# Towards Learning-based Sensor Management

Karsten Steinhaeuser, Nitesh V. Chawla and Christian Poellabauer
Department of Computer Science and Engineering
University of Notre Dame, IN 46556
{ksteinha,nchawla,cpoellab}@cse.nd.edu

## ABSTRACT

The management of wireless sensor networks in the presence of multiple constraints is an open problem in systems research. Existing methods perform well when optimized for a single parameter (such as energy, delay, network bandwidth). However, we might want to establish trade-offs on the fly, and optimize the information flow/exchange. This position paper shall serve as a preliminary proof-of-concept that techniques and algorithms from the machine learning and data mining domains can be applied to network data to learn relevant information about the routing behavior of individual nodes and the overall state of the network. We describe two simple examples which demonstrate the application of existing algorithms and analyze the results to illustrate their usefulness.

## 1. INTRODUCTION

The explosion of wireless networks over the last decade has spawned several entirely new fields of research with their own unique challenges and complexities [10]. *Wireless sensor networks* are particularly compelling because of their multitude of applications and deployment scenarios, with each application and scenario potentially imposing its own unique requirements of the network. That is, the same set of sensors can be required to behave, communicate, and adapt differently, based on the application and scenario. In [12], Younis et al. provide an excellent overview of the architectural requirements and design issues in wireless sensor networks, as well as a discussion of QoS challenges in sensor networks.

Modern sensors are often complete embedded systems that can contain complex components such as microphones or cameras, as well as local storage and processing power. These advanced features imply additional demands on the system and deploying such sensors in a distributed environment introduces several new constraints, for instance those of communication delay and power consumption. In addition, these resources offer an opportunity for placement of advanced modules — learning algorithms — that have the capability to process the available data and steer the progress.

The two problems of communication delay and power consumption have been examined separately and several solutions have been proposed, including Quality-of-Service (QoS) and low-energy routing algorithms [5][6][1]. We believe it is important to consider multiple constraints concurrently and consider the "health" of the network. He et al. presented SPEED, a real-time routing protocol for sensor networks [4], which is one attempt at meeting the latency constraints. However, we believe the implementation of learning methods can vastly mitigate the rigidity of the approach, while at the same time, satisfying multiple constraints.

In this paper, as a proof-of-concept, we explore approaches to studying sensor networks with the goal of better understanding routing behavior and the overall state of the network. Specifically, we have applied techniques from machine learning domains to *(i)* classify the routing decisions made by a simple existing algorithm and *(ii)* study the structure and state of the network. We simulated a small network using ns-2 for demonstration. Different routing algorithms typically focus on a single constraint such as energy, number of hops, etc. Our eventual goal is to learn multiple routing functions with different constraints, and develop a hybrid model that can take decisions based on the constraint(s) presented. We want to explore larger, more dynamic wireless sensor networks.

## 2. PROBLEMS AND MOTIVATIONS

We believe the dynamic, wireless sensor networks can largely gain from machine learning algorithms, and likewise the interaction enables fascinating problems for both fundamental and applied research in machine learning. In this section, we describe the types of problems that have motivated our work on this project at the interface between systems and machine learning in more detail.

### 2.1 The Systems Perspective

Current routing algorithms are generally focused on optimizing some particular aspect of the system. For example, one may attempt to minimize the end-to-end delay between two points of communication, while another may attempt to select a routing path along the nodes with the highest energy level [4]. However, each of these approaches exhibits a certain degree of rigidity as they can not easily be combined to perform both functions simultaneously.

Therefore, an active area of interest is flexible routing al-

gorithms and placaments within the context of wireless sensor networks. One of the unique demands of sensor networks is that the traffic patterns generated are often sporadic [10]. [15] propose a data driven algorithm for sensor placements at informative and cost-effective locations. This is due to the unpredictability of the environment in which the sensor operates. For example, there may be little or no data to communicate during periods of low activity recorded by a sensor. However, when there are "interesting" changes in the environment causing the sensor readings to fluctuate rapidly, it may be necessary to communicate a large quantity of data in a short period of time to alert a central control unit.

For this scenario, imagine an algorithm that could perform both delay-sensitive routing to ensure that critical information can be communicated quickly, and energy-aware routing to ensure that sensors can remain in the field without assistance for the longest possible time. Then take this idea one step further and envision an algorithm that can dynamically switch between different modes. For instance, we may want to preserve battery power during periods of low sensor activity and relax our requirements for end-to-end delay. However, if an important event is detected we want to switch to a more aggressive mode that minimizes delay at the expense of energy. Such applications serve as the underlying motivation for our work. The goal is to construct a hybrid "learned" model of multiple routing functions, via simulation, before deployment on a real network.

## 2.2 The Machine Learning Perspective

The applications posited in the previous subsection have in common the underlying problem of learning in dynamic, uncertain, and heterogeneous environments. Moreover, the (independent, identically, distributed data) i.i.d. assumption will not be valid, requiring algorithmic innovations in learning from relational data. Studying relational data (databases, networks, etc.) is emerging as a frontier of research in the machine learning domain. Most commonly, relational data is represented in the graphical format [2][7][3][13]. However, one of the most serious roadblocks to work in this area is the lack of sufficient real data to test these methods on. Large-scale wireless sensor networks generate vast quantities of relational, dynamic data, allowing for robust evaluation of machine learning methods.

## 2.3 Mutual Benefits

Based on the problems described above, it is apparent that there may be benefits for both sides from work at the intersection of systems and knowledge discovery. On one hand, new algorithms may be developed (or existing algorithms extended) from the machine learning domain to aid the study of networks and produce improved routing algorithms. On the other hand, the systems realm may be an untapped resource for the machine learning community, holding a wealth of data for learning and analysis.

In the following sections, we present two novel examples in which we apply machine learning techniques to networking problems to yield interesting results. These examples are not meant to represent a complete body of work, but rather serve as a proof-of-concept that the methods used produce meaningful results in small-scale applications, and therefore should serve as motivation for future work in this area.

# 3. NETWORK MODEL

This section leads into the examples by briefly describing the network and stating some underlying assumptions. We implemented a very basic network topology to underline the role of learning.

## 3.1 Network Topology

We chose to work with a fairly small network as we wanted to obtain tangible results that could easily be visualized and understood by the reader. In addition, a small network should suffice for a proof of concept. A larger network would likely be appropriate for the complete study, but this is beyond the scope of this position paper.

The network consists of twelve nodes: eleven sensor nodes labeled 1-11 and one the sink node labeled 0. Figure 1(a) shows the topology. There are several important assumptions about this arrangement. First, all the sensor nodes send their data exclusively to the sink node. Second, the sink node has a global view of the network (i.e., it has knowledge about the properties of connections and nodes in the network). Third and last, we assume that the network topology remains fixed.

## 3.2 Connection Properties

Each connection has two parameters associated with it: a cost (the "price" to use it), and a time (the delay experienced when crossing it). The link cost is used to model the energy required for a packet to be sent across a connection. However, link cost could also be an aggregate of multiple constraints computed by a cost function. The reason for attaching the energy to the links rather than the nodes lies in the limitations of the available network simulators. All connections in the network are assumed to be of the same type. This means that they are identical in the beginning and that they behave alike as traffic passes through the network. However, the actual parameter values may change.

The type of data we needed were network traces, basic information such as source, destination, and size about each packet sent and received, as well as energy and delay. We simulated the network model using ns-2[1]. Another fringe benefit of using ns-2 was that it supports output formatted for NAM, a network animator that visualizes the network and packet flow.

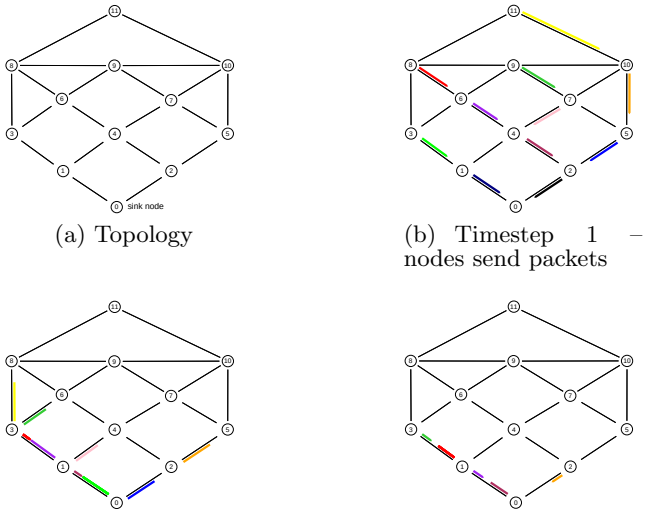
## 3.3 Simulation Setup

The scenario is as follows: every connection starts with a cost of 100 (an arbitrary value equivalent to 'full energy'). Each of the eleven sensor nodes sends packets of 1000 bytes to the sink at 100-millisecond intervals for 12 seconds. For every packet that crosses a connection, the cost of using that connection increases by 1. This information is updated every 2 seconds and the routing tables are recomputed accordingly. The routing protocol used is a least-cost model that finds the path with the lowest total cost.

# 4. ROUTING CLASSIFICATION

Our first example demonstrates that a simple learning algorithm can effectively "learn" a model of the routing function used based on observation of past behavior (i.e. packet traces). The task can be transformed into a *classification problem* which looks at each node's routing decisions and

[1]http://www.isi.edu/nsnam/ns/




(a) Topology

(b) Timestep 1 – nodes send packets

(c) Timestep 2 – packets are routed

(d) Timestep 3 – traffic converges on sink

**Figure 1: Sample NAM output of an ns-2 simulation – packets are represented as arrows and colored for readability**

builds a per-node local model. This represents a critical step towards the successful study of routing using machine learning methods. We used C4.5rules for this set of experiments [9]. Obviously, this approach has a computational limitation. But we wanted to take a first step in empirically understanding and demonstrating the interface.

## 4.1 Data

The input consisted of the same data sets as was used to generate the decision trees: the original data was partitioned by the source node, each outgoing connection represents a class, and cost and delay are the features that determine the classification.

## 4.2 Experiment

We ran C4.5rules on each data set to construct one set of decision rules per node in the network (except for the sink because it does not send any data). The default options were used and all advanced features disabled. See Figure 2 for an example of a rule set.

## 4.3 Results

We center our discussion around node 4 as it is one of the more interesting nodes in the network. Note that once again the size of the output (in this case the rule set) is not fixed, but is heuristically determined by the algorithm as a function of the input data. Figure 2 shows the rule set generated for node 4.

Let us now see how to use these rules for classification. Assuming the same hypothetical packet we used for the decision tree with parameters cost = 100 and delay = 0.02, we would scan down the list of rules until we found one where all conditions are satisfied. Rule 1 would fail, because delay $> 0.017$. Rule 2 would fail, because delay $> 0.019$. Rule 3 would *succeed*, because both delay $> 0.019$ and cost $\leq 100$ are true. Therefore, we have determined that this packet belongs to class 2, and hence should be routed to node 2.

```
Rule 1:
        delay <= 0.017
        cost <= 100
        ->  class 2  [71.6%]
Rule 2:
        delay > 0.017
        delay <= 0.019
        cost <= 120
        ->  class 1  [61.5%]
Rule 3:
        delay > 0.019
        cost <= 100
        ->  class 2  [96.6%]
Rule 4:
        cost > 100
        cost <= 120
        ->  class 1  [98.3%]
Rule 5:
        cost > 120
        cost <= 160
        ->  class 2  [98.6%]
Rule 6:
        cost > 160
        ->  class 1  [96.9%]
```

**Figure 2: C4.5 Decision Rule Set for Node 4**

Applying the C4.5rules algorithm to network packet trace data yields one set of decision rules per node, which model the routing behavior of that node. These rules are too specific to be used literally as a routing table. However, they could be abstracted into a more general form, which may be useful to make routing decisions. For example, the packet delays should be measured relative to other packets rather than an absolute time. Likewise, the energy level of those nodes may be converted into discrete values (such as high, medium, low) rather than continuous values. This could help avoid "overfitting" the test data, which results in too many rules that work great for the test data, but are meaningless in the general case.

Finally, it is worth pointing out that these rules can be changed dynamically either by re-evaluating (or tweaking) them as new data is collected or by storing multiple rule sets for different constraints. This is one area that has great potential but must be further explored.

# 5. SUBDUE AND NETWORK STATE

We now shift focus to our second example, a very different, though related, type of application, which demonstrates a method to "learn" important characteristics about the overall state of the network based on its structure and properties. This learning task is transformed into a subgraph search problem by mapping the network to a labeled graph. The algorithm we use searches graph-structured data for the most common substructures (patterns) within the data. From this information, we can deduce information about the network status. We hypothesize that it could extract relevant summaries from large scale networks, for example detecting links or paths in networks that are decaying faster or healthy links for routing decision. The purpose is to conduct an analysis of the large scale, dynamic networks.

## 5.1 Algorithm

The learning algorithm we used is a subgraph discovery algorithm called SUBDUE [2]. Its goal is to achieve the highest possible compression of a graph by replacing the most common substructure in a graph with a single node, a method that can be applied iteratively if desired. However, for this purpose we are only interested in its ability to find the most "interesting" substructure, or the one that appears most frequently in the graph.

SUBDUE works as follows. Given a set of labeled nodes and a set of edges, it constructs competing subgraphs which are matched to the original graph and grown until either no improvements can be made or the computational complexity becomes too large (in the latter case the iterative method should be used). SUBDUE uses the MDL principle as a heuristic for determining the best subgraph, which is based on the authors' argument that the best one is the one that produces the highest compression. For a complete description of SUBDUE, refer to [2].

## 5.2 Data

We wanted to apply the algorithm to the same network we used for the first experiment. Therefore, we had to describe our network as a labeled graph, which was quite simple based on the visual output produced by NAM. However, we would have to generate multiple input files at different times since we wanted to demonstrate that SUBDUE can be useful to determine the state of the network.

Therefore, we partitioned the simulation into time slices of 2 seconds each. We labeled all nodes "node" because they are essentially identical entities (except for the sink, which we labeled as "sink"). However, for the edge labels we took a snapshot of the network data at every time slice and used the connection cost for labeling. Because using the exact costs as labels would lead to overfitting the data, we translated them to discrete values using the following rules:
cost = 100 → label = full
100 < cost < 200 → label = high
200 ≤ cost < 300 → label = med
300 ≤ cost < 400 → label = low
cost = 400 → label = out
This nomenclature is based on the interpretation that the cost inversely relates to the energy level, i.e., a low cost implies a high level of energy available. We created six such data sets as input for the algorithm. See Figure 3 for a sample output.

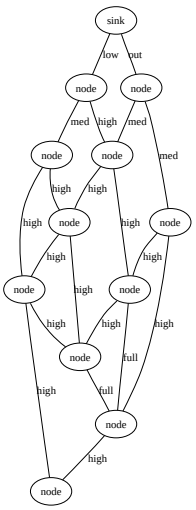



**Figure 3: Sample GraphViz output of the network as a graph in SUBDUE after 10 seconds of simulation**

## 5.3 Experiment

We ran SUBDUE on each data set to determine the most common substructure found in the graph (SUBDUE actually reports the 3 highest ranked substructures by default, but we only consider the top choice). Figure 4 shows the three different substructures identified as the most interesting patterns in the network at different times.

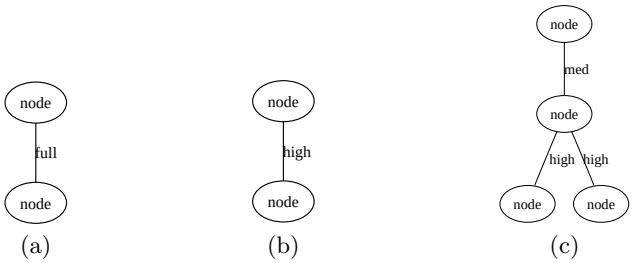



**Figure 4: Most common substructures identified by SUBDUE at different times during the simulation**

## 5.4 Results

Let us now take a closer look at the substructures discovered by SUBDUE and provide an intuitive explanation why these particular patterns were chosen. In addition, we clarify their relevance to better understanding the state of the network.

Figure 4 (a) shows the highest-ranking substructure at time t = 0. This is more of a reality check as we would expect the most common substructure in the original graph to be two nodes linked by a full connection (recall that cost always starts at 100, therefore energy starts at full for all connections).

Figure 4 (b) shows the highest-ranking substructure for the time 0<t<9. Again, this is relatively intuitive as a small amount of traffic has passed through most connections (except those close to the sink). Therefore, it follows that the energy level most of connections away from the sink is currently at high.

Figure 4 (c) is a more interesting substructure because it involves more than just two nodes (the input from which this substructure was derived is actually the graph in Figure 3). Notice that one of the connections is only of medium energy, although it is connected to two other nodes by high-energy connections.

Now that we have examined the three substructures, we must establish their relevance to understanding the network. To this end, we argue that these substructures (potentially combined with other metrics) can provide 'the big picture' about the overall state of the network. For example, if the most common substructure found is one of those in Figures 4 (a) or (b), we have a fairly good idea that the majority of connections in the network have a high energy level available. However, if the highest-ranking pattern is similar to that in Figure 4 (c), then we can deduce that some of the nodes in the network are beginning to reach lower energy levels. We expect that this trend would continue for extended simulation data.

Given a snapshot of network data in graph format, SUBDUE can provide us with the most common substructure(s) in the graph. These subgraphs are representative of patterns in the graph, which have relevance with regards to the state of the network. We have shown that for a small network, we obtain different substructures at varying times during the simulation. By comparing the subgraphs to the network from which they were derived (or by simple reasoning), we

can verify that they are intuitively correct.

Further, we show that by running the simulation for a longer period of time and passing it to SUBDUE, we obtain substructures of decreasing energy levels. This should (correctly) imply concerns regarding the reliability of the network. And while this may all seem intuitive because we can visually verify the graphical output, imagine a sensor network of hundreds or thousands of nodes, which is not at all unreasonable for modern applications. We argue that for a human observer to get a sense of the state of the network in this scenario would be nearly impossible. However, SUBDUE could provide us with the most common substructures in the graph, which are an important indication regarding the overall health of the network.

## 6. DISCUSSION AND FUTURE WORK

Using management of a simple wireless sensor network as a framework, we have shown that applying machine learning methods to systems problems can, and does in fact, yield interesting results. First, we have shown that a rule classifier is capable of learning the behavior of a routing algorithm. Second, we have shown that subgraph discovery algorithms can be used to infer knowledge about the general "health" of the network.

From a systems perspective, the results presented here only scratch the surface of the open problems. However, we believe that they are promising in the sense that they indicate directions for future work. Routing in wireless sensor networks is an active area of research with open questions that machine learning methods may be able to answer. To advance research on the routing problem, the next step must be to obtain real-world packet trace data.[2] From there, one could perform a more thorough analysis and construct a generalized model that is capable of providing the aspired dynamic, multi-constraint routing functionality.

From a machine learning perspective, the dynamic, relational, and heterogeneous data present its own unique challenges, opening up the door for significant algorithmic progress. The compelling aspect is the real data with real problems available from systems to actually evaluate the learning algorithms. Learning in wireless sensor networks can be transformed into a graph problem, and therefore machine learning methods may be applicable to aid in the discovery of structure and substructures for clustering, routing, and location. Another research area is the detection of network partitions, which has applications for routing and data management. SUBDUE, for instance, could be run on a graph labeled with signal strengths to detect areas of the network that are about to lose connectivity.

We believe that a plethora of other applications will emerge from continued collaboration and through reciprocal education about the types of open problems and available solutions.

## 7. REFERENCES


[1] K. Akkaya, M. Younis, *A Survey on Routing Protocols for Wireless Sensor Networks*, Ad Hoc Networks, Vol. 3, Issue 3, pp. 325-249, November 2003.

[2] D. J. Cook, L. B. Holder, *Substructure Discovery Using Minimum Description Length and Background Knowledge*, Journal of Artificial Intelligence Research, Volume 1, pp 231-255, 1994.

[3] N. Friedman, L. Getoor, D. Koller, A. Pfeffer, *Learning Probabilistic Relational Models*, Proceedings of the 16th International Joint Conference on Artificial Intelligence, 1999.

[4] T. He, J. A. Stankovic, C. Lu, T. Abdelzaher, *SPEED: A State-Less Protocol for Real-Time Communication in Sensor Networks*, Proceedings of the International Conference on Distributed Systems, 2003.

[5] W. Heinzelman, A. Chandrakasan, H. Balakrishnan, *Energy-Efficient Communication Protocol for Wireless Microsensor Networks*, Proceeding of the Hawaii Conference on System Sciences, 2000.

[6] W. Heinzelman, J. Kulik, H. Balakrishnan, *Adaptive Protocols for Information Dissemination in Wireless Sensor Networks*, Proceedings of the 5th annual ACM/IEEE International Conference on Mobile Computing and Networking, 1999.

[7] S. A. Macskassy, F. J. Provost, *A Simple Relational Classifier*, Proceedings of the 2nd Workshop on Multi-Relational Data Mining at KDD, 2003.

[8] J. Neville, D. Jensen, *Iterative Classification in Relational Data*, Proceedings of the AAAI Workshop on Learning Statistical Models from Relational Data, 2000.

[9] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Textbook. Morgan Kaufmann Publishers, San Mateo, California, 1993.

[10] S. Sahni, X. Xu, *Algorithms for Wireless Sensor Network*, International Journal of Distributed Sensor Networks, Vol. 1, 2005.

[11] R. Van Renesse, K. P. Birman, W. Vogels, *Astrolabe: A Robust and Scalable Technology for Distributed System Monitoring, Management, and Data Mining*, ACM Transactions on Computer Systems, Vol. 21, No. 2, pp. 164-206, 2003.

[12] M. Younis, K. Akkaya, M. Eltoweissy, A. Wadaa, *On Handling QoS Traffic in Wireless Sensor Networks*, Proceedings of the HAWAII International Conference on System Sciences, 2004.

[13] M. I. Jordan, *Graphical model*, Statistical Science (Special Issue on Bayesian Statistics),Vol. 19, pp. 140–155, 2004.

[14] M. Younis, M. Youssef, K. Arisha, *Energy-Aware Routing in Cluster-Based Sensor Networks*, Proceedings of the 10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2002.

[15] A. Krause, C. Guestrin, A. Gupta, J. Kleinberg, *Near optimal Sensor Placements: Maximizing Information while Minimizing Communication Cost*, 5th International Conference on Information Processing in Sensor Networks, 2006.


[2] An initiative to this end has been launched at Dartmouth, see http://crawdad.cs.dartmouth.edu/ for more information