Towards Adaptive Resource-Driven Routing

Angela Dalton
The Johns Hopkins University Applied Physics Laboratory
Email: Angela.Dalton@jhuapl.edu

Christine Julien
Department of Electrical and Computer Engineering
The University of Texas at Austin
Email: c.julien@mail.utexas.edu

***Abstract*—In pervasive computing environments, applications find themselves in constantly changing operating conditions. Such applications often need to discover locally available resources on-demand. Communication protocols have been developed that base discovery not on the unique address of the destination but on application-level characteristics of the destination host. Previous work has focused almost exclusively on purely on-demand protocols to achieve these resource connections. However, because the types of resources desired may be common across applications, the discovery and routing tasks can benefit from some degree of proactivity. In this paper, we describe our adaptive approach to incorporating resource advertisement in an application-driven routing protocol. We describe the adaptation mechanism in our protocol that allows the proactive component to dynamically tune its behavior to operating conditions.**

I. Introduction

Networks that support pervasive computing applications, especially those in highly dynamic environments, depart significantly from the traditional Internet model. Such networks entail opportunistic clouds of communicating nodes in which multihop routes are created on-demand as needed. In these networks, every node serves as both an application-supporting node and a router. In addition to the different network structure, emerging pervasive computing applications are driven by the need for *context-aware services*. Specifically, these applications desire to locate and interact with dynamic sets of locally available resources based on the applications' current locations and changing requirements.

Consider a network that supports mobile vehicles and pedestrians in an "un-wired" city. Wireless communication connects these users in a city-spanning network in addition to connecting them to a network of static kiosks. A visitor's application may request a connection to a mapping resource or an information kiosk for restaurant reviews. City residents may connect to traffic sensors to retrieve real-time information about traffic incidents or congestion.

Common solutions place a dedicated resource discovery service between the application and a traditional routing protocol. Before initiating communication, an application must first contact this lookup service to resolve the data or resource required into a unique address (in much the same way DNS serves the Internet). Approaches tailored to highly dynamic networks have distributed this lookup server in the network, increasing the resiliency to failures and movements and decreasing the latency of lookup time [1], [2]. Fundamentally, however, this approach requires an extra phase of communication to perform resource discovery and has been shown to discover resources that are not necessarily local [3]. Approaches that instead tailor the communication model to the content-based requirements of applications perform better in terms of overhead, latency, and locality of resource discovery [4], [5], [3].

Existing content-based routing approaches function entirely on-demand, exchanging information about available resources only when an application asks for a connection. While content-based routing offers better performance on many application-level metrics, there is also much benefit to registry based approaches. For example, in a simple content-based routing protocol, many requests for a commonly used resource may traverse the same paths, generating excessive overhead for information that could be shared. In this paper, we build on content-based routing approaches to incorporate a degree of proactivity in conjunction with the standard on-demand nature of these protocols. The idea is to advertise the availability of resources within a local region, bootstrapping the discovery process. We then move a step further, enabling our local advertisement area to dynamically adapt to application and network conditions.

We extend a particular instance of content-based communication, Cross-layer Discovery and Routing (CDR) [3], with proactive behavior. Our extension, Hybrid-Adaptive CDR (CDR-HA), advertises resources within a dynamically adjusted radius in response to changing network conditions around the resource provider.

In the next section, we present a brief overview of Cross-layer Discovery and Routing (CDR) and in Section III we present our adaptive model. In Section IV we describe a concrete instantiation of the adaptation. Section V concludes.

II. Cross-Layer Discovery and Routing

In this paper, we use an exclusively reactive-style protocol, Cross-Layer Discover and Routing (CDR) [3], as a starting point and hybridize it. This overview is brief; further details of CDR, its formal behavior, and its evaluation can be found in [3].

CDR enables discovery of distributed resources based solely on the target's attributes, capabilities, or data. CDR is based on source routing, in which each data packet carries with it the entire route it will traverse. To locate a resource, a node creates a request packet that contains a specification of the desired content. This packet is flooded through the network; all nodes forward the packet, and nodes that can provide the requested resource generate a response that traverses the reverse path back to the requester. As part of this process, source routes are generated that each contain a list of the nodes connecting the source to the potential provider. Subsequent data packets use these cached routes; as routes break due to node mobility, intermediate nodes generate error messages which cause routes to be updated. When there are no longer any viable routes, the node reinitiates route discovery.

In comparison to traditional routing protocols for pervasive computing networks, CDR not only provides communication matched to application requirements, but it does so with a very limited amount of increased overhead [3]. This overhead is the result of having to carry application-level information inside routing packets instead of fixed-length addresses. In addition, CDR reduces the latency of discovering resources by removing the level of indirection required by the use of discovery servers. Finally, CDR ultimately discovers resources that are closer in proximity to the requester. In pervasive computing applications that entail mobile devices, these resources are often connected by routes that are less likely to break. They are also often resources more well-suited to the application, since they reflect the application's immediate local environment.

III. The Hybrid CDR Protocol Model

In this section, we describe a proactive extension to CDR based on our hypothesis that a small amount of proactivity in the form of resource advertisement can increase the performance of discovery and routing with respect to several application-level metrics. Our extensions take a hybrid approach similar to that of Zone Routing [6], [7]. Building on CDR, we add proactive advertisement of resources within a locally adjustable region. This section focuses on the proactive component of the protocol extensions necessary to specify and maintain variable advertisement regions.

A. Resource Advertisement Data Structures

In our protocol, a node locally determines the region to which it will advertise each of its resources. Each node maintains a *resource table* that contains entries for each of the node's own resources and the other resources in the network the node knows about (because it received advertisements for them). Fig. 1 shows the information contained in an entry in the resource table. In addition to the obvious information stored about a resource (e.g., its description (*spec*), the region in which it is advertised, indicated by *ttl*, and how to reach it (*route*)), an entry in the resource table also contains a boolean flag indicating whether the table entry contains the preferred instance of the resource. The preferred instance can be determined in a number of ways, for example, the freshest known instance or the one with the shortest route, as is the case in our implementation. The *timestamp* associated with a resource in the table indicates the last time the node received an advertisement for this resource (it is the current time for the node's own resources). This allows the protocol to determine the freshness of advertisement information and is used to construct forwarded advertisement packets as discussed later.

Our hybrid CDR requires Resource Advertisement (RA) packets that both advertise a node's own resources and forward cached advertisements. Each node periodically decides whether to broadcast an RA packet. A node that provides one or more resources computes the advertisement region for each resource. We provide more detail about how the advertisement region is determined in Section IV. An RA packet may also contain advertisements for resources provided by other nodes; in this case the advertisement region has been designated by the provider. The packets carry the application-level specifications of resources from the node's resource table, as well as control information to limit forwarding and enable future optimizations to region determination. We limit the communication overhead incurred sending advertisements by consolidating all of a node's resource information into a single packet, where each resource advertised is represented by an entry with the information shown in Fig. 2. The advertisement entry's time-to-live (*ttl*) is initialized to the size of the region and is decremented each time the resource information is forwarded. The route record contains the information necessary to connect the requester to the provider.

spec	semi-structured description of the resource
ttl	size of the resource's advertisement region
route	route to the resource (from advertisement)
source_id	the source node advertising the resource
timestamp	timestamp of most recent advertisement
preferred	boolean designating whether this is the preferred instance of the resource.

Fig. 1. Resource Table Entry

$\langle ttl, source\_id, route\_record, spec \rangle$
contains a time to live (determined by the radius function), the source's unique id, the route record, and the resource specification.

Fig. 2. Resource Advertisement Structure

B. Proactive Protocol Behavior

Our primary addition to CDR is the periodic broadcast of resource advertisements. A periodic timer triggers creation of a resource advertisement packet from the node's resource table. For each resource the node provides, the advertisement region is calculated; resources provided by other nodes are advertised to the region specified by the provider, which is stored in the resource table. An Ad Structure (as in Fig. 2) is built for each resource advertised. All resource advertisements are packaged into a single packet, which is then broadcast to the network.

Upon receipt of a resource advertisement packet, a node extracts each individual Ad Structure from the packet. Should an entry already exist in the node's resource table for a resource in the advertisement, the entry is updated to reflect the newly received advertisement. New entries are created for resources not previously known by the node. Unlike the resource discovery and route reply packets generated reactively to application needs in CDR, an advertisement with a remaining time to live is not immediately forwarded. Advertisements for resources are instead propagated when the node periodically initiates a resource advertisement action.

Fig. 3 shows the behavior of our proactive extension pictorially. In the figure, node **A** is a resource provider with an advertisement radius of two hops, and nodes **B** and **C** are requesters looking for the resource **A** provides. Dashed arrows indicate the transmission of advertisement packets; heavy arrows indicate transmission of request packets. The grayed nodes store the advertisement for **A**'s resource. Starred nodes indicate locations in which a request matched a stored resource advertisement. In this example, **B**'s request packets found two matches for the same resource provider; **B** will select the one that gives the shortest path. **C**'s request matched an advertisement stored locally.

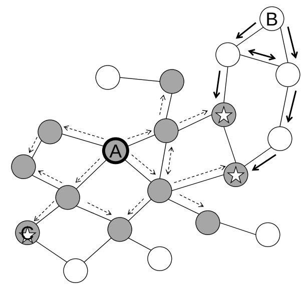

Fig. 3. A two-hop ad radius

Adding proactive advertisement to CDR results in the distribution of readily available routes to regional resources and the ability to adapt the region to respond to changing characteristics of the network. The latter is accomplished by defining different behaviors for the *Region Function* that is used to compute the time to live for the advertisement. In the next section, we describe our adaptive approach to determining the advertisement region in general and give a specific example of an adaptive radius function.

IV. Advertisement Region Determination

A key distinction of the adaptive protocol presented in this paper is the ability for a node to adapt the degree of its proactive behavior in response to changing network characteristics and application requirements. In our protocol, the function establishing the advertisement region can be easily redefined to suit the needs of a given application or deployment. Moreover, it can be adjusted to incorporate additional measures of network and application context that may become available in the future.

A. Defining Region Functions

In our current implementation, the advertisement region is defined as a radius, specified by a time-to-live (ttl), for each resource. The determination is performed locally by each node for each of the resources it wishes to advertise.

A node can determine its initial radius by performing a route discovery for the resource type it will provide. Using CDR, this results in a number of routes to other nodes offering the same resource. The initial radius can then be based on the number of similar resources in the network and the distances to them. A simple initial radius might be half the number of hops in the shortest route provided by the route discovery. Alternately, the initial radius could be set to zero, defaulting to purely reactive routing. Another approach is to set the initial radius to an arbitrary number of hops simply as a starting point. In all three cases, the advertisement radius will be adjusted over time to suit operating conditions of the network.

Given an initial value for a resource's advertisement radius, the job of the region function is to dynamically reevaluate network and application conditions to adjust the advertisement region. Network density and degree of mobility are two context metrics that could provide particularly useful information. For example, in highly mobile environments, advertising in a wide area makes little sense because advertisements and the routes they include quickly become outdated.

In addition to adapting the region to which a node forwards advertisements, the frequency of sending advertisements can also be adjusted to decrease discovery latency or overhead; our initial implementation sends advertisement packets at a static frequency. It is also possible for a node to provide more than one resource. In such a case, the radius for advertising the node's resources should be based on the most frequently requested resource. Including all resources in the advertisement adds minimal overhead when compared with the cost of sending a single packet.

Finally, we can also adjust the shape of the advertisement region. So far, we have only considered symmetric regions, defined by a single time-to-live, centered on the resource provider. However, in many instances, it may make sense to advertise more widely in one direction and less so in another. For example, nodes along a highway may function as information kiosks. It may make sense for the kiosks to exchange advertisements among themselves (i.e., within the static portion of the network) but not to the cars that move rapidly out of range. On the other hand, a caravan or cars on the highway might form a subnet with relatively static network topology; exchanging advertisements within this caravan makes sense, while distributing advertisements to cars moving in the opposite direction does not.

B. An Example Function

We provide herein a discussion of the capabilities of this adaptation framework by defining an example region adaptation function based on information gleaned from requests and advertisements a node handles during network operation. Our sample function begins by defining a static initial radius for all resources. To adapt its radius, our function uses two pieces of application-level information. Perhaps most intuitively we can look at the frequency of requests received for a resource a node is advertising. We increase the radius proportionally to requests because information about a route to a popular resource should be readily available throughout the network. The distance in hops from requesters can also provide an indication of how large a radius should be used. We may want to adjust the radius in proportion to a weighted moving average of number of hops to requesters. We need to limit the radius, however, to prevent creating overhead without resultant benefits. We evaluate these tradeoffs in the next section.

Advertisements from other nodes in the network providing similar resources supply information about the redundancy of a node's provision of a particular resource. We decrease the radius in the presence of other nearby providers, with the effect of limiting each provider's region to that most proximate to itself. Given additional context information about other providers we might be able to tailor the region even more effectively, for example by directing advertisements away from nearby nodes providing the same resource such that each one forwards only to a geographically opposed region. This is one possible example of a more intelligent adaptation, and one that presupposes context information not currently available.

We instantiate this adaptation as follows. To provide smooth adaptation, we use standardized scores for these two parameters. The value for the radius is calculated periodically; every period these two parameters are reevaluated for each advertised resource, and the advertisement radius for each resource is recalculated. Each period, the standardized score for the number of requests seen for a resource in a given period (Z_n^r) is calculated as:

$$Z_n^r = \frac{(N_n^r - \mu_r)}{\sigma_r}$$

where N_n^r is the raw number of requesters counted in a given period, μ_r is a substitution for the population mean, and is instead a running average over all periods where this parameter was observed; σ_n is the standard deviation associated with this data. If Z_n^r is positive, the number of requesters is *on the increase*; if it is negative, the number of requesters is *on the decrease*. We calculate the standardized score for the number of similar observed advertisements (Z_n^a) in the same manner. The function for the radius to be used in period n is:

$$ad\_radius_n = ad\_radius_{n-1} + sign * \frac{|Z_n^r|}{|Z_n^a|}$$

where ad_radius_{n-1} is the advertisement radius used in the previous period, and *sign* determines whether the advertisement radius is increased or decreased. Just the magnitudes of Z_n^r and Z_n^a are used to determine the amount to adjust the radius. In our implementation, *sign* controls the direction in which the adjustment happens. In this example, *sign* is positive if Z_n^r is on the increase *and* Z_n^a is on the decrease or remains the same. Otherwise *sign* is negative.

Fig. 4 shows the effect this adaptive radius function can have on the network previously shown in Fig. 3. This figure uses the same network as before, except that node **D** has begun advertising the same resource as node **A**. The figure shows the state of the network after both **A** and **D** discover that the other offers the same resource, In this example both have adapted their advertisement radius to one hop. **C** will now discover both resources, but will use **A** since it is closest. **B** will discover and use **D**'s resource.

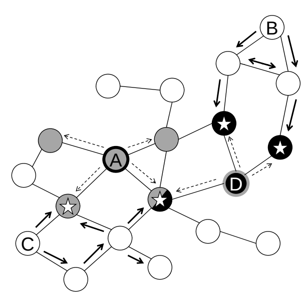

Fig. 4. An adaptive radius function

Using an adaptive radius function allows content-based communication to adjust its behavior in response to the kinds of dynamic operating conditions present in pervasive computing environments. This relieves a developer from having to know *a priori* the nature of the environment in which his protocol will function and allows the protocol to perform well even when the environment changes.

V. Conclusions

By examining pervasive computing application requirements, it becomes immediately apparent that traditional forms of communication based on unique identifiers do not match application behavior. Content-based communication protocols address this shortcoming by allowing indirection in the discovery process. Our new protocol, Hybrid-Adaptive CDR, retains the advantages of content-based CDR while adding a framework through which applications can dictate how the protocol determines when and to what extent to advertise nodes' resources. This radius of advertisement can depend on many aspects of the network and physical environment. In this paper, we explored a simple function that relates the scope of advertisement to the number of similar resources in the network and the number of other nodes requesting access to the resource.

We are working towards enhancements that can both help mitigate overhead from resource advertisement and increase the applicability of the protocol. For the former, the simplest way to reduce the cost of advertisement is to reduce the frequency with which advertisements are sent. With respect to increasing the expressiveness of the approach, we expect to incorporate additional network and environmental context information into the adaptation functions, enabling better adaptive definitions of the advertisement radius.

Acknowledgement

The authors would like to thank the Center for Excellence in Distributed Global Environments for providing the research facilities and the collaborative environment. This research was funded, in part, by the National Science Foundation (NSF), Grant # CNS-0626777 and by the Defense Advanced Research Projects Agency (DARPA), Grant # HR0011-07-1-0024. The conclusions herein are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

References

[1] P. Engelstad, Y. Zheng, T. Jonvik, and D. V. Thanh, "Service discovery and name resolution architectures for on-demand manets," in *Proc. of the Int'l. Wkshp on Mobile and Wireless Networks*, 2003, pp. 736–742.

[2] U. Kozat and L. Tassiulas, "Service discovery in ad hoc networks: An overall perspective on architectural choices and network layer support issues," *Ad Hoc Mobile Networks*, vol. 2, pp. 23–44, 2002.

[3] C. Julien and M. Venkataraman, "Cross-layer discovery and routing in reconfigurable wireless networks," in *Proc. of MASS*, 2006.

[4] C. Frank and H. Karl, "Consistency challenges of service discovery in mobile ad hoc networks," in *Proc. of MSWiM*, 2004, pp. 105–114.

[5] I. Gruber, R. Schollmeier, and W. Kellerer, "Performance evaluation of the mobile peer-to-peer service," in *Proc. of CCGrid*, 2004, pp. 363–371.

[6] Z. J. Haas, "A new routing protocol for the reconfigurable wireless networks," in *Proc. of the* 6^{th} *Int'l. Conf. on Universal Personal Communication*, 1997, pp. 561–566.

[7] Z. J. Haas, M. R. Pearlman, and P. Samar, "The zone routing protocol (ZRP) for ad hoc networks," IETF MANET Working Group Internet Draft, July 2002.