Processing Distributed Compound-Data Streams

Dongdong Zhang[1], Jianzhong Li[1,2], Weiping Wang[1], Jinbao Li[1,2], Longjiang Guo[1,2]

[1]School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China
[2] School of Computer Science and Technology, Heilongjiang University, Harbin 150080, China
{zddhit, lijzh, wpwang, lijinbao, guolongjiang}@hit.edu.cn

Abstract. In the environment of distribute data stream systems, the available communication bandwidth is a bottleneck resource. It is significant to reduce the communication overhead as possible for improving the availability of communication bandwidth with the constraint of the precision of queries. In this paper, we propose a new method for transferring data streams in distributed data stream systems, named as compound-data streams. The idea is that raw data streams are grouped and merged into compound-data streams, and then compound-data streams, instead of raw data streams, are transferred to the central processor node. By this way, the communication overhead can be reduced greatly.

1 Introduction

In many recent application systems, data takes the form of continuous *data streams*, rather than finite stored data sets. Examples include stock ticks in financial applications, performance measurement in network monitoring and traffic management, log records or click-streams in Web tracking and personalization, data feeds from sensor applications, network packets and messages in firewall-based security, call detail records in telecommunications, and so on.

The technique of processing distributed data streams has attracted the researchers in community of database [12,16,17,21]. In the environment of distributed data stream systems, the distributed remote data source nodes produce and collect data, and transfer the data to the central processor node with the form of stream. Furthermore, the central processor node deals with the queries and returns the answers to users. The data stream from the remote data source nodes can be denoted as an unlimited set with the model of $<r, t>$, which we call as ***Single-Data streaMs*** (**SDM**) in this paper, where r denotes a tuple and t denotes a timestamp which is determined by the time when r enters the data stream system or when the remote data source node produces r [26]. In distributed data stream systems, the available communication bandwidth becomes a bottleneck resource [12]. In addition, the electric energy, sensors used to transmit data, is also a bottleneck resource in the application of wireless sensor network [19,20]. So, one issue of processing distributed data streams is how to reduce the size of transferred data as possible with the constraint of the precision of queries, in order to save the communication cost or the electric energy of sensors. At present, there are two kinds of solutions. One solution is discarding a portion of data, which should be transferred to the central processor node, by installing filters at remote data source nodes [12]. The other solution is pushing some operations down to the remote data source nodes to run [19]. However, both two solutions have drawbacks.

(1) The first method can't support ad-hoc queries in real-time, because the remote data source nodes only transmit the data satisfying the queries, e.g. $A \geq 10$. If a new ad-hoc query need access all the data of the recent time, the central processor node can't give the answer since the data satisfying the condition of $A<10$ is absent.

(2) The second method restricts the sharing of querying-results since the operations pushed down to the remote data source nodes may be complex. Moreover, some operations, such as JOIN, can't be pushed down to remote data source nodes to run, and the raw data streams still need to be transferred to the central processor node, so that it causes the redundant data during the transferring

In a word, the previous work on distributed data streams focuses on single-data stream, and the efficiency of method is dependent on the category of queries. Thus, we consider drawing out a common operation from all kinds of queries and pushing it down to the remote data source to run, so that it can not only overcome the drawback of above methods, but also reduce the communication overhead.

We propose a new method to transfer the data streams in the environment of distributed data stream systems, named as ***Compound-Data streaMs*** (**CDM**). The main idea is transferring CDM to the central processor node, while items in SDM are grouped and merged into items in CDM at the remote nodes. Following is the advantage of transferring CDM instead of SDM. (1) Since one item in CDM has replaced several items in SDM, the communication cost is reduced.

(2) In the application of wireless sensor network, the electric energy, which the sensor needs to transmit the data, can be saved. (3) Since CDM is the partial result of processing SDM, the load of the central processor node can be reduced when dealing with CDM.

The contribution of this paper can be described as follows:

(1) We give the definition of CDM and define the operations on it.

(2) We present the join and aggregation algorithm based on CDM.

(3) Some special issues about the generation of CDM are discussed, including the span of CDM, the delay of queries, etc.

The remainder of this paper is organized as follows: Section 2 describes the model of distributed data stream system based on CDM. The querying algorithms over CDM are studied in Section 3. Section 4 discusses some special issues about the generation of CDM. Section 5 presents related work. Finally, Section 6 concludes the paper.

2 Model of Distributed Data Stream System

2.1 Compound-Data Streams

An item in SDM with the model of $<r, t>$ consists of a tuple r and a timestamp t, where r contains p dimension-attributes and q measure-attributes, denoted by $D_1,..,D_p$ and $M_1,\ldots,M_q$ respectively. However, an item in CDM consists of a tuple, deviation range, time interval and volume.

Definition 1. Suppose R is a set of items in SDM and $R=\{<r_i,t_i>|\ r_i.D_k=r_{i+1}.D_k=v_k \wedge t_i\le t_{i+1} \wedge 1\le i<n \wedge 1\le k\le p\}$, then all items in R can be merged into an item in *Compound-Data streaMs* (CDM) with the model of $< z, Dev, N, V >$, denoted by s. Following explains the content of s in detail.

(a) $z=(v_1,\ldots, v_p, u_1,\ldots,u_q)$, $u_i=\text{AVG}\{r_k.M_i|1\le k\le n\}$;

(b) $Dev=\{(minV_i, maxV_i)|1\le i\le q\}$ describes the boundary of deviation of measure-attributes, where $minV_i=\max\{u_i-r_k.M_i \mid 1\le k\le n\}$ and $maxV_i=\max\{r_k.M_i - u_i \mid 1\le k\le n\}$;

(c) $N=[t_1, t_n]$ describes a time interval, where $N.b=t_1$ and $N.e=t_n$. Meanwhile, we call the value of $N.e-N.b$ as the *span* of s, denoted by $|N|$;

(d) $V=n$ is the volume of s, which describes the number of items in SDM *absorbed* by s.

For easy to discuss, we assume $p=q=1$ in remainder of this paper. Meanwhile, we denote the dimension-attribute and the measure-attribute of r_i[1] in SDM as $r_i.D$ and $r_i.M$ respectively. We also denote the dimension-attribute, the measure-attribute, the time interval and the deviation's bound of s_i in CDM as $s_i.D$, $s_i.M$, $s_i.N$, $s_i.maxV$, $s_i.minV$ respectively. When there is no confusion, above symbols can be simplified as D_i, M_i, N_i, $maxV_i$ and $minV_i$ respectively.

From Definition 1, we can know that the aggregation of items in SDM absorbed by an item s in CDM can be evaluated without any error based on the content of s. Suppose R is the set of items in SDM absorbed by s, then $\text{SUM}(R)=s.M\times s.V$, $\text{COUNT}(R)=s.V$, $\text{AVG}(R)=s.M$, $\text{MAX}(R)=s.M+s.maxV$ and $\text{MIN}(R)=s.M-s.minV$.

2.2 Precision-Metrics of CDM

Since each item in CDM is generated through several items in SDM, some precision-metrics of CDM are needed to evaluate the performance of this process. The span and the volume of CDM can be regarded as two explicit precision-metrics, which explain the valid time range and

---

[1] When no confusion, the item $<r,t>$ in SDM can be denoted by r and the item $<s,Dev,N,V>$ in CDM can be denoted by s.

the number of items in SDM absorbed by the item in CDM respectively. In addition, the following is another two precision-metrics which character the deviation degree from different aspects between the average and the values of measure-attributes of items in SDM absorbed by the item in CDM.

Definition 2. Suppose s=<r, Dev, N, V> is an item in CDM, then the value of max{$s.minV$, $s.maxV$} / $s.M$ is called as the ***Convergent- Degree*** of s, denoted by $s.\eta$.

The Convergent-Degree describes the maximum deviation from the value of items in SDM absorbed by the item in CDM to their average. Meanwhile, Convergent-Degree can be used to adjust the precision of queries (see section 3.4).

2.3 Our Model of Distributed Data Stream System

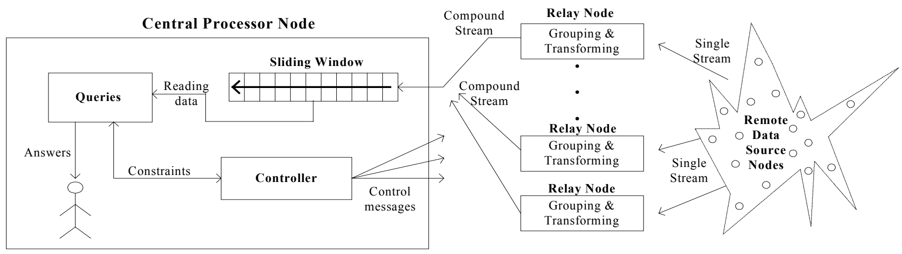

Fig. 1. The model of distributed data stream system

Figure 1 illustrates our model of distributed data stream system based on CDM. There are three kinds of processing nodes, including the central processor node, relay nodes and remote data source nodes. Remote data source nodes produce or collect items in SDM, and transfer SDM to relay nodes. While receiving SDM, the relay nodes group and merge the SDM into CDM, and transmit CDM to the central processor node. The central processor node deals with user's queries and returns the answers.

Based on the system load and the requirement of queries, the controller on the central processor node sends the control messages to relay nodes. Meanwhile, the relay nodes adjust the parameters of CDM in light of these control messages in order to reduce the communication overhead and improve the precision of queries as possible.

From Definition 1, we know that CDM can be regarded as the results of subsection Group-AVG operation over SDM. Actually, the effect of transferring CDM to the central processor node corresponds with that of the central processor node pushing a common operation (i.e. Group-AVG operation) of all queries down to the relay nodes for long running, only that the former method can provide more abundant information to the central processor node.

3 Generating and Querying CDM

3.1 Generating CDM

The relay nodes control the process of generating CDM based on the parameters of control messages such as the Convergent-Degree η_0 and the maximum span T_u, etc., which can be described as follows: While the relay nodes receiving SDM from remote data source nodes, they divide SDM into groups (called as ***branch-groups***) in terms of dimension-attributes and make each branch-group absorb as much as items in SDM. With the constraint of parameters of control messages, the relay nodes transform each branch-group into an item in CDM with the maximum volume. After transmitting an item in CDM to the central processor node, the relay node restarts the next proceeding of the corresponding branch-group. Following is the algorithm of generating CDM on relay nodes.

Algorithm Generating_Compound_Stream :

Input: SDM, parameters such as η_0 and T_u.

Output: CDM.

(1) WHILE (Receiving a new item r in SDM)

(2) Evaluate the value of dimension-attribute of r, then find the corresponding branch-group G_i and the item z_i in CDM;

(3) On the basis of z_i, transform $G_i \cup \{r\}$ into an new item s in CDM;

(4) IF ($s.\eta > \eta_0$ OR OR $|s.N| > T_u$)

(5) Transmit z_i to the central processor node; /*since z_i is of maximum volume */

(6) Clean and set $G_i := \{r\}$, then generate a new item z_i in CDM through G_i;

(7) ELSE

(8) $G_i := G_i \cup \{r\}$, $z_i := s$,

(9) END IF

(10) END WHILE □

In the above algorithm, the time complexity of clauses (2)~(9) is $O(d)$, and the space complexity is also $O(d)$, where d is the number of branch-groups. Thus, we can conclude that when d is not large, the resource expense is low during generating CDM. Furthermore, in the application of wireless sensor network, sensors are capable of generating CDM with limited memory and energy resource.

3.2 SPJ Operations of CDM

In spite of CDM ‘s infinity, the operations of CDM can still be restricted within a valid time interval.

Definition 3.(Select Operation, Project Operation) Suppose $R=\{<r_i, Dev^{(r)}_i, N^{(r)}_i, V^{(r)}_i>|1\leq i\leq n\}$, $O\in\{\sigma, \prod\}$, N_q is the valid time interval of O, P is the predicts or the projected attributes of O, then $O_P(R)=\{<s_k, Dev_k, N_k, V_k>|\ s_k=O_P(r_k), Dev_k=Dev^{(r)}_k, N_k=N^{(r)}_k\cap N_q, V_k=V^{(r)}_k\cdot|N_k|/|N^{(r)}_k|, <r_k, Dev^{(r)}_k, N^{(r)}_k, V^{(r)}_k>\in R\}$.

Definition 4.(Join Operation) Suppose $R=\{<r_i,Dev^{(r)}_i,N^{(r)}_i,V^{(r)}_i>\ |\ 1\leq i\leq n\}$ and $S=\{<s_i,Dev^{(r)}_i,N^{(s)}_i,V^{(s)}_i>|\ 1\leq i\leq m\}$, then $R\bowtie S=\{<t_k, Dev_k, N_k, V_k>\ |\ t_k=r_i\bowtie s_j, Dev_k=Dev^{(r)}_i\cup Dev^{(s)}_j, N_k=[\ \max\{N^{(r)}_i.b, N^{(s)}_j.b\}, \max\{N^{(r)}_i.e, N^{(s)}_j.e\}\ ], V_k=V^{(r)}_i\cdot V^{(s)}_j, <r_i, Dev^{(r)}_i, N^{(r)}_i, V^{(r)}_i>\in R, <s_j, Dev^{(s)}_j, N^{(s)}_j, V^{(s)}_j>\in S\}$.

Suppose V is the average volume of CDM, C_S and C_C are the time cost of operations of SDM and CDM respectively, then we can know $C_C(\sigma)/C_S(\sigma)=1/V$, $C_C(\prod)/C_S(\prod)=1/V$, $C_C(\bowtie)/C_S(\bowtie)=1/V^2$.

3.3 Join Algorithm of CDM

At present, most of join algorithms of data streams are based on sliding windows [2,3,4,23] and are divided into two classes. One is based on the idea of Nested-Loop (denoted by NLJoin), and the other is based on the idea of Hash (denoted by HJoin). The join process over sliding windows is separated into three phases [23]: inserting an item, probing and joining items, invaliding items.

Suppose A and B are two sliding windows which preserve the recent arrival of items in CDM. Each sliding window is corresponding to a data stream. Then, the process of $A\bowtie B$ can be described as follows.

(1) Inserting an item: When a new item s in CDM arrives, NLJoin algorithms insert s into A (or B) directly. However, HJoin algorithms should evaluate the hash-value of joining attribute of s, then insert s into the corresponding bucket of A (or B). The time complexity of this process is both $O(1)$.

(2) Probing and joining items: After a new item s in CDM is inserted into A (or B), the items in B (or A) should be probed and joined with s, then the join-results are output based Definition 4. In this process, NLJoin algorithms need scan all items in B (or A) and the time complexity is $O(n)$. However, HJoin algorithms only scan the item in one bucket of B (or A) and the time complexity is $O(n/B)$, where B is the number of buckets.

(3) Invaliding items: Suppose T is the valid time interval of the sliding window, then the condition of invaliding the item $<s,Dev, N, V>$ is that $Now–T>N.e$ holds, where Now is the current system clock. In spite that both immediate-invalid strategy and timing-invalid strategy can be

used to invalid the items in CDM, all items in sliding windows should be scanned. The time complexity is $O(n)$.

3.4 Aggregation of CDM

CDM supports all kinds of standard aggregation including SUM, COUNT, AVG, MAX and MIN. For each aggregation, we can evaluate the results and the precision of aggregation based on the content of items in CDM. Suppose N_q is the valid time interval of aggregation, $R=\{<s_i, Dev_i, N_i, V_i> \mid N_q \cap N_i \neq \phi \wedge 1 \leq i \leq n\}$ is the set of items in CDM satisfying the condition of aggregation. Then, during the central processor node dealing with the aggregation of CDM, each time interval N_i of the item s_i in R should be compared with N_q. If $N_i \subseteq N_q$ holds, we can obtain the aggregate result of items in SDM absorbed by s_i without any error. Otherwise, i.e. $N_q \cap N_i \neq N_i$, we should use the content of s_i to estimate the aggregate result of items in SDM covered by the time subinterval $N_q \cap N_i$. In light of the relation between N_q and the time interval of s_i, the set R can be divided into two subset of R_1 and R_2, where $R_1=\{<s_i, Dev_i, N_i, V_i> \mid N_i \subseteq N_q \wedge 1 \leq i \leq n\}$, $R_2=\{<s_i, Dev_i, N_i, V_i> \mid N_q \cap N_i \neq N_i \wedge 1 \leq i \leq n\}$, $R_1 \cup R_2 = R$ and $R_1 \cap R_2 = \phi$.

The aggregation on R can be defined as follows:

(1) SUM(R)= $\Sigma_{s_i \in R_1} s_i.M \times s_i.V + \Sigma_{s_i \in R_2} s_i.M \times s_i.V \times |N_q \cap N_i| / |N_i|$.

(2) COUNT(R)= $\Sigma_{s_i \in R_1} s_i.V + \Sigma_{s_i \in R_2} s_i.V \times |N_q \cap N_i| / |N_i|$.

(3) AVG(R)=SUM(R)/COUNT(R).

(4) MAX(R)=$\max\{\max\{s_i.M+s_i.maxV \mid s_i \in R_1\}, \max\{s_i.M \mid s_i \in R_2\}\}$.

(5) MIN(R)=$\min\{\min\{s_i.M-s_i.minV \mid s_i \in R_1\}, \min\{s_i.M \mid s_i \in R_2\}\}$.

4 Adjusting the Precision-metrics of CDM

Relay nodes transfers CDM, instead of SDM, can reduce communication cost, which is influenced by the value of precision- metrics of CDM. When the value of precision-metrics become larger, the data size transferred to the central processor node is reduced more. However, the value of precision-metrics can't be large enough, otherwise, not only the precision of queries on the central processor node descends, but also it causes much delay of queries' responding time. Thus, it is critical for relay nodes to adjust the value of precision-metrics of CDM reasonably based on the status of the system.

There are several factors impacting upon the setting of precision-metrics of CDM, including queries' tolerable delay of responding time, the constraint of queries' precision, the available bandwidth of network, etc. In this section, we will discuss the adjustment of precision-metrics based on these factors.

4.1 Span of CDM and Tolerable Delay of Queries

Suppose $\{<r_i,t_i>|\ t_i\leq t_{i+1} \wedge 1\leq i<n\}$ is the set of items in SDM absorbed by the item s in CDM, then the span of s is $T=t_n-t_1$. Based on Generating_Compound_Stream algorithm, we know that the relay node transmits s to the central processor node only after it has received the item $<r_n,t_n>$ in SDM. Without considering the time cost of transferring s, the delay of transferring the information of the item $<r_1,t_1>$ in SDM to the central processor node is t_n-t_1. In the same way, we can know that transferring the information of the item $<r_i,t_i>$ in SDM to the central processor node can cause the delay of t_n-t_i. Therefore, when an item in CDM with the span of T is transferred to the central processor node, it will cause the internal maximum delay of T.

Since the delay of transferring data could impact upon the precision and correctness of queries, the central processor node should be capable of dealing with the problem of delay. Thereby, based on the requirement of queries, the central processor node should determine a maximum tolerable delay τ, which is sent to relay nodes by the controller. Then, relay nodes will adjust the span T of items in CDM with the constraint of $T\leq\tau$.. On the other hand, the central processor node may also predict the future items based on the arrived items to deal with queries.

4.2 Span of CDM and Available Bandwidth

The limited bandwidth resource determines that there is the low bound of the span of items in CDM. Suppose B is the available bandwidth and ω is the number of relay nodes, then, the maximum data size transmitted by each relay node in unit time is B/ω. For any relay node i, suppose L_i is the number of branch-groups, α_i is the size of each item in CDM and T_i is the span, then we should have the constraint of $\alpha_i\cdot L_i/T_i\leq B/\omega$ in order to avoid the jam in network. Therefore, the low bound of the span T_i of CDM is $\alpha_i\cdot L_i\cdot\omega/B$.

5 Related Work

There is a good survey about issues of data streams in [5,6]. Several famous projects have attracted many researchers on data streams, such as STREAM [7], Telegraph [8,9], Aurora [10] projects. STREAM is a general data stream management system, which focuses on issues about query language and precise semantics of stream queries [5], about scheduling the operators [11], and about management of resources [7]. Telegraph [13] can deal with the streaming data adaptively. Aurora [10] schedules the operators and controls the flow direction of streaming data among the operators with the model of work flow system, so that it can optimally utilize the resource and enhance QoS [10,14,15]. [1,2,3,4] discuss how to improve the efficiency and precision of queries. However, above work was limited to processing data streams on single processor.

The research about distributed data stream systems has risen and wireless sensor network can be regarded as a special application environment of distributed data stream systems. [16] describes some method to processing the distributed data streams in Aurora*. [12] studies the problem of saving the bandwidth in distributed data steam system through adjusting the filters on the remote data sources. [17] discusses the issue of Top-K queries in the environment of distributed data stream systems. [18] presents an implement of data stream management system based on wireless sensor network. [19] discusses the problem of processing queries in wireless sensor network. [20] studies how to deal with acquisitional-queries in TinyDB. [21] gives some distributed stream algorithm and [22] studies the routing strategy of tuples among distributed operations. However, above work is based on SDM, and doesn't consider the case of processing CDM.

6 Conclusion and Future Work

The main idea of this paper is to transfer CDM in network in order to reduce the communication overhead. We push the common operation (i.e. Group-AVG operation) down to remote nodes, which can reduce the communication cost greatly in distributed data steam systems. In future work, we expect to add more useful information into CDM and find more efficient common operations pushed down to remote nodes, so that the precision of queries is improved and the communication overhead is reduced greatly.

References

1. S. Guha, N. Koudas. Approximating a Data Stream for Querying and Estimation: Algorithms and Performance Evaluation. ICDE, 2002.
2. Lukasz Golab. M.Tamer Ozsu. Processing Sliding Window Multi-Joins in Continuous Queries over Data Streams. VLDB, 2003.
3. S. D. Viglas, J. F. Naughton, J. Burger. Maximizing the Output Rate of Multi-Way Join Queries over Streaming Information Sources. VLDB, 2003.
4. M. A. Hammad, M. J. Franklin, W.G. Aref and A.K. Elmagarmid. Scheduling for shared window joins over data streams. VLDB, 2003.
5. B. Babcock, S.Babu and M.Datar. Model and Issues in Data Stream Systems. PODS, 2002.

6. L. Golab and M.T. Ozsu. Issues in Data Stream Management. SIGMOD Record, Vol. 32, No. 2, June, 2003.
7. R. Motwani, J. Widom and A. Arasu. Query Processing, Approximation, and Resource Management in a Data Stream Management System. CIDR, 2003.
8. S. Madden and M. J.Franklin. Fjording the Stream: An Architecture for Queries over Streaming Sensor Data. ICDE, 2002.
9. S. Chandraskearan and M. J.Franklin. Streaming Queries over Streaming Data. VLDB, 2002.
10. D. Carney, U. Cetintemel and M. Cherniack. Monitoring Streams-A New Class of Data Management Applications. VLDB, 2002.
11. B. Babcock, S. Babu, M. Datar, and R. Motwani. Chain: Operator Scheduling for Memory Minimization in Data Stream Systems. SIGMOD, 2003.
12. C. Olston, J. Jiang, and J. Widom. Adaptive Filters for Continuous Queries over Distributed Data Streams. SIGMOD, 2003.
13. S. Madden, M. Shah, J. M. Hellerstein, and V. Raman. Continuously Adaptive Continuous Queries over Streams. SIGMOD, 2002.
14. N. Tatbul, U. Cetintemel, and S. Zdonik. Load Shedding in a Data Stream Manager. VLDB, 2003.
15. D. Carney, U. Cetintemel, A. Rasin, S. Zdonik, M. Cherniack, and M. Stonebraker. Operator Scheduling in a Data Stream Manager. VLDB, 2003.
16. M. Cherniack, H. Balakrishnan, M. Balazinska. Scalable Distributed Stream Processing. In Proc. 1st Biennial Conf. On Innovative Data Syst. Res (CIDR), 2003.
17. B. Babcock and Chris Olston. Distributed Top-K Monitoring. SIGMOD, 2003.
18. S. Madden, M. J. Franklin. Fjording the Stream: An Architecture for Queries over Streaming Sensor Data. ICDE, 2002.
19. Y. Yao, J. Gehrke. Query Processing for Sensor Networks. CIDR, 2003.
20. S. Madden, M. J. Franklin, and J. M. Hellerstein. The Design of and Acquesitional Query Processor For Sensor Networks. SIGMOD, 2003.
21. P. B. Gibbons, S. Tirthapura. Distributed Streams Algorithms for Sliding Windows. SPAA, 2002.
22. F. Tian, D. J. DeWitt. Tuple Routing Strategies for Distributed Eddies. VLDB, 2003.
23. J. Kang. J. F. Naughton, S. D. Viglas. Evaluating Window Joins over Unbounded Streams. ICDE, 2003.
24. M.Arlitt and T. Jin. 1998 world cup web site access logs. Aug. 1998. Available at http://www.acm.org/sigcomm/ITA.
25. S. Floyd and V. Paxson. Wide-area Traffic: The Failure of Poisson Modeling. IEEE/ACM Transactions on Networking, 3(3), 1995.
26. A. Araru, S. Babu, J. Widom. An Abstract Semantics and Concrete Language for Continuous Queries over Streams and Relations. Technical Report, Stanford University Database Group. Nov. 2002. Available at http://dbpubs.stanford.edu/pub/2002-57.