Bayesian Reasoning for Sensor Group-Queries and Diagnosis

Ankur Jain[1], Edward Y. Chang[2], and Yuan-Fang Wang[3]

[1] Computer Science UC Santa Barbara, CA 93106 ankurj@cs.ucsb.edu
[2] Electrical and Computer Engg. UC Santa Barbara, CA 93106 echang@ece.uscb.edu
[3] Computer Science, UC Santa Barbara, CA 93106 yfwang.cs.ucsb.edu

Abstract. As large-scale sensor networks are being deployed with the objective of collecting quality data to support user queries and decision-making, the role of a scalable query model becomes increasingly critical. An effective query model should scale well with large network deployments and address user queries at specified confidence while maximizing sensor resource conservation. In this paper, we propose a *group-query* processing scheme using Bayesian Networks (BNs). When multiple sensors are queried, the queries can be processed collectively as a single group-query that exploits inter-attribute dependencies for deriving cost-effective query plans. We show that by taking advantage of the *Markov-blanket* property of BNs, we can generate resource-conserving group-query plans, and also address a new class of *diagnostic queries*. Through empirical studies on synthetic and real-world datasets, we show the effectiveness of our scheme over existing correlation-based models.

1 Introduction

Sensor network research is strategically positioned at the exciting confluence of sensing, computation, and communication. A sensor network can employ numerous small, inexpensive sensors of the same or different modalities (e.g., biological, chemical, mechanical, and electrical) to perform many useful tasks such as collecting seismic data, monitoring environment, measuring traffic flows, and safeguarding security, to name just a few. These sensors must be carefully managed to achieve two performance goals: 1) conserving power for prolonging useful life, and 2) collecting reliable data for supporting user queries, despite transient noise, sensor failures, and malicious attacks.

A query engine for a typical sensor network supports multiple users, where each user may query for probabilistic estimates of one or more sensor attribute values with some specified confidence level. Recent research efforts have shown that correlations are prevalent between sensor attributes, and queries on costly sensors can be answered efficiently by acquiring data from cheap sensors [9, 8]. In this paper, we advance the traditional correlation model in two directions. First, we employ Bayesian Networks (BNs) for characterizing sensor networks. BNs provide a compact representation of dependencies and offer effective inferencing methodologies. Such a model captures both the stochastic characteristics of, and the statistical relations between, sensor attributes. The use of BNs can provide efficient query-plan generation for traditional aggregate queries, as well as for diagnostic queries. Second, we consider the problem of *group-query processing*. When multiple queries are issued, instead of processing each query individually, we exploit inter-query relations to further reduce the overall resource usage.

To illustrate the advantages of using the BN over the traditional correlation model [9, 8], we present two simple examples generated using data from the National Data Buoy Center (NDBC) [25] and the Intel Lab [3], respectively. Figure 1 shows that a correlation model must maintain all pairwise correlations. In contrast, Figure 2(a) shows that the BN provides a much more compact representation of essential relations between sensor nodes. In general, for n attributes, with an average in-degree of d, the number of probability values required to represent the joint probability for a BN is $O(nd)$, in contrast to $O(n^2)$ required by the correlation model. Since $d \ll n$ for typical real-world sensor networks, the BN provides a much more succinct representation of a sensor network.

The work of [9, 8] successfully points out that a query on an expensive sensor can be answered using other inexpensive and statistically correlated sensors. When n is large, however, generating a good plan using such a correlation model can be time-consuming. The BN model can reduce the

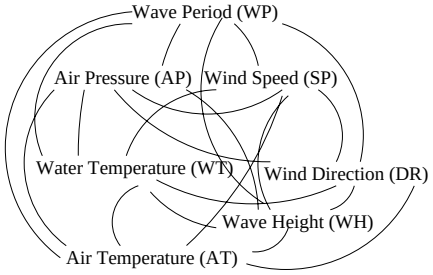
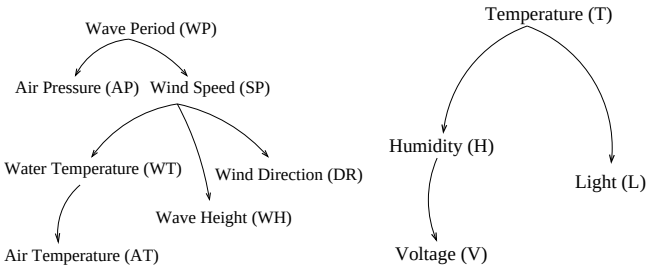


Fig. 1. Correlation model for NDBC data



(a) NDBC data (b) Intel Lab. data

Fig. 2. Compact representation using BN

search space of a node (representing a sensor query) from O(n) to the nodes in its *Markov blanket* [24], which consists only of the node's immediate parents, its immediate children, and other parents of its children. Let us revisit the example in Figure 2(a). Suppose a user queries "water temperature." Its Markov blanket tells us that an alternate plan should consider only "air temperature" or "wind speed," as they have the most direct and significant influence on "water temperature," instead of all attributes as shown in Figure 1. Moreover, a correlation model [9, 8] could make suboptimal choices in the plan. For example, in Figure 2(b) even though "voltage" is highly correlated with "temperature," "voltage" is conditionally independent of "temperature" given "humidity." Hence, given the value of "humidity", "temperature" has no effect on "voltage." More importantly, "humidity" could be a cheaper source to query than "temperature." The model of [9, 8] would miss this better choice in its query-plan generation. Thanks to the *Markov blanket* property of the BN, our model can not only efficiently reduce the search space for generating a query plan, but also generate a more effective plan. We can take further advantage of the Markov-blanket property to conduct *group-query* processing for making even more effective query plans. When the Markov blankets of multiple queries overlap, we are provided with more inter-query relations to further reduce the overall resource usage. For instance, Figure 2(a) shows that when two queries arrive over "water temperature" and "air temperature," respectively, the BN can tell us that we can treat these two queries as one *group query* because of their overlapping Markov blankets. The BN model thus can offer scalable performance to a large number of queries, as well as a large number of sensors.

In addition to supporting traditional *aggregate queries* in a more efficient and effective way, the employment of BN enjoys two additional benefits. First, a new class of *diagnostic queries* can be supported. A BN can readily use the sensor dependencies to determine the cause of abnormality in sensor data. This is because from the sensor-attribute relationships, we know the dependencies between different attributes, which can be used to detect an exception and then determine its type. Let us revisit Figure 2(a). The attribute "air temperature" is dependent on "water temperature." When readings of the "air temperature" sensor are out of its normal range, the BN tells us that we can query the values of "water temperature" (i.e., the only node in its Markov Blanket) to determine whether the "air temperature" is truly abnormal, or the air-temperature-sensor has malfunctioned. If the readings of "water temperature" are also "abnormal," but air- and water-temperature readings exhibit high conditional probability, then we can say that the air-temperature-sensor is normal and that the air temperature is abnormal. A traditional correlation-based scheme must verify the correlations with all sensors, and thus does not scale well. Second, the intuitive BN representation can be used to identify "hot spots" and replicate inexpensive and highly acquired sensors to improve both query efficiency and network reliability. For example, a sensor attribute with a high node degree (say "wind speed" in Figure 2(a)) in the BN and with low acquisition cost is likely to be acquired more often as it provides information about many other nodes at a reduced cost. Since such a *hot* node is likely to experience heavy network traffic, it should be replicated in the network.

In summary, this paper makes the following three important contributions:

1. We propose a compact and accurate abstraction of sensor networks based on the BN model. The employment of BN also permits us to tap into the work of sequential BN update, which can adapt BN structure and parameters to newly arrived data.

2. We devise a query plan generation algorithm that can save precious communication and computation resources in answering *group queries* with desired accuracy. Our experimental results show that our proposed scheme outperforms the correlation-based model for exploiting sensor dependencies (thanks to the Markov-blanket property) to conserve resources.
3. In addition to traditional aggregate queries, our model can answer a new class of *diagnostic queries* for fault detection and data inference, and can also assist sensor deployment and configuration.

The rest of the paper is organized as follows. We discuss related work in Section 2. In Section 3, we describe our sensor network architecture showing how a sensor network in its most general form can be represented as a BN and can be used to address queries. Section 3.2 presents the details of our query-plan-generation algorithm using the Markov-blanket property. We describe adaptations of Bayesian inference mechanisms to address general user queries as well as diagnostic queries. The experimental testbed and validations are described in Section 4. Section 5 presents concluding remarks.

2 Related Work

Our work is most related to the research work proposed in [9, 10]. The authors propose to learn a probabilistic model that captures the correlations that might exist between different sensor attributes. The learned model then aids in generating plans for answering queries at a lower cost. Such a correlation model builds one joint distribution table over all the sensor attributes and infers the probabilistic value of an attribute by conditioning all the other attributes on it. This approach does not scale well, as we have pointed out in Section 1. Our BN offers a compact representation, which not only makes query-plan generation efficient and scalable, but also allows *query grouping* to conserve even more resources.

Approximate query answering methods such as approximate caching [23] or DKF [17] can also be used to predict the value of a queried variable within bounded thresholds using the temporal dependencies. However, such schemes continuously monitor the streaming variable (to check if it is within the bounds) making them less effective in terms of energy conservation.

The probabilistic approach of query answering has also been studied in the recent past for moving object databases in [4]. While these efforts can produce effective solutions for providing probabilistic query results (typically object location) over imprecise data, their solutions are not directly applicable to sensor networks that operate in a resource-constrained environment. Furthermore, these solutions are limited to query inference, whereas we also present algorithms to generate query plans using a confidence-driven principle.

Recent works in [20, 26] propose sensor-database query models for the declarative, SQL-like query paradigm. The focus of these efforts is also to maximize in-network query processing to reduce sensor resource usage while still meeting the query precision specifications. These query models do not exploit sensor-attribute dependencies and hence cannot handle *group-query-plan generation* or *diagnostic queries* efficiently.

BNs have been used in traditional database systems mainly for attribute-selectivity estimation [13] and data-mining purposes. For sensor networks, BNs have primarily been used for sensor fusion and estimation [21]. Recent work in [2] employs BNs for probabilistic inference in sensor networks to determine if a sensor is surrounded by *enemy agents*. Our work, to the best of our knowledge, is the first that employs BNs for sensor-query processing.

BNs have found applications in diverse fields like biology, computer vision, computer software and decision support systems to name a few [22]. Since it interests a lot of different communities, BN research has been looked into from the perspectives of structure learning [11, 16], parameter estimation [12], efficient inference schemes [15, 19, 24], batch-mode and online updating of structure and parameters [1, 11]. Due to space limitations we cannot have a detailed discussion of all of them however we provide pointers to interested readers.

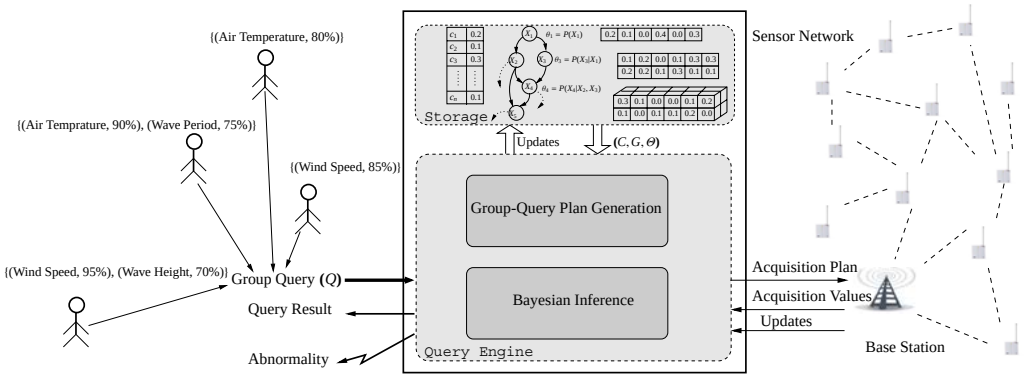


Fig. 3. Sensor Network Architecture

3 Architecture and Model

Figure 3 presents a typical sensor-network architecture, which consists of sensors (on the right-hand side), a query engine (in the middle), and queries issued by multiple users (on the left-hand side). The queries are formulated into a group-query **Q**, which we define as follows:

Definition 1**: Group Query**. *A group query* **Q***, in a sensor network of n attributes can be represented as:*

$$\mathbf{Q} = \{(X_i, \delta_i) | (X_i \in \mathbf{X}) \wedge (0 \leq \delta_i \leq 1) \wedge (1 \leq i \leq n)\} \text{ s.t.}, \delta_i < max_l P(X_i = x_{il}) \quad (1)$$

where $\mathbf{X} = \{X_1, X_2, X_3, \cdots, X_n\}$ *is the set of sensor attributes,* δ_i *is the confidence requirement for reporting the value of* X_i *and,* $P(X_i = x_{il})$ *is the probability with which* X_i *assumes a value of* x_{il}*. (A variable* X_i *in the* BN *is discretized into* k_i *bins. The subscript variable l takes values such that* $1 \leq l \leq k_i$*.)*

For group-query **Q**, the query engine generates a query plan, and then acquires data from the sensors. To generate a query plan for **Q**, the query engine consults the BN, which models the sensor network as a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$. The vertex set **V** is a set of random variables $\{X_i\}$ (one for each sensor), and the directional edge set **E** captures the statistical relations between sensors. **E** is a subset of $\mathbf{e}_{X_i,X_j}$, where $\mathbf{e}_{X_i,X_j} = \{(X_i, X_j), 1 \leq i, j \leq n, i \neq j\}$, represents the *influence relation* between (X_i, X_j). The cost set $\mathbf{C} = \{c_1, c_2, c_3, \cdots, c_n\}$ holds the acquisition cost associated with each node. The values in the cost set are functions of the routing and sensing costs. Changes to the BN and the cost sets (though rare) are updated when necessary. Based on the BN and data-acquisition costs, the query engine generates a query plan for group query **Q** such that the confidence requirement (δ_i) in reporting the values of queried attributes (X_i) can be met while consuming minimum sensor resources.

In the remainder of this section, we first briefly present the BN construction details. We then present our approach to generate efficient query plans for addressing group-queries. Finally, we present how BN can be used to answer traditional aggregate and diagnostic queries. Due to space constraints, we present only the critical steps of our methods. Detailed descriptions can be found in [18].

3.1 Bayesian Network Construction

A BN in its most general form consists of two parts: *model structure* and *model parameters*. The *model structure* is a directed graph in which the vertices are random variables and the directed edges represent the causal relationships between variables. To reduce the learning and inference complexity on BNs [11], we restrict the graph structure to be a polytree, which is a Directed Acyclic Graph (DAG) where an internal node can have any number of parents and children as long as there is no cycle in the corresponding undirected graph. The *model parameters* are mainly the Conditional Probability Tables (CPTs), where each CPT summarizes the conditional probability distribution of a nodal variable given its parents. The CPT of a node without any parent is its prior.

Bayesian inference uses a likelihood definition which states that a good model is one that is likely and can explain data well. Whether a model fits data well can usually be validated objectively based on a cost function. In information theory [6], causality is measured by the reduction in the

uncertainty or *entropy* of a random variable, given others. Hence, a reasonable definition of a good fit (a reasonable cost function) for a Bayesian network is the total entropy reduction (or the amount of entropy left) given the set of chosen edges. For any variable X_i, the entropy $H(X_i)$ is defined as follows:

$$H(X_i) = -\sum_{l=1}^{k_i} P(X_i = x_{il}) * log(P(X_i = x_{il})). \quad (2)$$

Structure learning is mathematically a constrained optimization problem: maximizing entropy reduction subject to the desire of using a simple network. We therefore construct our Bayesian tree minimizing the total entropy in the network($\sum_{i=1}^{n} H(X_i|Parents(X_i))$), where $H(X_i|Parents(X_i))$ measures the entropy of X_i given its parents. We first construct a completely connected digraph such that $w_{X_i,X_j} = H(X_i|X_j)$ where w_{X_i,X_j} is the weight of the edge e_{X_i,X_j} and then build a directed minimum spanning tree in $O(n^2 log(n))$ time using the algorithm described in [5].

After the network structure has been learned, the remaining uncertainty lies only in specifying the CPTs (parameter learning). We model conditional distributions in a CPT using probabilistic distributions in the exponential family (e.g., binomial, multinomial, and beta distributions for discrete random variables, or Gaussian and multivariate-Gaussian distributions for continuous variables) [16]. The advantage of using distributions in the exponential family is that a *closed-form* solution to parameter learning problem can be obtained using either maximum-likelihood or Bayesian learning.

There have been many independent research efforts on algorithm development for learning a BN model (e.g., [11]), and for conducting Bayesian inference (e.g., [7, 16, 24]). Since the focus of this work is to use the BN to conduct sensor queries, but not to devise new BN-related algorithms, we employ representative algorithms for BN generation [5] and inference [4].

3.2 Query Plan Generation

We now present our approach for generating group-query plans, and our techniques for addressing *diagnostic queries*. One of the most important properties of the BN that forms the theoretical foundation of our proposed algorithms is the conditional independence on the *Markov blanket* [24] (also called Markov Condition), which is defined as follows:

Definition 2**: Markov Blanket**. *Given a* BN *G, the Markov blanket* $MB(X_j)$ *of a node* X_j *in G is the set of nodes that are made up of* X_j*'s parents, its children and other parents of its children.*

Given $MB(X_j)$, X_j is conditionally independent of all other variables in G. This can be mathematically represented as shown in Equation 3:

$$P(X_j|G) = P(X_j|MB(X_j)) = \frac{P(X_j, MB(X_j))}{P(MB(X_j))}. \quad (3)$$

A straight result from Definition 2 is that variables outside the Markov blanket of a variable are not directly relevant for inferring its state once the values of *all* the Market-blanket variables are known. However, the influence of one node on any other node in a BN monotonically decreases (entropy monotonically increases) as the node distance between them increases. This observation is formally stated in the following lemma: **Lemma** 1**:** Given a node X_i and a set of arbitrary nodes $\mathbf{Y}$ in a BN s.t. $MB(X_i) \not\subseteq \{Y \cup X_i\}$, the conditional entropy of X_i given $\mathbf{Y}$ is at least as high as that given its Markov blanket or $H(X_i|\mathbf{Y}) \geq H(X_i|MB(X_i))$.

Proof: We exploit the property of conditional entropy which states that any additional information about a variable cannot cause an increase in its entropy [14], i.e. $H(X_i|X_j, X_k) \leq H(X_i|X_j)$ *or* $H(X_i|X_k)$. Further, separating $MB(X_i)$ into two parts: $\mathbf{MB_1} = MB(X_i) \cap \mathbf{Y}$ and $\mathbf{MB_2} = MB(X_i) - \mathbf{MB_1}$, and denoting $\mathbf{Z} = \mathbf{Y} - MB(X_i)$, we have:

$$\begin{aligned}
H(X_i|\mathbf{Y}) &= H(X_i|Z, \mathbf{MB_1}) && \because\ Y = \mathbf{Z} \cup \mathbf{MB_1} \\
&\geq H(X_i|\mathbf{Z}, \mathbf{MB_1}, \mathbf{MB_2}) && \because \text{Additional information cannot increase entropy} \\
&= H(X_i|\mathbf{Z}, MB(X_i)) && \because\ MB(X_i) = \mathbf{MB_1} \cup \mathbf{MB_2} \\
&= H(X_i|MB(X_i)). && \because \text{Markov-blanket definition}
\end{aligned}$$

Equipped with lemma 1 and the conditional independence property of the Markov blanket shown in Equation 3, we can now present our group-query plan generation algorithm. A group-query plan

[4] We employ Pearl's message passing [24] algorithm for Bayesian inference.

replaces the original queries with queries on alternate sensors that can meet the specified query confidence-levels at a reduced cost. Which sensors to query should depend on at least two factors: how expensive it is to query a particular sensor, and how much information the new sensor data can provide about the queried attributes. While the answer to the first question is readily available from the cost set **C**, the answer to the second question derives from Lemma 1: For a query attribute X_i in G, its state is influenced directly by every variable in its Markov blanket. Thus, to improve the confidence level of X_i, we analyze attributes from its Markov blanket. Since variables in the Markov blanket can in turn be affected by variables in *their* Markov blankets we might need to analyze Markov blankets recursively. However, since the confidence level typically dies out quickly beyond the immediate Markov blanket of the query object, recursive attribute-analysis rarely takes place in practice.

We consider a confidence-driven query paradigm where a group-query **Q** may query several attributes, each with some minimum confidence requirement. The amount of information that sensor $\mathbf{X}_i$ (in the Markov Blanket of a queried sensor) can provide about another sensor $\mathbf{X}_j \in \mathbf{Q}$, is quantitatively available as the conditional entropy reduction i.e. $H(\mathbf{X}_j) - H(\mathbf{X}_j|\mathbf{X}_i)$. Thus, we decide upon which sensors to query, by selecting them in a greedy fashion so to maximize the overall entropy reduction at least possible acquisition cost, such that the confidence requirement of all queried attributes are met. The details of this sensor selection algorithm can be found in [18]. Once the states of the variables (or sensor attributes) to be acquired are available, we can use that information to answer a wide variety of queries. The ability to answer a variety of queries will hinge upon one critical element: the ability to determine the likely state of a BN. As the BN description is essentially a probabilistic one, the state descriptions will also be expressed in terms of a likelihood function conditioned upon prior, current state, and sensor data.

We partition the set of nodes in a BN into three classes: the set of queried attributes (denoted as $\mathbf{X_q}$), the set of attributes whose values are known (denoted as $\mathbf{X_e}$), and the rest of the attributes (denoted as $\mathbf{X_h}$). Both $\mathbf{X_e}$ and $\mathbf{X_h}$ can be empty. In the case that $\mathbf{X_e}$ is empty, we do not have any sensor data, so the inference will rely completely on the prior (or historical data and trend). In the case that $\mathbf{X_h}$ is empty, all sensors are to be queried, and we obtain very detailed knowledge of the network to make inferences. In other cases (which are the typical cases), $\mathbf{X_q}$ is obtained from user query **Q**, and $\mathbf{X_e}$ is obtained from Υ. The probability of a queried variable $X_i \in \mathbf{X_q}$ can then be obtained as:

$$P(X_i|\mathbf{X_e}, G) = \frac{P(\mathbf{X_e}|X_i, G)P(X_i|G)}{P(\mathbf{X_e}|G)}, \tag{4}$$

where $P(X_i|G)$ is the prior probability of the queried variable, $P(\mathbf{X_e}|G)$ is the marginal probability of the evidence, and $P(\mathbf{X_e}|X_i, G)$ is likelihood[5].

The *pdf*s of the queried attributes obtained in Equation 4 (as a result of Bayesian inference) can then be used to answer a variety value and aggregate queries (refer [8] for details). For example, a range query to compute the probability of $\mathbf{X_i}$ lying in range $[l_i, u_i]$ can be computed as: $P(X_i \in [l_i, u_i]) = \int_{l_i}^{u_i} p(x_i)dx_i$.

3.3 Answering Diagnostic Queries

We perceive data abnormality as an event whose likelihood is suspiciously small, given the historical trends and current network state. For example, in the NDBC datasets, if the historical trends suggest that "sea temperature" is always 5 – 10% lower than the "air temperature," then situations would be abnormal when the temperature difference between the two sensors exceeds the threshold. The two main reasons for such abnormality are:

- Failures: A fault in the sensor or communication mechanism causes arbitrary values to be reported at the server, or
- Emergence or dissolution of statistical relations: New attribute dependencies may have evolved affecting the historical likelihood of events.

[5] A variety of real-time Bayesian inference algorithms are available [15], and the choice of a particular inference algorithm is beyond the scope of this paper.

Suppose the BN topology and the CPTs do not change. A naive method for detecting abnormality is to bound the expected sensor values or attribute correlations, and reporting abnormalities if the observed measures fall out of bounds. This approach does not work well for at least three reasons. First, this method is not scalable in large networks where an attribute is dependent on many other attributes. Second, historical data can reveal that high temperature differences *might* occur. Third, an abnormal reading on one sensor in an extreme weather condition can be an accurate reading, not necessarily resulting from the sensor's failure or a communication fault. Therefore, abnormality depends on the joint likelihood of several events *under a given* BN *state*.

We propose a *trigger & verify* approach to detect abnormalities in our sensor network architecture. Each time a value from a sensor is received at the query evaluator, it checks to see whether such a value is likely to be seen under the *current BN state*. This likelihood measure is available at no extra cost from the Bayesian inference engine (shown in Equation 4). If the likelihood measure for any attribute X_i is suspiciously small, the query evaluator *triggers* a request for abnormality diagnosis at each of the nodes in $MB(X_i)$. All nodes receiving a request for abnormality diagnosis capture a continuous sequence of data values and compute the likelihood of observing such a sequence as follows:

Suppose sensor X_j, discretized into bins $\{b_{j1}, b_{j1}, b_{j3}, \cdots, b_{jk_j}\}$, is serving an abnormality diagnosis request. It first captures a sequence S_j of continuous observation[6] and then obtains the event counts, i.e., the number of times X_j falls in bin b_{j1} and so on. Let the counts be denoted by $\{\alpha_{j1}, \alpha_{j2}, \alpha_{j3}, \cdots, \alpha_{jk_j}\}$, then the likelihood of observing sequence S_j given G is

$$L(S_j|G) = (\sum_{i=1}^{k_j} \alpha_{ji})! \prod_{i=1}^{k_j} \frac{(p_{ji})^{\alpha_{ji}}}{\alpha_{ji}!} \quad , \tag{5}$$

where, $p_{ji} = P(x_j = b_{ji})$ and $\sum_{i=1}^{k_j} \alpha_{ij} = |S_j|$. If $L(S_j|G)$ is small enough then the event is *verified* as abnormal and is reported back to the server. A direct consequence of such a trigger & verify approach is that 1) a trigger generated due to transient communication breakdown will not be verified as an abnormality, 2) broken sensors (those with a faulty sensing device) will verify abnormalities, and 3) sensors in the Markov blanket of the broken sensor will not verify the abnormality.

If an abnormality is detected in some sensor readings, the next logical questions to ask are "What caused the abnormality?" and "How will the abnormality affect other sensor readings?" While we already addressed the first question, the second question can be addressed using the *model structure* on the BN. When an X_j sensor is detected as broken, we infer its value from its Markov blanket in the BN graph. Furthermore, X_j is no longer used to infer values of variables that contain X_j in their Markov blankets.

4 Experimental Validation

We evaluated the performance of our query engine using BN on three datasets (depicted in Section 4.1). In this paper, our experimental analysis is organized into four parts, we have a more detailed analysis available in [18]. The first experiment examined the effect of group query size and confidence requirements respectively, on resource conservation (Section 4.2). We also compared the resource savings against those obtained using correlation model under varying query-confidence levels (Section 4.2). In the second experiment we analyzed the query answer quality achieved using our proposed approach (Section 4.3). The third experiment analyzed the abnormality detection ability of our proposed model (Section 4.4). The last experiment studied selectivity of attributes under different query conditions (Section 4.5).

4.1 Experiment Setup

We used two real datasets and one synthetically generated, as described below:

[6] Here, we make two simplified assumptions on S_j. First, S_j is a sequence of independent, identically distributed random variables forming a multinomial distribution. Second, the state of the network shows little or no variation for the duration of collecting S_j.

NDBC Data

Attribute	WP	AP	SP	WT	AT	WH	DR
*Rel. Cost (***C**)	0.14	0.08	0.22	0.27	0.09	0.10	0.10

Intel Data

Attribute	T	H	L	V
*Rel. Cost (***C**)	0.328	0.328	0.344	0.001

Table 1. Relative Costs of Sensors for Real Datasets

- *NDBC Dataset* – This real-world dataset was obtained from the National Data Buoy Center (NDBC) [25]. The sensor network consists of numerous ocean buoys streaming data of different modalities every hour to a base station. The data have seven attributes: "average wave period" (WP), "air pressure at sea level" (AP), "wind speed" (SP) , "water temperature" (WT), "air temperature" (AT), "wave height" (WH) and "wind direction" (DR) with relative costs shown in Table 1. We used data from three buoys in the San Francisco area (Station IDs 46012, 46013, 46026) in all our experiments. Historical data dating from year 1981 to 2003 were used for learning (with discretization into 4 bins), and segments of year 2004 data were used for testing.
- *Intel Data* – This real dataset (also used in [9][8]) was obtained from the Intel Research, Berkeley Lab [3]. The data were collected using 54 sensors providing "temperature" (T), "humidity" (H), "light" (L) and "voltage" (V) measurements (relative costs are shown in Table 1). We used 50% of the data for testing after discretizing each attribute into 8 bins.
- *Synthetic Data* – The synthetic datasets were constructed for the purposes of rigorous testing and for evaluating the performance under ideal conditions. We tested the system on several such datasets; but due to space constraints we report results on one which had the following properties: The BN was generated with 50 nodes having a maximum node degree of eight. Each node was discretized into five bins, and the CPTs were generated according to the Dirichlet distribution [7] with $\lambda = 0.01$.

Cost functions associated with different attributes were available for the real-world datasets. For synthetic data we tested the performance for randomly generated cost functions.

Our testbed consisted of $1,500$ group queries for NDBC data, $5,000$ group queries for Intel data, and $2,500$ for synthetic data; all were selected randomly from the testing data such that no two successive queries were separated by more than 10 units of time. The random selection ensures that the results are not biased toward particular temporal query patterns. Once a sensor value is available at the central server, we let its uncertainty value grow exponentially with time, which is taken into account using Bayesian inference.

We first define a few parameters that were used in the experimental setup to evaluate the performance of the system:

- Group-query size ($|\mathbf{Q}|$) – The number of sensor attributes whose values are required by one or more users at a time. The maximum value of $|\mathbf{Q}|$ is the number of nodes in the BN (e.g. $max(|\mathbf{Q}|) = 7$ for NDBC data).
- Confidence requirement – (δ_{min}) – The confidence required in reporting the values of the attributes in $|\mathbf{Q}|$[8].

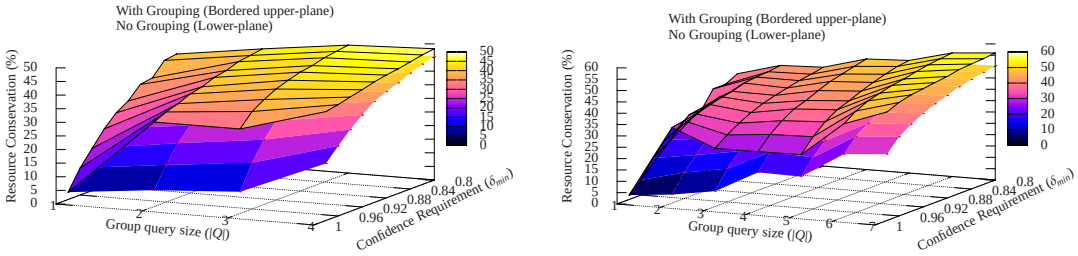
4.2 Resource Conservation

We define resource conservation as the percentage of the total resources saved in the sensor network to address all queries over the resource consumption if all the queried attributes in **Q** were to be acquired directly.

Effect of Grouping Queries In Figure 4, we compare the resource conservation achieved in addressing group queries using our group-query plan algorithm (the upper plane with solid lines) as opposed to processing them individually (the lower plane with dotted lines). We show the results for

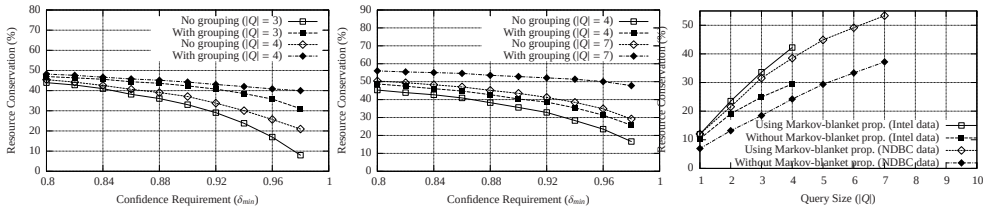
[7] If p_i is the probability of choosing a collection of items of size i from an item-set of size n, then $\{p_1, p_2, \cdots, p_n\}$ are the parameters of the multinomial distribution. The Dirichlet distribution with parameter λ is the conjugate prior on the parameters of the multinomial distribution. $\lambda \ll 1$ encourages "deterministic" CPTs (one entry near 1, the rest near 0), $\lambda = 1$ causes entries to be drawn from $U[0, 1]$ and $\lambda \gg 1$ causes all entries to near $1/k$ where k, is the discretization size [12].

[8] To maintain uniformity we always choose to query $|\mathbf{Q}|$ costliest attributes when the group-query size is less than its maximum value. We query all attributes in **Q** with the same value of δ_{min}. For example, for $|\mathbf{Q}| = 2$ in NDBC data, we always choose WT and SP and query both of them with 90% confidence for $\delta_{min} = 0.90$



(a) Intel Data (b) NDBC Data

Fig. 4. Resource conservation as a function of δ_{min} and |**Q**|



(a) Intel Data (b) NDBC Data

Fig. 5. Resource conservation as a function of δ_{min}

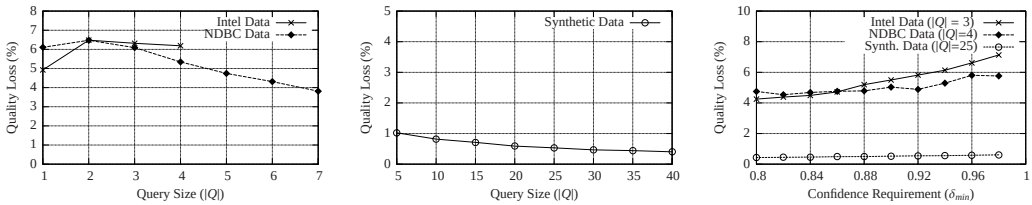
Fig. 6. Resource conservation with $\delta_{\mathbf{min}} = \mathbf{0.90}$

both real datasets as we vary both |**Q**| (from 1 to max(|**Q**|) and δ_{min} (from 0.80 to 0.98). In the figure, the vertical axis shows the resource conservation; the lighter the shaded color on the plane, the higher the resource conservation achieved. Note that even when issuing queries individually (group-query size |**Q**| = 1, along the left-edge of the figure), using BN inferencing can already save significant resources. (We will present 2D figures shortly to highlight some results.) When queries were grouped using our group-query algorithm, the resource conservation was more significant.

To facilitate a better view, Figure 5 provides a 2D view on the resource conservation with different group sizes. Figure 5(a) shows that when |**Q**| = 4 for the Intel dataset, the savings at various confidence levels are consistently achieve above 30%. This is because a larger group size provides the algorithm with more room to use the inter-attribute dependencies more effectively. On the NDBC data, Figure 5(b) shows that the savings can be above 50% when |**Q**| = 7 and $\delta_{min} \leq 0.96$.

BN vs. Correlation Model An important question to answer is "how does resource conservation of using BN compare with not using BN (or using the traditional correlation model)?" It is evident that using the correlation model incurs higher computational cost due to the model's larger search space for alternate sources. We were curious to find out whether the employment of BN could improve resource conservation; and if so, what factors contributed to the improvement.

We compared the resource conservation obtained for group-queries using our proposed approach against the one that uses a joint probability distribution over all the variables. This is similar to one proposed in [8] (which is equivalent to having completely connected graph structures as shown in Figure 1). Figure 6 shows the percentage of resource conservation on the two reallife datasets for $\delta_{min} = 0.90$. For Intel data, at |**Q**| = 1 (i.e. when there is no possibility of grouping queries), using BN conserves about 3% more resources than the correlation model. When we increase the group size, the conservation increases (at |**Q**| = 4, using BN achieves 12 additional percentile of conservation). A similar pattern is obtained from the NDBC dataset. The additional conservation achieved by BN is due to the fact that two correlated attributes might be conditionally independent on a *cheaper* attribute, a property that the Markov blanket decodes successfully, but the correlation-based model fails to decode. (Setting δ_{min} at different levels achieves the same result: BN outperforms the correlation model in resource conservation.)



(a) Real data $\delta_{min} = 0.90$ (b) Synthetic data $\delta_{min} = 0.90$ (c) Effect of δ_{min} on quality-loss

Fig. 7. Quality loss

4.3 Query Answer Quality-Loss

Our BN framework provides query answers based on probabilistic estimates. Therefore, as with the correlation model proposed in [8], 100% query precision cannot be attained. An effective query plan is one that conserves resources and meets the confidence values of the queries most of the time. We define *loss* as an event when the confidence of a queried attribute has not met the requirement after the execution of the query plan. For a particular **Q**, we calculated the *quality-loss* as the per-attribute mean loss percentage (or the percentage of the total losses over all attributes to the total number of times they were queried). Quality-loss depends on how tightly the data distribution fits the one used in the BN. If the data dependencies are modeled accurately in the CPTs, the quality-loss should not be affected adversely by $|\mathbf{Q}|$ and δ_{min}.

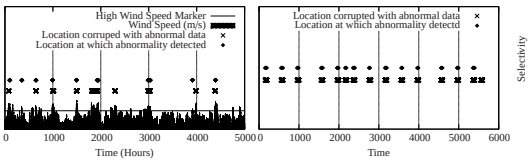
We show the quality-loss observed for different datasets in Figure 7 for $\delta_{min} = .90$. All other testing parameters were the same as described for the earlier experiments. Since real-data distribution can be modeled only to a certain degree of accuracy, we observe slight variations in quality-loss as we increase $|\mathbf{Q}|$. However, as seen in Figure 7(a), the loss always stays well under 7% in all cases. For the synthetic dataset (Figure 7(b)) we observe much less variation (always less than 1%), because the data distribution was modeled accurately. As discussed earlier, large $|\mathbf{Q}|$ allows our group-query algorithm to exploit sensor dependencies more effectively for producing stronger probabilistic estimates, and hence achieving lower quality-loss.

There is a trade-off between the quality-loss and the confidence requirement. High-confidence queries would cause the quality-loss to rise, though graceful degradation is desired. In another experiment, we studied the effect on quality-loss of increasing δ_{min}. The results obtained for all three datasets as we increased the confidence requirement from 0.80 to 0.98 are shown in Figure 7(c). The slow degradation shows the effectiveness of our approach, with a loss of less than 8% even at 98% confidence requirement for real datasets. The quality-loss for synthetic data always remains under 1%.

4.4 Abnormality

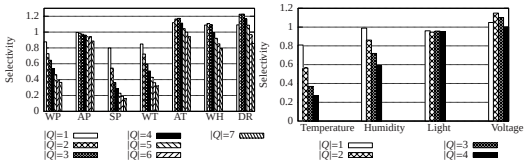
We tested the abnormality-detection ability of our model as proposed in Section 3.3. This abnormality experiment required domain knowledge on the data. We tested our proposed approach on the NDBC real-world dataset as follows: The NDBC facilitates the search for extreme weather conditions over its entire historical database. Since these extreme weather conditions are so classified by domain experts, we can safely tag them as "abnormal events." We searched for extreme "high wind" conditions over the entire historical dataset in San Francisco County and corrupted a normal testing dataset (used in the experiments described earlier) with 12 extreme conditions at randomly selected locations. We then modified the query-plan generation algorithm such that the "wind speed" attribute was always selected to be acquired by the query plan. We validated our approach by observing if (1) our algorithm could detect all the abnormal events, and (2) if it would correctly detect the time at which the abnormality occurred. Figure 8(a) shows the abnormality detection behavior of our algorithm with $|S_j| = 10$. The horizontal line shows the threshold for the discretization used for "wind speed" such that all values above it would fall in the same discretization bin. As seen in the figure, our model catches all the abnormal events except for one (the sixth from left) and reports one *normal* event (the second from left) as an abnormality.

Though there are many locations in the graph where the wind speed exceeds the threshold, our algorithm detects only those that were corrupted manually. Thus, our model is quite effective in de-



(a) NDBC data (b) Synthetic data

Fig. 8. Abnormality Detection



(a) NDBC data (b) Intel data

Fig. 9. Selectivity with $\delta_{\mathbf{min}} = \mathbf{0.90}$

tecting abnormalities and in reducing false positives. Abnormality-detection results for the synthetic dataset are shown in Figure 8. We generated a BN similar to the one used in testing but with a different Dirichlet distribution ($\lambda = 1$) for the CPTs. We corrupted the normal testing data with data sampled from the new BN for a randomly selected attribute at 15 different locations. Verification sequence was set to $|S_j| = 10$. As seen in the figure, our algorithm captures all the abnormalities except for one (the last one).

4.5 Selectivity

The selectivity of an attribute is the ratio of the number of times it is acquired to the total number of times it appeared in some query. The selectivity pattern can be extremely useful in improving network reliability and identifying "hot-spots" as discussed in Section 1. Sensors with high selectivity attributes should be replicated more in the network, and the communication overlay network could be adjusted so that energy consumption is reduced. A cheap attribute, with a high node degree in the BN graph is likely to show high selectivity since it lies in the Markov blanket of many other nodes and thus provides information about other nodes at a low cost. On the other hand, a costly attribute (with a high node degree) is likely to show low selectivity if there are other less inexpensive nodes lying in its Markov blanket. We show the selectivity behavior of the two real-world datasets in Figure 9. The experiment parameters were the same as those used in the resource conservation experiments. We expect to reduce the selectivity as we increase $|\mathbf{Q}|$, since as it allows scope for better optimization. As seen in Figure 9(b) the selectivity of "temperature" and "humidity"(having node degree 2) drop significantly as compared to "voltage". This is due to the fact the relative acquisition cost of "voltage" was very low, making its acquisition more frequent than costly attributes. Selectivity for "light" does not drop because it does not have any low cost attribute in its Markov blanket. Figure 9(a) shows the selectivity graph for the NDBC dataset. We observe that the selectivity of "speed" and "wave period" (costly attributes) drops significantly with the increase in $|\mathbf{Q}|$, as they have high node degrees. The node degree of "speed", being the highest (Figure 2(a)), shows the sharpest fall.

5 Conclusions and Future Work

In this paper, we have proposed using BNs for characterizing sensor networks for probabilistic query answering and sensor diagnosis. We proposed a greedy algorithm for saving sensor resources by grouping individual queries into one group-query. Our approach uses a Bayesian inferencing scheme which, in addition to providing probabilistic estimates of the queried variables, also provides effective methods for the sensor network diagnosis. The *pdf*'s of the queried variable can be used to address a wide range of value and aggregate queries. The BN structure also helps in improving the sensor network infrastructure by providing an intuitive model of the inter-attribute dependencies. Through examples and experiments on both real and synthetic datasets, we demonstrated that the BN is more effective in saving sensor resources than the previously proposed simplistic probabilistic models using correlations. Our model provides significant improvement in resource conservation of 15 – 20% over traditional models. We also showed the effectiveness of our model in capturing abnormalities and predicting attribute selectivity.

We plan to extend our work to address *what-if* queries. Bayesian inference allows us to reason about hypothetical scenarios given some counterfactual evidence. Such queries are called *what-if* queries. Such queries can be extremely useful predicting state of the network in hypothetical conditions to trigger alarms and issue warnings.

References

1. E. Bauer, D. Koller, and Y. Singer. Update rules for parameter estimation in Bayesian networks. In *UAI'97: Proc. of the Thirteenth Conf. on Uncertainty in Artificial Intelligence*, pages 3–13, August 1997.
2. R. Biswas, S. Thrun, and L. J. Guibas. A probabilistic approach to inference with limited information in sensor networks. In *IPSN'04: Proc. of the 3rd Intl. Symposium on Information Processing in Sensor Networks*.
3. P. Bodik, W. Hong, C. Guestrin, S. Madden, M. Paskin, and R. Thibaux. Intel lab data, http://db.lcs.mit.edu/labdata/labdata.html.
4. R. Cheng, D. V. Kalashnikov, and S. Prabhakar. Evaluating probabilistic queries over imprecise data. In *SIGMOD'03: Proc. of the 2003 ACM SIGMOD Intl. Conf. on Management of data*.
5. C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.
6. T. Cover and J. Thomas. *Elements of Information Theory*. John Wiley and Sons, Inc., 1996.
7. P. Dagum and M. Luby. Approximating probabilistic inference in Bayesian belief networks is NP-Hard. *Artificial Intelligence*, 60(1):141–153, 1993.
8. A. Deshpande, C. Guestrin, and S. Madden. Using probabilistic models for data management in acquisitional environments. In *CIDR'05: Proc. of 2nd Biennial Conf. on Innovative Data Systems Research*, Asilomar, CA, USA, January 4-7 2005.
9. A. Deshpande, C. Guestrin, S. Madden, J. M. Hellerstein, and W. Hong. Model-driven data acquisition in sensor networks. In *VLDB'04: Proc. of 30th Intl. Conf. on Very Large Data Bases*, Toronto, Canada, September 2004.
10. A. Deshpande, C. Guestrin, S. Madden, and W. Hong. Exploiting correlated attributes in acqusitional query processing. In *ICDE'05: Proc of 21st Intl. Conf. on Data Engineering*, Tokyo, Japan, April 5-8 2005.
11. N. Friedman and D. Koller. *Tutorial: Learning Bayesian networks from data*. NIPS'01: Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 2001.
12. A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian data analysis*. Chapman & Hall CRC, 1995.
13. L. Getoor, B. Taskar, and D. Koller. Selectivity estimation using probabilistic models. In *SIGMOD'01: Proc. of the 2001 ACM SIGMOD Intl. Conf. on Management of data*, pages 461–472, New York, NY, USA, 2001.
14. R. M. Gray. *Entropy and Information Theory*. Springer-Verlag, New York, 1990.
15. H. Guo and W. H. Hsu. A survey of algorithms for real-time Bayesian network inference. In *AAAI/KDD/UAI'02: Joint Workshop on Real-Time Decision Support and Diagnosis Systems*, Edmonton,Alberta, Canada, July 29 2002.
16. D. Heckerman. Tutorial on learning with Bayesian networks. Technical report, Microsoft Research, March 1995.
17. A. Jain, E. Y. Chang, and Y.-F. Wang. Adaptive stream resource management using Kalman filters. In *SIGMOD'04: Proc. of the 2004 ACM SIGMOD Intl. Conf. on Management of data*, pages 11–22, New York, NY, USA, 2004.
18. A. Jain, E. Y. Chang, and Y.-F. Wang. Efficient group and diagnostic queries on sensor networks (http://www.cs.ucsb.edu/ ankurj/bayesTR.pdf). Technical report, Computer Science, UC Santa Barbara, August 2006.
19. M. I. Jordan, Z. Ghahramani, T. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.
20. S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TinyDB: An acquisitional query processing system for sensor networks. *ACM Transactions Database Systems*, 30(1):122–173, 2005.
21. J. Moura, L. Jin, and M. Kleiner. Intelligent sensor fusion: A graphical model approach. In *ICASSP'03: Intl. Conf. on Acoustics, Speech, and Signal Processing*, volume 6, pages 6–10, Hong Kong, April 2003.
22. K. P. Murphy. *Dynamic Bayesian Networks: Representation, inference and learning*. PhD thesis, University of California, Berkeley, Fall 2002.
23. C. Olston, J. Jiang, and J. Widom. Adaptive filters for continuous queries over distributed data streams. In *Proc. of ACM SIGMOD Intl. Conf. on Management of Data*, San Diego, California, USA, June 2003.
24. J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.
25. N. W. Service. National Data Buoy Center, http://www.ndbc.noaa.gov/.
26. Y. Yao and J. Gehrke. Query processing in sensor networks. In *CIDR'03: First Biennial Conf. on Innovative Data Systems Research*, Asilomar, CA, January 2003.