Multi-dimensional Phenomenon-aware Stream Query Processing

Ashish Bindra
Institute Of Technology,
University of Washington,
Tacoma, WA, USA
abindra@u.washington.edu

Ankur Teredesai
Institute Of Technology,
University of Washington,
Tacoma, WA, USA
ankurt@u.washington.edu

Mohamed H. Ali
Microsoft Corporation,
One Microsoft Way
Redmond, WA, USA
mali@microsoft.com

Walid G. Aref
Department of Computer
Science, Purdue University
West Lafayette, IN, USA
aref@cs.purdue.edu

ABSTRACT
Geographically co-located sensors tend to participate in the same environmental phenomena. Phenomenon-aware stream query processing improves scalability by subscribing each query only to a subset of sensors that participate in the phenomena of interest to that query. In the case of sensors that generate readings with a *multi-attribute* schema, phenomena may develop across the values of one or more attributes. However tracking and detecting phenomena across all attributes does not scale well as the dimensions increase. As the size of sensor network increases, and as the number of attributes being tracked by a sensor increases this becomes a major bottleneck. In this paper, we present a novel *n-d*imensional *P*henomenon *D*etection and *T*racking mechanism (termed as *nd-PDT*) over *n*-ary sensor readings. We reduce the number of dimensions to be tracked by first dropping dimensions without any meaningful phenomena, and then we further reduce the dimensionality by continuously detecting and updating various forms of functional dependencies amongst the phenomenon dimensions.

Categories and Subject Descriptors
H.2.4 [Database Management]: Systems - Query processing; H.2.8 [Database Management]: Database Applications - Spatial databases and GIS;

General Terms
Algorithms, Measurement, Performance, Design, Experimentation

Keywords
Data Streams, Query Processing, DSMS, Sensors Networks

1. INTRODUCTION
With the advances in sensor network technology and with the growing ubiquity of sensor deployments, large amounts of sensor data is readily gathered; yet processing this data efficiently and accurately still poses significant challenges. Phenomenon-aware query processing [1][2] alleviates this problem by taking into consideration the fact that although sensor data exhibit tremendous variations in reading values globally across the entire region of a sensor network, sensor readings tend to be similar across geographically co-located sensors over a period of time. The domain of Phenomenon-aware query processing addresses the challenges in; (a) the continuous detection and tracking of phenomena as they appear, move, and disappear from the sensor field, and (b) the optimization of subsequent user queries using the detected phenomenon. The two goals are achieved by two components that reside at the core of a phenomenon-aware system: the Phenomenon Detection and Tracking (*PDT*) module and the phenomenon-aware optimizer, respectively.

The *PDT* module continuously monitors correlation amongst readings that are arriving from near-by sensors. The phenomenon-aware optimizer then assesses the interest of a user-given query in all detected phenomena. It ensures that a query is remotely deployed over (and only over) the sensors that participate in phenomenon that the query is interested.

In most cases, efficiency is achieved because the number of sensors that participate in various phenomena is significantly less than the total number of sensors. Also, the number of phenomena that are of particular interest to query Q_i is a subset of all observed phenomena. Therefore, query Q_i is deployed only over a small subset of sensors thereby reducing the number of queries being processed by each sensor increasing throughput.

The *PDT* module has been presented in literature in [1][3][4], while phenomenon-aware query optimization has been presented in [2]. Currently, all previously conducted research assumes a single-attribute sensor schema, where sensors read single value types, e.g., temperature sensors, light intensity sensors, etc. This remains true even if the sensor's schema is a multi-attribute schema. This severely limits the applicability of phenomenon aware query processors.

In this paper, we alleviate this problem by proposing a phenomenon aware query processing system that can handle sensors that generate multi-attribute readings, where phenomena may develop across some or all of the attributes. Meanwhile, there are hundreds or thousands of standing queries that are featured by conjunctive predicates over the values of some or all of these attributes. The naive approach is to deploy *n* independent *PDT* systems, one per attribute (or *dimension*). Then, the phenomenon-aware query optimizer deploys query Q_i over sensor S_j if there is a consensus among the all dimensions, that sensor S_j is of interest to query Q_i. Note that on one hand, each monitored dimension incurs some cost in the *PDT* process while on the other hand; it results in early filtering of some sensors that seem irrelevant to query Q_i. The challenge then is to find the right set of dimensions that balance between the *PDT* cost and the gained reduction in the query deployment map.

Our contributions can be summarized as follows:

1. We enhance phenomenon-aware query processors with the ability to detect phenomena across multiple dimensions.
2. We present a two-step dimensionality reduction technique. The first step filters-out the dimensions where no interesting

This work was supported in part by the National Science Foundation under Grant IIS-0811954.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage, and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ACM GIS '09, November 4-6, 2009. Seattle, WA, USA
(c) 2009 ACM ISBN 978-1-60558-649-6/09/11...$10.00

phenomena are detected and the second step then eliminates *functionally dependent* dimensions.

3. We further extend the multi-dimensional PDT and make it dynamic such that it can handle behavioral changes in the streaming environment by *continuously* adjusting the monitored set of dimensions. Dimensions are added and removed dynamically to balance between the cost of phenomenon detection across multiple dimensions and gained benefit of deploying queries over smaller subsets of sensors.

The remainder of this paper is organized as follows. Section 2 provides the background on phenomenon-aware systems. Section 3 outlines the issues of designing multidimensional phenomenon-aware query processing systems. Section 4 focuses on the proposed dimensionality reduction techniques and presents the system's ability to be adaptive and responsive to changes in the behavior of the underlying stream sources. Section 5 concludes the paper.

2. BACKGROUND

In this section we present a definition for a phenomenon and overview the general architecture of phenomenon-aware systems. Then, we discuss the cost-benefits trade-off of such systems.

A *phenomenon* takes place only when a set of sensors S in SN report similar reading values over a period of time [1], [4]. Definition 1 below provides a formal definition of a phenomenon that captures the notion of similarity among sensors. The notion of similarity has been extensively studied by several research groups, yet under different terminologies, e.g., phenomena, isobars, homogenous regions, deformable 2D objects, etc (][,[7],[8],[9],[10],[11]).

Definition 1 *A phenomenon* P_τ *at time instant* τ *is a binary tuple* (R_τ, B_w)*, where* R_τ *is the bounding region of phenomenon* P_τ *at time instant* τ *and* B_w *is the representative behavior of phenomenon* P_τ *over the most recent time window of size w, S.T. stream* $S_i \in R_\tau$, $Prob(|B_w(S_i) - B_w| \geq \varepsilon) \leq \alpha$.

Based on *Definition 1*, a phenomenon has to be associated with a time instant τ because a phenomenon may change its location (R_τ) over time. Also, the representative behavior of a phenomenon B_w is captured over a window of time (w) to ensure its persistency and to avoid the effect of noise. A stream source S_i that lies in the phenomenon region R_τ should report a behavior ($B_w(S_i)$) that is similar to the phenomenon representative behavior B_w with high probability (i.e., $Prob(|B_w(S_i) - B_w| \geq \varepsilon) \leq \alpha$). B_w, the phenomenon representative behavior, captures the intrinsic features of the underlying phenomenon, e.g., values, frequencies, and trends of tuples contributing to the phenomenon.

Figure 1 illustrates the architecture of phenomenon-aware systems. Phenomenon-aware systems receive, as input, a set of phenomenon definitions. These definitions are registered and stored in a special system catalog called phenomenon definition catalog. The continuous *PDT* query is distributed over the sensor network to be executed in-network. The outcome of this query is a set of detected phenomenon on the format of specified by Definition 1. The detection algorithms, both the centralized and distributed versions, are presented in [3].

The phenomenon-aware optimizer assesses the similarity between the expected query result and the phenomena stored in the detected phenomenon catalog. The optimizer binds each query to a subset of the detected phenomena that are believed to contain sensor readings of interest to the query. Equivalently, the optimizer binds each query to (and only to) the sensors that participate in the query's subset of interesting phenomena. These selected sensors are termed the query's working set of sensors. Then, the query dispatcher deploys each query remotely to (and only to) the sensors that belong to the query's working set, and hence, achieves an efficient *query deployment map*. The phenomenon-aware query processing has been implemented in the context of the *Nile* [8] data stream management system developed at Purdue University.

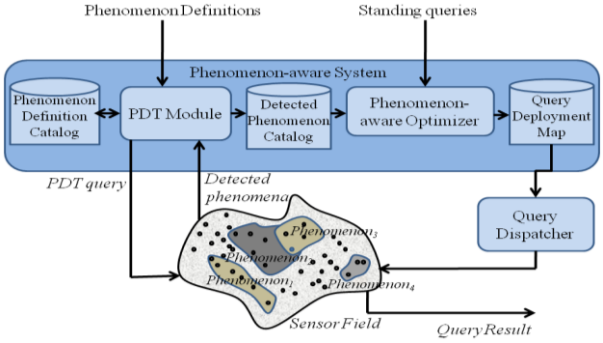

Figure 1. Architecture of a phenomenon-aware system.

3. MULTI-DIMENSIONAL PHENOMENON-AWARE QUERY PROCESSING

For better understanding of the multi-dimensional phenomenon-aware system and to build the right expectations for the performance of such systems, we devote this section to formalize the expected behavior of the system in terms of processing cost and the output and term this as the system throughput. Table 1 provides a summary for the symbols used in these costing equations.

Table 1. Summary of the costing formula symbols.

n_d	Number of PDT monitored dimensions
n_q	Number of standing queries at the sensor
n_{input}	Number of input events
$T_{acquisition}$	Acquisition time: total time taken to acquire n_{input} events from the environment and enqueue them at the sensor's input buffers
$T_{processing}$	Processing time: total time taken to process n_{input} events for n_q standing queries
$t_{processing}$	Processing time for a single tuple per single query
T_{PDT}	The cost of a single PDT measured in time units
Sel_i	Selectivity of query number i

$$Thruput_{acquisition} = \frac{n_{input}}{T_{acquisition}} \quad (1)$$

$$Thruput_{input} = \frac{n_{input} \times n_q}{T_{acquisition}} \quad (2)$$

$$Thruput_{output} = \frac{n_{input} \times \sum_i^{n_q} Sel_i}{T_{acquisition} + T_{processing}} \quad (3)$$

$$T_{processing} = n_{input} \times n_q \times t_{processing} + n_d \times T_{PDT} \quad (4)$$

$$Thruput_{output} = \frac{n_{input} \times \sum_i^{n_q} Sel_i}{\mathrm{T}_{acquisition} + (n_{input} \times n_q \times t_{processing} + n_d \times T_{PDT})} \quad (5)$$

We differentiate amongst three notions of throughput: (1) The *acquisition throughput*, which is the total number of tuples acquired by the sensor per time unit (Equation 1). (2) The *input throughput*, which is the acquisition throughput multiplied by the number of standing queries (Equation 2). The input throughput represents the total number of tuples that need to be processed by the sensor's query processor. (3) The *output throughput*, which is the number of output tuples coming out of the sensor's query processor. Equation 3 shows that the output throughput equals number of input events (after imposing the selectivity of each standing query) divided by the total time (acquisition time and processing time). Higher output throughput indicates that the sensor's processing cycles are invested efficiently to direct input tuples to the right queries. Equation 4 captures the processing time as: time spent in queries and time spent in the *PDT* modules, which is considered an overhead for phenomenon-aware query processors. Substituting the $T_{processing}$ in equation 3, we get equation 5.

$$n_q \propto \frac{1}{F_1(n_d)} \quad (6)$$

$$Sel_i \propto F_2(n_d) \quad (7)$$

Equation 6 and 7 show that the number of standing queries per sensors goes down as we increase the number of monitored *PDT* dimensions. Selecting the right set of monitored dimensions has a significant impact on the output throughput. Section 4 focuses on selecting the right set of dimensions to monitor the appearance and disappearance of phenomena.

4. DIMENSIONALITY REDUCTION

The efficiency of a multi-dimensional phenomenon-aware system relies on the careful choice of the *monitored dimension set*: (1) how many dimensions to monitor and (2) which dimensions out of the full set of dimensions to select. In this section, we address the choice of the *monitored dimension set* (*MDS*) as follows:

- **Step1:** Given a *MDS* of size n_d, what is the optimal choice of another *MDS'* of size n_d*-1*
- **Step 2**: Given an initial *MDS* of size n_d*.initial*, what is the optimal choice of a desired *MDS* of size n_d*.desired*

Step 1 is a single step dimensionality reduction process that reduces n_d one dimension at a time. Step 2 gives the stopping criterion of where to stop the dimensionality reduction process. The remainder of this section presents two algorithms to perform step 1. Section 4.1 presents an algorithm that eliminates dimensions with no (or "few") phenomena that are of interest to standing queries and introduces a concrete measure for what we mean by "few" phenomena. Section 4.2 eliminates dimensions that are functionally dependent on other dimensions. Section 4.3 presents the stopping criterion for the dimensionality reduction process.

4.1 Eliminating dimensions with no phenomena

Running the *PDT* modules on dimensions that have no (or few) phenomena incur additional overhead on the sensor's CPU without any significant reduction in the query deployment map. We quantify the effectiveness of running a *PDT* module on a given dimension in terms of the sensor's expected throughput. We make use of two input parameters that are computed continuously by the PDT module:

- Sensor Participation Ratio (*SPR*): the average number of phenomena a single sensor participates in divided by the total number of phenomena.
- Tuple Participation Ratio (*TPR*): the average number of tuples that participate in a phenomenon divided by the total number of tuples.

Also, based on the systems workload, we assume the availability of the following parameters:

- Query-Per-Phenomenon (*QPP*): the average of number of queries interested in each given phenomenon.
- Phenomenon-Per-Query (*PPQ*): the average of number of phenomena that each query is interested in.

SPR quantifies the phenomena a sensor will be participating in, while QPR represents how many query are interested in each phenomenon. The product of *SPR* and *QPR* gives us the expected number of queries that are expected to run on a single sensor. Thus,

$$n_q = SPR \times QPP \quad (8)$$

The expected selectivity of each query is the number of phenomena that are of interest to the query multiplied by average percentage of tuples that participate in a phenomenon.

$$Sel_i = TPR \times QPR \quad (9)$$

We substitute for n_q and Sel_i using equations (8) and (9) in equation (5) to evaluate the expected throughput for each dimension. Then, we eliminate the dimension with the least expected throughput. Note that for the correctness of the equations above, there are two implicit assumptions: (1) Assume that sets of queries that are interested in $phenomenon_i$ and $phenomenon_j$, respectively, are disjoint. This assumption allows us to assume that the overall selectivity is the summation of all individual query selectivities. This is not always a valid assumption in practice but it simplifies the model without much distortion to the expected behavior. Accounting for overlap in the query's interesting set of phenomena is possible and is omitted for brevity. (2) Assume that every query is interested in one or more phenomena. This allows us to exclude the cases where some queries are interested in sensor values that are not detected to be part of any phenomena since such queries will typically be deployed over all sensors anyway.

4.2 Eliminating functionally dependent dimensions

Generally speaking, in multi-attribute sensors, the trend of one attribute readings is related to the trend of another attribute reading by a functional correlation *F* (i.e., $Attr_1.value = F(Attr_2.value)$). Such correlations have been previously exploited to develop model based approximate query processors [14]. For simplicity, we limit the correlation function *F* to monitor correlations to follow a linear relationship on the form of:
$Attr_1.value = a \times Attr_2.value + b$, where a, b are constants. We assume that linear correlation is sufficient for a wide range of applications. However, extending the model to higher orders of correlation is also straightforward. Since, deriving the functional dependence between attributes is not the focus of this paper, we assume that if such dependencies exist they can be identified and expressed within the PDT process. In the remainder of this section, we answer two interesting questions: First, given two

correlated dimensions, which dimension is to be removed to increase efficiency? Second, given that we removed the dimension $Attr_1$ (say) which is correlated with $Attr_1$, how the phenomenon-aware query optimizer could use $Attr_1$ to filter the query predicates against $Attr_2$.

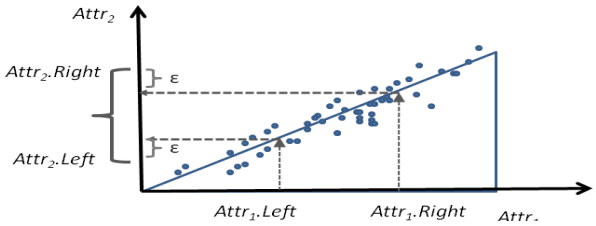

Figure 2. Transforming predicates across correlated dimensions.

Between two correlated dimensions, we eliminate one dimension or the other based on the throughput equation (Equation 5). We evaluate the throughput of the system assuming that $Attr_1$ has been removed. We repeat the throughput evaluation assuming that $Attr_2$ has been removed. Then, we decide to eliminate the attribute whose removal increases the throughput the most. In other words we remove the dimension whose phenomenon detection cost is higher, or the dimension with tuples that satisfy fewer query predicates.

Given a query predicate on Attr1 on the form of $Attr_1.Value_{Left} < x < Attr_1.value_{Right}$, where $Attr_1.value = a \times Attr_2.value + b$ and where $Attr_1$ has been eliminated from the system. Figure 2 shows that the predicates can be transformed to the other dimension. Then, the query interest in the detected phenomena is carried over to the retained dimension.

$$Attr_2 = \frac{Attr_1 - b}{a} \pm \varepsilon, \text{where } \varepsilon \text{ is the correlation error}$$

4.3 The stopping criterion for dimensionality reduction

Unfortunately, it is not trivial to correlate all dimensions with each other leading to an optimal set. Hence we suggest the following strategy to determine the stopping criterion for determining the optimal size of the monitored set.

Step 1: Start with d dimensions. Measure the throughput in Equation 5.

Step 2: Reduce one dimension ($d_{new}=d-1$) where no phenomena are found or where correlation is detected. Measure the throughput again.

Step 3: If the

$\Delta thruput = Thruput_{output}(d) - Thruput_{output}(d-1) < \varepsilon, where\ \varepsilon \approx 0$, then we stop iterating, otherwise perform step 2 again.

This strategy is easy to implement and meets the needs of incrementally reducing the dimension set.

5. CONCLUSION

In a geographically distributed sensor network, processes that exhibit similarity in behavior over time are termed as phenomena. We can obtain significant efficiency gains by developing systems capable of detecting and tracking phenomena occurring in multiple dimensions and deploying queries intelligently to run on limited nodes that participate in the phenomenon of interest. Moreover, functional correlations between dimensions can be exploited to further improve the performance of such query processors.

In this paper, we presented a novel *n-d*imensional *P*henomenon *D*etection and *T*racking mechanism that reduces both the number of sensors and the number of dimensions over n-ary sensor readings on which a continuous query is deployed. We performed dimensionality reduction from n to n' by dropping dimensions where no meaningful phenomena were detected. We then reduce the dimensionality further, from n' to n'' by detecting various forms of functional dependencies amongst the phenomenon dimensions. We then enhance the performance of this dimensionality reduction by making it an adaptive continuous process that dynamically determines the number of monitored dimensions. We addressed the design issues for each of these advances and developed the metrics that can be used to judge the utility of the proposed system.

6. REFERENCES

[1] Mohamed H. Ali, Mohamed F. Mokbel, Walid G. Aref, Ibrahim Kamel: Detection and Tracking of Discrete Phenomena in Sensor-Network Databases. In *SSDBM*, 2005.

[2] Mohamed H. Ali, M. F. Mokbel, and W. G. Aref. Phenomenon-aware stream query processing. In *MDM*, 2007.

[3] Mohamed H. Ali, Walid G. Aref, Ibrahim Kamel: Scalability Management in Sensor-Network PhenomenaBases. In SSDBM, 2006.

[4] Mohamed H. Ali, Walid G. Aref, Raja Bose, Ahmed K. Elmagarmid, Abdelsalam Helal, Ibrahim Kamel, Mohamed F. Mokbel: NILE-PDT: A Phenomenon Detection and Tracking Framework for Data Stream Management Systems. In *VLDB* 2005.

[5] Goce Trajcevski, Hui Ding, Peter Scheuermann, Roberto Tamassia, Dennis Vaccaro: Dynamics-aware similarity of moving objects trajectories. In *GIS*, 2007.

[6] Guang Jin, Silvia Nittel: Tracking deformable 2D objects in wireless sensor networks. In *GIS*. 2008.

[7] Liqian Luo, Aman Kansal, Suman Nath, Feng Zhao: Sharing and exploring sensor streams over geocentric interfaces. In *GIS*, 2008.

[8] M. A. Hammad, M. F. Mokbel, Mohamed H. Ali, and et al. Nile: A query processing engine for data streams. In *ICDE*, 2004.

[9] Mohamed H. Ali, Walid G. Aref, Cristina Nita-Rotaru: SPASS: scalable and energy-efficient data acquisition in sensor databases. In *MobiDE*, 2005.

[10] P. Kalnis, N. Mamoulis, and S. Bakiras. On discovering moving clusters in spatio-temporal data. In *SSTD*, 2005.

[11] R. Nowak and U. Mitra. Boundary estimation in sensor networks: Theory and methods. In *IPSN*, 2003.

[12] Crossbow, Inc. Wireless sensor networks. http://www.xbow.com

[13] A. Helal, H. Zabadani, J. King, Y. Kaddoura, and E. Jansen. The gator tech smart house: A programmable pervasive space. *Cover Feature, IEEE Computer*, 38(3):64–74, March 2005.

[14] A. Desphande, C. Guestrin, W. Hong, and S. Madden. Exploiting correlated attributes in acquisitional query processing. Technical report, Intel-Research, Berkeley, 2004.