On Sensor Selection in Linked Information Networks

Charu C. Aggarwal
IBM T. J. Watson Research Center
Email: charu@us.ibm.com

Amotz Bar-Noy
CUNY Graduate Center
Email: amotz@cuny.edu

Simon Shamoun
CUNY Graduate Center
Email: srshamoun@yahoo.com

***Abstract*—Sensor networks are often redundant by design in order to achieve reliability in information processing. In many cases, the relationships between the different sensors are known *a-priori*, and can be represented as virtual linkages among the different sensors. These virtual linkages correspond to an information network of sensors, which provides useful external input to the problem of sensor selection. In this paper, we propose the unique approach of using external linkage information in order to improve the efficiency of very large scale sensor selection. We design efficient theoretical models, including a greedy approximation algorithm and an integer programming formulation for sensor selection. Our greedy selection algorithm provides an approximation bound of $(e-1)/(2 \cdot e-1)$, where e is the base of the natural logarithm. We show that our approach is much more effective than baseline sampling strategies. We present experimental results that illustrate the effectiveness and efficiency of our approach.**

I. Introduction

In recent years, sensor networks have become very popular because of their numerous military and industrial applications. In many cases, a network may contain tens of thousands of sensors producing continuous streams of data. The large amount of data produced and transmitted is a challenge from a variety of cost-driven perspectives. For example, sensors have limited battery supplies, and so it is often desirable to minimize the total power consumption by the sensors.

Since the different sensor readings are often related, it is not necessary to use all sensors in the data collection process. In many cases, sensors have predictability relationships with each other that can be used in deciding which sensors to use to approximate the streams of others. For example, a bird call heard by one sensor will be replicated by another sensor nearby, albeit with some lag. Similarly, visual sensors located within a few meters of one another often produce almost the same data. Such relationships can be represented in the form of *logical*, *information-network*, or *virtual links* between sensors, some examples of which are as follows:

(1) Two sensors may be located a small distance apart. In such a case, a virtual link can be created between them that includes information about the distance between them.

(2) In fluid tracking applications, the topology of the pipe network provides external information about relationships of sensor readings at different nodes. In such cases, the pipe topology may be used to create virtual links.

When a sensor stream is not available, one or more of its linked sensors can be used to predict it with the use of numerical regression analysis [1], [2], [3]. The resulting loss of information is essentially the *error* of not using all the sensors. We note that the best choice of sensors depends on the strength of the relationships between the sensors and the topology of the corresponding relationships. The graph representations of such informative relationships between different data objects are referred to as *information networks* [4]. Information networks have proven to be a useful logical construct to analyze a wide variety of web and social network scenarios, though the approach has not been used in the context of sensor networks to the best of our knowledge. This paper takes a first approach to treating sensors as data objects and the corresponding relationships among them as information network relationships. We will see that such a systematic approach also allows us to derive a general approximation algorithm for the problem.

The goal of this paper is to design a *link- and data-driven* technique to determine which sensors are most critical for overall sensing quality, especially when the real-time budget constraints are tight. Such budget constraints can be resource constraints, such as power consumption, storage capacity, and bandwidth; or costs of operation, though the model is general enough to accommodate any kind of cost in the context of a sensor network. We design an approximation algorithm for this optimization problem and show how this solution overall can be used to make effective predictions about the data under cost-based budget constraints.

This paper is organized as follows. In Section II, we introduce the overall sensor selection problem in the context of linked information networks. In Section III, we study the problem of link error modeling, which is required in order to assign values to the variables of the optimization problem. In Section IV, we show that the problem is NP-hard, and design effective integer programming and approximation algorithms. Section V presents the experimental results. Finally, Section VII presents the conclusions.

II. The Sensor Selection Problem

In this section, we introduce the sensor selection problem in the context of linked information networks. We note that the network of relationships between sensors can be represented as a graph. In general, let the simple directed graph $G = (V, E)$ represent a network of n sensors and m logical links between sensors. Each vertex $v_i \in V$ represents a sensor and its associated stream, and each edge $e_{ij} = (v_i, v_j) \in E$ represents a logical link between sensors. These logical links can be designed on the basis of proximity or any other domain-specific property that affects predictions between sensors. In general, regression modeling [5] can be used to predict one sensor time series from another selected sensor. We note that it is computationally impractical to examine the behavior of all sensor pairs in order to make decisions about which sensors to select for the prediction process. For example, a network of 10^4 sensors contains 10^8 possible vertex pairs. Such a large number of possible node pairs can make the sensor selection problem computationally impractical. The logical links between sensors provide *external domain-specific information* which can *significantly reduce the computation required* in the selection process.

Each vertex v_i is associated with a cost of selection c_i, an importance u_i, and a prediction error q_i. Each edge e_{ij} is associated with a prediction error p_{ij}. We note that c_i and u_i are set in an application-specific manner. The cost c_i may reflect the consumption of power or some other resource by a sensor during the data collection process. The importance u_i reflects the importance of a sensor in providing useful information to the underlying application. The error of the regression when using v_i to predict v_j is denoted p_{ij}. The error of not predicting v_i at all is denoted q_i. The problem of selecting the optimal set of sensors in order to make predictions about all other sensors can be formulated either as a *minimization problem* or a *maximization problem.* In the minimization formulation, we minimize the weighted error of selecting a particular set of sensors. In the maximization formulation, we maximize the weighted *reduction* in error that arises from selecting a particular set of sensors over the trivial solution of not selecting any sensors at all.

The objective in the first (minimization) formulation of the problem is to select a set of sensors $S \subseteq V$ which minimizes the total prediction error of all streams and whose total cost does not exceed some budget constraint B. In this case, we define the *weighted non-predictability error* z_i as the importance-weighted error of not being able to predict sensor v_i at all; therefore, z_i is set to $u_i \cdot q_i$. Similarly, we create a weighted version of the link prediction error, which is set according to the importance u_j of the target sensor v_j. The weighted error w_{ij} of predicting sensor v_j from sensor v_i is set to $w_{ij} = u_j \cdot p_{ij}$. The idea here is that the error of predicting sensor v_j from sensor v_i is *magnified* by the importance u_j of sensor v_j. The objective function for a set of sensors S is calculated by iterating over each node v_j and computing its contribution to the total prediction error as follows: **(1)** If v_j is in S, it contributes 0 to the error. **(2)** If v_j is not in set S, but there is an edge from at least one vertex in S to v_j, then the error it contributes is the minimum weighted prediction error $w_{ij} = u_j \cdot p_{ij}$ of sensor v_j from any linked sensor $v_i \in S$. The implicit assumption here is that a one-to-one regression model is used from sensor v_i to sensor v_j to predict the stream in sensor v_j from the stream in sensor v_i. **(3)** If v_j is not covered by any linked sensors, then the error it contributes is the weighted error $z_j = u_j \cdot q_j$ of not predicting it at all.

In the second (maximization) formulation of the problem, the objective is to select a set of sensors $S \subseteq V$ that maximizes the total *reduction* in error and whose total cost does not exceed some budget constraint B. This is the dual of the minimization problem, since maximizing the total reduction in error is equivalent to minimizing the total error. The key difference is in the definition of the edge weights. The weight of edge e_{ij} is denoted by w'_{ij} and reflects the importance-weighted error *reduction* of sensor v_j when the stream at sensor v_i is used to predict it. Specifically, w'_{ij} is set to $u_j \cdot (q_j - p_{ij})$. We note that the implicit assumption here is that $p_{ij} \leq q_j$, since q_j represents the maximum error of sensor v_j, which would arise when sensor v_j is not predicted at all. The idea is that if v_i could accurately estimate v_j in the first (minimization) formulation, then v_i eliminates all error in estimating v_j in the second (maximization) formulation, such that $w'_{ij} = z_j$. Now the objective function is computed by summing the error reductions over all vertices $v_j \in V$ as follows: **(1)** If v_j is in S, it contributes $z_j = u_j \cdot q_j$ to the error reduction. **(2)** If v_j is not in set S, but there is an edge from at least one vertex in S to v_j, then the error reduction it contributes is the maximum weighted prediction error reduction $w'_{ij} = u_j \cdot (q_j - p_{ij})$ of sensor v_j from any linked sensor $v_i \in S$. **(3)** If v_j is not covered by any linked sensors, then it does not contribute anything to error reduction. Therefore, its contribution is 0.

III. Link Error Modeling

In order to use the aforementioned formulations, it is necessary to model the errors on the links between sensors. In this section, we discuss the process of link-error modeling. The error on the link from v_i to v_j is denoted p_{ij}. We use regression modeling between the time series of both streams to estimate the value of p_{ij}. If the time series at v_i and v_j are given by $x_1 \ldots x_t$ and $y_1 \ldots y_t$, respectively, then we can model $y_1 \ldots y_t$ as a function of $x_1 \ldots x_t$ with the following linear regression model for a window size of length w:

$$y_m = \sum_{n=1}^{w} a_n \cdot x_{m-w+n} \text{ for } w \leq m \leq t \qquad (1)$$

Here, $a_1 \ldots a_w$ are *regression coefficients*, which are used to model the dependence of $y_1 \ldots y_t$ on $x_1 \ldots x_t$. The values of these coefficients are specific to the particular pair of nodes that are being tested. Historical samples from the underlying streams are used in conjunction with mean-square regression analysis to estimate the coefficients [5]. Once these regression coefficients have been estimated, they are used to estimate the error of prediction. Here, the *Recursive Least Square (RLS)* method [1] is used to estimate the coefficients. In this method, if there are n sample pairs consisting of vectors $\vec{x}_i$ of w independent values and dependent values y_i, then a *gain matrix* G_i and coefficient vector $\vec{a_i}$ are incrementally updated using $\vec{x}_i$, y_i, G_{i-1}, and $\vec{a}_{i-1}$, such that $\vec{a} = \vec{a}_n$. This method also allows for an optional *forget exponent*, which exponentially decreases the effect of samples on the coefficients with time. This can be done in order to make the approach dynamic, in terms of continuously updating the regression coefficients with changes in the underlying streams.

Once the coefficients have been estimated, they are used to compute an *estimated value* y'_m. The estimated values are compared to the true values in order to estimate the underlying error. The square error of prediction is given by $r_m^2 = (y_m - y'_m)^2$, and the root mean-square error of prediction over the entire window is $\sqrt{\frac{\sum_{m=w}^{t} r_m{}^2}{t-w}}$. This is the error-value p_{ij} which is assigned to the link between v_i and v_j. We note that the value of p_{ij} only needs to be computed between those vertices v_i and v_j for which a link e_{ij} is present in the information network.

IV. Sensor Selection Algorithms

In this section, we design the algorithms for selection of sensor streams. First, we prove the hardness of the problem of stream selection.

Theorem 1: The stream selection problem is NP-hard.

Proof: We prove the hardness by a reduction from the knapsack problem to the error reduction (maximization) problem. Consider a knapsack with capacity W and n items with weight $w(i)$ and value $v(i)$, for each item i. The objective is to determine if there is a subset of the items whose total weight does not exceed W and whose total value is at least V. This can be modeled as a sensor selection problem in which there are only sensors and no edges (predictability relationships between sensors). Each sensor vertex v_i corresponds to a knapsack item i, with $c_i = w(i)$ and $z_i = v(i)$. The equivalent decision problem is to determine if there is a subset of sensors whose total cost does not exceed W and whose total error reduction is at least V. ■

The relationship of the sensor selection problem to other NP-hard problems is a useful exercise in cases where the objective functions show exact value correspondence, because it provides an idea of the kinds of algorithms which one may use in order to provide effective heuristic solutions. Fortunately, the link-based sensor selection problem has an interesting relationship with the *generalized maximum coverage problem* [6]. This relationship suggests the use of a carefully designed greedy mechanism [6], which defines and uses the concept of *residual density* to regulate the sensor selection process. We will show that such an approach provides an approximation factor of $\frac{e-1}{2\cdot e-1}$ for sensor selection, where e is the base of the natural logarithm. We propose two classes of solutions to the sensor selection problem. The first is integer programming, which can be used for either the minimization or maximization problems. We also present a greedy approximation algorithm for the maximization problem. We now describe both of these classes of solutions in detail.

A. Integer Programming Formulation

An integer programming formulation is a natural way of representing the problem. One advantage of using integer programming formulations is that they can be solved quite efficiently by a number of "off-the-shelf" tools. Furthermore, such tools can also provide an idea of the quality of the solution, since they often use linear programming relaxations in order to provide an optimistic bound on solution quality.

The minimization problem can be solved by transforming it to an integer programming formulation that is analogous to that of the p-medians problem [7]. We use two sets of variables: one for describing the selection of sensors, and another for describing the predictability between different sensors. The set of variables y_i indicate whether or not the stream from sensor v_i is selected for collection. Here, y_i is a binary variable which takes on the value of 1 when the sensor v_i is selected for collection. The binary variable x_{ij} indicates whether or not the stream from sensor v_i is used to predict v_j. The variables x_{ij} are defined only for the cases when the edge e_{ij} is present in the network.

The objective function is to minimize the sum of the weight of the sensors which cannot be predicted at all (because the sensors are not selected and no neighboring sensors predict them), and the weighted prediction error of the edges x_{ij} that are used to make regression-based predictions of sensors that are not selected. The former is the first term in the objective function of the minimization integer program below, and the latter is the second term in the objective function. A number of constraints are defined in order to establish the budget constraints and the relationship between the sensor selection and edge-based regression. The first constraint ensures that the cost of the selected sensors is at most equal to a budget B. The second constraint ensures that a sensor can be predicted by either its direct selection or by prediction from *at most* one neighboring (inlinking) sensor. The third constraint ensures that an edge may be used for prediction only if the source of the edge is included in the set of selected sensors. The final constraint ensures that all variables are integer.

$$\begin{aligned}
\text{Min.} & \sum_{v_i \in V} u_i \cdot q_i \cdot (1 - y_i - \sum_{j: e_{ji} \in E} x_{ji}) + \sum_{e_{ij} \in E} u_j \cdot p_{ij} \cdot x_{ij} \\
\text{subj. to:} & \sum_{v_i \in V} c_i \cdot y_i \le B, \qquad y_i + \sum_{j: e_{ji} \in E} x_{ji} \le 1 \ \forall v_i \in V \\
& x_{ij} \le y_i, \quad y_i, x_{ij} \in \{0,1\} \qquad \forall v_i \in V, e_{ij} \in E
\end{aligned}$$

The maximization problem can be solved by a similar integer programming formulation, in which the objective is to maximize the total reduction in the error as a result of sensor selection. We use the same set of decision variables y_i and x_{ij}. The reduction in error as a result of selecting sensors can be of two types, which is reflected in the two kinds of the terms in the objective function below: **(1)** For the case of selected sensors, the reduction in error is equal to the maximum importance weighted error, which is also equal to $z_j = u_j \cdot q_j$. This is reflected in the first term in the objective function of the formulation below. **(2)** For the case of sensors which are not selected, but are predicted by a neighboring sensor, the reduction in error is equal to $u_j \cdot (q_j - p_{ij})$. This is

```
Algorithm GreedySelect(SensorSet: $V$; EdgeSet: $E$;
        Budget: $B$; ImportanceVector: $\overline{u}$; CostVector: $\overline{c}$;
      MaxSensorErrors: $\overline{q}$; RegressionErrorVector: $\overline{p_{ij}}$);
begin
  $L = \{\}$; { Current Sensor Set }
  repeat
    Determine the sensor with largest value of
      residual density $R(L, v_i)/c_i$, if one exists
      such that $L \cup \{v_i\}$ is within budget $B$;
    if such a $v_i$ exists, then add to $L$;
  until(no vertex $v_i$ can be added to $L$);
  Determine single vertex $v_s$ with
      largest value of $R(\{\}, v_s)$;
  return the better of $L$ and $\{v_s\}$
    in terms of overall regression based error;
end
```

Fig. 1. Greedy Sensor Selection

reflected in the second term of the objective function below. Therefore, the decision problem may be formulated as follows:

$$\text{Maximize} \sum_{v_i \in V} u_i \cdot q_i \cdot y_i + \sum_{e_{ij} \in E} u_j \cdot (q_j - p_{ij}) \cdot x_{ij}$$

$$\text{subj. to:} \sum_{v_i \in V} c_i \cdot y_i \leq B, \quad y_i + \sum_{j: e_{ji} \in E} x_{ji} \leq 1 \quad \forall v_i \in V$$

$$x_{ij} \leq y_i \quad y_i, x_{ij} \in \{0, 1\} \qquad \forall v_i \in V, e_{ij} \in E$$

The constraints in the maximization and minimization formulations are similar, since they use the same variables and constraints. The main difference is in the formulation of the objective function. An advantage of formulating the problem as a maximization problem is that it lends itself to the use of other algorithms which can be implemented efficiently and also have a number of nice approximation properties. This also provides us with bounds on the quality of the solutions. We will discuss one such algorithm below, which leverages the use of the maximization formulation.

B. Greedy Approximation Algorithm

The aforementioned relationship of the maximization version of the sensor selection problem to the maximum coverage problem [8], [6] provides us with some ideas about the approximation algorithms that one may use to solve the problem. We will leverage this relationship to design an algorithm with a constant approximation factor.

In this algorithm, the sensor with the highest *residual density of importance-weighted error reduction* among the not-yet-selected sensors is added to the set of selected sensors in each round if its cost does not violate the budget constraints. Therefore, we need to define the concept of *density of importance-weighted error reduction*. For a given set of sensors L that have already been selected, we can calculate the optimal assignments for regression-based prediction and use it to compute the importance-weighted error of prediction for this set of sensors. We can then calculate the further (or *residual*) importance-weighted reduction in error of prediction by adding sensor v_i to L, which we denote $R(L, v_i)$. In general, the value of $R(L, v_i)$ is non-increasing with increasing size of L. We note that this reduction in error is computed in terms of the regression errors p_{ij}, the sensor importance u_i, and the maximum sensor errors q_i. The residual density of a sensor is the *residual error reduction* $R(L, v_i)$ divided by the cost c_i of sensor v_i. In other words, the residual density of sensor v_i *with respect to* the current sensor set L is given by $R(L, v_i)/c_i$.

It turns out that this approach may sometimes *not* have an approximation bound because of special cases in which a single sensor may provide excellent prediction of all other sensors, yet has a high cost. In order to neutralize the effect of such cases, we simply consider a solution in which we pick the *single* vertex v_i (within the budget constraints) that has the largest value of $R(\{\}, v_i)$. Note that we are not dividing by the cost c_i in this case. The *best set* among the two possibilities (the greedy solution and the singleton set) is reported as the optimal solution. The overall algorithm for sensor selection is illustrated in Figure 1.

A key step in this algorithm is the computation of the residual error reduction $R(L, v_i)$ in each step. We note that each sensor v_j is either selected in the set L, not predicted at all, or is predicted by the vertex $v_k \in L$ with the smallest value of p_{kj} and a corresponding importance weighted error of $u_j \cdot p_{kj}$. If the sensor v_j is currently predicted by v_k, and v_i has lower error p_{ij} of regression-based predictability than the current value p_{kj}, then this reduces the error by $u_j \cdot (p_{kj} - p_{ij})$. On the other hand, if sensor v_j is currently not predicted at all by any sensor, then the error is reduced by $u_j \cdot (q_j - p_{ij})$. Otherwise, there is no error reduction of sensor v_j. Therefore, the residual error reduction is defined as the sum of these values over all the sensors. The residual error reduction density is then determined by dividing this value by the cost c_i. It remains to show that the above algorithm has an approximation factor of $\frac{e-1}{2 \cdot e - 1}$

Theorem 2: The greedy algorithm has a constant approximation factor of $\frac{e-1}{2 \cdot e - 1}$ of the optimal solution, where e is the base of the natural logarithm.

Proof Sketch: The proof of the approximation bound may be derived by modeling the sensor selection problem as a *generalized maximum coverage problem* [6]. In the generalized maximum coverage problem, there is a set of elements $\mathcal{E}$ (where the i^{th} element is denoted by l_i) and a set of bins $\mathcal{B}$ (where the i^{th} bin is denoted by b_i). Each element is assigned a unique positive profit and non-negative weight for assignment to each bin. Additionally, each bin is assigned a weight as overhead for using the bin. The objective is to find a selection of bins and an assignment of elements to those bins with maximum profit and whose total weight is within some budget constraint B. The sensor selection problem can be modeled as an instance of the generalized coverage problem as follows. We associate each vertex v_i with a bin b_i and an element l_i, and set the weight of each b_i to c_i. If the edge e_{ij} exists or if $i = j$, it is possible to use sensor v_i to predict v_j. In this case, we set the weight of assigning l_j to b_i to 0 and set the corresponding profit to $u_j \cdot q_j$ when $i = j$ and $u_j \cdot (q_j - p_{ij})$ when $i \neq j$. Otherwise, since e_{ij} does not exist, then v_i can not predict v_j, and so we set the weight of assigning l_j to b_i

to ∞ to prevent such an assignment.

It can be shown that there is a one-to-one correspondence between solutions to instances of the two problems with identical objective function values. Furthermore, application of the greedy algorithm for the generalized maximum coverage problem [6] to the sensor-based instance of the maximum-coverage problem results in the same sequence of steps as proposed by the greedy sensor selection scheme. Therefore, the approximation factor of the sensor selection scheme is same as that of the greedy algorithm in [6], which is $\frac{e-1}{2\cdot e-1}$. ■

Note that is sufficient to select only the best singleton set in addition to the greedy solution to guarantee this approximation bound, although adding sensors will further improve the solution. If desired, the approximation bound can be further improved to $\frac{e-1}{e}$ by adapting a partial enumeration-based method [6], [9] to the sensor selection problem. Although this improves the worst-case bound, partial enumeration is time-consuming in practice and makes the approach less efficient. Furthermore, our experimental results show that our earlier approach can obtain nearly optimal results most of the time, as is evidenced by the comparison of our results with that of an integer programming solver. Therefore, we retain our aforementioned (simpler) approach with approximation bound of $\frac{e-1}{2\cdot e-1}$ in order to obtain more practical and efficient results. We further note that the $\frac{e-1}{e}$ approximation is an upper bound on the approximability of the optimal solution by any algorithm, unless certain (unlikely) conditions in complexity theory hold. This provides an idea of the quality of our approximation with respect to the best achievable results.

Theorem 3: No algorithm for sensor selection can achieve an approximation factor better than $\frac{e-1}{e}$, unless $NP \subseteq DPTIME(n^{\log\log n})$.

Proof Sketch: This is derived by a reduction from the budgeted maximum coverage problem to the maximized error reduction problem. We associate each set $S_i \in S$ with a vertex v_i with cost c_i and weight 0. We associate each element x_i with a vertex v_i with cost ∞ and weight w_i. For each set S_i and each $x_j \in S_i$, we assign an edge e_{ij} between the corresponding vertices v_i and v_j with weight w_j.

A solution to such an instance of the sensor selection problem will only select vertices associated with sets in S. Furthermore, only edges to vertices associated with elements covered by the sets associated with selected vertices contribute to the error reduction. Therefore, a solution to such an instance determines which sets maximize the weight of the covered elements within the budget constraints.

According to the reduction above, the budget and maximum error reduction in the sensor selection problem equals the budget and maximum weight of the corresponding coverage problem. It was already proven that the best achievable approximation factor for the latter problem is $\frac{e-1}{e}$, unless $NP \subseteq DPTIME(n^{\log\log n})$ [8]. The result follows. ■

V. Experimental Results

The goal of the testing was to show that the use of virtual information network links is useful for performing sensor selection and also results in great efficiency improvements. The data sets needed some further preparation in order to create the virtual links. While these data sets contain multiple time series, they do not contain virtual links as in an information network. Fortunately, we can use the meta-information associated with these data sets in order to create virtual links.

A. Data Sets

We divided each stream into a *training set* and a *test set*. The training set was used to derive the regression coefficients. The coefficients were then used to calculate the prediction error on test data. The data sets used are as follows:

1) Intel Berkeley Lab Data: The Intel Berkeley Lab data set [10] contains temperature, humidity, and light data collected from 54 motes placed in the Intel Berkeley Research lab. Each reading is assigned an *epoch number* that corresponds to the time at which the reading was taken. Only data up to epoch 60,000 is available. We divided this data set in half, using the first 30,000 epochs for training data and the remaining epochs for test data. We used the distance between the motes in order to generate the information network links that were used for predictions. That is, two motes were linked if the distance between them was lower than a given threshold. Links were allowed only between sensors of the same type. The logic in allowing such links was that they provided domain knowledge about the actual relationships between different sensors, which were also useful for the purposes of predictability.

2) EPANET Water Data: We used EPANET 2.0 to simulate water flow in a water pipe network with 126 junctions between various pipes [11]. This topology provides useful information about the relationship between chlorine concentrations at the junctions and was used in order to generate the information network links for predictive purposes. We sampled the chlorine levels at these junctions every 2 minutes over 480 hours, in which the demand at each junction changes every 30 minutes and follows a cyclic pattern that repeats itself every 48 hours.

3) Further Data Preparation: We normalized the values in each stream to the number of standard deviations from the mean and set the maximum error of prediction for each vertex to 10, the difference between 5 standard deviations above or below the mean. This is essentially equal to the maximum error q_i of each sensor. In addition, we needed to assign costs and importance values to the vertices. All importance values u_i for the sensors were set to 1, whereas costs differed between two scenarios **(a)** In the first scenario, all costs were uniform. **(b)** In the second scenario, the costs were designed to be non-uniform, and made to vary from a Zipf distribution. Thus, the sensors were randomly ordered and assigned a corresponding index i. The cost of the i^{th} sensor was then assumed to be $1/(i+1)^{\theta}$, $\forall 1 \leq i \leq n$.

B. Baseline Algorithms

We also used a sampling strategy as a baseline in order to test the effectiveness of the algorithm. The sampling strategy is to select vertices from V one by one at random and add them to S, on condition that they do not violate the budget constraints, until there are no more vertices to add. In order to further improve the effectiveness of sampling, we performed the random selection multiple times and picked the best of these selections. For the purpose of this paper, we used fifty samples.

C. Evaluation Measures

The stream selection mechanism was tested for effectiveness, efficiency, and sensitivity. In each case, we tested both the complete graph of links as well as the information network graph in order to show the efficiency advantages of encoding domain knowledge in the selection process. We used the selected sensor streams (and corresponding prediction assignments) in order to perform actual stream-to-stream regression and thereby predict the non-selected streams. The error of actual prediction is reported over different budgets. To evaluate efficiency, we determined the time required for sensor selection with the use of different techniques. We compare the running time over different budgets. We show that the use of carefully selected information network links can be useful in greatly improving the efficiency of the selection technique. We also tested the sensitivity of the technique to varying costs.

D. Effectiveness and Efficiency Results

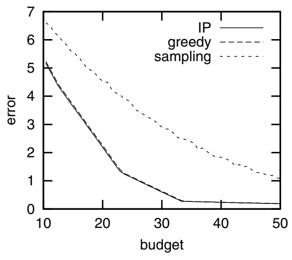
We used the integer programming, greedy, and sampling methods to derive sensor selections with the use of virtual links. These were utilized in order to compute the optimal selection, which was then run over the test data to get the regression-based prediction errors. The IP solver was set to search for an integer optimal solution or terminate after ninety seconds. In the case when the solver terminated, the best solution found so far was used, which is therefore an approximation to the optimal solution. In only a few cases did the solver timeout. In each case, the actual error prediction is computed and reported. We first compare how the error and computation time of all three methods vary with the budget, ranging from 10% to 50% of the total cost of all vertices. We then examine the sensitivity of the method to the skew in the costs across different sensors.

1) Intel Berkeley Lab Data: Unless otherwise mentioned, the default window size was set to 8. We present in which the temperature, humidity, and light streams were combined into one logical information network structure, thus tripling the number of streams. Virtual links were allowed between like streams only. In the simulations, we established virtual links according to the distance between sensors. In the following figures, we compare results for using links between sensors that are within five meters of each other ("5m links"), twenty meters ("20m links"), and links between all sensors ("complete links"). Figure 2 presents the effectiveness and efficiency results for the Intel Berkeley data over different link types. In this case, uniform costs were used. In Figure 2(a), we present the error rates with increasing budget for the case in which links were placed between sensors only when they were at a distance of 5 meters or less. On the X-axis, we present the budget as a percentage of the total sensor cost, and on the Y-axis we present the error of regression modeling. It is evident that the error decreases as the budget is increased and more sensors are available to perform the prediction. Another observation is that the curve for the greedy strategy overlaps that of the IP strategy. Since the IP strategy often terminated with an optimal solution, it follows that the greedy strategy was close to optimal as well in practice. Furthermore, both strategies are significantly superior to the sampling strategy over the entire range of values for the budget.

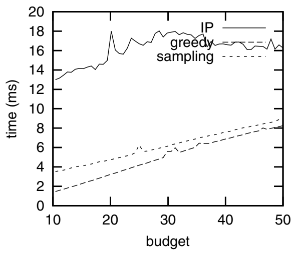
The efficiency results for the Intel Berkeley data are presented in 2(b). The budget is presented in the X-axis, and the running times are illustrated on the Y-axis. It is evident that the greedy algorithm was extremely efficient and provided lower execution times than the other methods. This trend was true in the case of all the different linkage-based representations. The greater efficiency of the greedy method is in spite of the fact that the greedy technique provided almost optimal results in most cases. This suggests that the greedy method provides the best tradeoff between effectiveness and efficiency. In addition, both the greedy and the sampling method had running times which increased gradually with the budget. This was because of the natural iterative way in which the nodes were added to the solution in the case of the greedy and sampling strategies. In the case of the IP solution, the running times were much more erratic, because the running time of the solver was dependent upon the time which it took to arrive at a near-optimal solution. This often depended on the starting point and many other factors that were more important than the size of the problem. In fact, the IP solver often required less time for larger budgets, because the optimal solution was much easier for the solver to find in such cases. For larger budgets, the IP solver sometimes performed more efficiently than the greedy algorithm. However, since the resource constrained scenarios are the more challenging and interesting case, the greedy solution turned out to be the most valuable on an overall basis.

Analogous results for the case when the links were defined by 20m (or less) distances and the complete set of links are illustrated in Figures 2(c)–(f). It is evident that the use of a larger number of links encodes a greater amount of information, and therefore reduces the error. However, such an error reduction comes at a great cost, as the execution times increase tremendously as the number of links are increased. For example, the use of 20m links is almost 20 times slower than using 5m links, whereas the use of the complete graph is two orders of magnitude slower than using 5m links. Since the sensor selection problem often may need to be repeatedly applied on changing cost and importance scenario, the efficiency of the method is paramount to using the method successfully in a variety of scenarios.

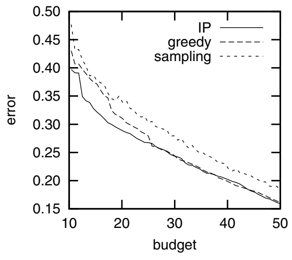
2) EPANET Water Data: Since the pipes are actual connections between junctions, their presence has predictability power in terms of the chlorine concentrations measured by sensors. Therefore, we used the topology of the pipe junctions as our underlying network. In addition, we also tested using the predictability power of all pairs of junctions (which we refer to as the "complete graph"). Figure 3 presents the



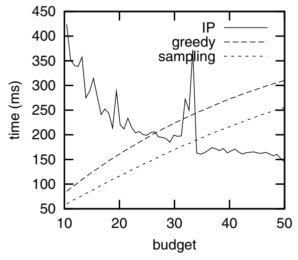
(a) 5m links



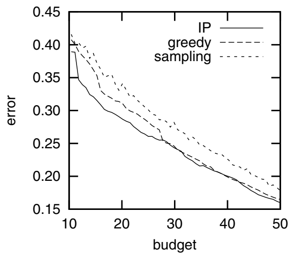
(b) 5m links



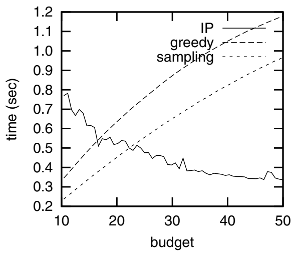
(c) 20m links



(d) 20m links

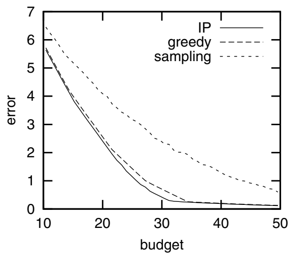


(e) complete links

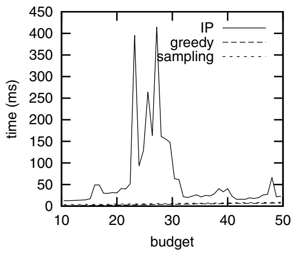


(f) complete links

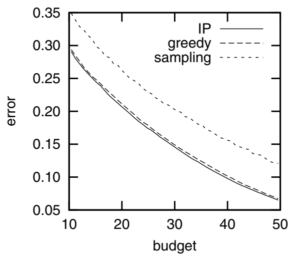
Fig. 2. Errors and execution times for Intel Berkeley data



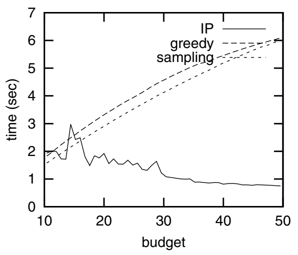
(a) junction topology graph



(b) junction topology graph

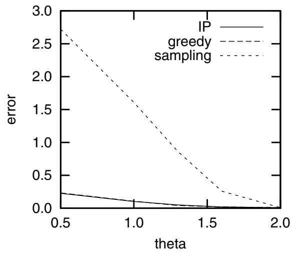


(c) complete graph

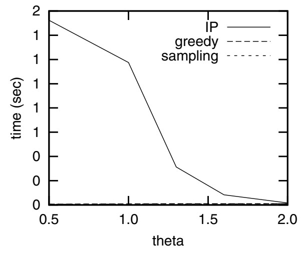


(d) complete graph

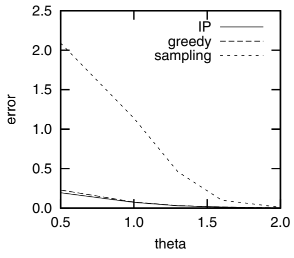
Fig. 3. Effectiveness and efficiency results for EPANET data



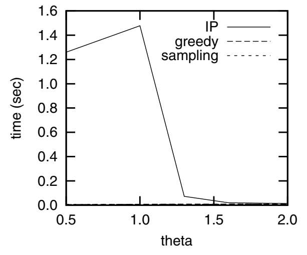
(a) 5m (Intel combination)



(b) 5m (Intel combination)



(c) junction topology (EPANET)



(d) junction topology (EPANET)

Fig. 4. Sensitivity with cost skew (budget fixed at 30%)

results for both cases with uniform costs. The effectiveness and efficiency results for the case where the pipe topology is used are presented in Figures 3(a) and 3(b), respectively. The cases in which the complete graph is used are presented in Figures 3(c) and 3(d), respectively. We make two main observations in this case. The first is that the prediction error of the greedy strategy matched the error of the IP strategy almost exactly in both scenarios. In particular, the baseline sampling strategy performed quite poorly in this case. This is another demonstration of the fact that the greedy strategy not only has an approximation bound, but is also quite effective in practice. The second is that the IP strategy is much more erratic in this case as compared to the Intel data. For the case in which the link topology was used, we found that the running time for the IP solver peaked suddenly at intermediate values of the budget. This is because intermediate values of the budget provided the most realistic combinations to the IP solver. As a result, the running times could also be greater in these cases. Because of the erratic nature of the IP solver in terms of efficiency, it is evident that the greedy approach is much more desirable, especially in cases in which the importance of the sensors varies over time and the solution needs to be repeatedly recomputed.

E. Cost Sensitivity Analysis

We also tested the case in which the costs of the sensors are non-uniform and drawn from a Zipf distribution. The cost of the i^{th} sensor was assumed to be $1/(i+1)^{\theta}$, $\forall 1 \leq i \leq n$. The value of θ varied in the range $[0.5, 2.0]$. The budget was fixed at 30% of the total cost of all sensors. The results are presented in Figure 4. In each case, the Zipf parameter is represented on the X-axis. The errors and execution times for the Intel combination data are presented in Figures 4(a) and

4(b), respectively, whereas that for the EPANET data set are presented in Figures 4(c) and 4(d), respectively. One trend that is evident from the different figures is that the error differences between the sampling and the other two strategies are greater for lower levels of skew. This is because when the skew level is very high, the effectiveness is greatly dominated in the inclusions of one or two sensors with very high costs. Since the sampling strategy picks the best set over many iterations, it was able to pick out those few sensors in those cases. Nevertheless, the greedy strategy turned out to be quite robust in this case, in terms of both execution and running times. The IP strategy again exhibited its unpredictable behavior in terms of the running times, even in terms of the variation over different values of the skew parameter θ. This again suggests that certain instances of the problem can be more difficult for the solver, a result of which they require larger running times.

VI. Related Work

The sensor selection and placement problem has been addressed independently by the stream mining and sensor placement communities. The problem of optimal sensor *placement* has been studied in [12], [9]. The work in [13] studies the problem of sensor selection when the costs and benefits of placing a sensor at a given location have already been modeled externally. The problem of sensor selection has been studied in detail in [14], [15]. The work in [14] is similar to [13], in that it examines the problem of sensor selection when the benefits of picking particular sets of sensors can be externally modeled. None of these techniques discuss models for determining the optimal sensor sets in a data-driven manner. The work in [15] is somewhat data-driven, in that it uses the current data in conjunction with external utility functions for the selection process. Many of the aforementioned techniques are not *data-driven* and require external feedback about sensor benefits. Further, the use of information network linkages to model sensor relationships is unique to the model of our paper.

The problem of data driven stream selection has been studied extensively by the database community [16], [17], [18]. The work in the database community is somewhat distinct from that in the sensor community in that the problem of selection is studied from the perspective of *storage constraints* rather than *stream collection costs*. Nevertheless, the two ways of looking at this problem are quite closely related. The work in [2], [3] examines the cross-predictability of streams with the use of online latent variable detection and its use for predictive purposes. Considerable work [5] has been done in using regression modeling for stream prediction and selection. However, none of this work studies the advantages of studying the stream selection problem in the context of a large information network of linkages, as is studied in this paper.

VII. Conclusion

In this paper, we studied the problem of sensor selection in linked information networks. The idea is to use the domain-specific information which is available in linked information networks in order to greatly prune the size of the logical graph which needs to be explored for sensor selection. We presented a greedy algorithm and an integer programming algorithm with the help of this formulation. We showed how to optimize the sensor selection problem and use this solution in order to make predictions about the different time series. We showed the advantages of our methods over natural baseline schemes such as sampling.

Acknowledgment

Research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] B. Yi, N. Sidiropoulos, T. Johnson, H. Jagadish, C. Faloutsos, A. Biliris, "Online data mining for co-evolving time sequences," *ICDE*, 2000.

[2] Y. Sakurai, S. Papadimitriou, and C. Faloutsos, "Braid: Stream mining through group lag correlations," in *SIGMOD*, 2005.

[3] J. Sun, S. Papadimitriou, and C. Faloutsos, "Online latent variable detection in sensor networks," in *ICDE*, 2005.

[4] J. Leskovec, "Tutorial summary: Large social and information networks: opportunities for ml," in *ICML*, 2009.

[5] C. Aggarwal, *Data Streams: Models and Algorithms*. Springer, 2007.

[6] R. Cohen and L. Katzir, "The generalized maximum coverage problem," *Inf. Process. Lett.*, vol. 108, no. 1, pp. 15–22, 2008.

[7] C. ReVelle and R. Swain, "Central facilities location," *Geographical Analysis*, vol. 2, pp. 30–42, 1970.

[8] S. Khuller, A. Moss, and J. Naor, "The budgeted maximum coverage problem," *Inf. Process. Lett.*, vol. 70, no. 1, pp. 39–45, 1999.

[9] A. Krause, A. P. Singh, C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *Jour. of Machine Learning Res.*, vol. 9, pp. 235–284, 2008.

[10] S. Madden. (2004, Jun.) Intel Lab Data. [Online] Available: http://db.csail.mit.edu/labdata/labdata.html

[11] A. Ostfeld(et.al.), "The battle of the water sensor networks (bwsn): A design challenge for engineers and algorithms," *Jour. of Water Resources Planning and Mgmt*, vol. 134, no. 6, pp. 556–568, 2008.

[12] A. Krause, C. Guestrin, A. Gupta, and J. M. Kleinberg, "Near-optimal sensor placements: maximizing information while minimizing communication cost," in *IPSN*, 2006.

[13] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. M. VanBriesen, and N. S. Glance, "Cost-effective outbreak detection in networks," in *KDD*, 2007.

[14] A. Krause and C. Guestrin, "Near-optimal observation selection using submodular functions," in *AAAI*, 2007.

[15] D. Golovin, M. Faulkner, and A. Krause, "Online distributed sensor selection," in *IPSN*, 2010.

[16] A. Deligiannakis and Y. Kotidis, "Data reduction techniques in sensor networks," *IEEE Data Eng. Bull.*, vol. 28, no. 1, pp. 19–25, 2005.

[17] A. Deligiannakis, Y. Kotidis, and N. Roussopoulos, "Compressing historical information in sensor networks," in *SIGMOD*, 2004.

[18] G. Cormode and M. N. Garofalakis, "Sketching streams through the net: Distributed approximate query tracking," in *VLDB*, 2005.