A Content-Based Networking Protocol For Sensor Networks

Cyrus P. Hall Antonio Carzaniga Jeff Rose Alexander L. Wolf

Software Engineering Research Laboratory
Department of Computer Science
University of Colorado
Boulder, Colorado 80309-0430 USA
{hallcp,carzanig,rosenj,alw}@cs.colorado.edu

University of Colorado
Department of Computer Science
Technical Report CU-CS-979-04
August 2004

Abstract

An ideal sensor network would minimize communication by routing information only to those nodes requiring the information. We are exploring the use of a content-based network for this purpose, where messages containing sensor readings and associated metadata are relayed from source nodes to destination nodes based solely on the fact that the destination nodes have expressed interest in specific message content. Routing uses a *distance vector* protocol augmented to construct both primary and alternate routes. Forwarding uses content-based matching together with a special process called *dynamic receiver partitioning*. The protocol, called DV/DRP, is specifically designed for wireless sensor networks or other similarly constrained network configurations. We present simulations showing that the protocol scales to large networks while minimizing the resource consumption of individual nodes. We also show that the protocol is robust with respect to both transient and permanent node and communication failures.

1 Introduction

A sensor network is typically composed of a set of nodes that are capable of measuring various types of phenomena. The kinds of data, as well as the rates at which those data can be usefully consumed, are characteristics of the applications built on top of a sensor network. Sensor nodes themselves are increasingly seen as general-purpose, commodity items that simply produce generic data. Therefore, as others have argued [12], the message traffic in a sensor network should be driven by the dynamic "interests" of the application, rather than by the particular static capabilities or configurations of the sensors. The application might, for example, be interested only in receiving temperature readings from a particular region that exceed a certain threshold or, because of its knowledge of how the data are to be used, it might be interested only in receiving periodic messages containing a temperature reading.

Sensor networks are an ideal application of the *content-based networking* model of communication [7]. In this model a message is transmitted from a sender to one or more receivers without the sender having to address the message to any specific receiver. Receivers express interest in the kinds of messages they would like to receive, and the network delivers to the receivers any and only messages matching those interests. Interests are expressed by receivers through *predicate advertisements*. In this receiver-driven style of communication, the network is responsible for efficiently applying predicates to the content of messages so as to minimize the computational and communication costs of the network.

We propose a protocol that implements the content-based networking model specifically for wireless sensor networks and other similar network configurations. Its design is based on the following assumptions.

- *Primitive communication infrastructure:* Nodes in a sensor network have direct access only to their immediate neighbors and communicate with them through a point-to-point or local-broadcast link-layer communication service. There is no multi-hop network service such as unicast, multicast, anycast, or flooding (broadcast) available.
- *Resource-constrained nodes:* Nodes are severely limited in the amount of state they can maintain, in the amount of messages they can send and receive, and in the amount of energy they can consume.
- *Many producers and few consumers:* Most of the nodes will act as information producers, while only a few, resource-rich nodes will act as information consumers. The producers (i.e., message senders) are the sensor nodes. The consumers (i.e., message receivers) are commonly referred to as *base stations*, providing functions such as data correlation and gateways to higher-level applications.

In addition, we are currently assuming that the nodes are stationary. Although mobility should not have a major impact on the protocol, we have not studied its behavior in the presence of mobile nodes.

The key features that together distinguish the protocol from others proposed for sensor networks are as follows.

- *An energy-efficient distance-vector routing protocol that can tolerate both transient and permanent network failures.* For each receiver, the protocol creates forwarding state corresponding to a shortest-path spanning tree and maintains a configurable array of alternatives that can be used in case the best route fails.
- *A forwarding scheme that efficiently evaluates message content against receiver predicates.* Predicates are applied to a message once, at the node where the message enters the network.[1] Using a compact bit vector, the message is annotated with an indication of the intended receivers, where each bit represents a different receiver. Relay nodes need only evaluate the bit vector, rather than the full predicate.
- *A routing scheme that avoids loops and redundant paths without requiring the storage and maintenance of message caches or additional routing state.* When a node replicates a message so as to send it down two or more paths (i.e., acting as a path splitter within the topology of the network), each replica is annotated with a partition of the intended receiver space. This simple and local mechanism prevents loops and redundant paths without the need for a costly reverse-path flooding broadcast.

[1] Sensor nodes are entry points for the messages they produce for their own data, as well as relay nodes for the messages produced by other sensor nodes; they must be capable of exhibiting both functionalities.

- *A mechanism to limit the rate at which messages are delivered, rather than to limit the rate at which messages are produced.* Following the principle that application interests should drive message traffic, the protocol allows receivers (i.e., base stations) to specify delivery rates as part of predicate advertisements. The protocol spreads and integrates the rate information just as it does the predicates. This allows the network to optimize traffic flow under the given rate limitations by adaptively integrating production rates across the many sensors located around the network.

We call our protocol the *distance vector / dynamic receiver partitioning* (DV/DRP) protocol.

In this paper we present the design of DV/DRP and give results of a quantitative evaluation conducted over a number of simulated scenarios. The simulations show that: (1) the service is functional, stable with respect to growing networks, and robust in the face of transient and permanent network failures and (2) the service holds resource consumption to a minimum. For example, in a network of 500 sensors with 10 base stations and in the presence of intermittent link failures for 10% of the nodes, the average level of control traffic remains under 60 packets per second throughout the entire network. In terms of power consumption, we estimate that this amounts to only 1.5nA (5400nAh/s) over the entire network.

2 Related Work

The sensor-networking community has studied a wide variety of network protocols. A large portion of this work involves techniques for optimizing various aspects of link-layer communications. For example, network data aggregation [13, 14], node clustering techniques [11] for minimizing radio listening time, and energy efficient MAC protocols [17, 18] have all been explored. Many of these techniques can be applied to a network running DV/DRP to further improve its performance.

At a higher level of communication, directed diffusion [12] shares many of our goals. Directed diffusion is a data-driven network service that provides a similar interface to that of DV/DRP. The general routing strategy underlying directed diffusion is based on three processes, not all neccesarily used together: (1) receiver interests are "diffused" throughout a sensor field to establish so-called "gradients"; (2) sensor data are sent toward receivers either in a flooding fashion or by following gradients; and (3) gradients are dynamically modified by applications through reinforcement of paths.

Although originally conceived as a particular routing protocol, directed diffusion is now presented as a general conceptual framework for a family of protocols [10]. Many fundamental protocol decisions are not defined, but instead delegated to the application (e.g., gradient establishment and reinforcement), giving great flexibility to the protocol designer.

DV/DRP is rather different from directed diffusion in this respect. DV/DRP is a fully specified protocol with an application interface that clearly isolates the application from routing decisions (see Section 3). We chose this architecture for two main reasons. First, our target is a network of extremely simple and resource-constrained nodes. In such networks, deploying and executing application-specific logic on each node may be either infeasible or undesirable. Second, and perhaps more importantly, we see potential problems in delegating routing decisions to the application developer. In particular, doing so offers little or no guarantees in terms of interoperability, global optimality, or even stability.

In recent work, the general directed diffusion framework has been instantiated as three classes of protocols [16]. Two of these protocol classes, "push" and "two-phase pull" diffusion, are based on a common scheme in which sensor data are flooded throughout the network, and receivers establish preferred routes through positive reinforcement. The primary benefit of this scheme is that by keeping multiple paths alive (i.e., by replicating messages) the network is probabilistically more reliable. There has also been work done on using multipath routing in directed diffusion [9]. However, that solution still replicates messages when there are no failures in the primary path.

In DV/DRP we adopt a rather different approach to route recovery. In particular, DV/DRP deals with transient failures using *local alternate routes* and permanent route failures using a *reactive repair* mechanism. While we have not yet performed a detailed analysis, we hypothesize that the overhead involved in continuously inflating traffic on a per-message basis, as done in push and two-phase pull diffusion, is greater than the cost of maintaining local alternate routes and repairing broken paths on a per-receiver basis.

DV/DRP is most similar to the third directed diffusion protocol class, "one-phase pull". One-phase pull is purely a publish/subscribe scheme, where interests (subscriptions) are flooded to all nodes, and where data (publications) follow backwards the paths established by subscriptions. Algorithms for such a scheme have been extensively studied in the publish/subscribe literature [2, 5] and in the area of content-based networking [4, 6, 8]. While forming the theoretical foundation of DV/DRP, the algorithms in the work described here are substantial adaptations for use in the resource-constrained and primitive environment of a wireless sensor network.

3 Routing and Forwarding

The content-based networking model [7] offers a promising approach to energy-efficient, network-level communication services in resource-constrained sensor networks. Instead of using explicitly configured and addressed end points to form communication paths, routes are formed by receivers advertising a message selection predicate to the network. A message containing data that match one or more selection predicates is forwarded toward the receivers advertising those predicates.

Consider a base station in a sensor network monitoring wildland fire conditions. The base station may want to be notified (on behalf of some higher-level application) if the temperature or wind speed have reached critical values. The base station might advertise the selection predicate shown in Figure 1. The horizontal line in this

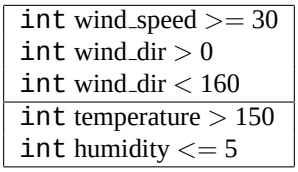


Figure 1: Example Receiver Predicate

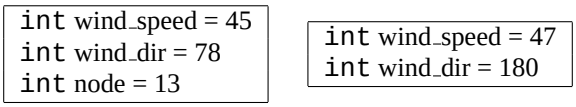


Figure 2: Example Messages

example indicates that the predicate is a disjunction of two expressions; each expression is itself a conjunction of more primitive constraints on individual values.[2] Sensor nodes might produce messages like the ones shown in Figure 2. The messages are routed toward receivers that have advertised matching predicates.

Clearly, this form of network-level service model is akin to the application-level communication model known as *publish/subscribe*. Similar models have been proposed and used in the design of sensor networks [12].

In the context of a multi-hop network protocol, forwarding information is derived, in part, from the predicates [8]. Predicates are used in the construction of forwarding tables residing at individual nodes [6]. The forwarding algorithm performs a match against these predicates. If a message matches the constraints associated with one or more outbound interfaces, then the message is forwarded to the next node(s) along the path(s) to the intended receiver(s). In the example above, the first message matches the predicate, while the second one does not.

Figure 3 shows the formats of the message and predicate advertisement packets used in DV/DRP. The various fields are described below.

3.1 Routing

DV/DRP uses a straightforward distance-vector routing protocol augmented to maintain an array of alternate next hops. When a receiver node r advertises a predicate p_r, the routing protocol propagates p_r to every node in the

[2] Attribute names, such as "wind_speed", need not be strings; we envision using enumerated integers for actual deployment.

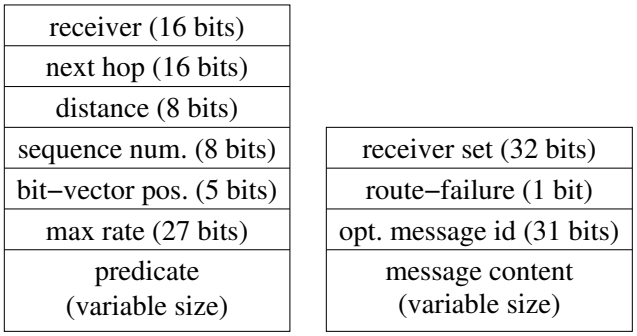


Figure 3: Predicate Advertisement (a) and Message (b) Packet Formats

network, thereby forming a content-based forwarding tree rooted at r. This can be thought of as the equivalent of a route advertisement. The content-based forwarding tree "attracts" messages matching p_r toward r.

The routing protocol uses a table to represent a distance vector. For convenience, the same table also stores receiver predicates and, therefore, serves as a forwarding table. For each receiver r in the network, the table stores path information as well as path-independent information. The path-independent information consists of the receiver predicate p_r, a sequence number s_r, a bit-vector position b_r, the minimum inter-arrival interval Δ_r, and the time of the last matching message t_r.

Path information consists of an array of distance-vector entries $(n_r, l_r), (n'_r, l'_r), (n''_r, l''_r), \ldots$ The length of this array is a static parameter of the protocol. The first entry in the array represents the usual distance-vector information. In particular, n_r is the next-hop node on the shortest path toward r, and l_r is the length of that path. The other entries represent the *local second-best path*, the *local third-best path*, and so on, which are used as alternate paths in case of path failure. We define the *local n^{th}-best path* as the n^{th}-best path computed by a traditional distance-vector protocol. In practice, an alternate path goes one hop away from the current node (along a sub-optimal path), and then follows a primary (or, if necessary, alternate) path from there. This definition is exemplified by the graph of Figure 4.

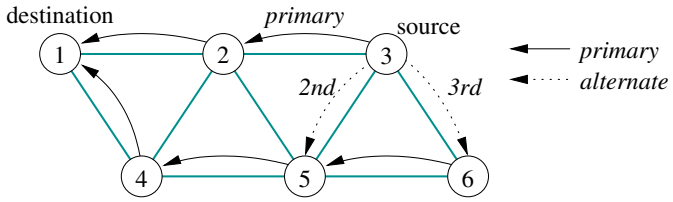


Figure 4: Primary and Alternate Paths

Figure 5 shows examples of routing/forwarding tables at two points in time. The figure shows a network of five nodes, two of which (nodes 1 and 2) become receiver nodes. Initially, node 1 advertises predicate p_1. The predicate propagates through the network, generating a content-based forwarding tree rooted at node 1, represented by the solid arrows in Figure 5a. Later, node 2 advertises predicate p_2. This predicate propagates throughout the network, creating the forwarding tree represented in Figure 5b with dashed arrows. Also shown are the states of the routing/forwarding table of node 5.

The propagation of predicates follows a simple variation of a traditional distance-vector protocol. A predicate is new when it refers to a new receiver, or when it carries a sequence number greater than the sequence number stored in the table for that receiver. This allows the receiver to change its interests dynamically. In the first case, a new entry is added to the table. In the second case, the existing entry is updated with the new predicate, its primary path is updated, and all the alternate paths are cleared (and then rebuilt). In both cases, the predicate is broadcast to all the neighboring nodes. A predicate is obsolete, and therefore immediately dropped, if its sequence number is lower than the one already stored in the table. If a predicate comes in with the current sequence number for its receiver, it is

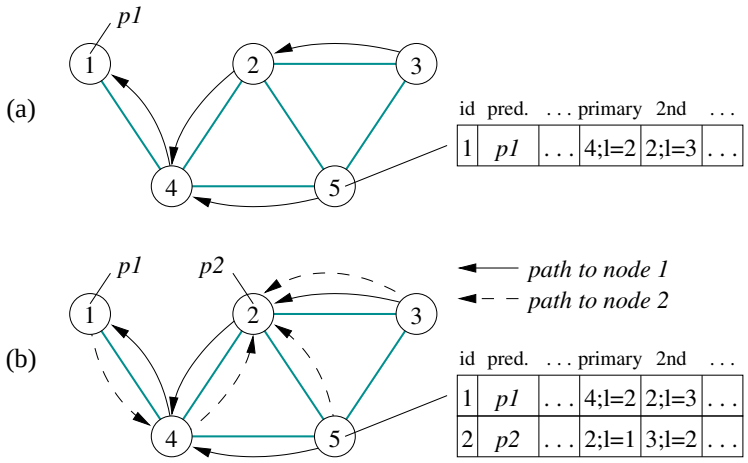


Figure 5: Routing/Forwarding Tables

inserted in the path array according to its distance, and propagated if it is inserted as the primary path. A zero sequence number in the predicate is used to reset the sequence counter in the routing/forwarding table. A "null" predicate can be used to remove a route entry if a receiver is no longer interested in messages.

3.1.1 Recovery

Sensor networks are subject to failures that can be modeled as transient and permanent node failures. For example, wireless links exhibit fading and other types of intermittent interference, which appear as node failures to neighbors. Also, nodes may be permanently damaged by environmental agents, or they may simply run out of power, causing permanent failures.

DV/DRP deals with failures by detecting errors in data transmissions and by reacting to those errors. At first, DV/DRP tries to route messages around the failed node. If the failure persists, DV/DRP tries to repair the broken paths by reissuing predicate advertisements. In particular, the forwarding algorithm of DV/DRP starts by sending a message to the next-hop node on the primary path to the given receiver. If communication to that node fails, DV/DRP (1) sets a *route-failure* flag in the message, (2) attaches a random message identifier to that message, (3) inserts that message identifier into a local message cache, (4) tries to send the message along one of the alternate routes, and (5) if all available paths fail, it sends the message as a flood packet. When a node receives a message that has the route-failure flag set, the node must first check if that message identifier is not in its cache. If the node finds the message identifier in its cache, then it concludes that the packet got trapped in a loop by following one or more alternate routes. In this case, the node falls back to step (5), sending the message as a flood packet. In essence, DV/DRP tries to route a message around failed nodes by first trying alternate routes, and resorting to flooding only as a last option.

In our experiments, we have observed that usually less than 5% of all route deviations result in loops. We have also studied the prevalence of loops in random topologies of various size. Due to lack of space, we do not report our study here. However, in over 50 500-node topologies studied, we observed loops in at most 15% of all sender/receiver combinations.

If and when the message gets to its final destination, the route-failure flag will inform the receiver node that one or more branches of its forwarding tree have failed. After seeing a certain number of failures, the receiver node will attempt to repair broken routes by reissuing a predicate advertisement. The policy that controls this reactive re-advertisement process is a configurable parameter of the protocol. In addition to this reactive recovery method, DV/DRP implements a proactive method based on heart-beat re-advertisements. In our experience, reactive recovery is quite effective, so that in many situations the rate of proactive re-advertisements can be reduced to a minimum or completely turned off.

To avoid congestion and to limit the power consumption due to flood packets caused by failed routes, DV/DRP can limit the maximum rate at which nodes are allowed to issue flood packets. This does not affect the fault tolerance

of the network unless the rate is greater than the period between failures. The maximum rate is a configurable protocol parameter.

3.1.2 Data Rate Limitation

By default, predicate advertisements do not impose a limit on the rate at which messages are delivered. This behavior, however, can be undesirable and even destructive, especially in large networks of sensors producing many messages matching the predicates. In these cases, a receiver and its surrounding relay nodes can be overwhelmed by the message flow.

In order to adapt the content-based service to the needs and capabilities of applications, DV/DRP allows receivers to specify a limit on the message delivery rate. This specification is given as an integral part of a predicate advertisement. A receiver r can advertise a predicate p_r and a minimum inter-arrival interval Δ_r ($\Delta_r = 0$ means no rate limitation). Minimum inter-arrival intervals are then propagated together with the advertisement, and recorded in the routing/forwarding tables. The semantics of this rate limitation service is not intended to uphold fairness across sources. In some circumstances, sources may continually overshadow others, which would lead to a non-representative view of the network from the base station(s).

3.2 Forwarding

The forwarding state produced by the DV/DRP routing protocol is intended to "attract" matching messages toward receiver nodes. A node should proceed to forward an incoming message by matching the message against all the predicates in the routing/forwarding table. The basic idea is that a message m matching predicates $p_1, p_2, \ldots, p_i$ is forwarded to all the next-hop nodes $n_1, n_2, \ldots, n_i$. This matching process is also called *content-based forwarding* [8].

3.2.1 Short-Circuiting Predicate Evaluation

Every node has complete knowledge of all the (small number of) receivers in the network. Therefore, the forwarding algorithm can apply predicate evaluation for a message m, once and for all, at the node where m first enters the network. The results of the evaluation are stored in a header field of the message, called the *receiver set*. At every subsequent hop in the path, the predicate evaluation can be avoided. Instead, all that is necessary is an evaluation of the receiver set against the forwarding information in the routing/forwarding table.

One way to represent a receiver set is as an array of receiver identifiers. This approach is very simple, and can be used directly with existing statically assigned node identifiers, such as MAC addresses. The problem, however, is that it introduces a prohibitive communication overhead, due to the potentially large size of the array. Another approach would be to use Bloom filters [3] to obtain a compact representation of a set of receiver addresses. Unfortunately, Bloom filters offer only a probabilistic membership test.

Our solution is to use a simple fixed-size bit vector, where each receiver is represented by a dynamically assigned bit position. The obvious advantage of this solution is that it offers a compact representation that incurs no collisions. In principle, the fixed size of the bit vector could be a serious disadvantage, as it imposes an upper bound on the number of active receivers in a network. In practice, such a limitation does not affect most applications in sensor networking, where the vast majority of the nodes are senders (i.e., sensors), and only a few nodes act as receivers (i.e., base stations).

The disadvantage of the bit-vector solution is that it requires nodes to somehow negotiate their bit-vector position. Although static assignment of the position could be performed before deployment, a better approach is to support dynamic assignment. Fortunately, this can be done with a minor extension to the routing protocol, and with a simple local conflict-resolution protocol. Specifically, predicate advertisements are extended to carry the bit position of the receiver. When a node r advertises a predicate p_r for the first time, r randomly chooses its own bit-vector position, b_r, among the ones that are not already in use by other receivers. The node then sends out the predicate advertisement, following the usual distance-vector protocol, with the receiver identifier r, the predicate p_r, and the bit position b_r.

Because it takes some time for predicate advertisements to propagate through the network, it is possible that two or more nodes will pick the same bit-vector position. This rare event is detected by the conflicting nodes as soon as they receive each other's respective advertisements. When a conflict is detected, the node with the lowest node identifier is

given priority and, therefore, keeps its chosen bit-vector position. All the other nodes in conflict are forced to choose a different bit-vector position, and resend their advertisements.

3.2.2 Dynamic Receiver Partitioning

It it easy to see that content-based forwarding delivers messages to interested receivers. However, content-based forwarding alone will also exhibit duplicate deliveries and route loops. As an example, consider the scenario of Figure 5b. Assume a message m matching both p_1 and p_2 is sent by node 5. Following both the content-based paths of p_2 (dashed) and p_1 (solid), the message gets to nodes 2 and 4. Then, the copy that went to node 2 is sent again to node 4, and vice versa, thereby creating duplicates. Loops also occur between nodes 2 and 4, and between nodes 1 and 4. The loops can be avoided by a forwarding algorithm that does not send a message back to the node where it came from. However, it is easy to construct examples with loops of three or more nodes, where that simplistic control would not be effective. To avoid duplications and loops, we have designed a forwarding protocol that augments the basic content-based forwarding with a process called *dynamic receiver partitioning*.

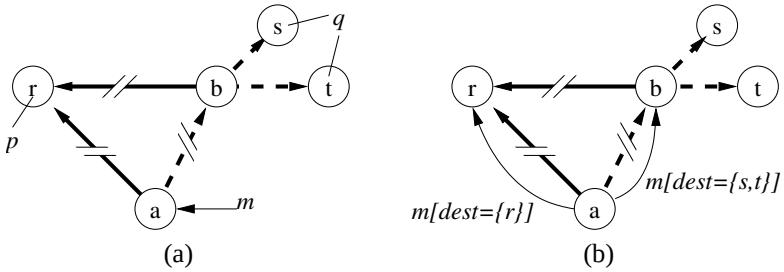


Figure 6: Dynamic Receiver Partitioning

We observe that duplicate paths to a receiver r occur when two copies of a message m cross two distinct branches of the content-based spanning tree of r. Figure 6a shows a scenario that would cause such a duplication. The figure represents a fragment of a larger network. For generality, we consider content-based paths rather than direct links, as symbolized by broken arrows. The solid and dashed lines represent the content-based tree of predicates p and q, respectively. Predicate p is advertised by node r, while predicate q is advertised by nodes s and t. The example shows a message m, sent by node a, matching both p and q.

Figure 6b illustrates the basic idea of dynamic receiver partitioning. Node a first computes the set of receivers $D = \{r, s, t\}$ interested in m. Then a partitions D according to the next-hop node on the path to each receiver. The result consists of two partitions $D_1 = \{r\}$ and $D_2 = \{s, t\}$, which a attaches to the copies of m going towards r and b, respectively. The idea is that the receiver set attached to a message limits the scope of the forwarding for downstream nodes. In the example, node b receives a packet containing m and $D_2 = \{s, t\}$, which limits the propagation of m to s and t.

3.2.3 Forwarding Algorithm

Figure 7 sketches the content-based forwarding algorithm used in DV/DRP. The algorithm makes use of one of two procedures, depending on where within the network the forwarding is to take place. The first procedure, cb_drp_init_forward, takes a message as a parameter, and is executed at the node where the message enters the network. The second procedure, drp_forward, takes a message and a receiver set as parameters, and is executed at each subsequent hop in the routes to the receivers. The procedures are similar in structure, differing only in that the first procedure performs the full predicate evaluation, while the second instead performs only the receiver-set membership test.

4 Evaluation

We evaluated the DV/DRP protocol by implementing it in simulation and then analyzing its behavior under several scenarios. The primary goals of our analysis were to: (1) assess the ability of DV/DRP to implement the content-

```
proc cb_drp_init_forward(message m) {
  map<node, set<node>> P := ∅
  foreach r in fwd_table {
    if time() − t_r > Δ_r and match(m,p_r) {
      P[n_r] := P[n_r] ∪ r
      t_r := time()
    }
  }
  foreach n in P { //for each next-hop node
    send(m,P[n],n)
  }
}
```

```
proc drp_forward(message m, set<node> D) {
  map<node, set<node>> P := ∅
  foreach r in fwd_table {
    if time() − t_r > Δ_r and r ∈ D {
      P[n_r] := P[n_r] ∪ r
      t_r := time()
    }
  }
  foreach n in P { //for each next-hop node
    send(m,P[n],n)
  }
}
```

Figure 7: Sketch of the Forwarding Algorithm

based delivery model; (2) profile the costs incurred by DV/DRP, and thereby evaluate its applicability to networks of resource-constrained nodes; and (3) make sure that the protocol is stable and responds gracefully to application demands, as well as to network failures.

4.1 Experimental Setup

Our experiments were conducted on random network topologies of 10, 25, 50, 100, 250, and 500 nodes. Topologies were generated by placing nodes randomly within a square target area, and by connecting to each node the other nodes within a given range. For all topologies, we maintained a constant density of nodes so as to obtain a constant average fan out (edge connectivity) of between five and six neighbors. Finally, we discarded all partitioned topologies. These topology parameters conform to those used in similar simulation studies [16].

On top of each network topology, we have defined several application scenarios. A scenario is a complete definition of the behavior of each node. In particular, a scenario defines: (1) when nodes send messages, and the content of those messages; (2) when nodes advertise predicates, and the content of those predicates; and (3) how often nodes are subject to communication failures, and for how long.

The messages and predicates we used are similar to those shown in figures 1 and 2. The rate at which senders inject messages is modeled as a Poisson process. Receivers exhibit two types of behavior: some advertise a predicate at the beginning of the simulation, never changing that advertisement, while others change their predicate advertisement periodically throughout the simulation. The parameters that control node failures are the mean time between failures and the durations of those failures, both modeling a Poisson distribution.

For our simulations, we assumed a CSMA MAC that reports failures to the network protocol. However, DV/DRP can also run on top of "send-and-forget" MACs by using application-level acknowledgments with timeouts.

4.2 Content-Based Delivery

The first experiment we conducted was designed to assess the main functionality of the content-based service implemented by DV/DRP. The scenario that defines this experiment is characterized by a network of 100 nodes, in which each node generates messages at a mean rate of one every 10 seconds. The network contains five receivers that change their predicates every 30 minutes. The scenario has no failures.

The results of this first experiment, plotted in Figure 8, are quantified by the percentage of *false negatives* and *false positives* over the total number of messages sent.

We record a false negative whenever a message does not reach one of its intended receivers. The false negatives that are observed at the beginning are due to the natural latency of the network as it propagates the initial receiver predicates. During the rest of the simulation, we observe minor occurrences of false negatives due to latencies in propagating changes to receiver predicates.

False positives occur when a message is dropped at one of its final destinations because it does not match the predicate currently advertised by that destination. In practice, false positives are also caused by the latency of the

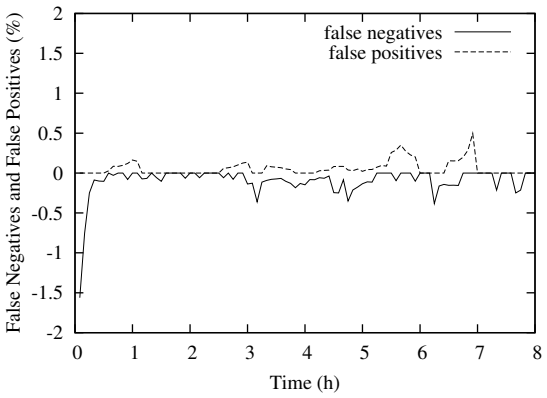


Figure 8: Functionality: False Negatives and False Positives

propagation of advertisements.

The primary conclusion we draw from this experiment is that DV/DRP is effective in implementing the content-based service, showing levels of inaccuracy that remain well below 1% of the overall message traffic. The false positives and false negatives that occur are largely unavoidable, due to the natural latency of network communication.

4.3 Control Traffic

The experiment above assessed the main functionality of DV/DRP, which is to deliver to receivers all and only the messages matching their predicates. The following experiment analyzed the amount of control traffic incurred by DV/DRP in accomplishing this goal.

Figure 9 shows the data and control traffic for the same scenario as Figure 8. The values on the Y-axis represent the number of packets of each kind going through the network at any given time. The first curve represents the message (i.e., "data") traffic. The fluctuations are due to basic application behavior: the changing predicates and the variation in message content produced by the sensors.

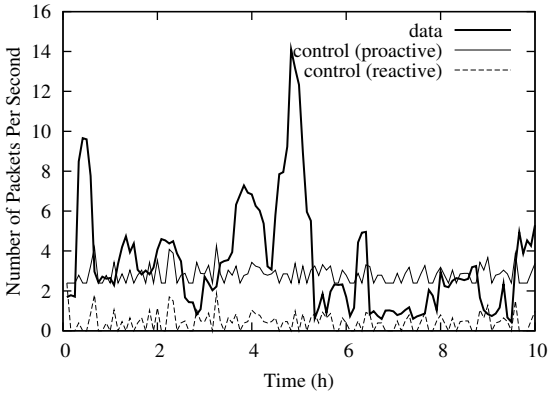


Figure 9: Data and Control Traffic

The second two curves represent the control traffic when using, respectively, the proactive and reactive route recovery methods (see Section 3.1.1). Notice that the two curves are identical in form. This is due to the fact that the scenario contains no failures and the fluctuations in control traffic are caused solely by receivers advertising new predicates. However, the proactive method incurs a fixed higher cost than the reactive method due to its use of periodic re-advertisements whose rate is determined by a (fixed) configuration parameter.

4.4 Scalability

The purpose of our scalability experiments was to assess the protocol as both the size of the network and the number of receivers increases. We also wanted to verify that DV/DRP remains scalable in the presence of transient failures. Figure 10 shows experiments conducted over such scenarios. Each node generates messages at a mean rate of one every 12 seconds and each receiver changes its predicate at a mean rate of once every 10 minutes. In practice, receiver predicates are likely to be much more stable, so the experimental rate serves to heavily stress the protocol.

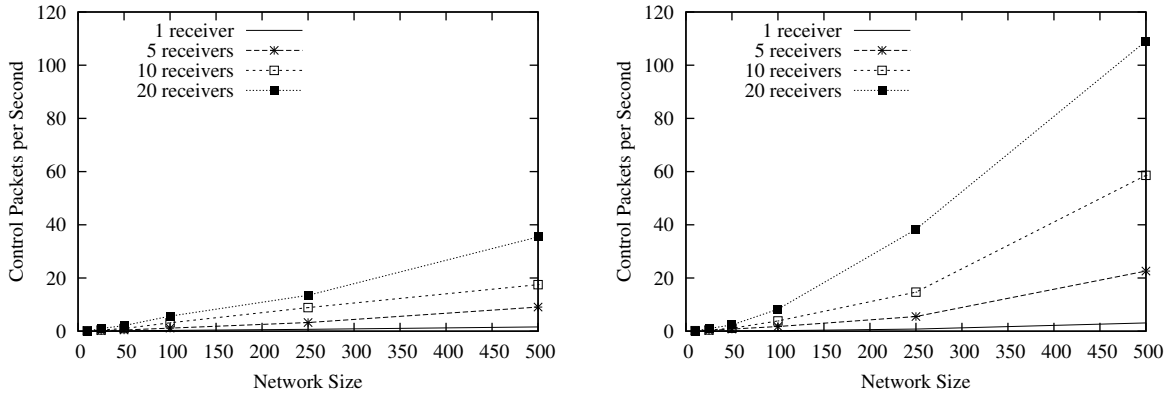


Figure 10: Scalability in Network Size and Number of Receivers

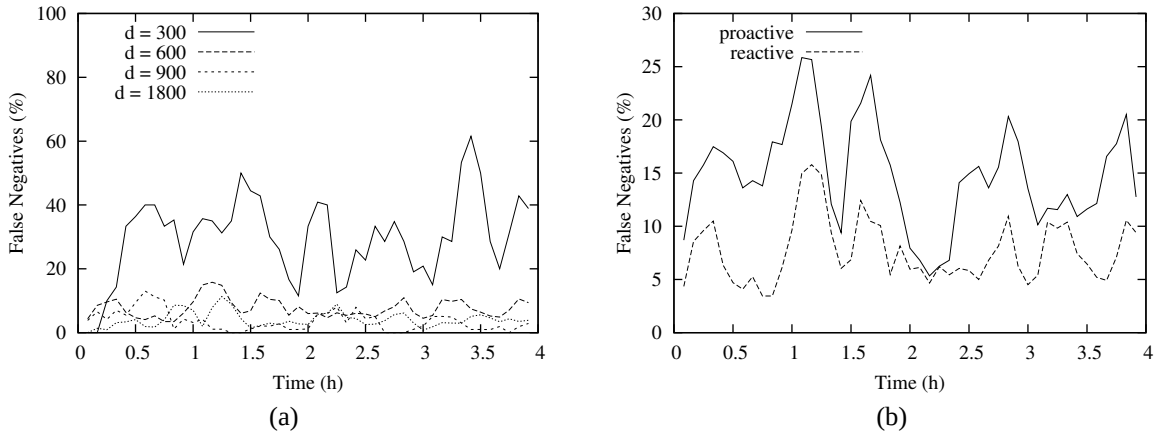
Figure 10a shows, in the absence of failures, that the amount of control traffic in the entire network at any given time grows essentially linearly or less with growth in the size of the network. Figure 10b shows what happens when 10% of the nodes experience failures at any given time and the network uses the reactive recovery method. Even in the case of a 500-node network with 20 base stations, each node processes only one control packet approximately every five seconds caused by the combination of application-level predicate changes and reactive re-advertisements issued to repair broken paths.

4.5 Transient Failures

In this set of experiments we assessed how well DV/DRP recovers from transient failures in the network. The mean time between failure for each node in the network is given as the value d, the failures resolve after a mean time of 60 seconds, and we assume that base stations do not fail. In a real network, failures would be likely to cluster in the same region of the network. However, our simulation does not yet model that scenario. Nevertheless, the parameters we used are realistic enough to get a feel for how DV/DRP performs.

We ran the experiments on the 100-node topology used in sections 4.2 and 4.3. A mean rate of one message every 30 seconds is generated by sensor nodes, and receiver predicates do not change. Figure 11a shows the percentage of false negatives over time, where each curve represents a different mean failure rate. The reactive recovery method was used in all cases. For $d = 300$, where nodes are down approximately 20% of the time, recovery is still quite good, remaining under 40% false negatives for most of the run. In the case of $d > 300$, DV/DRP is able to keep the false negatives below 15% on average.

Figure 11b compares the reactive and proactive recovery methods for $d = 600$. The time between re-advertisements for proactive recovery was set so as to create the same average amount of control traffic as reactive recovery. As expected, reactive recovery results in significantly fewer false negatives. This means that reactive recovery is using approximately the same amount of energy to produce a much better result.

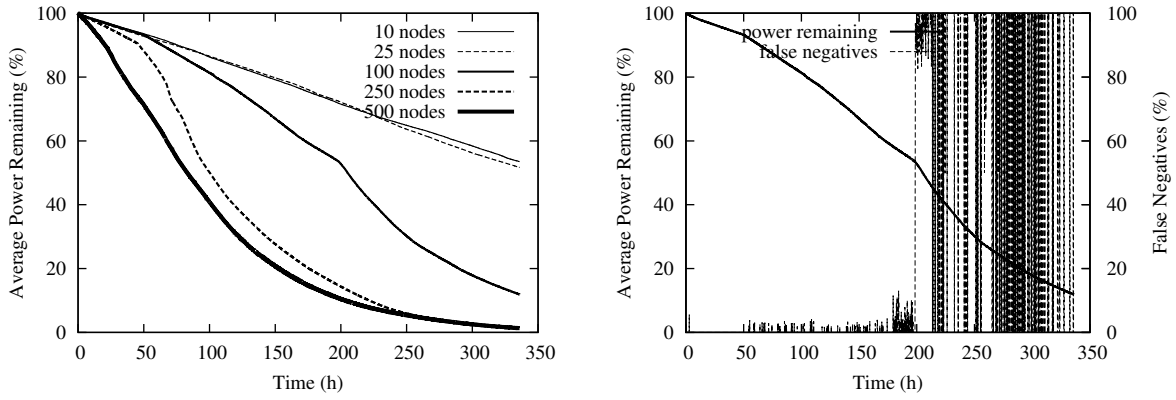


(a) (b)

Figure 11: Behavior in the Presence of Transient Failures

4.6 Permanent Failures

Protocols such as DV/DRP can exacerbate the problem of power-constrained sensor-network nodes by directing a majority of total network traffic toward a small set of nodes, usually clustered around the base stations. In our next experiments, we assessed the tolerance of DV/DRP during periods when nodes are failing permanently due to power exhaustion. Using a given set of power metrics [1, 15], we calculated the amount of battery charge expended each time a message is sent or received, and each time a sensor value is read. We also accounted for current leakage [15]. Figure 12 shows our results.



(a) (b)

Figure 12: Power Drain for Different-Sized Topologies (a) and Tolerance to Permanent Failures (b)

Each node was given the same, very limited amount of initial charge (100000 nAh), except for base stations, which were given unlimited power. Figure 12a shows the percentage of total remaining power in the network for topologies of different sizes. Once the network reaches a critical point, the linear power drain changes slope as the network begins to drain power at a faster rate.

We have traced this critical point to when nodes around the base stations begin to fail. Recall that it is through these neighbor nodes that messages are forwarded to a base station. When a neighbor node fails, another neighbor will be selected (see Section 3.1). However, by the time one node fails, other neighbors of the base station will likely be low on power reserves themselves. There are two reasons for this. First, we assume that the MAC layer is active for all messages, not just those bound for a given node. Second, nodes around the base station have a high likelihood

of being the next-hop for a large percentage of traffic in the network and so experience a high level of usage. After the first node fails, a remaining node is chosen, and over time paths become longer, and more power is used, both to deliver messages and maintain routes.

A well-behaved protocol should minimize the rate of undelivered messages until a base station has no neighbors left and has been completely partitioned from the network. Figure 12b shows both the power remaining and false negatives for the 100-node topology. DV/DRP handles the degrading state of the network well, with less than 20% false negatives until the base station is completely isolated after 200 hours. The remainder of the graph is dominated by noise: false negatives fluctuating from 0% to 100% caused by time periods where no messages matching the base station's predicate are generated. If the graph were to continue, this effect would eventually end and false negatives would reach zero after all nodes had expired.

5 Conclusion

We have presented a content-based networking protocol that solves many of the problems inherent in sensor-network communication. DV/DRP maximizes proper message delivery and minimizes power consumption through techniques that maintain shortest paths, yet avoid the flooding and loops found in previous protocols. Moreover, the techniques are designed to tolerate both transient and permanent network failures. Extensive simulation studies substantiate our claims.

Work remains to improve and further assess DV/DRP. We have not addressed in-network aggregation or mobility, both of which can be critical capabilities of a sensor network. We also need to explore improved local recovery mechanisms. We plan to deploy DV/DRP on a testbed sensor network in the near future and to report the results of using the protocol in the field.

Acknowledgments

The authors would like to thank Rick Han for his insightful comments. The work of the authors was supported in part by the National Science Foundation and Army Research Office under agreement numbers ANI-0240412 and DAAD19-01-1-0484.

References

[1] H. Abrach, S. Bhatti, J. Carlson, H. Dai, J. Rose, A. Sheth, B. Shucker, J. Deng, and R. Han. MANTIS: System support for multimodal networks of in-situ sensors. In *2nd ACM International Workshop on Wireless Sensor Networks and Applications*, pages 50–59, Sept. 2003.

[2] G. Banavar, T. D. Chandra, B. Mukherjee, J. Nagarajarao, R. E. Strom, and D. C. Sturman. An efficient multicast protocol for content-based publish-subscribe systems. In *19th IEEE International Conference on Distributed Computing Systems*, pages 262–272, Austin, Texas, May 1999.

[3] B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, July 1970.

[4] F. Cao and J. P. Singh. Efficient event routing in content-based publish-subscribe service networks. In *IEEE INFOCOM 2004*, Hong Kong, China, Mar. 2004.

[5] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems*, 19(3):332–383, Aug. 2001.

[6] A. Carzaniga, M. J. Rutherford, and A. L. Wolf. A routing scheme for content-based networking. In *IEEE INFOCOM 2004*, Hong Kong, China, Mar. 2004.

[7] A. Carzaniga and A. L. Wolf. Content-based networking: A new communication infrastructure. In *NSF Workshop on an Infrastructure for Mobile and Wireless Systems*, number 2538 in Lecture Notes in Computer Science, pages 59–68, Scottsdale, Arizona, Oct. 2001. Springer-Verlag.

[8] A. Carzaniga and A. L. Wolf. Forwarding in a content-based network. In *SIGCOMM 2003*, pages 163–174, Karlsruhe, Germany, Aug. 2003.

[9] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *Mobile Computing and Communications Review*, 2002.

[10] J. Heidemann, F. Silva, and D. Estrin. Matching data dissemination algorithms to application requirements. In *First International Conference on Embedded Networked Sensor Systems*, pages 218–229, Los Angeles, California, Nov. 2003.

[11] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *33rd Hawaii International Conference on System Sciences*. IEEE Computer Society, 2000.

[12] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed diffusion for wireless sensor networking. *IEEE/ACM Transactions On Networking*, 11(1):2–16, Feb. 2003.

[13] B. Krishnamachari, D. Estrin, and S. B. Wicker. The impact of data aggregation in wireless sensor networks. In *22nd International Conference on Distributed Computing Systems*, pages 575–578. IEEE Computer Society, 2002.

[14] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: A tiny aggregation service for ad hoc sensor networks. *SIGOPS Operating Systems Review*, 36(SI):131–146, 2002.

[15] J. R. Polastre. Design and implementation of wireless sensor networks for habitat monitoring. Master's thesis, University of California at Berkeley, 2003.

[16] F. Silva, J. Heidemann, R. Govindan, and D. Estrin. Directed diffusion. Technical Report ISI-TR-2004-586, USC/Information Sciences Institute, Jan. 2004.

[17] T. van Dam and K. Langendoen. An adaptive energy-efficient MAC protocol for wireless sensor networks. In *First International Conference on Embedded Networked Sensor Systems*, pages 171–180, Los Angeles, California, 2003.

[18] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *IEEE INFOCOM 2002*, New York, New York, June 2002.