A Dual-Hybrid Adaptive Routing Strategy for Wireless Ad-Hoc Networks

A. Bruce McDonald\textsuperscript{1,\dagger} and Taieb Znati\textsuperscript{1,\ddagger}

\textsuperscript{1}University of Pittsburgh, Telecommunications Program
\textsuperscript{\dagger}University of Pittsburgh, Department of Computer Science
\textsuperscript{\ddagger}Children’s Hospital of Pittsburgh, Department of Neurophysiology

\textbf{Abstract}—Recent debate in the research community has focused around the following question: should the design of routing algorithms for ad-hoc networks be predicated upon the minimization of routing overhead, or the optimization of network paths? The answer to this question depends upon the mobility and traffic characteristics present in the network. It is doubtful that any one approach by itself can be optimal, or even sufficient when there are temporal and spatial changes in the network dynamics. Most proposed solutions have adopted a fixed approach. To address this shortcoming we propose a dynamic cluster-based routing strategy that shifts the characteristics of its routing approach according to localized network dynamics. This is achieved by dividing routing into intra-cluster and inter-cluster components utilizing a strategy that is hybrid in terms of both its route acquisition and path computation approaches. By adapting the cluster organization to node mobility the strategy senses the network and balances optimality and overhead. The reason for doing this is to achieve the best possible performance subject to a wide range of operating environments.

\section{Introduction}

Nodes in ad-hoc networks are expected to cooperate in order to compute routes and forward traffic on behalf of other nodes [1]. This is a difficult task because node mobility and unpredictable variation in link quality lead to variation in resource availability and frequent changes to the topology. Unfortunately, traditional routing protocols designed for infrastructured networks generate too much overhead and cannot converge rapidly enough when conditions are highly dynamic or when the number of nodes is large. To address these shortcomings, new algorithms have been proposed that attempt to strike a more effective balance between the competing need for responsiveness and efficiency [2].

This paper presents a strategy for adaptively balancing the tradeoff between path optimality and routing overhead using a dynamic cluster algorithm supporting a routing strategy that is hybrid along two-dimensions. First, it combines table-driven and on-demand routing strategies—dynamically adjusting the relative contributions of each in time and space. Secondly, it combines optimal path computation with least overhead routing in a manner that adjusts to the dynamics and size of the network. Specifically, using a probabilistic model that characterizes the future availability of network links, the cluster algorithm adapts to mobility, thus, the characteristics of the cluster organization are dynamically adjusted to reflect localized network dynamics. Consequently, the balance between proactive and reactive routing, and the tradeoff between path optimality and routing overhead are dynamically adapted.

The remainder of this paper is organized as follows: In Section-II, the debate between path optimality and routing overhead is discussed. Subsection-II(B.) focuses on the operational domains for different routing approaches, and it is suggested that neither overhead minimization, nor path optimization techniques by themselves provide a complete solution to the ad-hoc routing problem. The section concludes with an outline of the design objectives for a unified adaptive strategy that can adjust itself to achieve different objectives according to spatial and temporal variation in the network dynamics. Section-III presents a distributed clustering algorithm and dynamic cluster-based routing strategy that have been designed to meet these objectives.

\section{Frameworks for Ad-Hoc Routing}

Routing algorithms designed for ad-hoc networks can be classified according to three basic properties: The first of these is the route acquisition policy which characterizes the manner in which routes are established—this policy determines when routes are acquired to specific destination nodes, and the set of conditions under which the algorithm responds to network dynamics. Two orthogonal policies have been adopted, namely, proactive routing, also referred to as \textit{table-driven} routing, and reactive routing, also referred to as \textit{on-demand} routing. A third approach referred to as \textit{hybrid} routing includes both proactive and reactive components. The second property classifies the \textit{path computation} algorithm used to build and select paths. Garcia-Luna-Aceves and Spohn characterize two commonly used approaches [3], namely, the \textit{optimal routing approach} (ORA) and the \textit{least-overhead routing approach} (LORA). ORA seeks to achieve optimality with respect to a given routing metric, while LORA attempts to minimize the routing overhead without guarantee of path optimality. The third property used to classify ad-hoc routing algorithms characterizes the \textit{type of routing information} that is required to compute paths. Distance-Vectors (DV) are used to advertise reachability and aggregate cost to selected destinations; whereas, Link-State-Updates (LSU) are used to advertise status information about a set of links. In addition to these properties, algorithms can be classified according to the organizational structure of the network as being either flat or hierarchical.

A three-dimensional model of the classification framework just described is depicted in Figure-1(a). The figure shows how various combinations of route acquisition policy, path computation algorithm and routing information characterize different strategies. The table in Figure-1(b) classifies several proposed ad-hoc routing algorithms according to this framework.

Path computation based on optimal route computation requires the complete search of the network state space. Consequently, proactive routing is well adapted to this approach because up-to-date state information is available a priori. However, proactive schemes tend to be less well adapted for least-overhead rout-
**Reactive routing** establishes routes to a limited set of destinations on a demand basis;

**Least-overhead routing** suffices to any loop-free route without guarantee of optimality. Dissemination of routing information and response to network dynamics are limited.

**Multi-path routing** establishes multiple paths between a given source and destination. No response is required until a topology change eliminates all paths to a given destination.

**Cluster-based routing** dynamically partitions the network into smaller subnets to reduce the amount of routing information that must be exchanged and processed.

Despite advantages, each of these strategies may exhibit poor performance under certain conditions. Pure reactive routing requires flooding in order to discover new paths, and the overhead due to localized reactions may become excessive when the session-length-to-mobility ratio is high. Hence, reactive routing alone may not be sufficient to support routing over a wide range of network dynamics. Least-overhead routing is an important concept that can reduce routing traffic in highly dynamic environments, or when the number of required destinations is large. However, while sub-optimal paths are generally tolerable, the quality of the paths may degrade over time. Hence, the selected paths must be of sufficiently high quality to ensure that the overhead incurred by routing over these paths does not exceed the overhead required to compute optimal paths.

Multi-path routing can work well in dense networks. However, without sufficiently disjoint paths the benefits may be minimal. Finally, cluster-based techniques are very useful for limiting the scope of routing algorithm response to node mobility. Although clustering can improve network scalability and reduce routing overhead, it introduces a location management problem: given a desired destination node how do you find which cluster it resides in at any point in time? An effective clustering solution must minimize the overhead due to cluster management and produce a relatively stable cluster topology.

### B. Operational Domains

For a given network scenario it is possible to make a case for a particular route acquisition policy and path computation approach. However, ad-hoc network conditions are prone to change in time and space. Thus, as illustrated in Figure-2(a), fixed approaches require disparate strategies for different environments and are limited in terms of their ability to adapt to changing network dynamics. The figure shows the operational domains in terms of the level of topology dynamics and the size of the network for each possible strategy. The figure is intended to visually demonstrates that no fixed strategy by itself can be expected to operate efficiently under all possible conditions.

Figure-2(b) illustrates the desired behavior of a hypothetical unified strategy that adapts itself to changing network dynamics. The idea is to implement a scheme that incorporates proactive and reactive route acquisition, and is able to combine both optimal and least-overhead path computation. Rather than attempting to force a fixed balance whose tradeoffs cannot compensate for changing dynamics, such a unified scheme must sense the state of the network and adjust the relative contributions of each element in dynamic fashion.
A cluster-based routing strategy is specified that adapts to node mobility in order to dynamically divide routing into two components: *intra-cluster routing* for maintaining routes between nodes within the same cluster, and *inter-cluster routing* for establishing routes between nodes in different clusters.

Optimal intra-cluster routes are maintained by a level-1 routing protocol. Any ad-hoc routing protocol capable of proactively maintaining optimal routes can be utilized [4, 5, 3]. Inter-cluster routes are acquired on-demand using a combination of a proactive level-2 routing that maintains *least-overhead* cluster-level routing, and a *dynamic binding protocol* that reactively locate the current cluster of a desired destination. The Source-Tree-Adaptive Routing Algorithm (STARA) [3] is an example of a protocol that has been designed specifically for proactive LORA based operation.

The resulting combination of optimal and least-overhead routing together with reactive location management represents a dual-hybrid strategy in which the path computation algorithm and route acquisition policies are de-coupled. This allows the mix of proactiveness and reactiveness to be balanced independently of mix of optimal and least-overhead routing. This is a unique feature that differentiates this strategy from other hybrid strategies such as ZRP [6].

### A. Distributed Dynamic Clustering

The Distributed Dynamic Cluster Algorithm (DDCA) for the \((\alpha, t)\)–Cluster presented in this paper builds upon the ideas first proposed in [7]. The idea is to dynamically partition the network into non-overlapping clusters of nodes consisting of one parent with zero or more children. The affiliation of a node with a cluster depends upon meeting a lower bound on the probability of path survival between that node and the parent node of the cluster [8, 9, 10].

Cluster formation in DDCA uses mobility-based criteria. In order to join a cluster a node must be able to reach the parent node of the cluster along a cluster internal path \(^1\) that is expected to survive for a period of time \(t\) with a probability of at least \(\alpha\). The scheme proposed in [11] also utilizes a mobility-based criteria, however, it builds clusters that consist of a set of nodes that are adjacent to a cluster-head. An \((\alpha, t)\)–Cluster is effectively a multiple-hop subnet in which the maximum number of hops between any pair of nodes in the same cluster varies dynamically depending on the mobility characteristics of the nodes. Hence, the size and structure of each \((\alpha, t)\)–Cluster is adapted continuously to the environment.

DDCA is an event-driven adaptive algorithm that is specified in terms of a finite set of states, events and actions. Every node maintains its current state and determines its current cluster affiliation in response to both hard-state and soft-state events. Each active node seeks feasible clusters to join upon initial activation, disconnection from its previous cluster, or the detection of a cluster partition. The state-transition diagram depicted in Figure 3 specifies the complete distributed algorithm as it is executed asynchronously by each node in the network.

The following definitions and properties are required in order to characterize the \((\alpha, t)\)–Cluster and understand the DDCA:

---

\(^1\)See Definition 3.
Definition 1 Path Availability: Let $P_{i,j}^k(t)$ indicate the state of path $k$ from node $i$ to node $j$ at time $t$. $P_{i,j}^k(t) = 1$ if all the links in the path are active at time $t$, and $P_{i,j}^k(t) = 0$ if one or more links in the path are inactive at time $t$. The availability of path $k$ at time $t$, $\Pi_{i,j}^k(t)$ is defined as follows:

$$\Pi_{i,j}^k(t) \equiv Pr(P_{i,j}^k(t) = 1)$$

Definition 2 $(\alpha, t)$-path: Let $P_{i,j}^k(\tau)$ indicate the state of path $k$ from node $i$ to node $j$ at time $\tau$, $\Pi_{i,j}^k(\tau + t)$ be its availability at time $\tau + t$. Path $k$ is defined as an $(\alpha, t)$-path iff the following two conditions hold:

$$P_{i,j}^k(\tau) = 1$$

$$\Pi_{i,j}^k(\tau + t) \geq \alpha$$

Definition 3 Internal Path: Let $k$ be a path from node $i$ to node $j$, and let $N^k$ be the set of nodes that lie along that path. Path $k$ is defined as an internal path with respect to a set of nodes $S$, if $N^k \subseteq S$.

Definition 4 $(\alpha, t)$-availability: Two nodes $i$ and $j$ are defined as being $(\alpha, t)$-available at time $\tau'$, if there exists an $(\alpha, t)$-path, $k$, between them at time $\tau \leq \tau'$ and there is at least one active path between them at all times during the interval $(\tau, \tau')$.

Property 1 Node Covering: Let $N$ be the set of all nodes in the network and $Q$ be the set of all the clusters in the network. The union of all the clusters $C_i \in Q$ must equal $N$—the set of clusters cover the network.

Property 2 Cluster Exclusivity: Let $Q$ be the set of all the clusters in the network. Then the intersection of any pair of clusters $C_i, C_j, i, j, in Q$ must be empty: $C_i \cap C_j = \emptyset$.

Property 3 Identifier Uniqueness: All nodes in a given cluster share a common cluster identifier (CID). The CID must be unique among all the clusters in the network.

Definition 5 $(\alpha, t)$-cluster: Let $p^2$ be a node and $S$ be a set of nodes: An $(\alpha, t)$–Cluster is defined as the set of nodes $C = SU_p$, such that: For each $n \in S$, and $p$ are $(\alpha, t)$-available along an internal path with respect to $C$, and $C$ adheres to Properties 1-3.

Cluster characterization requires a model which quantifies $(\alpha, t)$-path availability. Path availability is a random process, which depends upon the mobility of the nodes that lie along a given path. Consequently, the mobility characteristics of the nodes play an important role in clustering. Some models appear in the literature [8, 9, 10]; however, more work is required in this area.
DDCA differs from the \((\alpha, t)\)-Cluster algorithm [7] in two ways: first, DDCA does not require mutual path availability between every pair of nodes in the cluster. Secondly, once a node joins a cluster it remains in that cluster until it can no longer reach its parent along a cluster-internal path. The changes were implemented to improve cluster stability and reduce overhead.

The main idea of DDCA is to utilize path availability information maintained by the proactive intra-cluster routing algorithm in order for a node to determine if a cluster is feasible to join. Specifically, the path availability from an given node to the parent of a cluster is a measure of cluster strength. A node can join a cluster if and only if the cluster strength is at least \(\alpha\).

Each un-clustered node seeks a feasible cluster by broadcasting a Join-Request message. If it receives no responses it creates a new cluster in which it is alone—an orphan. To prevent adjacent un-clustered nodes from creating new clusters, simultaneous requests are handled by forcing nodes with higher identifiers to back-off and try again. A node that receives at least one Join-Response message joins the maximum strength cluster from which a response was received.

A node joins a cluster by changing its state, setting its cluster identifier (CID) and initiating a level-1 routing exchange with its neighbors. As a child, each node must process and respond to Join-Request messages and detect if it has become disconnected from the cluster, or if a cluster partition has occurred.

The parent of every cluster is initially an orphan. Each orphan node periodically attempts to join an adjacent cluster until it detects that at least one child has joined its cluster. This can be detected by the reception of routing information and the subsequent increase in size of the intra-cluster routing table. Each parent node must process and respond to Join-Request messages and detect if it has become disconnected from the its children.

### B. Routing Methodology

In this section we describe the dual-hybrid routing strategy supported by the DDCA cluster organization. Routing is achieved utilizing a dynamic two-level hierarchical strategy consisting of optimal and least-overhead table-driven algorithms operating at each level. The logical relationships between the clustering algorithm, the routing algorithm, and the other network-layer entities are depicted in Figure 4(a).

In contrast to our framework presented in [7], this scheme implements a proactive least-overhead level-2 routing protocol in combination with a dynamic binding protocol to achieve its hybrid characteristics. The cluster algorithm resides logically between the routing-layer and the dynamic binding protocol. As such, the cluster algorithm presents a logical topology to the routing algorithm, and it accepts feedback from the routing algorithm in order to adjust that logical topology and make clustering decisions.

Figure 4(b) illustrates the relationship between mobility, routing, and clustering. Increased mobility increases routing overhead. As such, the mobility information provided indirectly to the the clustering algorithm—through routing information—is used to adjust the cluster size and membership. Hence, adapting the clusters can reduce the routing overhead by adjusting the balance between proactive and reactive routing.

Figure 5 depicts an example of a two-level cluster-based organization and shows the data structures required for intra-cluster and inter-cluster routing. The figure shows the tables associated with the source node \(A\). The level-1 routing table consists of one entry for each destination node within the same cluster and one entry for each neighboring cluster. The \(Next\) column indicates the node ID of the node adjacent to node \(A\) that is its successor along the optimal path to the corresponding destination. The level-2 routing table consists of one entry for each cluster in the network. The \(Next\) column in the level-2 routing table specifies the next cluster ID along the current active path toward the corresponding destination cluster. This can be resolved to a successor node using the level-1 routing table.

The level-2 protocol requires that one node generate an update on behalf of its cluster. When a level-2 update is generated it must be flooded to all the nodes in each neighboring cluster. Level-2 updates are not transmitted beyond the neighboring clusters. The node with the lowest node ID in each cluster is designated to generate level-2 updates. A node is easily able to determine if it has the lowest ID. Latency may result in more than one update being generated. This does not affect the correct operation of the protocol. Based on this strategy every node maintains level-2 topology information.
Figure 5: Two-level Routing and Dynamic Binding Tables.

In order for a source node to forward packets to a desired destination it must use the dynamic binding protocol to discover the current cluster ID associated with the destination. Once determined, this information is maintained in the dynamic cluster-binding cache at the source node. The dynamic binding protocol utilizes knowledge of the level-2 topology to efficiently broadcast a binding request to all the clusters. This is achieved using reverse-path forwarding with respect to the source cluster.

The binding process is similar to a reactive route discovery process; however, a priori knowledge of the cluster topology makes it significantly more efficient and simpler to accomplish. This is because the level-2 information can be used to infer a broadcast tree that can be used to forward a request to every cluster only once. Hence, there is no need for a complex query control scheme [12]. In the case of transient level-2 loops due to latency in the processing of routing information, a time-to-live (TTL) field can be used to ensure the process will terminate in finite time. Furthermore, each request need only be processed by one node in each intermediate cluster, and if the target is discovered along a given subtree early termination of the query thread on that subtree is easily achieved. Finally, the request provides binding information directly to the target of the request. Consequently, the response can be sent directly to the source of the request via unicast routing. As shown in the figure, the cluster binding cache at node A indicates the current clusters associated with destinations Y and Z. Should a destination change its cluster while another node is caching its binding information a combination of traffic redirection [13] and proxy binding can be used to efficiently update the stale information.

IV. Conclusions

This paper presents an in-depth discussion of the fundamental questions as to whether ad-hoc routing algorithms should adopt optimal path or least-cost routing approaches. It is suggested that the answer to this question depends upon the specific operational environment and that no approach by itself is well adapted to operate when there are temporal and spatial changes in the network dynamics. Consequently, an outline is presented of the desirable characteristics of a unified strategy that is capable of sensing these shifts in the environment, and dynamically adjusting the balance of routing approaches that are used. The reason for doing this is to achieve the best possible performance subject to a wide range of operating environments.

A unified strategy is proposed whose design is based upon achieving the objectives stated above. The proposed scheme utilizes a dynamic clustering algorithm that is adaptive with respect to node mobility. The adaptive cluster organization supports a dual-hybrid routing approach that dynamically balances the tradeoff between proactive and reactive routing, while also dynamically trading off route optimality for routing overhead in a unique and novel way.

Future work is needed to assess the performance of this strategy. Particularly in comparison to other hybrid strategies subject to a range of realistic network topologies and mobility characteristics. Additionally, the strategy presented in this paper did not directly reflect traffic characteristics. Future work is required to understand the impact of different traffic patterns on this and other hybrid routing strategies.

References