Attribute Based Routing in Sensor Networks*

A. Agarwal and T.D.C. Little
Department of Electrical and Computer Engineering
Boston University, Boston, Massachusetts
{*ke,tdcl*}*@bu.edu*

June 1, 2006

MCL Technical Report No. 06-01-2006

Abstract– Data-centric routing is enabled by forwarding rules that interpret the contents or labeling of messages in a Sensor Network. In this paper we describe a framework for achieving data-centric routing using attributes defined using an XML representation. We demonstrate the benefits of the framework and show a number of addressing schemes that can be cast onto the framework, illustrating the utility of the scheme. Finally, we describe the complexity of the scheme in relation to existing addressing mechanisms with respect to setup, maintenance, and scale.

This work is supported by the NSF under grant No. CNS-0435353. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

1 Introduction

Sensor networks have come to be characterized by large numbers of low-cost, minimally-powered wireless devices that are applied in sensing phenomena in the physical world and communicating them to coordinating applications or end users. As the cost of devices decreases, deployment density is expected to increase yielding spatial redundancies. However, the expectation will also be for frequent device failures due to energy consumption (battery depletion) exposure (environmental damage), and quality of devices (imprecise manufacturing). In the context of many redundant devices, the functional need for addressing individual devices is diminished in favor of location or sensed-parameter specific information. Class-representative values are appropriate for yielding measurement of phenomena and techniques are appropriate for exploiting this system characteristic. We focus on routing of these values using an attribute-based technique.

Sensor networks are often used in environmental monitoring applications where they are deployed in inaccessible locations with little or no management. Saving energy in the network is crucial for longevity of the applications and often the focus of energy conservation is in minimizing expensive communication costs. The use of spatial redundancy allows operating parts of the network in a sleep mode until device failures require these to awake to sustain operation. While nodes in the network use multi-hop communication to connect to other nodes, often the communication is highly localized in the network while the data are logged at a base station. Data are viewed as tightly coupled to time of capture and node location rather than network address. The implication here is that, especially in the presence of spatial oversampling representative values are more meaningful than device-specific ones. This leads us to mechanisms for extracting and routing data labeled with class information rather than conventional address (or device ID) information. This idea extends into applications requiring in-network processing across regions or within other subsets. For example, routing data among a group of nodes involved in the tracking of an animal in a sensor grid exhibits behaviors suited to labeled data sharing and communication.

In its more conventional form, routing involves passing of data from a source to its destination without interpreting the content of the data. This is address-centric routing in which the routing framework is optimized for source and destination address pairs. To achieve this routing there must be some form of route discovery – proactive or reactive – that facilitates the propagation of a message to its destination. The "best" route is based upon a cost function derived from different parameters such as signal strength, hop count or energy requirements. Once the route, or path, has been established, the path information is embedded in the message packet or cached at each node for corresponding source-destination pairs. Here, the path of each data packet is strongly associated with the address of origin and destination nodes.

In contrast, data-centric, labeled, or attribute-based routing implies evaluation of the contents of transmitted data at each hop in the network, that is, not just the source and destination headers. Because each node in a sensor network provides a routing function (unlike hierarchical wired networks), individual nodes can apply routing rules on the contents of a message to expedite or cancel its propagation. A data-centric routing technique is also consistent with the localization of message exchange in applications that use cooperative in-network processing. In a data-centric routing approach, the data and their context can be more important than the node address. Whereas, in conventional networks, the goal is often to minimize the cost in terms of path length, delay, energy consumed. The distinguishing feature from conventional networks here is the trade-off between conflicting metrics.

Routing protocols rely on a route discovery process to establish a relationship between the source and destination nodes to form a multi-hop path. However, the costs associated with route discovery, especially in the network-wide case, can be prohibitive and unmanageable for many scenarios. One approach is to localize communication using decentralized routing techniques. A distributed routing protocol relies on some form of relationship either static or dynamic which is present throughout the network and is in general sufficient to form the source to destination path without precise global information of the entire network. An example is a rooted tree structure, Fig. 1, where each node is aware of the location of its neighboring node relative to a root node, often the base-station. This information is sufficient to recognize parent and child nodes and communicate in a bi-directional flow up and down the tree. The formation of a rooted tree does not require extensive path formation strategies and communication costs are low. The communication is however, limited to root-node and node-root paths. Arbitrary node-node communication is not implemented without increasing complexity. It is efficient for data-logging applications, aggregating strategies and applications such as TinyDB. The common factor for implementing any distributed routing protocol is the relationship between nodes described in the form of a network wide parameter and a decision logic based on this parameter.

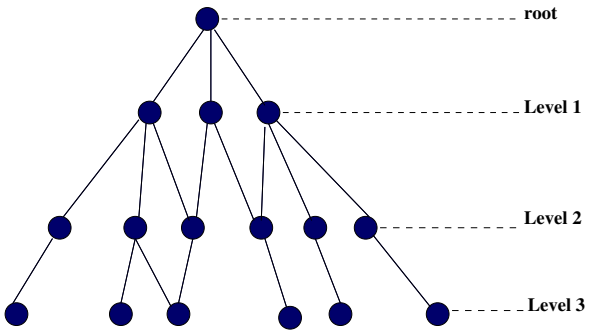

Figure 1: An Example of a Rooted Tree.

Attribute-based routing is a *data-centric* approach to routing in which the data exchanged between nodes forms the basis of the routing function. We adopt an approach which enables each node to parse each data message, draw a meaningful conclusion from the data and make a routing decision based on the collective set of attributes. There are notable motivations behind adopting such a design approach. Event detection [1], data aggregation [2], in-network query processing [3], data-centric storage [4], and consensus-based algorithms rely on the exchange of local sensed data for decision making. In an attribute-based routing scheme there is a potential to address a selection, set, or class of nodes with non-unique destinations. Attributing data, in addition to address logic, allows the implementation of distributed and adaptable algorithms thereby supporting more efficient routing.

In this paper we describe a framework for achieving attribute-based routing to meet the aforementioned goals. The scheme relies on attributes embedded in routed messages and requires sets of routing rules to be established and maintained at each node in the system. The most interesting features of our proposed scheme are (1) the ability to route data independent of device addresses, (2) class- or clique-based addressing using attributes, (3) the ability to cast arbitrary routing schemes that use data attributes onto a sensor network, such as GPSR or Directed Diffusion, (4) support for dynamic reconfiguration of the sensor network, and (5) support for the goal of permitting ef-

ficient routing through orthogonal, coexisting routing functions. The remainder of this paper is organized as follows – Section II describes related work and schemes for data-centric routing in sensor networks. Section III presents the proposed framework for attribute-based routing. Section IV demonstrates the applications built on the proposed framework. Section V provides an analysis or our scheme. Section VI concludes the paper.

2 Related Work

Related work in the area of local coordination in sensor networks has shown the value of localized communication. Saligrama et al. show that by exchanging sensor information in a local neighborhood, a node can detect an event normally inferred by viewing global data [1]. Consider the example of detecting a boundary in a network. The centralized approach to this problem is easier to implement but more communication intensive. The authors propose this can be computed by data exchange between local sensors achieving significant savings in communication costs. Whitehouse et al. demonstrate the capability to track an object or event in the network by exchanging status information in a neighborhood [5]. In the Duck Island project [6], the importance of aggregating data on a local basis has been highlighted to save communication costs over long distance links. Krishnamachari et al. demonstrate the advantages of data-centric routing scheme over an address-centric routing approach [7]. Coffin et al. [8] propose a simple scheme based on reverse path forwarding to form routing trees rooted at a base station to log data in a network. Govindan shows the gains over flooding a query by using a data-centric storage scheme [4]. While these papers show in great detail the advantages of localized communication, few have discussed the implementation details like assigning roles for individual sensor nodes and managing local coordination. Additionally, the data-centric approach precludes the requirement of a stringent naming scheme which is advantageous in arbitrary deployments with spatial redundancy, or heterogeneous nodes in the network. The naming scheme in a data-centric routing approach is less relevant to the routing function than in an address-centric routing approach. In the TinyDB design [9], the authors show a framework for executing queries and treating the sensor network as a database. However, it is limited in the extent in which general computation can be achieved towards in-network processing. TinyDB forms a routing tree rooted at the base station. Cougar, [3], is a query optimizer design that generates an efficient query plan for in-network processing that reduces resource usage and extends the lifetime of the network. In [10], the authors have proposed a sensor network modeled on a geographic hash table, creating specific data storage points in the network depending on the context of data. This greatly reduces the task of retrieving data from the network as there is a deterministic location of data in the network.

In the proposed work, *Abstract Regions* [11], relies on locality for supporting a set of communication primitives for data sharing. While the focus here is on a set of primitives which bind the programming, control and communication, our emphasis is on creating a routing framework to enable local coordination and communication within the network. In GPSR, [12], the authors have presented a distributed approach towards routing data using the destination location and information about the local topology. The framework proposed in this work is designed to support such distributed algorithms which adopt a data-centric approach towards routing. The aim is to provide a substrate that can support multiple routing techniques.

3 Proposed Attribute Based Routing Framework

In this section we describe our attribute-based routing framework and associated elements. We introduce the concept of using class identifiers for nodes in the network. In addition to node addressing, nodes are identified on the basis of properties in the network that identifies their class. In this context, we propose the use of class identifiers to create routing rules in the network which are locally formed to perform the global routing goal in the network. The class identifiers can be derived from locally information, sensed values, or state of the system. Examples of attributes include GPS location coordinates, network topology, sensing modality, residual energy, data types, and composite functions. The advantage of using class identifiers for nodes is that a set of rules can be devised which operating in a distributed fashion at distinct points in the network achieve the global routing goal. The address of a node loses its resolution at intermediate nodes in the network and the routing decision is based on attributes which identify the node class. Thus, the class identifier relates each packet to a unique node or a set of nodes in the network. We term this identifier as an *attribute* which describes a relation between the data and the node. An attribute or a set of attributes is used to compute the path at the intermediate nodes from the source to destination.

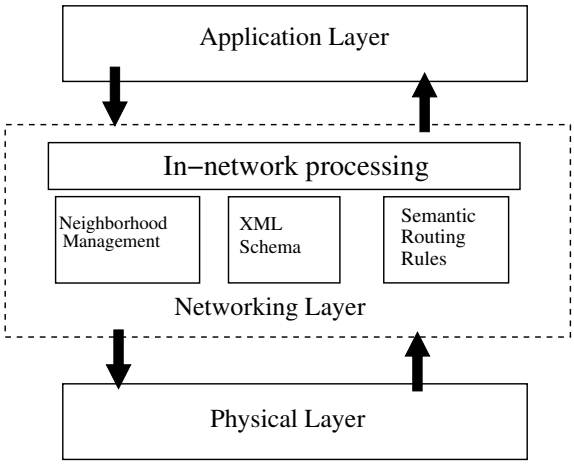

Figure 2: Framework for Implementing Attribute-Based Routing.

Individual node (device) addresses are still required to distinguish between nodes that are within close proximity or are otherwise indistinguishable due to identical attributes. However, this does not imply that device addresses are required for routing.

The proposed framework is designed to be modular as shown in Fig. 2. The networking component requires an implementation of neighborhood management, XML schema interpretation, in-network processing, and semantic routing rules. The Neighborhood Management Module seeks to generate a routing table based on the attribute-value pairs of its neighborhood. This is essential for the routing algorithms to apply semantic rules based on the attributes and make a routing decision. In this work, we specify XML to encode messages, as discussed later. For interpreting XML encoded messages, we require a repository for the XML Schema to interpret the form of incoming messages and to package outgoing ones. The in-network processing module is used by applications which perform distributed computation on attribute-value pairs and generate consensus on events or evaluate boundaries in the system. The semantic routing rules describe a set of

local rules applied at each node in the system to achieve the global routing goal. The details of these components follow.

3.1 Routing Tables and Message Structure

Because we associate attributes with nodes in the system rather than node addresses, each node is required to manage tables of attributes defining neighbors. This table can contain several attribute-value pairs or a single attribute depending upon the requirements of the sensing application. An example routing table of this type is shown in Table 1.

Table 1: Illustration of a Routing Table with Attribute-Value Pairs

Node-ID	*Attribute*	*Value*
0	attribute1	value1
1	attribute3	value1
0	attribute2	value4
2	attribute3	value2
3	attribute1	value2
2	attribute1	value1
...	...	...

We use XML to embed attribute-value pairs into the message packets. The use of XML in sensor networks has been explored by others, e.g., Tilak et al., as a common standard for information exchange among heterogeneous nodes [13]. Lightweight XML parsers are also feasible in this context [14]. Semantic rules are applied on the set of attributes derived by parsing the XML message attributes to derive routing information. By changing the semantic rules we can change the routing paradigm or even support multiple concurrent paradigms.The following shows an example XML message type.

```
<msg id> numerical value </msg id>
<attribute1>
  <attribute2> value2 </attribute2>
  <attribute3> value3 </attribute3>
</attribute1>
<data> message </data>
```

3.2 Protocol Operation

The application of the routing framework requires protocols for bootstrap and for subsequent message propagation. For bootstrap, neighboring nodes advertise their presence by periodic beacon-

ing. Nodes subsequently and periodically share attribute-value information to generate sufficient state to support routing. The attribute-value pairs exchanged are largely dictated by the sensing application (we show examples later).

Messages are decoded using an XML schema defined in the module and a routing table as shown in Table 1 is generated. The relevant data are passed to the in-network processing module and then to application-specific semantic rules that define message propagation. Data are either made available to the application layer for further processing or are forwarded to other nodes in the network. If the data are to be forwarded to other nodes in the network, the attributes of the message and the neighborhood information obtained from the routing table will be processed and a next hop node will be selected. The data will then be encoded using a suitable XML schema and transmitted to the appropriate neighboring nodes.

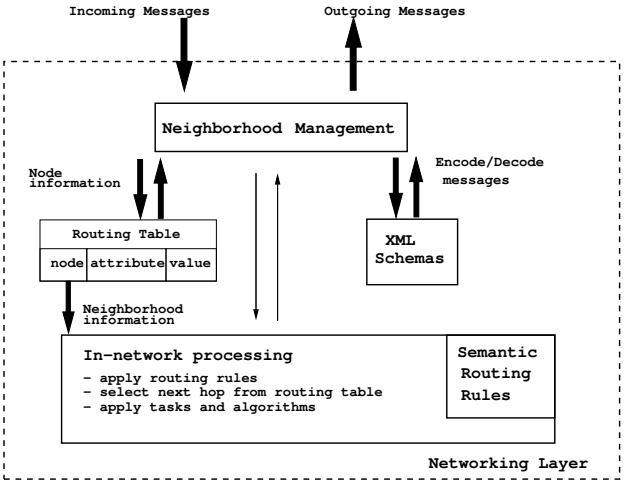

Figure 3: Interactions Among Different Modules of the Framework

4 Applications

In this section we describe several applications to demonstrate how the attribute-based routing framework can be used as a common substrate to enable multiple routing or application functions. These representative cases include a rooted tree (e.g., as formed in Directed Diffusion), greedy forwarding, directional (bearing) forwarding, and animal tracking.

4.1 Rooted Tree

Consider a arbitrary deployment of nodes on which a rooted tree is desired (e.g., Directed Diffusion or TinyDB [9]). The tree is usually rooted at a data sink node or base station. Fig. 4 shows such a case. The root serves to initiate the tree formation process and depending on the multi-hop distance from the root, each node in the network recognizes its hop distance from the root. This parameter becomes the depth of the node and each node is aware of the depth of its neighboring nodes. The communication in the network is anchored at the root or the base node. The message exchange in the system is of *query-response* form, in which the base queries a set of nodes, or *event-log* form,

where an event detected by a node is logged at the *base-station*. In this scenario, all messages can be classified as either downstream, from base to nodes at a certain depth or upstream, from nodes to the base. The routing table (e.g., Table 2) must store the hop count of the nodes in the neighborhood. The semantic rule created for routing recognizes the direction of flow for each message and selects one of the parent or child nodes.

Consider an application where messages are to be relayed to all nodes (or a subset of all nodes) at a depth of 5 hop-counts from the base. The message can be encoded as follows.

```
<msg id> 01 </msg id>
<from> base </from>
<to>
  <hop count> 5 </hop count>
</to>
<data> query </data>
```

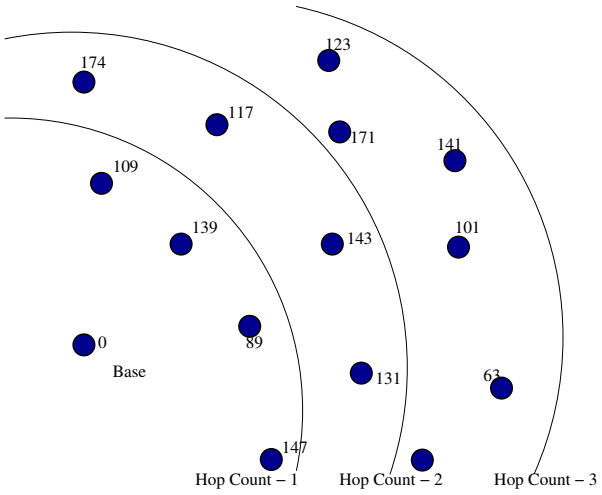

Figure 4: Sensor Network Requiring a Rooted Tree

At each level in the tree, a node selects nodes which are child nodes to propagate the message from the base. Nodes at level 5 recognize the hop-count limit and stop further propagation of messages. More complex tasks and relations can be suitably described as above using detailed XML messages and corresponding routing rules in the network. Correspondingly, the response is relayed to one of many parents for each node generating a response. The parent selection may be achieved by some local decision making algorithm based on metrics such as lifetime or signal strength. The following is an XML-encoded response from a node to the base.

```
<msg id> 01 </msg id>
<from> node-id </from>
<to> base </to>
<data> response </data>
```

Table 2: Example Routing Table with Depth 2

Node-ID	*Hop Count*	*Signal Strength*	*Residual Energy*
131	2	-75dBm	2.75V
117	2	-69dBm	3.10V
139	1	-70dBm	2.90V
89	1	-72dBm	2.81V
101	3	-80dBm	2.71V
171	3	-90dBm	2.45V
. . .	. . .	. . .	. . .

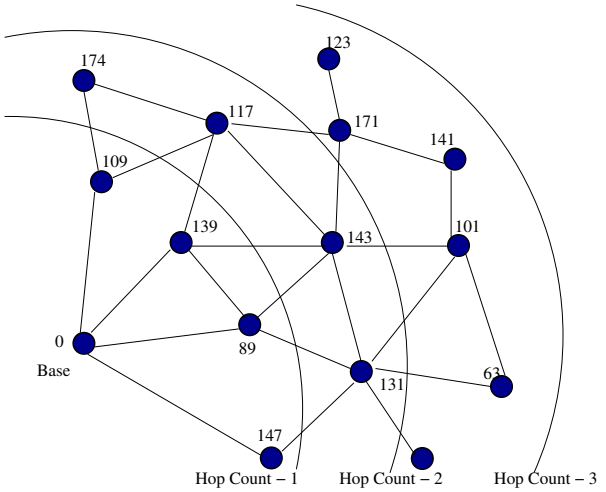

Figure 5: Rooted Tree with Neighborhood Links Across Hop Counts

4.2 Greedy Forwarding-Based Routing

GPSR or the Greedy Perimeter Stateless routing is a routing protocol proposed in [12] that uses the position of routers and a packet's destination to make greedy forwarding decisions using only local topology information. Each node in the network is assumed to be aware of the location of its neighbors. The location of the destination node is embedded in the packet and the routing algorithm selects a next hop node from its routing table such that the distance to the destination is minimized. In a contrast to other ad hoc routing protocols, this algorithm does not require path formation strategies or route caching. It is a distributed algorithm where the computation at each node is able to route data to the destination. The algorithm potentially modifies the packet's attributes to route around a void region where greedy forwarding is not possible. The routing protocol dictates that whenever a void is encountered, which is a local maxima, the contents of the packet are modified such that the greedy forwarding process can continue without the danger of returning to the prior local maximum. An example is illustrated in Fig. 7.

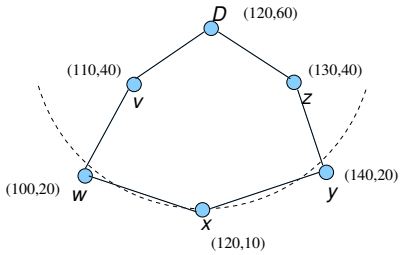

Figure 6: Greedy Forwarding in Location-Based Routing

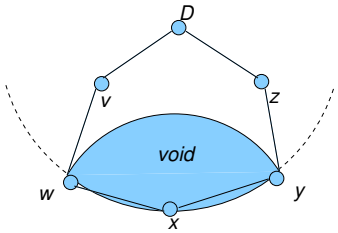

Figure 7: Node *x*'s Void with Respect to Destination *D*

In this case the routing table is formed by a bootstrap process in which nodes share their IDs and attribute-value pairs defining location. When a node receives a message with destination information, it is able to compute the next hop by minimizing the distance between the node and the destination described in the message. This minimization function can be defined as part of the semantic routing rules module. When a void is encountered, the message is modified to include additional attributes of L_f, L_p and e_0, which describe the action taken to circumnavigate the void. These attributes are specific to the routing protocol (GPSR) [12] and are encoded with the help of an XML schema such that the node at the next hop is able to parse this information and recognize the presence of a void in the region. The next hop can then compute the path based on the information propagated. By using a different schema, the nodes are able to relay information about the action taken to navigate the void and this information can propagate through additional hops. The

change in computation of the path is expressed by applying a different semantic rule corresponding to the XML schema used. This process continues at each hop in the network until the destination is reached .

Table 3: Illustration of the Routing Table Generated at Node *x*

Node-ID	*Location*
w	(100,20)
y	(140,20)
...	...

A packet in the network is encoded as shown below. The location of the destination *D* is embedded in the packet along with L_f, L_p and e_0 fields, which are populated when a void is encountered in the path of the message.

```
<msg id> 01 </msg id>
<destination>
  <node id> D </node id>
  <Lf> value(Lf) </Lf>
  <Lp> value(Lp) </Lp>
  <e0> value(e0) </e0>
</destination>
<data> fire-alert </data>
```

4.3 Directional Propagation in a VANET

VANETs or Vehicular Area Networks, essentially a network of vehicles on a highway to facilitate information propagation [15]. In VANETs there is often a high density of nodes in the environment. Using device addressing under these conditions difficult as the nodes are highly mobile and dense with frequent and random arrivals and departures from the network. With a goal to propagate information about dangerous road conditions such as ice or accidents, a VANET must facilitate routing under these conditions. Geographical (roadway) information and vehicular behavior are tightly coupled to position and thus location-based attribution can be exploited in these scenarios. In addition, a VANET is extremely dynamic in nature, suggesting that proactive routing schemes will be costly to maintain. Thus we explore the use of attributed location data to facilitate routing here.

Consider an application to propagate road information such as road surface conditions (e.g., water, ice, etc.) sourced from individual vehicle sensors. A vehicle sensing a dangerous condition generates a warning message that is valuable to other approaching vehicles. Therefore, the time and location of the dangerous condition are important attributes as is the radius of oncoming

vehicular traffic. This can be captured as attributes of source location of event (from GPS), road identifier (Route-1), heading (North), and destination information in the form of a time-to-live (TTL) parameter, and a function of time and distance relative to the source. The routing function at each node is then required to receive the message and parse the embedded attributes with respect to its own local information and select a relevant next-hop for the message. The cooperation of all nodes in this way achieved successful dissemination of the dangerous road condition only towards vehicles that will be arriving at the incident.

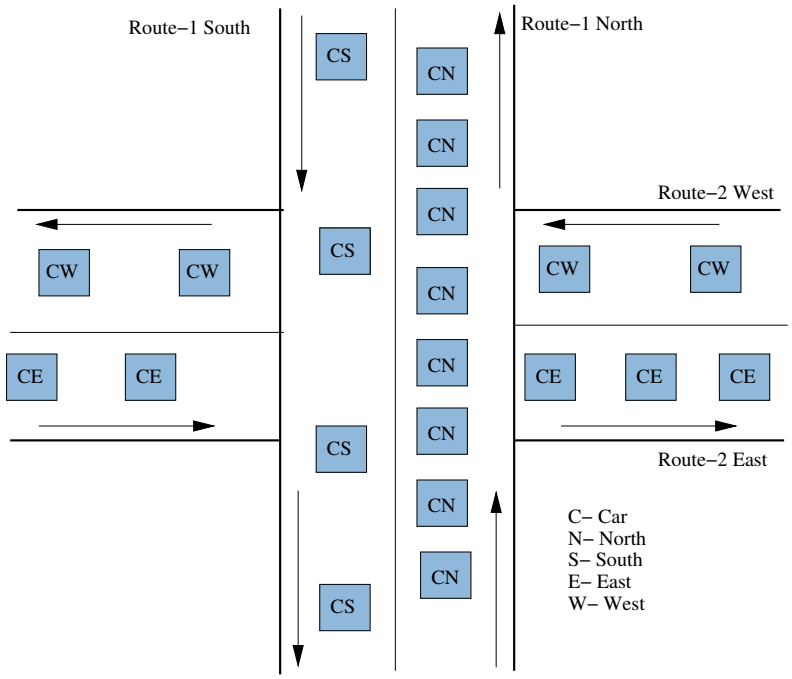

Figure 8: The VANET Scenario

The attributes embedded in the message appear as described in the XML message structure for VANET as shown below.

```
<id>109< /id>
<event>
  ice
  <location>
    <Latitude>4807.038,N</Latitude>
    <Longitude>01131.324,E</Longitude>
  </location>
  <road-id>Route-1</road-id>
  <heading>North</heading>
  <time> 19:32 </time>
  <TTL>
    <distance>5 miles</distance>
    <time>10 mins</time>
```

Table 4: Example Routing Table in the VANET Scenario

Node-ID	*Location*	*Heading*	*Road-ID*
103	$42.3604N71.0573W$	North	RT-1
119	$42.3604N71.0573W$	South	RT-1
119	$4737.216'N12219.75'W$	North	RT-1
107	$3746.35'N12224.233'W$	East	RT-2
159	$42.3604N71.0573W$	West	RT-2
177	$3746.35'N12224.233'W$	East	RT-2
200	$4737.216'N12219.75'W$	West	RT-2
...	...	...	...

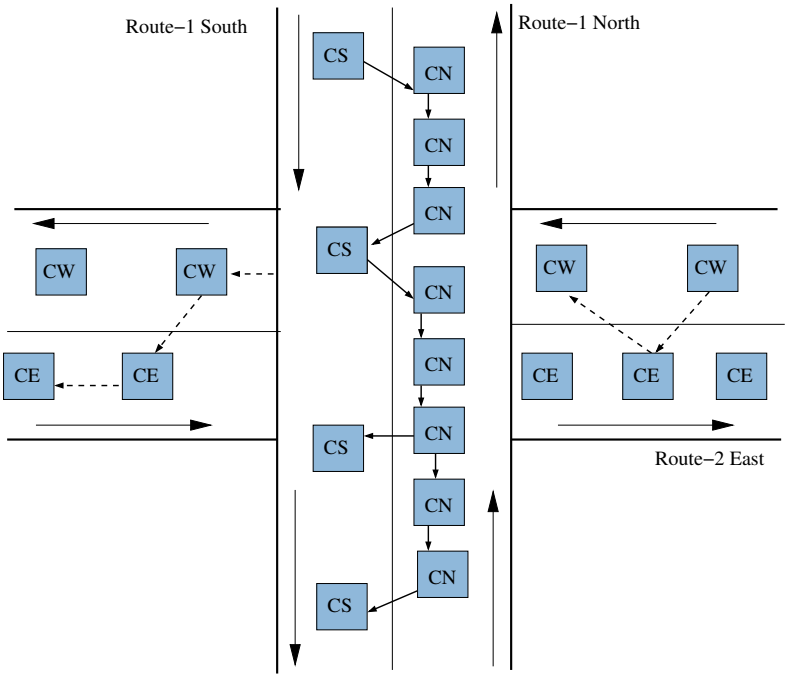

Figure 9: Message Propagation in the VANET Scenario

```
  </TTL>
</event>
....
```

A vehicle on the highway traveling along *Route-1* can be either north-bound or south-bound. Depending upon its direction of travel, a node relays messages to other vehicles forward or backwards, thus achieving directional propagation. The above message propagation can be achieved by defining a function upon the direction of travel of node, the location attributes of nodes in the neighborhood and the destination attribute of the message.

4.4 Animal Tracking

A popular application in sensor networks is to track the movement of an object (e.g., an animal) within a sensor network grid. Tracking is achieved by associating the object with a sensing modality (e.g., visual, aural, radio). The signal may be observed by multiple sensors around the object. A localization of the object is obtained by applying algorithms on the signal values observed at any given time. With an objective of tracking an object, identified target coordinates are disseminated to a base station in addition to neighbors to support tracking. This is an example illustrating the use of a rooted tree for collecting track information and the application of in-network processing to facilitate efficient tracking.

The network periodically observes the signal at each node to detect any activity of the object. The instant when an object enters the activity zone is recorded by a transition in the indicator signal in the network. The node with the maximum signal is declared as leader in the tracking process. All nodes share the indicator signal and a time-stamp indicating the signal capture time. Further, it is assumed that the rate of observation of indicator signal is greater than the rate of movement of the object. Thus, when an object moves within the network, the strength of indicator signal changes with the object movement. The *leader* is aware of this movement as all nodes in the vicinity share the indicator signal strength. The *leader* then relinquishes its post as the leader and assigns the node in its vicinity with the highest signal strength as the leader. At such time it records in its database the time-range or time stamp at which it was the leader. The new *leader* is thus aware of its predecessor leader and it continues the same process until the object moves and it assigns a new leader. Representative data generated inside each node is shown in Table 5.

The *base station* can bootstrap the network and setup a rooted tree structure by initiating communication and progressively setting up parent selection by virtue of hop-count relative to the base-station, as described in Section IV. The base station can flood a query in the network to find the leader at any time (e.g., via a TinyDB-like mechanism [9]). The nodes periodically share the indicator signal observed and the time of observation with neighboring nodes. The nodes also store information pertaining to the leader node in the vicinity. The routing table at each node may be described as shown in Table 6. Once the leader at a given time instant has been established, a trace function is executed at this leader to trace the path followed by the object. The trace function is executed by storing its indicator signal value and time stamp from the database and forwarding this to the successor relative to the time stamp. This can be achieved as in a message shown below.

```
<msg id>01</msg id>
```

Table 5: Leader Table Maintained at a Node in the Network

Node-ID	*Indicator Signal*	*Time Interval*
189	-75dBm	19:38-19:41
193	-72dBm	19:41-19:43
113	-69dBm	19:43-19:46
141	-65dBm	21:20-21:21
159	-73dBm	21:21-21:24
117	-70dBm	22:28-22:30
174	-65dBm	22:30-22:37
...	...	...

<type>*trace*</type>
<hopcount> 5
<node id>117</node id>
<time> $19:40-19:42$ </time>
<signal-strength> -70dBm
</signal-strength>
</hopcount>
<hopcount> 4
<node id>139</node id>
<time> $19:37-19:40$ </time>
<signal-strength> -72dBm
</signal-strength>
</hopcount>
...
<stop-condition>
<hopcount> 20 </hopcount>
<time> $20:00$ </time>
<signal-strength> 0dBm
</signal-strength>
</stop-condition>

This process is repeated at each node until a stop condition is achieved. The node at which the stop condition is achieved now has in store the path followed by the object. The node then modifies the header information and relays the path information to the base station. This objective is achieved by forwarding the data now to the parent nodes successively to reach the base station. This can be done in a message as shown below.

Table 6: Illustration of Node Routing Table

Node-ID	*Attribute*	*Value*	*Descriptor-Tag*
113	hopcount	4	child
141	hopcount	4	child
117	hopcount	3	peer
189	hopcount	3	peer
193	hopcount	2	parent
174	hopcount	2	parent
141	indicator-signal	-75dBm	leader
117	indicator-signal	-69dBm	predecessor
189	time-range	17:37-17:42	last-observation
193	time-range	18:23-18:24	new-observation
174	time-range	18:23-18:24	new-observation
...	...	...	...

```
<msg id>01</msg id>
<type>base log</type>
<hopcount> 5
  <node id>117</node id>
  <time> 19 : 40 − 19 : 42 </time>
  <signal-strength> −70dBm
    </signal-strength>
</hopcount>
<hopcount> 4
  <node id>139</node id>
  <time> 19 : 37 − 19 : 40 </time>
  <signal-strength> −72dBm
    </signal-strength>
</hopcount>
...
<stop-condition>
  <hopcount> 20 </hopcount>
  <time> 20 : 00 </time>
  <signal-strength> 0dBm
    </signal-strength>
</stop-condition>
```

Thus, as shown in Fig. 10 the dotted lines indicate the path traversed by the object and, hence,

the path followed by the trace function. The solid line indicates the path followed by the data logging function which reports the trace path back to the base station. The parent selection at each node is defined in the semantic routing rules module for each corresponding function. For a trace function, the next hop is selected as the successor from the database generated at a time described. While the *base-log* function requires selecting a parent from the routing table. The *leader-selection* function requires selecting the node with highest indicator signal strength as the leader. The different functions being performed in the network can be described in Table 7

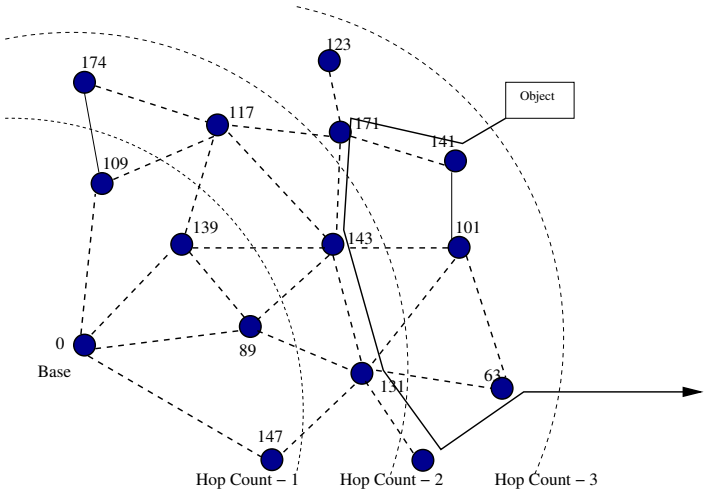

Figure 10: Illustration of Path of an Object Traced in the Network

Table 7: Message Exchange for Performing the Object Tracking Application

Message Type	*Description*
Beacon	All nodes in the neighborhood sharing hopcount, indicator signal from object, etc.
Leader Selection	Node with the highest indicator signal strength
Query	All child nodes satisfying the *time-range* criterion
Trace	Successor or predecessor nodes stored in the *Leader* table
Base Log	Parent node from the routing table

In this example, we demonstrated that the attribute-based routing structure can simultaneously support a number of different routing structures. By using different message schemas and embedded attributes the routing rules can generate different routing patterns. The *leader-selection* function is an instance of in-network processing where nodes compute the leader in the local neighborhood and assign a special role to the *leader*.

5 Discussion and Analysis

We described an attribute-based routing framework which seeks to provide a common framework for implementing data-centric and distributed routing protocols. The routing protocols rely on strictly localized communication unlike conventional networks that perform end-to-end multi-hop communication. This is essential for scalability in sensor networks as the cost for route establishment and maintenance can grow prohibitively with increase in nodes. Moreover, by implementing an in-network processing module and using XML schema within each node, we enable each node in the network to interpret data and interest messages. The network is reactive to changes in local conditions in the network such as the presence of voids in the GPSR example. Sensor nodes need not have globally unique identifiers, rather, nodes need only distinguish themselves among neighbors.

The rooted tree structure discussed in Section IV is an example of a distributed protocol that does not rely on end-to-end path formation strategies. While it has limited capability in terms of communication paths, it is a relatively simple algorithm to implement. Routing protocols which rely on end-to-end path formation strategies such as AODV and DSR require at least $12n^2 - 6n + 1$ transmissions to set up paths, while the rooted tree requires at most n transmissions to set-up the tree structure (*n* is the size of the network.) Furthermore, the overhead for route caching and route maintenance is relatively high as compared to the rooted tree structure which requires only periodic beaconing to maintain paths. Finally, the cost of storing XML an schema is a ***fixed*** cost and scalable with the the number of routing structures implemented while the cost of maintaining a routing table in AODV and DSR grows with the size of the network.

In the GPSR case, the implementation described is expected to perform at a level similar to that described in literature with the overhead being use of XML encoded messages. While the cost of using an XML structure in a sensor network may seem unreasonable, the cost can be minimized by using stronger encoding schemes and compression; both of which are valid techniques although with some loss in generality. The aim here is to harness the functional capabilities of XML including flexibility and adaptability.

The VANET example demonstrates the utility of semantic routing rules where the nodes achieve directional propagation along geographically constrained paths. Also interesting is the ability to isolate network traffic originating from orthogonal road traffic (intersecting roads) such that data is only propagated along the desired road path. This example demonstrates how information can be disseminated in a coordinated fashion even in the absence of globally unique identifiers and without flooding the network.

The tracking example demonstrates the support for multiple routing structures and in-network processing in the form of a tracking function. In evaluating the communication costs we assume asymptotic costs of $O(n)$ message transmissions for flooding and $O(\sqrt{n})$ for point-to-point routing where *n* is the the number of nodes. We compare the overall cost of executing a tracking query in a distributed implementation versus the cost in a centralized implementation. An address-based routing protocol cannot sufficiently emulate the behavior of a data-centric protocol due to established fixed end-to-end paths based on a cost criterion.

The cost of setting up a rooted tree structure is $O(n)$. A query can be executed in the network at a cost of $O(\sqrt{n})$ when using a TinyDB implementation or in the worst case $O(n)$ for flooding the network. The trace function involves *k* nodes from the network depending on the size and lifetime of the object in the network, while the cost of logging the path traced by the object would be the

cost of traversing up a tree which is at most $O(\sqrt{n})$. Thus, the total cost of executing a query on the network is $O(n+\sqrt{n}+n+\sqrt{n})$ which is $O(n)$. While in a centralized implementation, a query involves querying n nodes and getting n responses. Thus, the cost of executing a single query is $O(n^2)$. The costs are significantly different for large networks involving in-network processing.

6 Conclusion

In this paper describe a novel framework to implement content-based routing in sensor networks. The intent is to make the routing function aware of the role of the network and define routing in term of local rules applied on attributes to yield a global property. While in conventional networks, there are a large set of variables, sensor networks have limited number of attribute sets which can be exploited to design the routing structure. The framework demonstrates distributed control which is achieved by a set of modifiable local rules. Multiple routing schemes can potentially be overlaid in this framework with the use of XML schemas and corresponding routing rules. The framework does not introduce a significant penalty to the system and existing performance criteria can be matched. The gains achieved include ease of programmability and added functionality to support complex sensor network applications. Using attributes to associated with nodes in the network provides a layer of abstraction to the application layer. The application layer may define the destination or target as an attribute relationship and the networking layer can perform the function of mapping the intended relationship to a node or a set of nodes which satisfy the class property. An important aspect of the framework is the use of a general representational scheme such as XML to provide interpretation of message contents and attributes by different nodes in the sensor network.

References

[1] E.B. Ermis and Venkatesh Saligrama. A statistical sampling methods for decentralized estimation and detection of localized phenomena. In *Fourth Intl. Symp. on Information Processing in Sensor Networks (IPSN)*, pages 143–150, April 2005.

[2] Bhaskar Krishnamachari, Deborah Estrin, and Stephen Wicker. The impact of data aggregation in wireless sensor networks. In *Intl. Workshop on Distributed Event-Based Systems, (DEBS '02), held in conjunction with IEEE ICDCS*, July 2002.

[3] Yong Yao and Johannes Gehrke. The cougar approach to in-network query processing in sensor networks. *SIGMOD Rec.*, 31(3):9–18, 2002.

[4] Ramesh Govindan. Data-centric routing and storage in sensor networks. pages 185–205, 2004.

[5] Kamin Whitehouse, Cory Sharp, Eric Brewer, and David Culler. Hood: a neighborhood abstraction for sensor networks. In *MobiSys '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 99–110, New York, NY, USA, 2004. ACM Press.

[6] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. *In Proc. of the 1st ACM Intl. Workshop on Wireless Sensor Networks and Applications*, September 2002.

[7] B. Krishnamachari, D. Estrin, and S. Wicker. Modelling data-centric routing in wireless sensor networks. In *IEEE INFOCOM*, 2002.

[8] D. Coffin, D. V. Hook, S. McGarry, and S. Kolek. Declarative ad-hoc sensor networking. *SPIE Integrated Command Environments*, 2000.

[9] Samuel R. Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. Tinydb: an acquisitional query processing system for sensor networks. *ACM Trans. Database Syst.*, 30(1):122–173, 2005.

[10] Sylvia Ratnasamy, Brad Karp, Li Yin, Fang Yu, Deborah Estrin, Ramesh Govindan, and Scott Shenker. Ght: a geographic hash table for data-centric storage. In *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 78–87, New York, NY, USA, 2002. ACM Press.

[11] Matt Welsh and Geoff Mainland. Programming sensor networks using abstract regions. In *NSDI*, pages 29–42, 2004.

[12] Brad Karp and H. T. Kung. Gpsr: greedy perimeter stateless routing for wireless networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 243–254, New York, NY, USA, 2000. ACM Press.

[13] Sameer Tilak, Kenneth Chiu, Nael B. Abu-Ghazaleh, and Tony Fountain. Dynamic resource discovery for wireless sensor networks. In *IFIP International Symposium on Network-Centric Ubiquitous Systems (NCUS 2005)*, 2005.

[14] Tiny xml. URL. *http://sourceforge.net/projects/tinyxml/*.

[15] T.D.C. Little and A. Agarwal. An information propagation scheme for vehicular networks. In *Proc. of IEEE Intelligent Transportation Systems Conference (ITSC)*, Vienna, Austria, September 2005.