UNICAST ROUTING TECHNIQUES FOR MOBILE AD-HOC NETWORKS

ROBERTO BERALDI, ROBERTO BALDONI

Dipartimento di Informatica e Sistemistica

Universita' di Roma "La Sapienza" Via Salaria 113, Roma, Italy

{beraldi, baldoni}@dis.uniroma1.it

TEL +39 06 49918481, FAX + 39 06 85300849

ABSTRACT

A Mobile Ad hoc Network (MANET) is an autonomous system of functionally equivalent mobile nodes, which must be able to communicate while moving, without any kind of wired infrastructure. At this end, mobile nodes must cooperate to provide the routing service. Routing in mobile environments is challenging due to the constraints existing on the resources (transmission bandwidth, CPU time and battery power) and the required ability for the protocol to effectively track topological changes.

This paper discusses the issue of routing in mobile ad-hoc networks by focusing on the main solutions proposed in the literature to cope with mobility. The paper surveys the techniques adopted for the case of unicast routing, i.e. when it is required to send a packet from a source node to a destination node, by illustrating how they are introduced into the main representative protocols.

1 Introduction

The term MANET (Mobile Ad hoc Network) refers to a set of wireless mobile nodes that can communicate and move at the same time. No fixed infrastructures are required to allow such communications, rather all nodes cooperate in the task of routing packets to destination nodes. This is required since each node of the network is able to communicate only with those nodes located within its transmission radius R, while a source node S and a destination node D of the MANET can be located at distance much higher then R. When S wants to send a packet to D, the packets have to cross many intermediate nodes and for this reason MANETs belong to the class of the multi-hop wireless networks.

This paper considers the problem of *unicast routing*, responsible of routing packets from a single source node to a single destination node, and illustrates the main techniques adopted in designing routing protocols for such a wireless multi-hop environment. Let us first briefly review how the routing problem can be formulated at an

abstract setting. A network is usually modeled as a graph G=<V,E>, where V={1,…,n} is the set of nodes (the routers) of the graph and E $\subseteq$ V x V is the set of edges (the links) [1]. See FIG. 1.

When used to model MANETs, elements belonging to the vertex set V of G correspond to mobile stations (assumed fixed in number) and edges belonging to set of edge E correspond to wireless links in the network. A wireless link among two nodes *i* and *j* is established when, roughly, the physical distance PD(*i*,*j*) is less or equal to the transmission radius R (a typical value for R is 250 m). Due to node mobility, new wireless links are continuously created and the existing one deleted, leading to a so-called random *unit* graph.

A change in the set E of the graph is called a topological change and is driven by node mobility. Topological changes also occur in wired networks, but in this case they are a consequence of a link fault rather than the normal operation mode of the network.

A path P(S,D) from S to D is specified as a sequence of nodes P(S,D)≡<N_0,N_1,.., N_K>, where; N_0=S,N_K=D, $N_i \neq N_j$, (N_i ,N_{i+1}) $\in$ E (for i ≠ j , 0≤i≤k-1). The length of the path, |P(S,D)|, is the number of links (hops) a packet performs before reaching the destination node, thus |<N_0,N_1,..,N_K>|=K. The term route is also used as a synonymous of path. If the destination is composed of a set of nodes, multicast routing is required. This issue is out of the scope of this paper and will not further be considered.

The task of routing can be divided in two sub-tasks: (1) the computation of P(S,D) and (2) data packet forwarding along the route. Paths are calculated by a distributed algorithm running at network level which includes some form of path maintenance to deal with topological changes. Maintenance can provided by "sampling" the current network topology with periodic route updates and, optionally, sending updates upon a change in network topology (event-triggered update). In this way, the algorithm should track topological changes and consistently update paths. Also, in the very likely case many paths exist between S and D, the routing protocol has to calculate the best path P^*, with respect to some metric. For example, a widely used metric it the path length; in this case P^* is the shortest-path, i.e. $P^*(S,D) = \min_P \{|P(S,D)|\}$.

A node $N_i \neq D$ that receives a packet addressed to D, retransmits the packet on the link connected to the next-hop node of P(S,D), N_{i+1}. There are two main techniques to realize this forwarding function. In the first one each node has access to a local *routing table* (RT), with entries for all nodes in the network, indicating the next-hop node along the path. The next-hop node does not depend on the source node S. As an alternative, each packet

stores the full path into its header, so that the next-hop node - N_{i+1} - is learned from the packet itself. Since the path is stored in the packet by the source, this technique is called *source routing*.

Several important structural differences exit between wired and wireless networks that make routing very different in the two environments [4,5,6,7]:

(a) in a MANET the rate of topological changes is very high compared to wired networks. A conventional routing protocol should continuously be forced to send and receive topology updates to maintain routing tables up-to-date. In an event-triggered Link State protocol, for example, any topological change would trigger a flooding, resulting in a flooding rate equal to the topological change rate. In this scenario nodes cannot easily preserve their own battery power by switching to “stand-by” or “sleep” operation mode. Moreover, the protocol could not react fast enough to changes. On the other hand, consistent and fast route table updates is mandatory. The possible effects of delay or time skew in table updates are:

- a packet being routed through a non-optimal path (an increase in the end-to-end packet delay)
- a packet loss, due to a temporary inconsistency in routing tables (loops or a broken link along the path)

(b) links may be unidirectional: a node *i* can receive the signal (and thus a packet) from a node *j*, but the contrary can be not true (in this case G is a direct graph). As a consequence, a transmission that requires an handshake between *i* and *j* fails, while best effort transmissions - like broadcast, succeed

(c) several technological limitations on the use of the resources exist, namely battery power, transmission bandwidth and CPU time. The main two consequence of such limitations are: (1) a blind route update mechanism, both periodic or event-triggered, can waste resources, since updates are sent even when no data transmission at all occur in the network; (2) the definition of some suitable metrics to compare different paths that capture these costs. For example, a selection criteria based only on the path length could not be suitable: the shortest path can have a shorter "live" compared to a longer one and thus trigger more repairing activities. The remaining battery lifetime of the nodes of a path can also be an important factor to include in the metric: paths composed of nodes with a high value for the reaming battery lifetime should be preferred

to the ones with a lower value. On the other hand, fairness is also a concern. All nodes should equally contribute to the packet routing effort.

(d) Security is challenging in the MANET environment [8-10]. In the absence of any authentication mechanism, due to the nature of radio transmissions, a malicious node can easily corrupt route tables, caches, and other similar information, claiming false route information

When the topological changes are extremely high, little can be done to assure that routing algorithms converge fast enough to track topological changes, and routing can only be achieved by flooding [2,3]. In the other cases the design routing protocols is challenging.

This paper is organized as following. Section 2 provides a brief review of conventional routing protocols. Section 3 describes the main techniques adopted in routing protocols for MANETs; Section 4 discusses the issue of protocol performance. Finally, conclusions are given in Section 5.

2 A brief review of traditional routing protocols for wired networks

Conventional routing protocols are based on routing tables which store paths to all possible destinations. To guarantee that routing tables are up-to-date and reflect the actual network topology, nodes continuously exchange route updates and recalculate the paths.

Such protocols are divided into two "complementary" classes [1]:

- Distance Vector (DV) algorithms, a node sends to its neighbors the whole routing table (its distance vector)
- Link State (LS) algorithms, a node sends to all the other ones the state of the link with the current neighbors via a reliable flooding.

Route updates are sent periodically or when a topological change is detected. Path calculation in Distance Vector algorithms is based on a distributed version of the classical Bellman-Ford algorithm (DBF). Every node *i* maintains a set of distances for each destination *j* and for each neighbor node *k*. Let the distance be denoted as H^k_j. The next-hop node of the path from *i* to *j* is the node k^* such that $H^{k^*}_j=\min_k(H^k_j)$. Ties are broken arbitrarily. The succession of the next-hop nodes chosen in this manner leads to the shortest path from *i* to *j*.

A node running a distance vector protocol does not know the network topology, i.e. it does not store the graph G. A well known example of DV algorithm is RIP (Routing Information Protocol) [16].

In LS algorithms each node stores the whole network topology, and calculate the best path autonomously from the others. Information about the topology is advertised in the form of link-state. The link state indicate whenever the links with the neighbors are up or down and the cost associated to it.

Due to flooding, each node eventually receives all the link-states from all the other nodes of the network. A node can then build the graph G of the network and can calculate the path from itself to any other node, usually according to the Dijkstra's shortest path algorithm. A well known example of LS algorithm is OSPF (Open Shortest Path First) [17]. The following simple result guarantee that routes calculated independently by nodes are consistent.

LEMMA: Sub-paths of a shortest paths are also shortest path [1].

Routing information in the routing table of the nodes are stored in a distributed manner according to a "next-hop" fashion, so that globally they provide a shortest path spanning tree routed at the destination, see FIG 2.

DV protocols may badly react to a topology change since it suffers from very low convergence (count-to-infinity problem) and may create temporary loops. On the contrary LS converges faster then DV but it requires a higher overhead [1].

3 Unicast routing protocols for MANETs

A first attempt to cope with mobility is to use specific techniques aiming at tailoring the conventional protocols to the mobile environment, while preserving their nature. For this reason the protocols designed around such techniques are referred to as *table-driven* or *proactive* protocols. The term proactive refer to the ability of the protocol to calculate all possible paths independently of their effective use. Proposals for proactive protocols include: pure DV protocols, pure LS protocols, mixed LS and DV protocols.

A different approach in design a protocol is to calculate a path when strictly needed for data transmission. If data traffic is not generated by nodes, then the routing activity is totally absent. Protocols of this family are dubbed *reactive* protocols or *on-demand* routing protocols. Reactive protocols are characterized by the elimination of the conventional routing tables at nodes and consequently the need of their updates to track changes in the network topology. The computation of a path boils down to a discovery procedure based on flooding, used as

last resort when previous discovered routes are no longer valid. Proactive and reactive approaches are merged in *hybrid* protocols [11]. Also, reactive features can be added in the proactive framework [12].

Hierarchical routing protocols, that superimpose a logical topology on top of the physical one, is another approach proposed [2]. The hierarchy is a new dimension in the protocol design space, with solutions both in the proactive and reactive realms [13,14]. Finally, the knowledge of the location can help the task of routing. In the location-based routing protocol, the node location is the only criteria adopted in routing packets. For a survey on the position-based routing techniques the reader may refer to [15].

3.1 Proactive protocols

The advantage of these protocols is that a path to the destination is immediately available, so that no delay is experienced when an application needs to send packets. In some case this can be useful, as for interactive applications. The main mechanism adopted in proactive protocols are the following:

- to increase the amount of topology information stored at each node (to avoid loops and speed up protocol convergence)
- to vary dynamically the size of route updates and/or the update frequency
- to optimize flooding
- to combine DV and LS features

3.1.1 Tailoring Distance-Vector Protocols: DSDV

The Destination Sequenced Distance Vector (DSDV) protocol is one of the first attempt to customize the classical DBF protocol [18]. The key elements of DSDV are: (a) an aging mechanism based on monotonically increasing sequence numbers, that indicates the freshness of the route and which is used to avoid routing loops and the count-to-infinity problem; (b) the use full route updates, sent periodically every update interval, or incremental route update sent on topological changes; (c) the delay of route updates for routes that are likely to be unstable, i.e. those for which a new update is on the way towards a node.

Entries in the classical routing table are enriched with a sequence number. An odd number indicates a distance equal to infinite and is used for those destinations that become unreachable, while even numbers are used by the destination to stamp route updates.

For each route update for a destination *j*, the corresponding entry in the routing table is updated with the new received information if the route has a more recent sequence number or if the route has the same sequence number but a better metric. A route entry is deleted from the table if no update has been received for a given number of update intervals.

In DSDV, routes with a metric of ∝ are advertised without a delay, while the ones can be delayed according to an average *settling time*. By doing so, it is possible to prevent the advertisement of unstable routes, i.e. those routes that will very likely be replaced by a better route update received in the near future.

This case can happen since updates for the same destination can arrive to a node in any order, for example because received from different neighbors.

To calculate the average settling time, a node uses a table which stores, keyed by the first field:

- the destination address
- last settling time
- average settling time

Route updates is sent periodically (every an update interval of ΔT sec.) and incrementally as topological changes.

3.1.2 Tailoring Link-State Algorithm: OLSR

The Optimized Link State Routing (OLSR) protocol is an optimization over the classical link state protocol, tailored for mobile ad hoc networks [19,20]. The key idea of OLSR is the use of *multipoint relay* (MPR) nodes to flood the network in an efficient way by reducing duplicates packets in the same region. The protocol also selects bi-directional links for the purpose of routing, so that the problem of packet transfer over unidirectional links is avoided. Each node *i* selects, independently from the other nodes, a minimal (or near minimal) set of multipoint relay nodes, denoted as MPR(*i*), from among its one hop neighbors. The nodes in MPR(*i*) have the following property: every node in the symmetric 2-hops neighborhood of *i* must have a symmetric link towards MPR(*i*). In other words, the union of the one hop neighbor set of MPR(*i*) contains the whole 2-hops neighbor set. See FIG. 3. The MPR sets permit to realize flooding efficiently: when a node *i* wants to flood a message, it sends the message only to the nodes in MPR(*i*), which in turn send the message to their MPR nodes and so on.

The *Multipoint Relay Selector* set (MPR selector set) of a node *j* is composed of the set of neighbors that have selected it as MPR. Each node periodically floods its MPR Selector Set, using the flooding technique described above, and a special type of control message called Topology Control (TC) message. Using TC messages a node

announces to the network that it has reachablity to the nodes of its MPR selector set (it is its last-hop node). A TC-message is stamped with a sequence number, incremented when the MPR Selector set changes.

To increase the reaction to topology changes while limiting the protocol overhead, the time interval between two consecutive TC message transmissions can be decreased up to a minimum, if a change in the MPR Selector is detected. Also, if a node has an empty MPR selection set, it may not generate any TC message. However, when its MPR selection set becomes empty is should still send empty TC-messages for a period of time in order to invalidate the previous TC-messages. Information gained from TC messages are used to build the network topology and then the routing tables.

A node running OLSR thus adopts the following three data structures:

- neighbor sensing information base
- topology information base
- routing table

3.1.3 Merging Distance Vector and Link-State behaviors: FSR

The Fisheye State Protocol (FSR) is a proactive protocol based on the so-called "fisheye technique", proposed by to reduce the size of information required to represent graphical data [21].

The novelties in FSR are: (a) the transmission of link-state packets to the neighbors instead of by flooding (a method borrowed from the Global State Routing protocol, GSR [22]; (b) the introduction of the notion of scope to define region of the networks with different accuracy in routing information.

FSR is similar to link-state protocols as each node maintain a topology table of the network. A node also maintains a route table and a neighbor list.

Differently from link-state protocols, which flood route update to the network, in FSR link state packets are exchanged with neighbors only, while sequence numbers are used to indicate the freshness of the information, as in DSDV. A route update message includes of a sequence of <destination address, neighbors list> pairs. To realize the fisheye technique, FSR introduces the notion of scope. The scope at a node *i* is defined as the set of nodes that can be reached within *h* hops from *i*. See FIG 4.

Routing updates are generated at difference rates, with the higher frequency for nodes with the smaller scope. This produces a reduction in the number and size of update messages and thus in the protocol overhead. Basically, FRS is based on a route optimality - route cost tradeoff. It maintains distance and quality information about neighborhood of a node with progressively less details as the distance increases. Such an imprecise

knowledge of the best path to the destination can lead to use some non-optimal route; however the protocol has good potential to scale to large networks, since topological changes occurring in a regions located far away from a node do not produce the same amount of traffic as a single-scope proactive protocol. Moreover, when the packet gets closer to the destination, it reaches scopes with increasingly precise knowledge about the destination, and this mitigates the lack of precise route information. It is evident how the scope is a new dimension in the protocol design space.

3.1.4 Path-finding algorithms: WRP

The Wireless Routing Protocol (WRP) belongs to the general class of the Path-Finding Algorithms (PFA) [23-25], defined as the set of distributed shortest-path algorithms that calulate the paths using information regarding the length and second-to-last hop (predecessor) of the shortest path to each destination. PFA algorithms eliminate the count-to-infinity problem of DBF. WRP also addresses the problem of avoiding short term loops, that can be still present in such algorithms for the paths specified by the predecessor node [24]. The protocol uses the following four data structures:

- Distance table
- Routing table
- Link-cost table
- Message Retransmission List (MRL)

The distance table of node *i* is a matrix containing, for each destination *j* and each neighbor of *i* (say *k*), the distance to *j* and the predecessor node reported by *k*.

The Routing table of a node *i* is enriched with the predecessor and the successor of the path to the destination node, as well as a tag that specifies whether the entry corresponds to a simple path, a loop or to a destination that has not been tagged. With these information at hand, WRP is able to avoid the count-to-infinity problem and to highly reduce routing loops by forcing each node to perform consistency checks.

MRL is used to store the identification of those nodes that have not acknowledged a route update. After a timeout, route updates are retransmitted to them. Also, WRP relies on the transmission of HELLO messages to detect the connectivity with neighbors in case no route update is received within a suitable time interval.

3.2 Reactive protocols

A different approach for routing in mobile environments is routing on-demand. The approach is characterized by the elimination of the conventional routing tables at nodes and consequently the need of their updates to track changes in the network topology.

On-demand routing protocols calculate a path before data transmission. If data traffic is not generated by nodes, then the routing activity is totally absent. For such a reason they are also called *reactive protocols*.

A reactive protocol is characterized by the following procedures, used to manage paths:

- path discovery
- path maintenance
- path deletion (optional)

Data forwarding is accomplished according to two main techniques:

- source routing
- hop-by-hop

The discovery procedure is based upon a query-reply cycle that adopts flooding of queries. The destination is eventually reached by the query and at least a reply is generate. Path discovery is triggered asynchronously on-demand when there is a need for the transmission of a data packet and no path to the destination is known. Route discovery is not required for the transmission of every single data packet, since the discovered path is likly to be valid for a period of time that allow many successive transmissions to the same destination.

As a result of a path discovery, the network nodes acquire a new “routing state” which stores the paths learned during the discovery. Routing information are maintained by a maintenance procedure either until no longer used or explicitly deleted.

In the following, some representative routing protocols characterized by different maintenance procedures and the use of different techniques to record the routing state are described: route caches, temporary routing tables, logical structures.

3.2.1 DSR

The distinguish features of the Dynamic Source Routing protocol (DSR) are: (a) packet forwarding via source routing; (b) aggressive use of route cache that store full paths to destinations [26].

Source routing presents the following advantages:

- allows packet routing to be trivially loop-free
- avoids the need for up-to-date routing information in the intermediate nodes through which packets are forwarded
- allows nodes to cache route information by overhearing data packets

The DSR protocol is composed of two main mechanisms, Route Discovery and Route Maintenance. Route Discovery adopts route request (RREQ) - route reply (RREP) control packets and is triggered by a node S which attempts to send a packet to a destination node D and does not have a path into its cache. Discovery is based on flooding the network with a RREQ packet, which includes the following fields: the sender address; the target address; a unique number to identify the request; a route record [27].

On receiving a RREQ control packet, an intermediate node can:

- reply to S with a RREP, if a path to the destination is stored into its cache (in this case the returned path is the concatenation of the path accumulated from S to the node and the path in the cache from the node to D)
- discard the packet, if already received
- append its own id into the route record and broadcast the packet to its neighbors, in the other cases

On receiving a RREQ packet the destination replies to S with a RREQ packet. An RREQ packet is routed through the path obtained by reversing the route stored in the RREQ's route record (the links are assumed bi-directional) and containing the accumulated path. See FIGG 5 and 6. An extension of DSR, called RODA, that supports asymmetric links has been also proposed. In this case the destination triggers another route discovery that piggybacks the accumulated route in the RREQ packet [28].

The base mechanism for route maintenance is as follows. When an intermediate node detects that the link to its next-hop node toward the destination is broken, it removes this link from its route cache and returns a Route Error control message to S. The source S then triggers a new route discovery.

Several improvements to this base mechanism have been proposed; they include: (a) promiscuous mode operation - a node can run its radio interface in promiscuous mode to listen control packets not addressed to it to gain information useful for cache management; (b) salvaging - instead of sending an error route message back to the source, an intermediate node can use a path to the destination from its own cache when the link to the next hop is broken; (c) random delays - when sending RREP to avoid a reply storm [29].

Cache management is quite critical in DSR to assure good performance. Especially under high load conditions, if stale routes are not immediately removed from the cache, they can be used in the reply to other discovery requests and quickly be cached by other nodes. Several schemes have been proposed [30,31]:

- fixed lifetime
- adaptive lifetime
- negative cache
- wider error notification

The first case is straightforward: each entry receives the same constant value for its lifetime. However, care must be taken when selecting the lifetime value, since a wrong value can result in performance even worst than a DSR with no cache at all. Adaptive lifetime estimates the value of lifetime for each new entry through a heuristic based on significant events observed (for example, error-route notification, link breakage, etc.).

In the negative cache scheme a node caches the broken links seen recently via link layer feedback or error route packets. This information disables the writing of any new cache entry which includes reference to those broken links for a given time interval.

The wider notification scheme broadcasts error routes notification to all the source node (i.e. the ones that forwarded packets along the broken route) in a tree fashion starting from the point of failure. The mechanism is able to efficiently remove cached routes also from neighbors of a source node.

Simulation results showed that the adaptive lifetime scheme can achieve the same performance of a well tuned fixed lifetime. However, performance results with a combination of adaptive lifetime and active deletion with wider notification are superior to the ones obtained using deletion schemes separately.

3.2.2 AODV

The Ad hoc On Demand Distance Vector Routing (AODV) borrows the use of sequence number from DSDV to supersede stale cached routes and to prevent loops, while the discovery procedure is derived from the one adopted in DSR [32,33]. The main difference with DSR is that a discovered route is stored locally at nodes, rather than be included in the packet's header.

Route discovery process is triggered by a node S when it needs to send a packet to a node D for which it has no routing information into its routing table. Route discovery is based on flooding a RREQ packet similarly to DSR.

As a node forwards the packet request, it sets up a reverse path from itself to S by recording the address of the neighbor from which it received the first copy of the RREQ. Similarly, when a RREQ control packet is forwarded towards the destination, a node automatically set up the reverse path from all nodes back to the source. The other reverse paths are deleted after a timeout period. See FIGG. 7 and 8.

At the end of the discovery phase, as a result of the request-reply packet transmissions, a new routing state is created at nodes. The state is composed of the routing table entries that record the next-hop node to the destination of the active path. A forward path is deleted if not used within a given route expiration time interval. Each time the route is used, the expiration time interval is reset.

Route maintenance is based on the periodic transmission of HELLO messages. Upon detecting a link-failure, a node sends an Unsolicited Route Reply packet to all its active upstream neighbors invalidating all the routes using the broken link. Those nodes, in turn, relay the packets to their respective upstream nodes so that eventually all active source are notified. After receiving the Unsolicited Route Reply the source issues another route request.

3.2.3 ADV

The Adaptive Distance Vector (ADV) is a distance vector protocol that maintains routes for *active receiver* advertising route updates just like any distance vector algorithm [12]. In other words, the distance vector algorithm is dormant unless activated for particular nodes. Authors classify ADV as a distance-vector protocol that exhibits some on-demand characteristics.

In ADV a node is tagged as active receiver if it is the destination of any currently active connection. Before any data transmission can take place the source node notifies, through an *init-connection* control packet flooded to all other nodes in the network, that it needs to open a connection with the destination node.

Similarly, when a connection is closed, the source node broadcasts network-wide an *end-connection* control packet. The destination, in turn, if has no further active connection, broadcasts network-wide a *non-receive-alert control packet*. As a result of this mechanism, all nodes in the network are able to tag routing table entries with a receiver flag to indicate whenever a destination is an active receiver, i.e. the destination node becomes an active receiver.

ADV also reduces the routing overhead by varying the frequency and the size of routing updates in response to traffic and node mobility.

3.2.4 TORA

The Temporally-Ordered Routing Algorithm (TORA) belongs to a general family of "link reversal" algorithms [3,34]. TORA is designed to react efficiently to topological changes and to deal with network partitions. The name of the protocol is due to the assumption of having synchronized clocks (for example via GPS), required in ordering events occurring in the network

TORA provides routing by exploiting a completely different approach compared to the ones described so far. Route optimality is a secondary concern in TORA. The main goal is to find stable routes that can be quickly and locally repaired. The protocol builds a DAG routed at the desired destination for this purpose. The DAG is obtained by assigning a logical direction to the links, on the basis of a "height" or reference level assigned to nodes. If (*i*,*j*) is a direct link of the DAG, *i* is called the upstream node and *j* the downstream node. The DAG has the following property: there is only one sink node (the destination), while all other nodes have at least one, but usually many, outgoing links. See FIG 9. The destination node can be reached from a node following any of its outgoing link. Loops are trivially avoided due to the property of the DAG.

Protocol functioning can be divided into three separated phases: route discovery, route maintenance and route deletion.

The first phase relies on an exchange of short query-reply control packets. See FIG. 10. During the transmission ot the reply messages, which is also done by flooding, links receive a logical direction (upstream or downstream) based on their logical relative height, so that a DAG, routed at the destination, is created at the end of the phase. The graph is referred to as “destination oriented” graph. See FIG 9.

This routing state can be viewed as network of tubes, with water flowing downhill towards the destination node, that has the lowest height in the network.

Route maintenance is activated to maintain the DAG and is based on a finite sequence of "link reversal" operations. A key feature of TORA is that many topological changes may trigger no reaction at all. If fact, if one of the outgoing link of a node breaks, but the node has at least other downstream node, the destination can still be reachable through another path, and thus no repairing activities are required. See FIG. 11

On the contrary, when a node detects that it has no downstream nodes, it generates a new reference level in order to become a global maximum. The new reference level is propagated into the network, causing partial link reversal for those nodes which, as a result of the new reference level, have lost all routes to the destination. At the end of the repairing activities, localized near the node, the DAG is re-established. See FIG. 12.

TORA is able to detect network partitions. On detecting a network partition, a node floods a clear packet that resets the routing state.

3.2.5 ABR

The Associativity Based Routing protocol (ABR) is an on-demand protocol carefully designed to work in mobile environments [35]. The key idea is the use of the longevity of routes as the main selection criteria instated of the route length. In ABR a longer-lived route is preferred to a shorter-lived one, even if the length of the former is lesser than the latter. By doing so, the protocol uses the most stable routes, i.e. the ones that are likely to requires the least maintenance activities. A similar approach has also be proposed in the Signal Stability Adaptive protocol (SSA) [36]. The estimation of a route lifetime includes the combination of several straightforward measurements (remaining power lifetime, signal strength,…) and a new metric, the so-called "associativity" between nodes, which captures the degree of stability of one node with respect to another node over time and space.

The technique proposed to capture associativity is as follows. Each node generates a periodic beacon and counts the beacons received from its neighbors to update their "associativity ticks". The associativity ticks are reset if the beacon signal is not received for a suitable period of time.

The value of the associativity ticks allows to classify a node as having a high or a low mobility state with its neighbors. A high value of a node *i* with respect to a node *j* indicates, in fact, that the node *i* was able to receive many consecutive beacons from *j*. As such, the protocol assumems that it is very likely for the two nodes to remain close each other. The node then exhibits a low mobility state with respect to *j* and the link from *i* to *j* is classified as long-lived. On the contrary, a low value for the associativity ticks indicates a transiting neighbor and thus a high mobility state.

The protocol consists of three phases: route discovery phase, route reconstruction phase and the route deletion phase. As for other on-demand protocols route discovery is base on flooding. Intermediate nodes are not allowed to reply to a request. The path is selected by the destination node and is stored as active path at intermediate nodes in a hop-by-hop fashion.

A source node acquires a new route to a destination node by broadcasting a Broadcast Query (BQ) control packet. As a packet is propagated, intermediate nodes append their own id and route quality value, that includes the associativity ticks.

The destination node waits a suitable period of time after receiving the first BQ packet so that it can receive other request packets forwarded along other paths. In this way the node can select the best path according to the

following criteria. If a route is composed of nodes with a high value for the associativity ticks then the route is preferred to a path with a shorter number of hops. The number of hops is considered only to select among two routes with the same overall associativity degree.

Once a route has been selected, the destination sends a BQ reply control packet (BQ-reply) to the source along the selected route. As the BQ-reply is propagated backwards to the source, intermediate nodes involved into its retransmission are able to setup a route entry thus activating a forward path, as already described for AODV.

The route reconstruction phase (RRC) is invoked when the association stability relationship is violated. In some cases, the procedure can try to repair the corrupted sub-paths, triggering a new sub-path discovery, without notify the break to the source. In the worst case, a Route Notification (RN) control message is sent to the source and a new discovery phase is initiated. The final phase of ABR is the route deletion phase. This phase consists of flooding the network with a Route Deletion (RD) control packet. As an alternative to such costly deletion procedure, ABR also proposes a less expensive one, called the soft state approach, in which the route entries are deleted upon time out.

3.3 Hybrid Protocols: ZRP

ZRP is a hybrid routing protocol aims to combine the advantages of both the proactive and reactive approaches, i.e. mainly to reduce the latency necessary to acquire a new route and the protocol overhead [37]. Central for such a protocol is the notion of *zone.* A zone Z(k,n) for a node n with radius k, is defined as the set of nodes at a distance not greater than k hops

$$Z(k,n) = \{i \mid H(n,i) \leq k\},$$

where H(i,j) is the distance in number of hops between node i and node j. The node n is called the *central* node of the routing zone, while node b such that H(n,b)=k is called the *peripheral* node of n. See FIG 13.

The value for k is usually small compared to the network diameter can be optimized under different scenarios, characterized by various mobility and traffic degrees. The protocol's architecture is organized into four main components. The IntrAzone Routing Protocol (IARP); the IntErzone Routing Protocol (IERP); the Bordercast protocol (BRP) and a layer-2 Neighbor Discovery/Maintenance Protocol (NDP) [38,39].

The IARP provides routes proactively to those nodes located inside the source's routing zone. It can be based upon any proactive protocol with the difference that route updates are propagated to a distance no greater that *k* hops. IARP uses NDP to learn about node's neighbors.

For those nodes located at a distance $k'>k$ from the source, ZRP relies on the IERP to calculate on-demand an interzone path. The IERP uses a form of selective flooding to exploit the underlying zone structure generated by the IARP. Specifically, flooding is based on sending query packets only to the peripheral nodes (also called border nodes), using a special kind of multicast transmission, dubbed *bordercast* (see FIG 14). When a node receives the query packet, it can either reply to the source – if D is a member of its routing zone - or bordercast the query packet to its peripheral nodes. Eventually the query packet reaches a node having D as member of its zone, so that a reply control packet is generated and sent back to the source. A route to D can be accumulated in the query packet during forwarding (as for DSR) or – to reduce the query packet length – in the reply control packet during the reply phase.

Packet forwarding along an interzone path adopts a modified source routing. A routing path only contains the border nodes that have to be traversed. Forwarding along border nodes is table driven, since the distance between border nodes the nodes is *k*.

Since there is no coordination among nodes, zones heavily overlap. A node can be member as well as border node of many zones. In this way, the basic search mechanism can perform even worst than a standard flooding. Authors provide several solutions to deal with this problem by stopping and controlling redundant query threads.

Route maintenance is responsible for maintaining interzone paths. The use of a local repair procedure, aiming at repairing the broken link by a mini-path search - is also suggested to reduce the need of global route discovery.

An improvement over ZRP – the Distributed Dynamic Routing algorithm (DDR) - has also been proposed [40]. The protocol is based on the construction of a forest, of non-overlapping dynamic zones.

3.4 Position aided protocols: LAR

The novelty in the Location Aided Routing protocol (LAR) is the estimation of the position of the destination node, used to increase the efficiency of the discovery procedure [41,42]. Only a subset of nodes are queried during the discovery phase, specifically the ones in the so-called "request zone" nearby the estimate. LAR uses standard flooding as last resort when no estimations are available. The position can be obtained, for example, from the Global Position System (GPS).

The request zone is calculated from the "expected zone", which is defined as the zone where the destination node D should be located from the viewpoint of the source node S at the current time t_1, given that S known its position at some time t_0 in the past and its average velocity.

The request zone is defined as the smallest rectangle that includes the current location of S and the expected zone. See FIG 15. In the LAR scheme 1 the request zone (i.e. its four corners) is included into the route discovery packet. A node forwards a route discovery packet only if it belongs to the request zone and it is receiving the packet for the first time.

The destination D replies with a packet that contains its current position, the local time and – optionally – its current speed or its average value over some time interval.

In the LAR scheme 2, the discovery packet contains (a) the distance *Dist* from the node that is sending the packet to the destination and (b) the position (x,y) of the destination. Let D_i be the distance from a node *i* to the destination. When the source node sends the packet it sets $Dist = D_s$. When a node *i* forwards the packet it sets $Dist = D_i$ before the retransmission.

A node *j* that receives a request packet from a node *i* forwards the packet only if it was received for the first time and

$$Dist + \delta > D_j$$

i.e. if *j* is "at most δ farther" from (x,y) than node *i*. The value δ can be set to a value greater then zero to take into account location errors or to trade-off the probability of finding a route on the first attempt with the cost of finding the route. Simulations results showed as using location information results in significant lower routing overhead, as compared to algorithms that do not use the location information.

4 Protocol performance analysis

A protocol performance analysis aims to measure two complementary aspects. The first one is how efficiently the routing service is provided wrt a give set of cost indexes e.g. bandwidth, battery energy consumption, CPU cycles; the second aspect concerns application oriented metrics, i.e. the performance seen externally by an application in using the protocol.

To evaluate these two aspects, protocol complexity analysis and protocol simulation have been used. The aim of the complexity analysis is to measure - at a simplified and abstract setting - the resources required the protocol in performing an single "protocol operation", for example the reaction to a link breakage. More specifically:

(a) computation complexity is the number of computation steps executed by a node to perform of a protocol operation

(b) space complexity is the memory space required by a node to store routing information

(c) communication complexity is the number of packets exchanged among nodes in performing a single protocol operation.

(d) The values are obtained as function of some network parameter (i.e. the number of nodes and/or the diameter of the network, etc.) and usually assumes the worst case protocol's behavior and synchronous execution (i.e. all nodes are able to execute single computation steps at fixed points in time) [3,24].

Complexity analysis is not able to capture all the protocol performance issues, such as the measurement of other relevant application-oriented metrics and their relationships with many important factors - like the mobility pattern, communication efficiency and offered load. Indeed many proposals in the literature do not face it.
Such a study is normally carried-out through protocol simulations, as realistic analytical models are infeasible due to the high number of factors to be included. very diffult to define, unless many details are omitted. Protocol simulations aim at estimate statistics for several metrics that include [43]:

(a) throughput (number of bytes delivered per second)

(b) packet delivery ratio (packets delivered / packets generated)

(c) end-to-end data packet delay (time interval from when a packet is ready for the transmission to its delivery)

(d) routing overhead (number of control bits transmitted/data bit delivered; or control packets / data packets delivered)

(e) route discovery time (time required to compute a new route)

(f) route optimality (cost of the path used – cost of the optimal one)

(g) number of out-of-order packets

(h) power consumption (energy consumed per delivered bit or packet)

The first three metrics are the ones relevant for an application that adopts connectionless transmissions. The others provide insights on the efficiency of the routing service. It is important to note that when measured at TCP

level, the throughput is usually very poor - despite its excellent values measured for connectionless traffic. This is due to the frequent temporary link failure and route changes whose effect it to trigger many false congestion detection. The design of suitable TCP protocols for MANETs is an active research topic [44-47].

There are two main factors that influence the above metrics: the rate of topological changes (also called *mobility*) and the rate of packet transmission (*offered load*). By scanning the protocol performance in the mobility-offered load space, it is possible to determinate the domain where the protocol is suitable to work.

A fairly comparison of routing protocols through simulation, under realistic conditions, is not a trivial task. Simulations have to be carried out under a wide set of different scenarios as well as have to use realistic and validated models for many physical phenomena, like radio propagation and interference, 802.11 medium access control, etc and also. Widely used simulation tools include ns-2, developed at University of California at Berkely, with wireless extensions provided by the CMU Monarch Project [48] and GloMoSim developed at University of California at Los Angeles [49]. Detailed and homogeneous simulation studies have been carried out for the following protocols: DSDV, DSR, ADV, AODV, TORA. The main conclusions are given below, for an in-depth discussion of the results, the reader may refer to the original papers [50-52].

DSDV is suitable only for low mobility. For a high value of the mobility, the protocol fails to converge and thus tables contain stale routes. However, for low values of offered load, the periodic route update results in a high value for the normalized routing overhead.

TORA is suitable for low mobility degree and with low to medium offered load. Performance are negatively affected by the existence of short-lived routing loops, raising during link reversal, and by the proactive nature of route maintenance (paths are monitored and repaired even if not used).

As far as AODV and DSR are concerned, both protocols are suitable under high mobility and offered load. However, AODV is suitable also when traffic diversity (number of active connections) increases, a condition for which DSR is not able to cope with.

Simulation results also showed the benefit of combining both the proactive and reactive techniques. ADV outperforms AODV and DSR in terms of higher throughput (up to 50% in high mobility), lower packet delays, fewer routing and control overhead packets [12].

5 Conclusions

Many protocols have been proposed in the literature to provide routing in a mobile environment. In this paper we have illustrated the main techniques proposed to deal with mobility in the case of unicast flat routing protocols. Table 1 and 2 summarize the main characteristics of each protocol described in this paper. The experience gained in wired networks have influenced the design of mobile ad hoc routing protocols in many ways.

However, wireless mobile networks introduce several concerns that make the routing issue completely different from the one provided for the fixed counterpart. For example, the limitation on the battery lifetime and transmission bandwidth, as well as the notion of route stability lead to a new notion of "best" route. Also, fairness is an important property of a protocol, since using always the same route result in battery drain-out for the same mobiles.

Besides the above considerations, other more evident desirably properties include: (a) loop avoidance; (b) fast converge upon link changes, (c) localized reaction to topology changes, (d) multiple routes information, (e) unidirectional link support and also (f) Quality of Service (QoS) support; (g) network partition support.

As a final remark, it is evident that while many interesting ideas have been proposed, still other efforts are required to full understand how all the needs can be satisfied in a cost-effective way.

References

1. A. S. Tanenbaum, Computer Networks, 3rd edition, Prentive Hall Andrew 1996
2. C.-C. Chiang, G. Pei, M. Gerla and T.-W. Chen, ``Scalable Routing Strategies for Ad Hoc Wireless Networks,", JSAC99, IEEE Journal on Selected Areas in Communications, Vol. 17, No. 8, pages 1369-1379, August 1999.
3. M.S. Corson and A. Ephremides, "A Distributed Routing Algorithm for Mobile Wireless Networks," *ACM/Baltzer Wireless Networks*, vol.1, no.1, pp.61-81, February 1995
4. David B. Johnson, ``Routing in Ad Hoc Networks of Mobile Hosts," Proceedings of the Workshop on Mobile Computing Systems and Applications, IEEE Computer Society, Santa Cruz, CA, pages 158-163,December 1994.
5. Ad hoc Networking, C.E. Perkins, Addison-Wesley, 2000
6. C-K Toh, "Ad hoc Mobile Wireless Networks", Prentice Hall PTR, Upper Saddle River, 2002
7. R.Prakash, " Unidirectional links prove costly in Wireless Ad-Hoc Networks " , Proceedings of the Discrete Algorithms and Methods for Mobile Computing and Communications - Dial M '99, Seattle, WA, August 20, 1998

8. F. STAJNO and R.J. Anderson, “The resurrecting dukling: Security issues for ad-hoc wireless networks”, in 7th Security Protocols Workshop, vol 1796 of Lecture Notes in Computer Science pages 172-194
9. P. Papadimitratos and Z.J. Haas, “Secure Routing for Mobile Ad Hoc Networks,” *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002),* San Antonio, TX, January 27-31, 2002.
10. L. Zhou and Z.J. Haas, “Securing Ad Hoc Networks,” *IEEE Network Magazine*, vol,13, no.6, November/December 1999
11. Z.J. Haas and M.R. Pearlman, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks," Internet Draft, draft-ietf-manet-zone-zrp-02.txt, June 1999
12. R.V. Boppanam and S.P. Konduru, "An Adaptive Distance Vector Routing Algorithm for Mobile Ad Hoc Networks, Proceedings of *IEEE INFOCOM 2001* Vol.: 3, pp. 1753-1762
13. C.-C. Chiang, H.-K. Wu, W. Liu, M. Gerla, "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel,*" IEEE Singapore International Conference on Networks*, 1997
14. R. Sivakumar, P. Sinha, and V. Bharghavan, "CEDAR: A Core-Extraction Distributed Ad Hoc Routing Algorithm," *IEEE Journal on Selected Areas in Communications, Special Issue on Wireless Ad Hoc Networks,* vol.17, no.8, pp.1454-1465, August 1999
15. S. Giordano, I. Stojmenovic, Lj. Blazevic http://www.site.uottawa.ca/~ivan/routing-survey.pdf
16. C. Hedrick, "The Routing Information Protocol", RFC 1058, June 1988
17. J. Moy. OSPF version 2. RFC 1247, July 1991.
18. C.E. Perkins and P. Bhagwat, ``Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", Computer Communications Review, pages 234-244, October 1994.
19. T. Clausen et al., "Optimized Link State Routing Protocol,” *IETF MANET Workin Group Internet Draft "draft-ietf-MANET-olsrr -06.txt,"* September 2001
20. A. Qayyum, L. Viennot, A. Laouiti, “Multipoint relaying: An efficient technique for flooding in mobile wireless networks,” 35th Annual Hawaii International Conference on System Sciences (HICSS'2001)
21. M.Gerla, X. Hong, G. Pei, "Fisheye State Routing Protocol (FSR) for Ad Hoc Networks", draft-ietf-MANET-fsr -02.txt, IETF MANET Working Group - Internet Draft, Dicember 2001
22. T.-W. Chen and M. Gerla, "Global State Routing: A New Routing Scheme for Ad-hoc Wireless Networks," *IEEE ICC,* pp171-175, June 1998

23. P.A. Humblet, "Another Adaptive Shortest-Path Algorithm", IEEE Trans. Comm., Vol.39, No.6, June 1991, pp.995-1003.
24. S. Murthy and J.J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks," *ACM Mobile Networks and Applications Journal, Special Issue on Routing in Mobile Communication Networks,* October 1996
25. C. Cheng, R. Reley, S.P.R. Kumar and J.J. Garcia-Luna-Aceves, "A Loop-Free Extended Bellman-Ford Routing Protocol without Bouncing Effect," *ACM Computer Communications Review,* vol. 19, no. 4, 1989, pp. 224-236
26. D.B. Johnson and D.A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," in Mobile Computing, edited by T. Imielinski and H. Korth, chapter 5, pp.153-181, Kluwer Academic Publishers, 1996
27. David B. Jhonson, David A. Maltz, Yih-Chun Hu, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks " draft-ietf-manet-dsr-07.txt, IETF MANET working group, February 2002
28. D-K Kim, C-K Toh, and Y-H Choi, "RODA: A New Dynamic Routing Protocol using Dual Paths to support Asymmetric Links in Mobile Ad Hoc Networks", Proceedings of IEEE IC3N, October 2000
29. S.Y. Ni, Y.C. Tseng, Y.S Chen, and J.P. Sheu, "The Broadcast Storm Problem in Mobile Ad hoc Networks," Proceeding of MOBICOM 99, pp.151-162, August 1999
30. Yih-Chun Hu and David B. Johnson, "Caching Strategies in on-demand routing protocols for wireless ad hoc networks", Proceedings of IEEE/ACM MOBICOM 00, pages 231-242, August 2000.
31. Mahesh K. Marina and Samir R. Das, "Performance of Route Cache Strategies in Dynamic Source Routing", Proceedings of ICDCS-2001, April 2001, Scottsdale, Arizona.
32. C.E. Perkins and E.M.Royer, ``Ad-hoc On Demand Distance Vector Routing," Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, February 1999, pages 90-100.
33. C. E. Perkins, E. M. Belding-Royer, and S R. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing, " draft-ietf-manet-aodv-10.txt, IETF MANET working group, January 2002
34. Vincent D. Park and M. Scott Corson, ``A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," Proceedings of IEEE INFOCOM '97, Kobe, Japan, April 1997.
35. C.K. Toh, "Associativity-Based Routing for Ad-Hoc Mobile Networks, " Wireless Personal Communications, Vol. 4, No. 2, pp. 1-36, Mar. 1997.

36. R. Dube, C. Rais, K. Wang and S. Tripathi, “Signal Stability based Adaptive Routing (SSA) for Ad hoc Mobile Network”, In IEEE Personal Communications, vol 4, no 1, pp.36-45, February 1997.
37. M.R. Pearlman and Z.J. Haas, ``Determining the Optimal Configuration for the Zone Routing Protocol," IEEE Journal on Selected Areas in Communications, Vol. 17, No. 8, August 1999, pages 1395-1414.
38. Z J. Haas, M R. Pearlman, P Samar, “The Interzone Routing Protocol (IERP) for Ad Hoc Networks”, draft-ietf-manet-zone-ierp-01.txt, IETF MANET working group, December 2001
39. Z J. Haas, M R. Pearlman, P Samar,“The Intrazone Routing Protocol (IARP) for Ad Hoc Network,” draft-ietf-manet-zone-iarp-01.txt, IETF MANET working group, December 2001
40. N. Nikaein, H Labiod, C. Bonnet, "Distributed Dynamic routing algorithm for mobile ad hoc networks", Proccedings of MobiHOC 2000, pages 19-27
41. Y.-B. Ko and N.H. Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks," *ACM/IEEE MobiCom,* Dallas, Texas, 1998
42. Y-B. Ko and N. H. Vaidya, "Location-Aided Routing(LAR) in Mobile Ad Hoc Networks", in ACM/Baltzer Wireless Networks (WINET) journal, Vol.6-4, 2000
43. S. Corson and J. Macker, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations," RFC2501, Request for Comments, January 1999.
44. G. Holland and N.H Vaidya, "Analysis of TCP performance over mobile ad hoc networks," Proceedings of IEEE/ACM MOBICOM'99, pp. 219-230, Seattle, August 1999
45. M. Gerla, K. Tang and R. Bagrodia, "TCP Performance in Wireless Multi-hop Networks," in *Mobile Computing Systems and Applications*, pp.41-50, 25-26 Feb.1999
46. D. Kim, C-K. Toh and Y. Choi, "TCP BuS: Improving TCP Performance in Wireless Ad Hoc Networks", Journal of Communication and Networks, Vol.3 no.2, 2001
47. F. Wang and Y. Zhang, “Improving TCP Performance over Mobile Ad-Hoc Networks with Out-of-Order Detection and Response”, in proceeding of the Third ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'02), Lausanne, Switzerland, June 2002
48. http://www.monarch.cs.cmu.edu/cmu-ns.html
49. http://may.cs.ucla.edu/projects/glomosim/
50. Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu and Jorjeta Jetcheva, ``A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking, Dallas-Texas, October 1998.

51. S.R. Das, C.E. Perkins, and E.M. Royer, "Performance comparison of two on-demand routing protocols for ad hoc networks," *IEEE INFOCOM*, vol.1, pp.3-12, March 2000

52. Sung-Ju Lee and Mario Gerla, ``A Simulation Study of Table-Driven and On-Demand Routing Protocols for Mobile Ad Hoc Networks," IEEE Network, pages 48-54, July/August 1999.

FIGURES

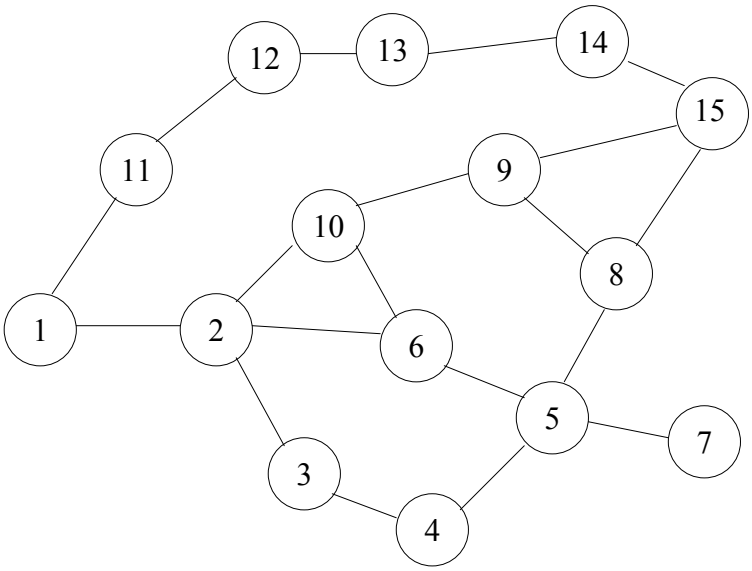


FIGURE 1:d A graph representation of a network.

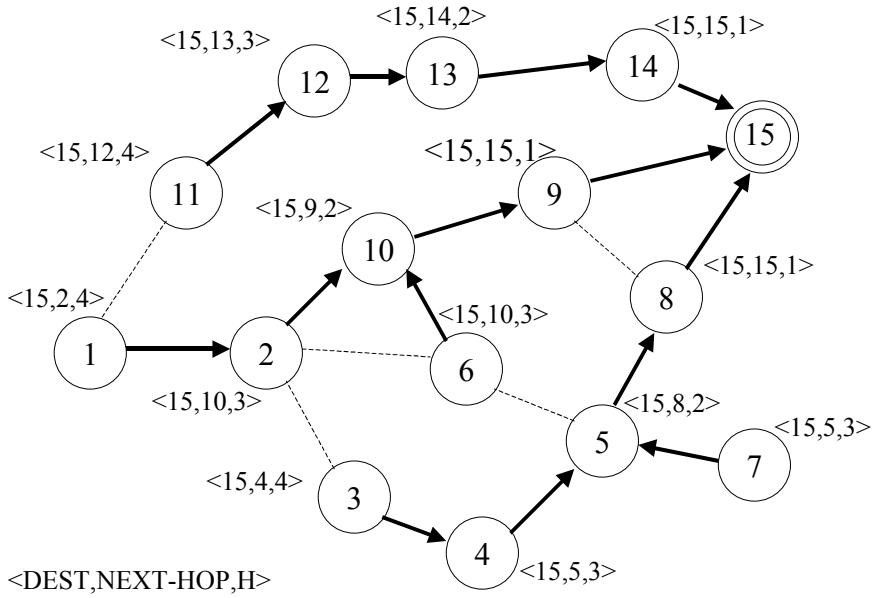


FIGURE 2: Route table entries for the destination node 15 showing the Minimum Spanning Tree (MST)

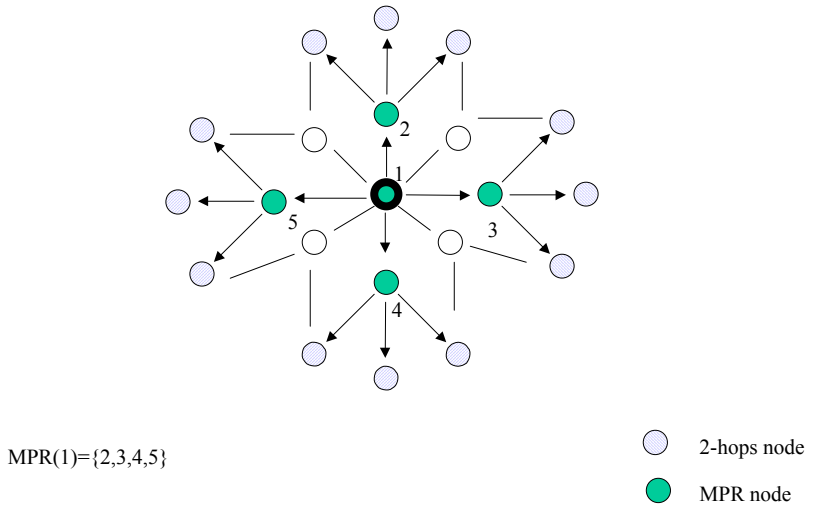


FIGURE 3 An example of flooding using MPR nodes

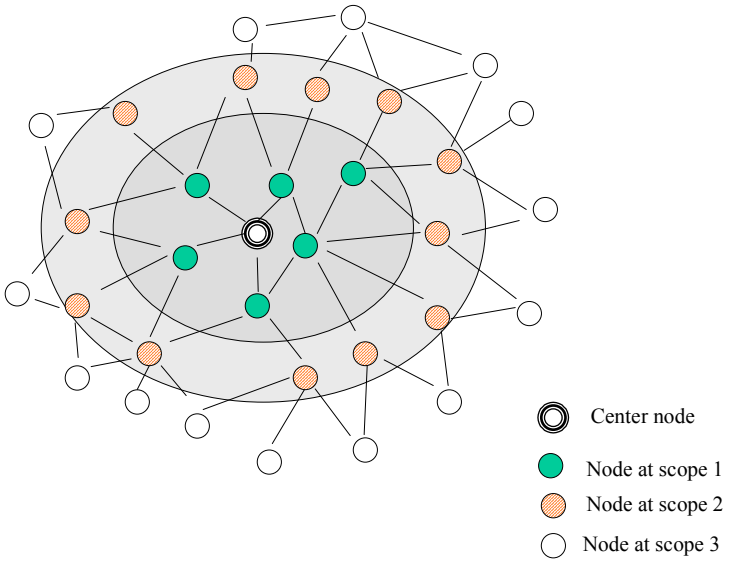


FIGURE 4: An example of scopes in the FSR protocol

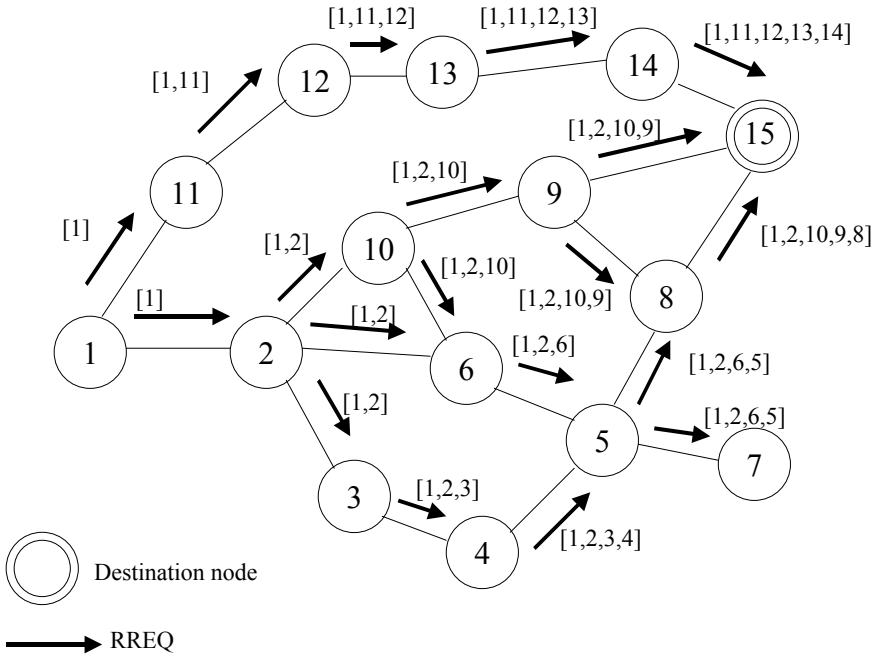


FIGURE 5: An example of routing propagation request in DSR

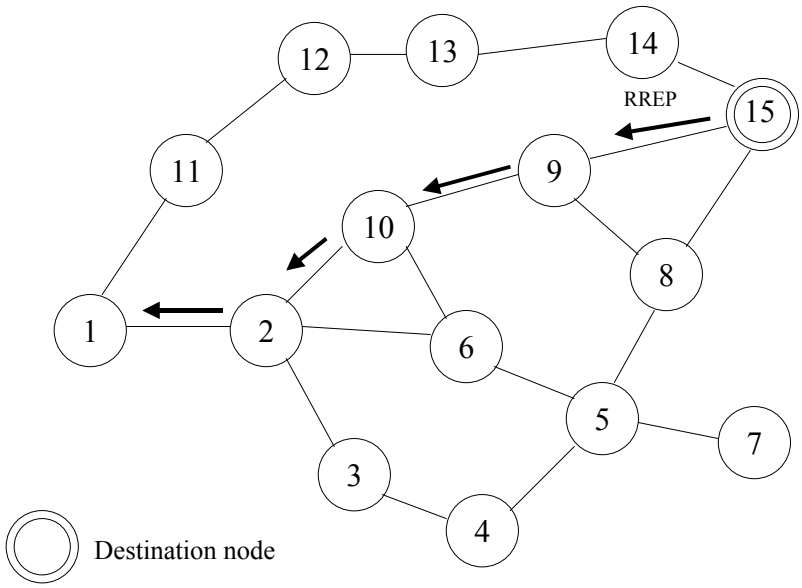


FIGURE 6: An example of route reply in DSR

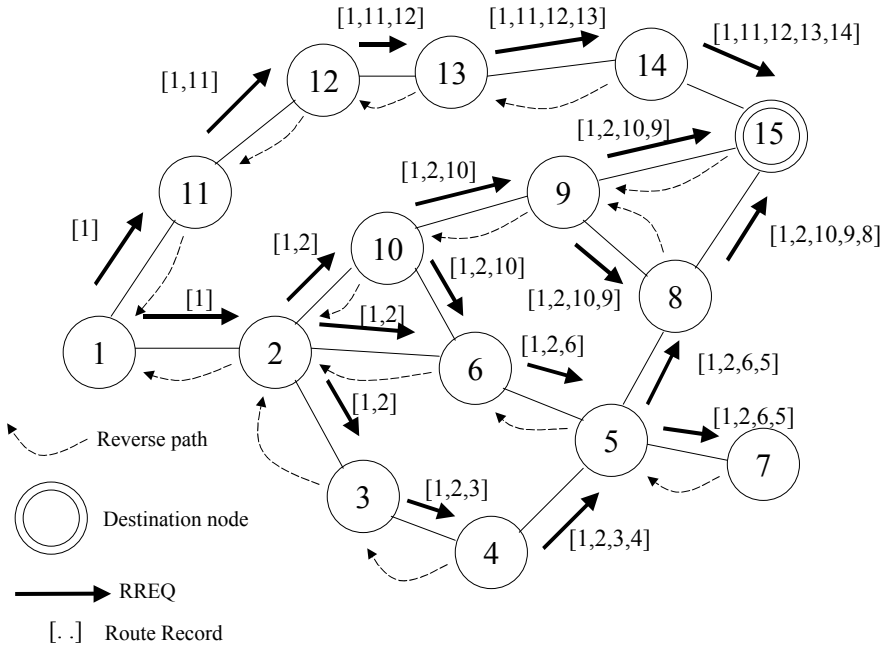


FIGURE 7: An example of propagation of RREQ in AODV and reserve path setup

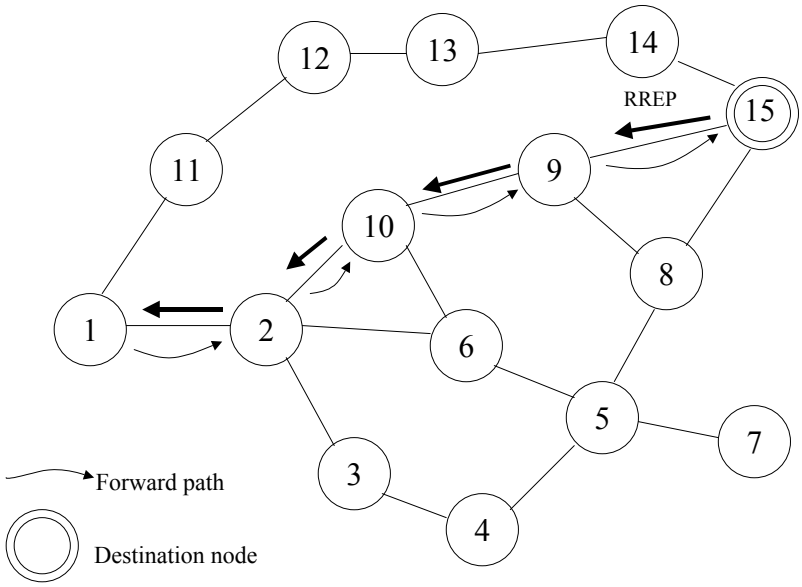


FIGURE 8: An example of forward path setup in AODV

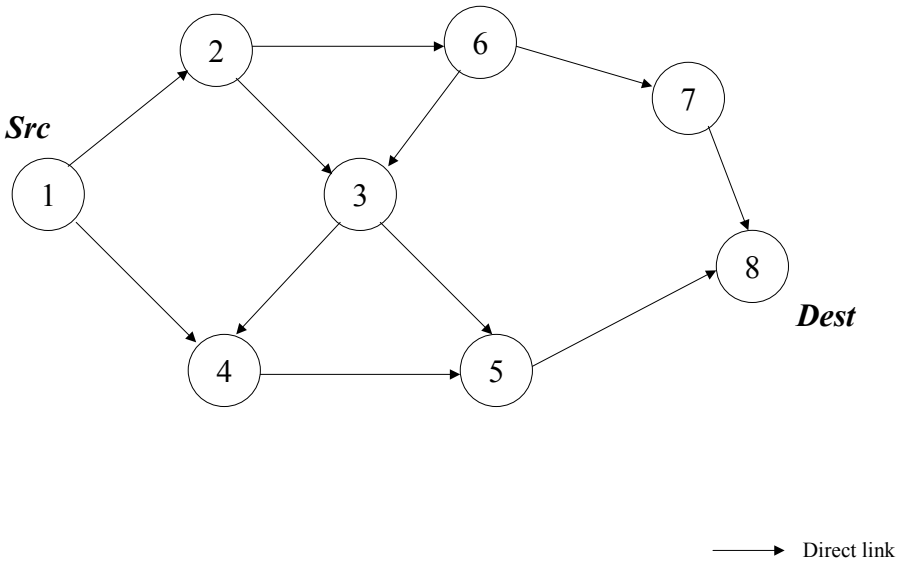
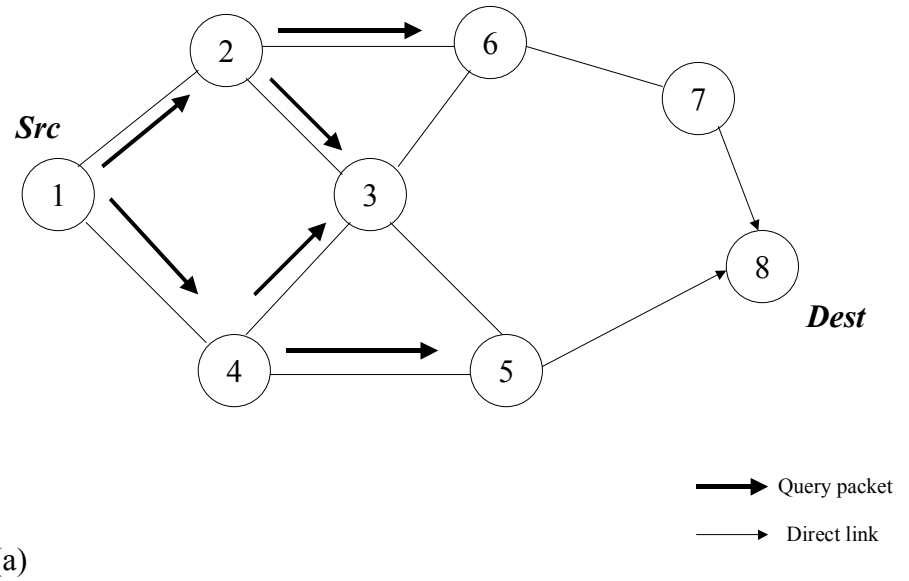
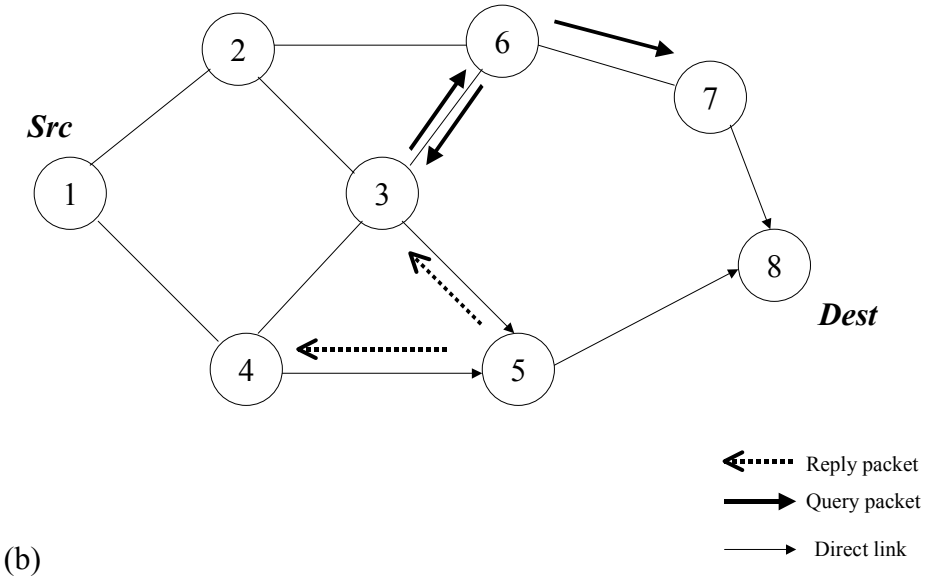


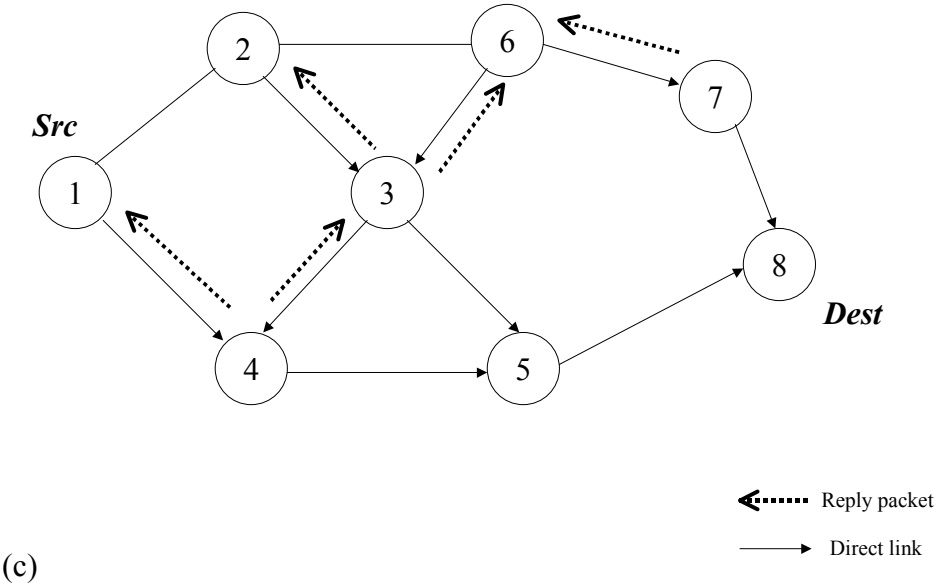
FIGURE 9: An example of DAG rooted at node 8



(a)



(b)



(c)

FIGURE 10: An example of query-reply cycle. (a) Generation of route request and its first propagation; (b) request propagation and initial reply; (c) reply propagation;

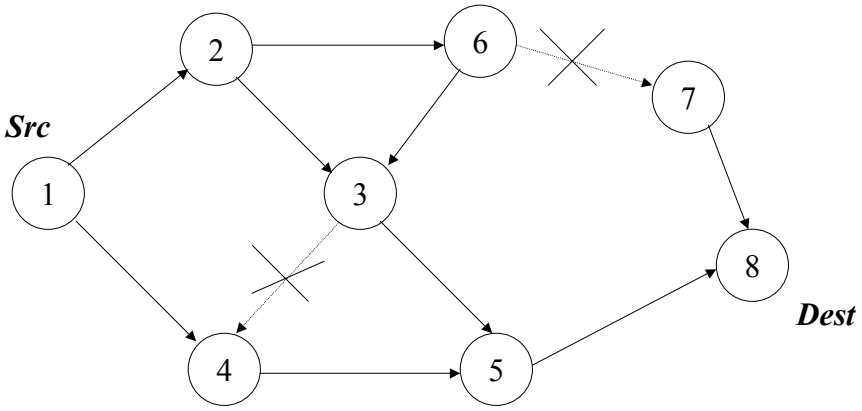
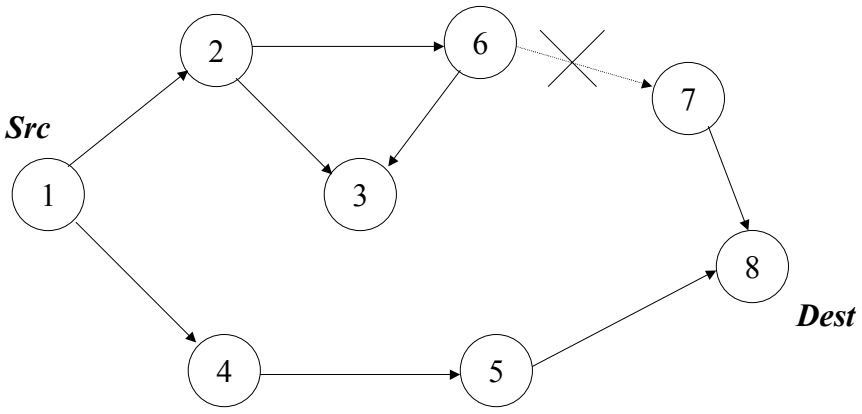
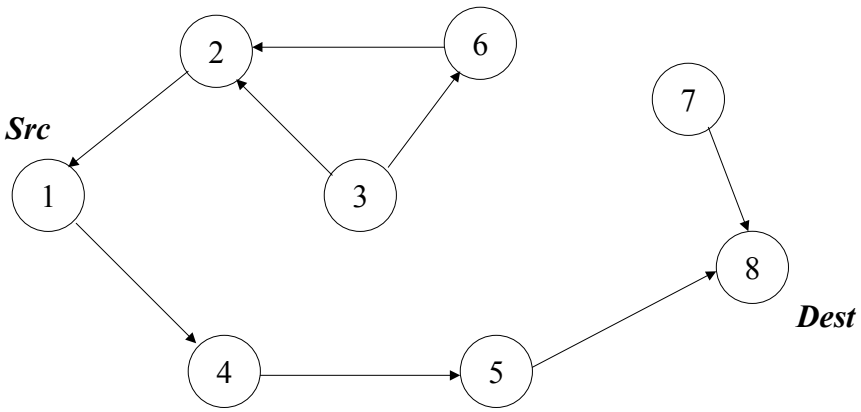


FIGURE 11: An example of topological changes that don't require maintenance



(a)



(b)

FIGURE 12: (a) an example of topological change that requires link reversal operations; (b) DAG obtained after the link reversal operations

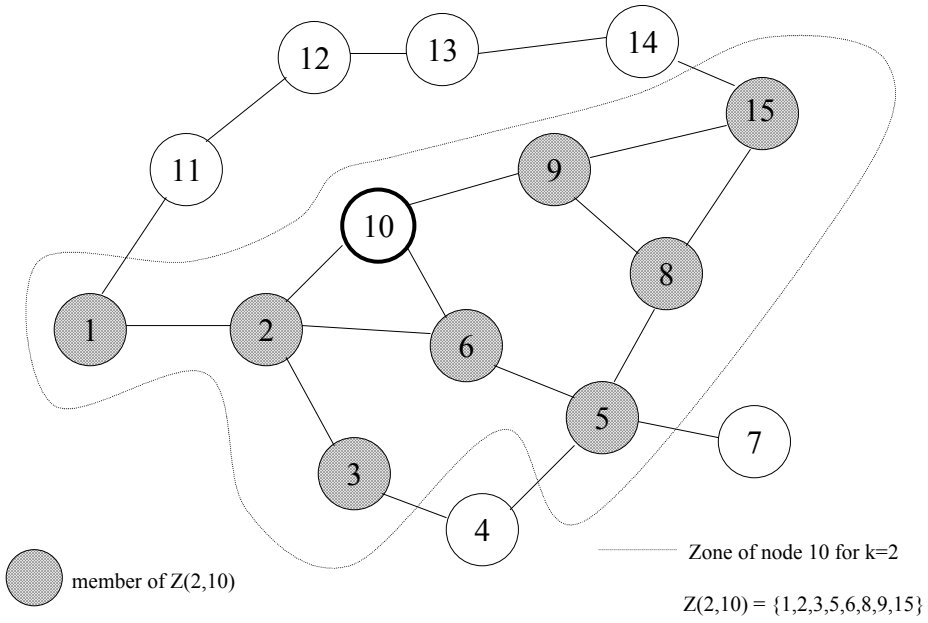


FIGURE 13 An example of zone in ZRP

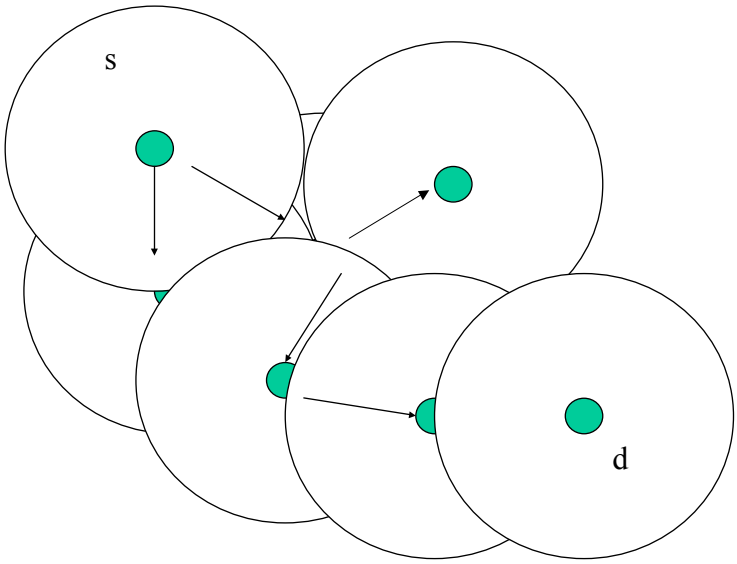


FIGURE 14: A sketch of query propagation using bordercast

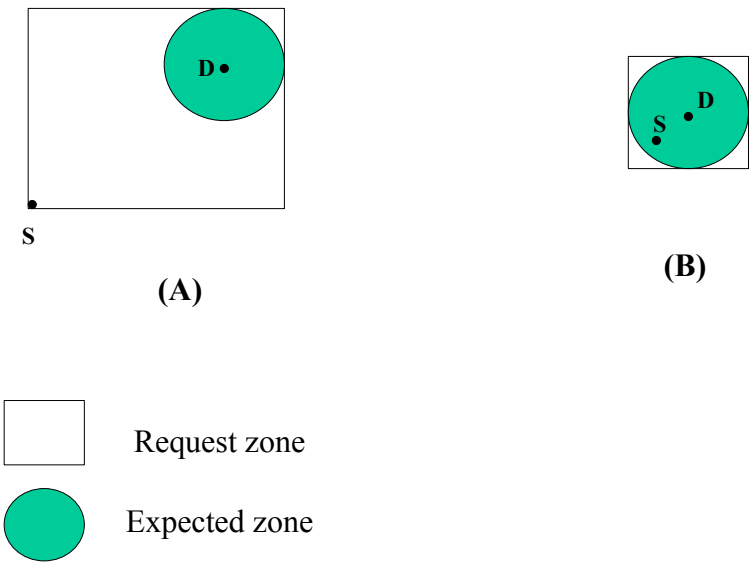


FIGURE 15: Examples of request and routing zones. (A) S outside the D's expected zone; S inside the D's expected zone