison (Network Intrusion dataset, horizon=256, stream_speed=200)	30
2.5	Quality comparison (Charitable Donation dataset, horizon=4, stream_speed=200)	30
2.6	Accuracy comparison (Network Intrusion dataset, stream_speed=80, buffer_size=1600, $k_{fit}=80$, $init_number=400$)	31
2.7	Distribution of the (smallest) best horizon (Network Intrusion dataset, Time units=2500, buffer_size=1600, $k_{fit}=80$, $init_number=400$)	31
2.8	Accuracy comparison (Synthetic dataset B300kC5D20, stream_speed=100, buffer_size=500, $k_{fit}=25$, $init_number=400$)	31
2.9	Distribution of the (smallest) best horizon (Synthetic dataset B300kC5D20, Time units=2000, buffer_size=500, $k_{fit}=25$, $init_number=400$)	32
2.10	Stream Proc. Rate (Charit. Donation data, stream_speed=2000)	33
2.11	Stream Proc. Rate (Ntwk. Intrusion data, stream_speed=2000)	33
2.12	Scalability with Data Dimensionality (stream_speed=2000)	34
2.13	Scalability with Number of Clusters (stream_speed=2000)	34
3.1	The ensemble based classification method	53
3.2	VFDT Learning Systems	54
3.3	On Demand Classification	54
3.4	Online Information Network System	55
3.5	Algorithm Output Granularity	55
3.6	ANNCAD Framework	56
3.7	SCALLOP Process	56
4.1	Karp <i>et al.</i> Algorithm to Find Frequent Items	68
4.2	Improving Algorithm with An Accuracy Bound	71

4.3	StreamMining-Fixed: Algorithm Assuming Fixed Length Transactions	73
4.4	Subroutines Description	73
4.5	StreamMining-Bounded: Algorithm with a Bound on Accuracy	75
4.6	StreamMining: Final Algorithm	77
5.1	The Forward Time Slice Density Estimate	89
5.2	The Reverse Time Slice Density Estimate	89
5.3	The Temporal Velocity Profile	90
5.4	The Spatial Velocity Profile	90
6.1	A tilted time frame with natural time partition	108
6.2	A tilted time frame with logarithmic time partition	108
6.3	A tilted time frame with progressive logarithmic time partition	109
6.4	Two critical layers in the stream cube	111
6.5	Cube structure from the m -layer to the o -layer	114
6.6	H-tree structure for cube computation	115
6.7	Cube computation: time and memory usage vs. # tuples at the m -layer for the data set $D5L3C10$	118
6.8	Cube computation: time and space vs. # of dimensions for the data set $L3C10T100K$	119
6.9	Cube computation: time and space vs. # of levels for the data set $D5C10T50K$	120
7.1	Data Flow Diagram	130
7.2	Illustration of Example 7.1	137
7.3	Illustration of Observation 1.4	138
7.4	Procedure $SetSamplingRate(x, R_x)$	139
8.1	Sliding window model notation	153
8.2	An illustration of an Exponential Histogram (EH).	160
9.1	Illustration of the Wavelet Decomposition	178
9.2	The Error Tree from the Wavelet Decomposition	179
10.1	Drifting normal distributions.	220
10.2	Example ECBs.	220
10.3	ECBs for sliding-window joins under the frequency-based model.	222
10.4	ECBs under the age-based model.	222
11.1	The system architecture for a multi-resolution index structure consisting of 3 levels and stream-specific auto-regressive (AR) models for capturing multi-resolution trends in the data.	240
11.2	Exact feature extraction, update rate $T = 1$.	241
11.3	Incremental feature extraction, update rate $T = 1$.	241

<i>List of Figures</i>	xiii
11.4 Approximate feature extraction, update rate $T = 1$.	242
11.5 Incremental feature extraction, update rate $T = 2$.	243
11.6 Transforming an MBR using discrete wavelet transform. Transformation corresponds to rotating the axes (the rotation angle = 45° for Haar wavelets)	247
11.7 Aggregate query decomposition and approximation composition for a query window of size $w = 26$.	249
11.8 Subsequence query decomposition for a query window of size $ Q = 9$.	253
12.1 Illustration of problem.	262
12.2 Illustration of updating w_1 when a new point \mathbf{x}_{t+1} arrives.	266
12.3 Chlorine dataset.	279
12.4 Mote dataset.	280
12.5 Critter dataset	281
12.6 Detail of forecasts on Critter with blanked values.	282
12.7 River data.	283
12.8 Wall-clock times (including time to update forecasting models).	284
12.9 Hidden variable tracking accuracy.	285
13.1 Centralized Stream Processing Architecture (left) Distributed Stream Processing Architecture (right)	291
14.1 (A) the area inside an ϵ circle. (B) Seven evenly spaced vectors - $\mathbf{u}_1 \dots \mathbf{u}_7$. (C) The borders of the seven half-spaces $\hat{u}_i \cdot \mathbf{x} \geq \epsilon$ define a polygon in which the circle is circumscribed. (D) The area between the circle and the union of half-spaces.	318
14.2 Quality of the algorithm with increasing number of nodes	319
14.3 Cost of the algorithm with increasing number of nodes	319
14.4 ASIA Model	322
14.5 Bayesian network for online distributed parameter learning	327
14.6 Simulation results for online Bayesian learning: (left) KL distance between the conditional probabilities for the networks $B_{ol}(k)$ and B_{be} for three nodes (right) KL distance between the conditional probabilities for the networks $B_{ol}(k)$ and B_{ba} for three nodes	328
15.1 An instance of dynamic cluster assignment in sensor system according to LEACH protocol. Sensor nodes of the same clusters are shown with same symbol and the cluster heads are marked with highlighted symbols.	336

- 15.2 Interest Propagation, gradient setup and path reinforcement for data propagation in *directed-diffusion* paradigm. Event is described in terms of attribute value pairs. The figure illustrates an event detected based on the location of the node and target detection. 336
- 15.3 Sensors aggregating the result for a MAX query *in-network* 337
- 15.4 Error filter assignments in tree topology. The nodes that are shown shaded are the *passive* nodes that take part only in routing the measurements. A sensor communicates a measurement only if it lies outside the interval of values specified by E_i i.e., maximum permitted error at the node. A sensor that receives partial results from its children aggregates the results and communicates them to its parent after checking against the error interval 339
- 15.5 Usage of duplicate-sensitive sketches to allow result propagation to multiple parents providing fault tolerance. The system is divided into *levels* during the query propagation phase. Partial results from a higher level (level 2 in the figure) is received at more than one node in the lower level (Level 1 in the figure) 339
- 15.6 (a) Two dimensional Gaussian model of the measurements from sensors S_1 and S_2 (b) The marginal distribution of the values of sensor S_1 , given S_2 : New observations from one sensor is used to estimate the *posterior density* of the other sensors 343
- 15.7 Estimation of probability distribution of the measurements over sliding window 344
- 15.8 Trade-offs in modeling sensor data 345
- 15.9 Tracking a target. The leader nodes estimate the probability of the target's direction and determines the next monitoring region that the target is going to traverse. The leaders of the cells within the next monitoring region are alerted 347

List of Tables

2.1	An example of snapshots stored for $\alpha = 2$ and $l = 2$	15
2.2	A geometric time window	17
3.1	Data Based Techniques	44
3.2	Task Based Techniques	44
3.3	Typical LWClass Training Results	49
3.4	Summary of Reviewed Techniques	53
4.1	Algorithms for Frequent Itemsets Mining over Data Streams	64
8.1	Summary of results for the sliding-window model.	165
9.1	An Example of Wavelet Coefficient Computation	177
12.1	Description of notation.	267
12.2	Description of datasets.	278
12.3	Reconstruction accuracy (mean squared error rate).	285

Preface

In recent years, the progress in hardware technology has made it possible for organizations to store and record large streams of transactional data. Such data sets which continuously and rapidly grow over time are referred to as data streams. In addition, the development of sensor technology has resulted in the possibility of monitoring many events in real time. While data mining has become a fairly well established field now, the data stream problem poses a number of unique challenges which are not easily solved by traditional data mining methods.

The topic of data streams is a very recent one. The first research papers on this topic appeared slightly under a decade ago, and since then this field has grown rapidly. There is a large volume of literature which has been published in this field over the past few years. The work is also of great interest to practitioners in the field who have to mine actionable insights with large volumes of continuously growing data. Because of the large volume of literature in the field, practitioners and researchers may often find it an arduous task of isolating the right literature for a given topic. In addition, from a practitioners point of view, the use of research literature is even more difficult, since much of the relevant material is buried in publications. While handling a real problem, it may often be difficult to know where to look in order to solve the problem.

This book contains contributed chapters from a variety of well known researchers in the data mining field. While the chapters will be written by different researchers, the topics and content will be organized in such a way so as to present the most important models, algorithms, and applications in the data mining field in a structured and concise way. In addition, the book is organized in order to make it more accessible to application driven practitioners. Given the lack of structurally organized information on the topic, the book will provide insights which are not easily accessible otherwise. In addition, the book will be a great help to researchers and graduate students interested in the topic. The popularity and current nature of the topic of data streams is likely to make it an important source of information for researchers interested in the topic. The data mining community has grown rapidly over the past few years, and the topic of data streams is one of the most relevant and current areas of interest to

the community. This is because of the rapid advancement of the field of data streams in the past two to three years. While the data stream field clearly falls in the emerging category because of its recency, it is now beginning to reach a maturation and popularity point, where the development of an overview book on the topic becomes both possible and necessary. While this book attempts to provide an overview of the stream mining area, it also tries to discuss current topics of interest so as to be useful to students and researchers. It is hoped that this book will provide a reference to students, researchers and practitioners in both introducing the topic of data streams and understanding the practical and algorithmic aspects of the area.

Chapter 1

AN INTRODUCTION TO DATA STREAMS

Charu C. Aggarwal
IBM T. J. Watson Research Center
Hawthorne, NY 10532
charu@us.ibm.com

Abstract

In recent years, advances in hardware technology have facilitated new ways of collecting data continuously. In many applications such as network monitoring, the volume of such data is so large that it may be impossible to store the data on disk. Furthermore, even when the data can be stored, the volume of the incoming data may be so large that it may be impossible to process any particular record more than once. Therefore, many data mining and database operations such as classification, clustering, frequent pattern mining and indexing become significantly more challenging in this context.

In many cases, the data patterns may evolve continuously, as a result of which it is necessary to design the mining algorithms effectively in order to account for changes in underlying structure of the data stream. This makes the solutions of the underlying problems even more difficult from an algorithmic and computational point of view. This book contains a number of chapters which are carefully chosen in order to discuss the broad research issues in data streams. The purpose of this chapter is to provide an overview of the organization of the stream processing and mining techniques which are covered in this book.

1. Introduction

In recent years, advances in hardware technology have facilitated the ability to collect data continuously. Simple transactions of everyday life such as using a credit card, a phone or browsing the web lead to automated data storage. Similarly, advances in information technology have lead to large flows of data across IP networks. In many cases, these large volumes of data can be mined for interesting and relevant information in a wide variety of applications. When the

volume of the underlying data is very large, it leads to a number of computational and mining challenges:

- With increasing volume of the data, it is no longer possible to process the data efficiently by using multiple passes. Rather, one can process a data item at most once. This leads to constraints on the implementation of the underlying algorithms. Therefore, stream mining algorithms typically need to be designed so that the algorithms work with one pass of the data.
- In most cases, there is an inherent temporal component to the stream mining process. This is because the data may evolve over time. This behavior of data streams is referred to as *temporal locality*. Therefore, a straightforward adaptation of one-pass mining algorithms may not be an effective solution to the task. Stream mining algorithms need to be carefully designed with a clear focus on the evolution of the underlying data.

Another important characteristic of data streams is that they are often mined in a distributed fashion. Furthermore, the individual processors may have limited processing and memory. Examples of such cases include sensor networks, in which it may be desirable to perform in-network processing of data stream with limited processing and memory [8, 19]. This book will also contain a number of chapters devoted to these topics.

This chapter will provide an overview of the different stream mining algorithms covered in this book. We will discuss the challenges associated with each kind of problem, and discuss an overview of the material in the corresponding chapter.

2. Stream Mining Algorithms

In this section, we will discuss the key stream mining problems and will discuss the challenges associated with each problem. We will also discuss an overview of the material covered in each chapter of this book. The broad topics covered in this book are as follows:

Data Stream Clustering. Clustering is a widely studied problem in the data mining literature. However, it is more difficult to adapt arbitrary clustering algorithms to data streams because of one-pass constraints on the data set. An interesting adaptation of the k -means algorithm has been discussed in [14] which uses a partitioning based approach on the entire data set. This approach uses an adaptation of a k -means technique in order to create clusters over the entire data stream. In the context of data streams, it may be more desirable to determine clusters in specific user defined horizons rather than on

the entire data set. In chapter 2, we discuss the micro-clustering technique [3] which determines clusters over the entire data set. We also discuss a variety of applications of micro-clustering which can perform effective summarization based analysis of the data set. For example, micro-clustering can be extended to the problem of classification on data streams [5]. In many cases, it can also be used for arbitrary data mining applications such as privacy preserving data mining or query estimation.

Data Stream Classification. The problem of classification is perhaps one of the most widely studied in the context of data stream mining. The problem of classification is made more difficult by the evolution of the underlying data stream. Therefore, effective algorithms need to be designed in order to take temporal locality into account. In chapter 3, we discuss a survey of classification algorithms for data streams. A wide variety of data stream classification algorithms are covered in this chapter. Some of these algorithms are designed to be purely one-pass adaptations of conventional classification algorithms [12], whereas others (such as the methods in [5, 16]) are more effective in accounting for the evolution of the underlying data stream. Chapter 3 discusses the different kinds of algorithms and the relative advantages of each.

Frequent Pattern Mining. The problem of frequent pattern mining was first introduced in [6], and was extensively analyzed for the conventional case of disk resident data sets. In the case of data streams, one may wish to find the frequent itemsets either over a sliding window or the entire data stream [15, 17]. In Chapter 4, we discuss an overview of the different frequent pattern mining algorithms, and also provide a detailed discussion of some interesting recent algorithms on the topic.

Change Detection in Data Streams. As discussed earlier, the patterns in a data stream may evolve over time. In many cases, it is desirable to track and analyze the nature of these changes over time. In [1, 11, 18], a number of methods have been discussed for change detection of data streams. In addition, data stream evolution can also affect the behavior of the underlying data mining algorithms since the results can become stale over time. Therefore, in Chapter 5, we have discussed the different methods for change detection data streams. We have also discussed the effect of evolution on data stream mining algorithms.

Stream Cube Analysis of Multi-dimensional Streams. Much of stream data resides at a multi-dimensional space and at rather low level of abstraction, whereas most analysts are interested in relatively high-level dynamic changes in some combination of dimensions. To discover high-level dynamic and evolving characteristics, one may need to perform multi-level, multi-dimensional on-line

analytical processing (OLAP) of stream data. Such necessity calls for the investigation of new architectures that may facilitate on-line analytical processing of multi-dimensional stream data [7, 10].

In Chapter 6, an interesting **stream.cube** architecture that effectively performs on-line partial aggregation of multi-dimensional stream data, captures the essential dynamic and evolving characteristics of data streams, and facilitates fast OLAP on stream data. Stream cube architecture facilitates online analytical processing of stream data. It also forms a preliminary structure for online stream mining. The impact of the design and implementation of stream cube in the context of stream mining is also discussed in the chapter.

Loadshedding in Data Streams. Since data streams are generated by processes which are extraneous to the stream processing application, it is not possible to control the incoming stream rate. As a result, it is necessary for the system to have the ability to quickly adjust to varying incoming stream processing rates. Chapter 7 discusses one particular type of adaptivity: the ability to gracefully degrade performance via “load shedding” (dropping unprocessed tuples to reduce system load) when the demands placed on the system cannot be met in full given available resources. Focusing on aggregation queries, the chapter presents algorithms that determine at what points in a query plan should load shedding be performed and what amount of load should be shed at each point in order to minimize the degree of inaccuracy introduced into query answers.

Sliding Window Computations in Data Streams. Many of the synopsis structures discussed use the entire data stream in order to construct the corresponding synopsis structure. The sliding-window model of computation is motivated by the assumption that it is more important to use recent data in data stream computation [9]. Therefore, the processing and analysis is only done on a fixed history of the data stream. Chapter 8 formalizes this model of computation and answers questions about how much space and computation time is required to solve certain problems under the sliding-window model.

Synopsis Construction in Data Streams. The large volume of data streams poses unique space and time constraints on the computation process. Many query processing, database operations, and mining algorithms require efficient execution which can be difficult to achieve with a fast data stream. In many cases, it may be acceptable to generate *approximate solutions* for such problems. In recent years a number of *synopsis structures* have been developed, which can be used in conjunction with a variety of mining and query processing techniques [13]. Some key synopsis methods include those of sampling, wavelets, sketches and histograms. In Chapter 9, a survey of the key synopsis

techniques is discussed, and the mining techniques supported by such methods. The chapter discusses the challenges and tradeoffs associated with using different kinds of techniques, and the important research directions for synopsis construction.

Join Processing in Data Streams. Stream join is a fundamental operation for relating information from different streams. This is especially useful in many applications such as sensor networks in which the streams arriving from different sources may need to be related with one another. In the stream setting, input tuples arrive continuously, and result tuples need to be produced continuously as well. We cannot assume that the input data is already stored or indexed, or that the input rate can be controlled by the query plan. Standard join algorithms that use blocking operations, e.g., sorting, no longer work. Conventional methods for cost estimation and query optimization are also inappropriate, because they assume finite input. Moreover, the long-running nature of stream queries calls for more adaptive processing strategies that can react to changes and fluctuations in data and stream characteristics. The “stateful” nature of stream joins adds another dimension to the challenge. In general, in order to compute the complete result of a stream join, we need to retain all past arrivals as part of the processing state, because a new tuple may join with an arbitrarily old tuple arrived in the past. This problem is exacerbated by unbounded input streams, limited processing resources, and high performance requirements, as it is impossible in the long run to keep all past history in fast memory. Chapter 10 provides an overview of research problems, recent advances, and future research directions in stream join processing.

Indexing Data Streams. The problem of indexing data streams attempts to create an indexed representation, so that it is possible to efficiently answer different kinds of queries such as aggregation queries or trend based queries. This is especially important in the data stream case because of the huge volume of the underlying data. Chapter 11 explores the problem of indexing and querying data streams.

Dimensionality Reduction and Forecasting in Data Streams. Because of the inherent temporal nature of data streams, the problems of dimensionality reduction and forecasting are particularly important. When there are a large number of simultaneous data streams, we can use the correlations between different data streams in order to make effective predictions [20, 21] on the future behavior of the data stream. In Chapter 12, an overview of dimensionality reduction and forecasting methods have been discussed for the problem of data streams. In particular, the well known MUSCLES method [21] has been discussed, and its application to data streams have been explored. In addition,

the chapter presents the SPIRIT algorithm, which explores the relationship between dimensionality reduction and forecasting in data streams. In particular, the chapter explores the use of a compact number of hidden variables to comprehensively describe the data stream. This compact representation can also be used for effective forecasting of the data streams.

Distributed Mining of Data Streams. In many instances, streams are generated at multiple distributed computing nodes. Analyzing and monitoring data in such environments requires data mining technology that requires optimization of a variety of criteria such as communication costs across different nodes, as well as computational, memory or storage requirements at each node. A comprehensive survey of the adaptation of different conventional mining algorithms to the distributed case is provided in Chapter 13. In particular, the clustering, classification, outlier detection, frequent pattern mining, and summarization problems are discussed. In Chapter 14, some recent advances in stream mining algorithms are discussed.

Stream Mining in Sensor Networks. With recent advances in hardware technology, it has become possible to track large amounts of data in a distributed fashion with the use of sensor technology. The large amounts of data collected by the sensor nodes makes the problem of monitoring a challenging one from many technological stand points. Sensor nodes have limited local storage, computational power, and battery life, as a result of which it is desirable to minimize the storage, processing and communication from these nodes. The problem is further magnified by the fact that a given network may have millions of sensor nodes and therefore it is very expensive to localize all the data at a given global node for analysis both from a storage and communication point of view. In Chapter 15, we discuss an overview of a number of stream mining issues in the context of sensor networks. This topic is closely related to distributed stream mining, and a number of concepts related to sensor mining have also been discussed in Chapters 13 and 14.

3. Conclusions and Summary

Data streams are a computational challenge to data mining problems because of the additional algorithmic constraints created by the large volume of data. In addition, the problem of temporal locality leads to a number of unique mining challenges in the data stream case. This chapter provides an overview to the different mining algorithms which are covered in this book. We discussed the different problems and the challenges which are associated with each problem. We also provided an overview of the material in each chapter of the book.

References

- [1] Aggarwal C. (2003). A Framework for Diagnosing Changes in Evolving Data Streams. *ACM SIGMOD Conference*.
- [2] Aggarwal C (2002). An Intuitive Framework for understanding Changes in Evolving Data Streams. *IEEE ICDE Conference*.
- [3] Aggarwal C., Han J., Wang J., Yu P (2003). A Framework for Clustering Evolving Data Streams. *VLDB Conference*.
- [4] Aggarwal C., Han J., Wang J., Yu P (2004). A Framework for High Dimensional Projected Clustering of Data Streams. *VLDB Conference*.
- [5] Aggarwal C, Han J., Wang J., Yu P. (2004). On-Demand Classification of Data Streams. *ACM KDD Conference*.
- [6] Agrawal R., Imielinski T., Swami A. (1993) Mining Association Rules between Sets of items in Large Databases. *ACM SIGMOD Conference*.
- [7] Chen Y., Dong G., Han J., Wah B. W., Wang J. (2002) Multi-dimensional regression analysis of time-series data streams. *VLDB Conference*.
- [8] Cormode G., Garofalakis M. (2005) Sketching Streams Through the Net: Distributed Approximate Query Tracking. *VLDB Conference*.
- [9] Datar M., Gionis A., Indyk P., Motwani R. (2002) Maintaining stream statistics over sliding windows. *SIAM Journal on Computing*, 31(6):1794–1813.
- [10] Dong G., Han J., Lam J., Pei J., Wang K. (2001) Mining multi-dimensional constrained gradients in data cubes. *VLDB Conference*.
- [11] Dasu T., Krishnan S., Venkatasubramaniam S., Yi K. (2005). An Information-Theoretic Approach to Detecting Changes in Multi-dimensional data Streams. *Duke University Technical Report CS-2005-06*.
- [12] Domingos P. and Hulten G. (2000) Mining High-Speed Data Streams. In *Proceedings of the ACM KDD Conference*.
- [13] Garofalakis M., Gehrke J., Rastogi R. (2002) Querying and mining data streams: you only get one look (a tutorial). *SIGMOD Conference*.
- [14] Guha S., Mishra N., Motwani R., O’Callaghan L. (2000). Clustering Data Streams. *IEEE FOCS Conference*.
- [15] Giannella C., Han J., Pei J., Yan X., and Yu P. (2002) Mining Frequent Patterns in Data Streams at Multiple Time Granularities. *Proceedings of the NSF Workshop on Next Generation Data Mining*.
- [16] Hulten G., Spencer L., Domingos P. (2001). Mining Time Changing Data Streams. *ACM KDD Conference*.
- [17] Jin R., Agrawal G. (2005) An algorithm for in-core frequent itemset mining on streaming data. *ICDM Conference*.

- [18] Kifer D., David S.-B., Gehrke J. (2004). Detecting Change in Data Streams. *VLDB Conference*, 2004.
- [19] Kollios G., Byers J., Considine J., Hadjieleftheriou M., Li F. (2005) Robust Aggregation in Sensor Networks. *IEEE Data Engineering Bulletin*.
- [20] Sakurai Y., Papadimitriou S., Faloutsos C. (2005). BRAID: Stream mining through group lag correlations. *ACM SIGMOD Conference*.
- [21] Yi B.-K., Sidiropoulos N.D., Johnson T., Jagadish, H. V., Faloutsos C., Biliris A. (2000). Online data mining for co-evolving time sequences. *ICDE Conference*.