Network Working Group
Request for Comments: 5614
Category: Experimental
August 2009

R. Ogier
SRI International
P. Spagnolo
Boeing

Mobile Ad Hoc Network (MANET) Extension of OSPF Using Connected Dominating Set (CDS) Flooding

Abstract

This document specifies an extension of OSPFv3 to support mobile ad hoc networks (MANETs). The extension, called OSPF-MDR, is designed as a new OSPF interface type for MANETs. OSPF-MDR is based on the selection of a subset of MANET routers, consisting of MANET Designated Routers (MDRs) and Backup MDRs. The MDRs form a connected dominating set (CDS), and the MDRs and Backup MDRs together form a biconnected CDS for robustness. This CDS is exploited in two ways. First, to reduce flooding overhead, an optimized flooding procedure is used in which only (Backup) MDRs flood new link state advertisements (LSAs) back out the receiving interface; reliable flooding is ensured by retransmitting LSAs along adjacencies. Second, adjacencies are formed only between (Backup) MDRs and a subset of their neighbors, allowing for much better scaling in dense networks. The CDS is constructed using 2-hop neighbor information provided in a Hello protocol extension. The Hello protocol is further optimized by allowing differential Hellos that report only changes in neighbor states. Options are specified for originating router-LSAs that provide full or partial topology information, allowing overhead to be reduced by advertising less topology information.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document specifies an extension of OSPFv3 [RFC5340] to support a new interface type for mobile ad hoc networks (MANETs), i.e., for broadcast-capable, multihop wireless networks in which routers and hosts can be mobile. Note that OSPFv3 is specified by describing the modifications to OSPFv2 [RFC2328]. This MANET extension of OSPFv3 is also applicable to non-mobile mesh networks using layer-3 routing. This extension does not preclude the use of any existing OSPF interface types, and is fully compatible with legacy OSPFv3 implementations.

Existing OSPF interface types do not perform adequately in MANETs, due to scaling issues regarding the flooding protocol operation, inability of the Designated Router election protocol to converge in all scenarios, and large numbers of adjacencies when using a point-to-multipoint interface type.

The approach taken is to generalize the concept of an OSPF Designated Router (DR) and Backup DR to multihop wireless networks, in order to reduce overhead by reducing the number of routers that must flood new LSAs and reducing the number of adjacencies. The generalized (Backup) Designated Routers are called (Backup) MANET Designated Routers (MDRs). The MDRs form a connected dominating set (CDS), and the MDRs and Backup MDRs together form a biconnected CDS for robustness (if the network itself is biconnected). By definition, each router in the MANET either belongs to the CDS or is one hop away from it. A distributed algorithm is used to select and dynamically maintain the biconnected CDS. Adjacencies are established only between (Backup) MDRs and a subset of their neighbors, thus resulting in a dramatic reduction in the number of adjacencies in dense networks, compared to the approach of forming adjacencies between all neighbor pairs. The OSPF extension is called OSPF-MDR.

Hello packets are modified, using OSPF link-local signaling (LLS; see [RFC5613]), for two purposes: to provide neighbors with 2-hop neighbor information that is required by the MDR selection algorithm, and to allow differential Hellos that report only changes in neighbor states. Differential Hellos can be sent more frequently without a significant increase in overhead, in order to respond more quickly to topology changes.

Each MANET router advertises a subset of its MANET neighbors as point-to-point links in its router-LSA. The choice of which neighbors to advertise is flexible, allowing overhead to be reduced by advertising less topology information. Options are specified for originating router-LSAs that provide full or partial topology information.

This document is organized as follows. Section 2 presents an overview of OSPF-MDR, Section 3 presents the new interface and neighbor data items that are required for the extension, Section 4 describes the Hello protocol, including procedures for maintaining the 2-hop neighbor information, Section 5 describes the MDR selection algorithm, Section 6 describes changes to the Interface state machine, Section 7 describes the procedures for forming adjacencies and deciding which neighbors should become adjacent, Section 8 describes the flooding procedure, Section 9 specifies the requirements and options for the contents of router-LSAs, and Section 10 describes changes in the calculation of the routing table.

The appendices specify packet formats, detailed algorithms for the MDR selection algorithm, an algorithm for the selection of a subset of neighbors to advertise in the router-LSA to provide shortest-path routing, a proposed option that uses non-ackable LSAs to provide periodic flooding without the overhead of Link State Acknowledgments, and simulation results that predict the performance of OSPF-MDR in mobile networks with up to 200 nodes. Additional information and resources for OSPF-MDR can be found at http://www.manet-routing.org.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

In addition, this document uses the following terms:

MANET Interface

A MANET Interface is a new OSPF interface type that supports broadcast-capable, multihop wireless networks. Two neighboring routers on a MANET interface may not be able to communicate directly with each other. A neighboring router on a MANET interface is called a MANET neighbor. MANET neighbors are discovered dynamically using a modification of OSPF’s Hello protocol.

MANET Router

A MANET Router is an OSPF router that has at least one MANET interface.

Differential Hello

A Differential Hello is a Hello packet that reduces the overhead of sending full Hellos, by including only the Router IDs of neighbors whose state changed recently.

2-Hop Neighbor Information

This information specifies the bidirectional neighbors of each neighbor. The modified Hello protocol provides each MANET router with 2-hop neighbor information, which is used for selecting MDRs and Backup MDRs.

MANET Designated Router (MDR)

A MANET Designated Router is one of a set of routers responsible for flooding new LSAs, and for determining the set of adjacencies that must be formed. The set of MDRs forms a connected dominating set and is a generalization of the DR found in broadcast networks. Each router runs the MDR selection algorithm for each MANET interface, to decide whether the router is an MDR, Backup MDR, or neither for that interface.

Backup MANET Designated Router (Backup MDR or BMDR)

A Backup MANET Designated Router is one of a set of routers responsible for providing backup flooding when neighboring MDRs fail. The set of MDRs and Backup MDRs forms a biconnected dominating set. The Backup MDR is a generalization of the Backup DR found in broadcast networks.

MDR Other

A router is an MDR Other for a particular MANET interface if it is neither an MDR nor a Backup MDR for that interface.

Parent

Each router selects a Parent for each MANET interface. The Parent of a non-MDR router will be a neighboring MDR if one exists. The Parent of an MDR is always the router itself. Each non-MDR router becomes adjacent with its Parent. The Router ID of the Parent is advertised in the DR field of each Hello sent on the interface.

Backup Parent

If the option of biconnected adjacencies is chosen, then each MDR Other selects a Backup Parent, which will be a neighboring MDR or BMDR if one exists that is not the Parent. The Backup Parent of a BMDR is always the router itself. Each MDR Other becomes adjacent with its Backup Parent if it exists. The Router ID of the Backup Parent is advertised in the Backup DR field of each Hello sent on the interface.

Bidirectional Neighbor

A bidirectional neighbor is a neighboring router whose neighbor state is 2-Way or greater.

Routable Neighbor

A bidirectional MANET neighbor becomes routable if the SPF calculation has produced a route to the neighbor and the neighbor satisfies a quality condition. Once a neighbor becomes routable, it remains routable as long as it remains bidirectional. Only routable and Full neighbors can be used as next hops in the SPF calculation, and can be included in the router-LSA originated by the router.

Non-Flooding MDR

A non-flooding MDR is an MDR that does not automatically flood received LSAs back out the receiving interface, but performs backup flooding like a BMDR. Some MDRs may declare themselves non-flooding in order to reduce flooding overhead.

2. Overview

This section provides an overview of OSPF-MDR, including motivation and rationale for some of the design choices.

OSPF-MDR was motivated by the desire to extend OSPF to support MANETs, while keeping the same design philosophy as OSPF and using techniques that are similar to those of OSPF. For example, OSPF reduces overhead in a broadcast network by electing a Designated Router (DR) and Backup DR, and by having two neighboring routers form an adjacency only if one of them is the DR or Backup DR. This idea can be generalized to a multihop wireless network by forming a spanning tree, with the edges of the tree being the adjacencies and the interior (non-leaf) nodes of the tree being the generalized DRs, called MANET Designated Routers (MDRs).

To provide better robustness and fast response to topology changes, it was decided that a router should decide whether it is an MDR based only on local information that can be obtained from neighbors' Hellos. The resulting set of adjacencies therefore does not always form a tree globally, but appears to be a tree locally. Similarly, the Backup DR can be generalized to Backup MDRs (BMDRs), to provide robustness through biconnected redundancy. The set of MDRs forms a connected dominating set (CDS), and the set of MDRs and BMDRs forms a biconnected dominating set (if the network itself is biconnected).

The following subsections provide an overview of each of the main features of OSPF-MDR, starting with a summary of how MDRs, BMDRs, and adjacencies are selected.

2.1. Selection of MDRs, BMDRs, Parents, and Adjacencies

The MDR selection algorithm is distributed; each router selects itself as an MDR, BMDR, or other router (called an "MDR Other") based on information about its one-hop neighborhood, which is obtained from Hello packets received from neighbors. Routers are ordered lexicographically based on the tuple (RtrPri, MDR Level, RID), where RtrPri is the Router Priority, MDR Level represents the current state of the router (2 for an MDR, 1 for a BMDR, and 0 for an MDR Other), and RID is the Router ID. Routers with lexicographically larger values of (RtrPri, MDR Level, RID) are given preference for becoming MDRs.

The MDR selection algorithm can be summarized as follows. If the router itself has a larger value of (RtrPri, MDR Level, RID) than all of its neighbors, it selects itself as an MDR. Otherwise, let Rmax denote the neighbor with the largest value of (RtrPri, MDR Level, RID). The router then selects itself as an MDR unless each neighbor can be reached from Rmax in at most k hops via neighbors that have a larger value of (RtrPri, MDR Level, RID) than the router itself, where k is the parameter MDRConstraint, whose default value is 3.

This parameter serves to control the density of the MDR set, since the MDR set need not be strictly minimal.

Similarly, a router that does not select itself as an MDR will select itself as a BMDR unless each neighbor can be reached from Rmax via two node-disjoint paths, using as intermediate hops only neighbors that have a larger value of (RtrPri, MDR Level, RID) than the router itself.

When a router selects itself as an MDR, it also decides which MDR neighbors it should become adjacent with, to ensure that the set of MDRs and the adjacencies between them form a connected backbone. Each non-MDR router selects and becomes adjacent with an MDR neighbor called its Parent, thus ensuring that all routers are connected to the MDR backbone.

If the option of biconnected adjacencies is chosen (AdjConnectivity = 2), then additional adjacencies are selected to ensure that the set of MDRs and BMDRs, and the adjacencies between them, form a biconnected backbone. In this case, each MDR Other selects and becomes adjacent with an MDR/BMDR neighbor called its Backup Parent, in addition to its Parent.

OSPF-MDR also provides the option of full-topology adjacencies (AdjConnectivity = 0). If this option is selected, then each router forms an adjacency with each bidirectional neighbor. Although BMDR selection is optional if AdjConnectivity is 0 or 1, it is recommended since BMDRs improve robustness by providing backup flooding.

Prioritizing routers according to (RtrPri, MDR Level, RID) allows neighboring routers to agree on which routers should become an MDR, and gives higher priority to existing MDRs, which increases the lifetime of MDRs and the adjacencies between them. In addition, Parents are selected to be existing adjacent neighbors whenever possible, to avoid forming new adjacencies unless necessary. Once a neighbor becomes adjacent, it remains adjacent as long as the neighbor is bidirectional and either the neighbor or the router itself is an MDR or BMDR (similar to OSPF). The above rules reduce the rate at which new adjacencies are formed, which is important since database exchange must be performed whenever a new adjacency is formed.

2.2. Flooding Procedure

When an MDR receives a new link state advertisement (LSA) on a MANET interface, it floods the LSA back out the receiving interface unless it can be determined that such flooding is unnecessary (as specified in Section 8.1). The router MAY delay the flooding of the LSA by a small random amount of time (e.g., less than 100 ms). The delayed flooding is useful for coalescing multiple LSAs in the same Link State Update packet, and it can reduce the possibility of a collision in case multiple MDRs received the same LSA at the same time. However, such collisions are usually avoided with wireless MAC protocols.

When a Backup MDR receives a new LSA on a MANET interface, it waits a short interval (BackupWaitInterval), and then floods the LSA only if it has a neighbor that did not flood or acknowledge the LSA and is not known to be a neighbor of another neighbor (of the Backup MDR) that flooded the LSA.

MDR Other routers never flood LSAs back out the receiving interface. To exploit the broadcast nature of MANETs, a new LSA is processed (and possibly forwarded) if it is received from any neighbor in state 2-Way or greater. The flooding procedure also avoids redundant forwarding of LSAs when multiple interfaces exist.

2.3. Link State Acknowledgments

All Link State Acknowledgment packets are multicast. An LSA is acknowledged if it is a new LSA, or if it is a duplicate LSA received as a unicast. (A duplicate LSA received as multicast is not acknowledged.) An LSA that is flooded back out the same interface is treated as an implicit acknowledgment. Link State Acknowledgments may be delayed to allow coalescing multiple acknowledgments in the same packet. The only exception is that (Backup) MDRs send a multicast Link State Acknowledgment immediately when a duplicate LSA is received as a unicast, in order to prevent additional retransmissions. Only Link State Acknowledgments from adjacent neighbors are processed, and retransmitted LSAs are sent (via unicast) only to adjacent neighbors.

2.4. Routable Neighbors

In OSPF, a neighbor must typically be fully adjacent (in state Full) for it to be used in the SPF calculation. An exception exists for an OSPF broadcast network, to avoid requiring all pairs of routers in such a network to form adjacencies, which would generate a large amount of overhead. In such a network, a router can use a non-adjacent neighbor as a next hop as long as both routers are fully adjacent with the Designated Router. We define this neighbor relationship as a "routable neighbor" and extend its usage to the MANET interface type.

A MANET neighbor becomes routable if it is bidirectional and the SPF calculation has produced a route to the neighbor. (A flexible quality condition may also be required.) Only routable and Full neighbors can be used as next hops in the SPF calculation, and can be included in the router-LSA originated by the router. The idea is that if the SPF calculation has produced a route to the neighbor, then it makes sense to take a "shortcut" and forward packets directly to the neighbor.

The routability condition is a generalization of the way that neighbors on broadcast networks are treated in the SPF calculation. The network-LSA of an OSPF broadcast network implies that a router can use a non-adjacent neighbor as a next hop. But a network-LSA cannot describe the general topology of a MANET, making it necessary to explicitly include non-adjacent neighbors in the router-LSA. Allowing only adjacent neighbors in LSAs would either result in suboptimal routes or require a large number of adjacencies.

2.5. Partial and Full Topology LSAs

OSPF-MDR allows routers to originate both full-topology LSAs, which advertise links to all routable and Full neighbors, and partial-topology LSAs, which advertise only a subset of such links. In a dense network, partial-topology LSAs are typically much smaller than full-topology LSAs, thus achieving better scalability.

Each router advertises a subset of its neighbors as point-to-point links in its router-LSA. The choice of which neighbors to advertise is flexible. As a minimum requirement, each router must advertise a minimum set of "backbone" neighbors in its router-LSA. An LSA that includes only this minimum set of neighbors is called a minimal LSA and corresponds to LSAFullness = 0. This choice results in the minimum amount of LSA flooding overhead, but does not ensure routing along shortest paths. However, it is useful for achieving scalability to networks with a large number of nodes.

At the other extreme, if LSAFullness = 4, then the router originates a full-topology LSA, which includes all routable and Full neighbors.

Setting LSAFullness to 1 results in min-cost LSAs, which provide routing along shortest (minimum-cost) paths. Each router decides which neighbors to include in its router-LSA based on 2-hop neighbor information obtained from its neighbors’ Hellos. Each router includes in its LSA the minimum set of neighbors necessary to provide a shortest path between each pair of its neighbors.

Setting LSAFullness to 2 also provides shortest-path routing, but allows the router to advertise additional neighbors to provide redundant routes.

Setting LSAFullness to 3 results in MDR full LSAs, causing each MDR to originate a full-topology LSA while other routers originate minimal LSAs. This choice does not provide routing along shortest paths, but simulations have shown that it provides routing along nearly shortest paths with relatively low overhead.

The above LSA options are interoperable with each other, because they all require the router-LSA to include a minimum set of neighbors, and because the construction of the router-LSA (described in Section 9.4) ensures that the router-LSAs originated by different routers are consistent. Routing along shortest paths is provided if and only if every router selects LSAFullness to be 1, 2, or 4.

2.6. Hello Protocol

OSPF-MDR uses the same Hello format as OSPFv3, but appends additional information to Hello packets using link-local signaling (LLS), in order to indicate the set of bidirectional neighbors and other information that is used by the MDR selection algorithm and the min-cost LSA algorithm. In addition to full Hellos, which include the same set of neighbor IDs as OSPFv3 Hellos, OSPF-MDR allows the use of differential Hellos, which include only the IDs of neighbors whose state (or other information) has recently changed (within the last HelloRepeatCount Hellos).

Hellos are sent every HelloInterval seconds. Full Hellos are sent every 2HopRefresh Hellos, and differential Hellos are sent at all other times. For example, if 2HopRefresh is equal to 3, then every third Hello is a full Hello. The default value of 2HopRefresh is 1; i.e., the default is to send only full Hellos. The default value for HelloInterval is 2 seconds. Differential Hellos are used to reduce overhead and to allow Hellos to be sent more frequently, for faster reaction to topology changes.

3. Interface and Neighbor Data Structures

3.1. Changes to Interface Data Structure

The following modified or new data items are required for the Interface Data Structure of a MANET interface:

Type
A router that implements this extension can have one or more interfaces of type MANET, in addition to the OSPF interface types defined in [RFC2328].

State
The possible states for a MANET interface are the same as for a broadcast interface. However, the DR and Backup states now imply that the router is an MDR or Backup MDR, respectively.

MDR Level
The MDR Level is equal to MDR (value 2) if the router is an MDR, Backup MDR (value 1) if the router is a Backup MDR, and MDR Other (value 0) otherwise. The MDR Level is used by the MDR selection algorithm.

Parent
The Parent replaces the Designated Router (DR) data item of OSPF. Each router selects a Parent as described in Section 5.4. The Parent of an MDR is the router itself, and the Parent of a non-MDR

router will be a neighboring MDR, if one exists. The Parent is initialized to 0.0.0.0, indicating the lack of a Parent. Each router advertises the Router ID of its Parent in the DR field of each Hello sent on the interface.

Backup Parent

The Backup Parent replaces the Backup Designated Router data item of OSPF. The Backup Parent of a BMDR is the router itself. If the option of biconnected adjacencies is chosen, then each MDR Other selects a Backup Parent, which will be a neighboring MDR/BMDR if one exists that is not the Parent. The Backup Parent is initialized to 0.0.0.0, indicating the lack of a Backup Parent. Each router advertises the Router ID of its Backup Parent in the Backup DR field of each Hello sent on the interface.

Router Priority

An 8-bit unsigned integer. A router with a larger Router Priority is more likely to be selected as an MDR. The Router Priority for a MANET interface can be changed dynamically based on any criteria, including bandwidth capacity, willingness to be a relay (which can depend on battery life, for example), number of neighbors (degree), and neighbor stability. A router that has been a (Backup) MDR for a certain amount of time can reduce its Router Priority so that the burden of being a (Backup) MDR can be shared among all routers. If the Router Priority for a MANET interface is changed, then the interface variable MDRNeighborChange must be set.

Hello Sequence Number (HSN)

The 16-bit sequence number carried by the MDR-Hello TLV. The HSN is incremented by 1 (modulo 2^16) every time a Hello packet is sent on the interface.

MDRNeighborChange

A single-bit variable set to 1 if a neighbor change has occurred that requires the MDR selection algorithm to be executed.

3.2. New Configurable Interface Parameters

The following new configurable interface parameters are required for a MANET interface. The default values for HelloInterval, RouterDeadInterval, and RxmtInterval for a MANET interface are 2, 6, and 7 seconds, respectively.

The default configuration for OSPF-MDR uses uniconnected adjacencies (AdjConnectivity = 1) and partial-topology LSAs that provide shortest-path routing (LSAFullness = 1). This is the most scalable configuration that provides shortest-path routing. Other configurations may be preferable in special circumstances. For example, setting LSAFullness to 4 provides full-topology LSAs, and setting LSAFullness to 0 provides minimal LSAs that minimize overhead but do not ensure shortest-path routing. Setting AdjConnectivity to 2 may improve robustness by providing a biconnected adjacency subgraph, and setting AdjConnectivity to 0 results in full-topology adjacencies.

All possible configurations of the new interface parameters are functional, except that if AdjConnectivity is 0 (full-topology adjacencies), then LSAFullness must be 1, 2, or 4 (see Section 9.3).

Differential Hellos should be used to reduce the size of Hello packets when the average number of neighbors is large (e.g., greater than 50). Differential Hellos are obtained by setting the parameter 2HopRefresh to an integer greater than 1, with the recommended value being 3. Good performance in simulated mobile networks with up to 160 nodes has been obtained using the default configuration with differential Hellos. Good performance in simulated mobile networks with up to 200 nodes has been obtained using the same configuration except with minimal LSAs (LSAFullness = 0). Simulation results are presented in Appendix E.

Although all routers should preferably choose the same values for the new configurable interface parameters, this is not required. OSPF-MDR was carefully designed so that correct interoperation is achieved even if each router sets these parameters independently of the other routers.

AdjConnectivity

If equal to the default value of 1, then the set of adjacencies forms a (uni)connected graph. If equal to the optional value of 2, then the set of adjacencies forms a biconnected graph. If AdjConnectivity is 0, then adjacency reduction is not used; i.e., the router becomes adjacent with all of its neighbors.

MDRConstraint

A parameter of the MDR selection algorithm, which affects the number of MDRs selected and must be an integer greater than or equal to 2. The default value of 3 results in nearly the minimum number of MDRs. Values larger than 3 result in slightly fewer MDRs, and the value 2 results in a larger number of MDRs.

BackupWaitInterval

The number of seconds that a Backup MDR must wait after receiving a new LSA before it decides whether to flood the LSA. The default value is 0.5 second.

AckInterval
The interval between Link State Acknowledgment packets when only delayed acknowledgments need to be sent. AckInterval MUST be less than RxmtInterval, and SHOULD NOT be larger than 1 second. The default value is 1 second.

LSAFullness
Determines which neighbors a router should advertise in its router-LSA. The value 0 results in minimal LSAs that include only "backbone" neighbors. The values 1 and 2 result in partial-topology LSAs that provide shortest-path routing, with the value 2 providing redundant routes. The value 3 results in MDRs originating full-topology LSAs and other routers originating minimal LSAs. The value 4 results in all routers originating full-topology LSAs. The default value is 1.

2HopRefresh
One out of every 2HopRefresh Hellos sent on the interface must be a full Hello. All other Hellos are differential. The default value is 1; i.e., the default is to send only full Hellos. If differential Hellos are used, the recommended value of 2HopRefresh is 3.

HelloRepeatCount
The number of consecutive Hellos in which a neighbor must be included when its state changes, if differential Hellos are used. This parameter must be set to 3.

3.3. Changes to Neighbor Data Structure

The neighbor states are the same as for OSPF. However, the data for a MANET neighbor that has transitioned to the Down state must be maintained for at least HelloInterval * HelloRepeatCount seconds, to allow the state change to be reported in differential Hellos. The following new data items are required for the Neighbor Data Structure of a neighbor on a MANET interface.

Neighbor Hello Sequence Number (NHSN)
The Hello sequence number contained in the last Hello received from the neighbor.

A-bit
The A-bit copied from the MDR-Hello TLV of the last Hello received from the neighbor. This bit is 1 if the neighbor is using full-topology adjacencies, i.e., is not using adjacency reduction.

FullHelloRcvd

A single-bit variable equal to 1 if a full Hello has been received from the neighbor.

Neighbor’s MDR Level

The MDR Level of the neighbor, based on the DR and Backup DR fields of the last Hello packet received from the neighbor or from the MDR-DD TLV in a Database Description (DD) packet received from the neighbor.

Neighbor’s Parent

The neighbor’s choice for Parent, obtained from the DR field of the last Hello packet received from the neighbor or from the MDR-DD TLV in a DD packet received from the neighbor.

Neighbor’s Backup Parent

The neighbor’s choice for Backup Parent, obtained from the Backup DR field of the last Hello packet received from the neighbor or from the MDR-DD TLV in a DD packet received from the neighbor.

Child

A single-bit variable equal to 1 if the neighbor is a child, i.e., if the neighbor has selected the router as a (Backup) Parent.

Dependent Neighbor

A single-bit variable equal to 1 if the neighbor is a Dependent Neighbor, which is decided by the MDR selection algorithm. Each MDR/BMDR router becomes adjacent with its Dependent Neighbors (which are also MDR/BMDR routers) to form a connected backbone. The set of all Dependent Neighbors on a MANET interface is called the Dependent Neighbor Set (DNS) for the interface.

Dependent Selector

A single-bit variable equal to 1 if the neighbor has selected the router to be dependent.

Selected Advertised Neighbor (SAN)

A single-bit variable equal to 1 if the neighbor is a Selected Advertised Neighbor. Selected Advertised Neighbors are neighbors that the router has selected to be included in the router-LSA, along with other neighbors that are required to be included. The set of all Selected Advertised Neighbors on a MANET interface is called the Selected Advertised Neighbor Set (SANS) for the interface.

Routable

A single-bit variable equal to 1 if the neighbor is routable.

Neighbor’s Bidirectional Neighbor Set (BNS)
The neighbor’s set of bidirectional neighbors, which is updated when a Hello is received from the neighbor.

Neighbor’s Dependent Neighbor Set (DNS)
The neighbor’s set of Dependent Neighbors, which is updated when a Hello is received from the neighbor.

Neighbor’s Selected Advertised Neighbor Set (SANS)
The neighbor’s set of Selected Advertised Neighbors, which is updated when a Hello is received from the neighbor.

Neighbor’s Link Metrics
The link metric for each of the neighbor’s bidirectional neighbors, obtained from the Metric TLV appended to Hello packets.

4. Hello Protocol

The MANET interface utilizes Hellos for neighbor discovery and for enabling neighbors to learn 2-hop neighbor information. The protocol is flexible because it allows the use of full or differential Hellos. Full Hellos list all neighbors on the interface that are in state Init or greater, as in OSPFv3, whereas differential Hellos list only neighbors whose status as a bidirectional neighbor, Dependent Neighbor, or Selected Advertised Neighbor has recently changed. Differential Hellos are used to reduce overhead, and they allow Hellos to be sent more frequently (for faster reaction to topology changes). If differential Hellos are used, full Hellos are sent less frequently to ensure that all neighbors have current 2-hop neighbor information.

4.1. Sending Hello Packets

Hello packets are sent according to [RFC5340], Section 4.2.1.1, and [RFC2328], Section 9.5, with the following MANET-specific specifications beginning after paragraph 3 of Section 9.5. The Hello packet format is defined in [RFC5340], Section A.3.2, except for the ordering of the Neighbor IDs and the meaning of the DR and Backup DR fields as described below.

Similar to [RFC2328], the DR and Backup DR fields indicate whether the router is an MDR or Backup MDR. If the router is an MDR, then the DR field is the router’s own Router ID, and if the router is a Backup MDR, then the Backup DR field is the router’s own Router ID. These fields are also used to advertise the router’s Parent and Backup Parent, as specified in Section A.3 and Section 5.4.

Hellos are sent every HelloInterval seconds. Full Hellos are sent every 2HopRefresh Hellos, and differential Hellos are sent at all other times. For example, if 2HopRefresh is equal to 3, then every third Hello is a full Hello. If 2HopRefresh is set to 1, then all Hellos are full (the default).

The neighbor IDs included in the body of each Hello are divided into the following five disjoint lists of neighbors (some of which may be empty), and must appear in the following order:

List 1. Neighbors whose state recently changed to Down (included only in differential Hellos).

List 2. Neighbors in state Init.

List 3. Dependent Neighbors.

List 4. Selected Advertised Neighbors.

List 5. Unselected bidirectional neighbors, defined as bidirectional neighbors that are neither Dependent nor Selected Advertised Neighbors.

Note that all neighbors in Lists 3 through 5 are bidirectional neighbors. These lists are used to update the neighbor’s Bidirectional Neighbor Set (BNS), Dependent Neighbor Set (DNS), and Selected Advertised Neighbor Set (SANS) when a Hello is received.

Note that the above five lists are disjoint, so each neighbor can appear in at most one list. Also note that some or all of the five lists can be empty.

Link-local signaling (LLS) is used to append up to two TLVs to each MANET Hello packet. The format for LLS is given in Section A.2. The MDR-Hello TLV is appended to each (full or differential) MANET Hello packet. It indicates whether the Hello is full or differential, and gives the Hello Sequence Number (HSN) and the number of neighbor IDs in each of Lists 1 through 4 defined above. The size of List 5 is then implied by the packet length field of the Hello. The format of the MDR-Hello TLV is given in Section A.2.3.

In both full and differential Hellos, the appended MDR-Hello TLV is built as follows.

- The Sequence Number field is set to the current HSN for the interface; the HSN is then incremented (modulo 2^16).

- The D-bit of the MDR-Hello TLV is set to 1 for a differential Hello and 0 for a full Hello.

- The A-bit of the MDR-Hello TLV is set to 1 if AdjConnectivity is 0 (the router is using full-topology adjacencies); otherwise, it is set to 0.

- The N1, N2, N3, and N4 fields are set to the number of neighbor IDs in the body of the Hello that are in List 1, List 2, List 3, and List 4, respectively. (N1 is always zero in a full Hello.)

The MDR-Metric TLV (or Metric TLV) advertises the link cost to each bidirectional neighbor on the interface, to allow the selection of neighbors to include in partial-topology LSAs. If LSAFullness is 1 or 2, a Metric TLV must be appended to each MANET Hello packet unless all link costs are 1. The format of the Metric TLV is given in Section A.2.5. The I bit of the Metric TLV can be set to 0 or 1. If the I bit is set to 0, then the Metric TLV does not contain neighbor IDs, and contains the metric for each bidirectional neighbor listed in the (full or differential) Hello, in the same order. If the I bit is set to 1, then the Metric TLV includes the neighbor ID and metric for each bidirectional neighbor listed in the Hello whose metric is not equal to the Default Metric field of the TLV.

The I bit should be chosen to minimize the size of the Metric TLV. This can be achieved by choosing the I bit to be 1 if and only if the number of bidirectional neighbors listed in the Hello whose metric differs from the Default Metric field is less than 1/3 of the total number of bidirectional neighbors listed in the Hello.

For example, if all neighbors have the same metric, then the I bit should be set to 1, with the Default Metric equal to this metric, avoiding the need to include neighbor IDs and corresponding metrics in the TLV. At the other extreme, if all neighbors have different metrics, then the I bit should be set to 0 to avoid listing the same neighbor IDs in both the body of the Hello and the Metric TLV.

In both full and differential Hello packets, the L bit is set in the Hello’s option field to indicate LLS.

4.1.1. Full Hello Packet

In a full Hello, the neighbor ID list includes all neighbors on the interface that are in state Init or greater, in the order described above. The MDR-Hello TLV is built as described above. If a Metric TLV is appended, it is built as specified in Section A.2.5.

4.1.2. Differential Hello Packet

In a differential Hello, the five neighbor ID lists defined in Section 4.1 are populated as follows:

List 1 includes each neighbor in state Down that has not yet been included in HelloRepeatCount Hellos since transitioning to this state.

List 2 includes each neighbor in state Init that has not yet been included in HelloRepeatCount Hellos since transitioning to this state.

List 3 includes each Dependent Neighbor that has not yet been included in HelloRepeatCount Hellos since becoming a Dependent Neighbor.

List 4 includes each Selected Advertised Neighbor that has not yet been included in HelloRepeatCount Hellos since becoming a Selected Advertised Neighbor.

List 5 includes each unselected bidirectional neighbor (defined in Section 4.1) that has not yet been included in HelloRepeatCount Hellos since becoming an unselected bidirectional neighbor.

In addition, a bidirectional neighbor must be included (in the appropriate list) if the neighbor's BNS does not include the router (indicating that the neighbor does not consider the router to be bidirectional).

If a Metric TLV is appended to the Hello, then a bidirectional neighbor must be included (in the appropriate list) if it has not yet been included in HelloRepeatCount Hellos since its metric last changed.

4.2. Receiving Hello Packets

A Hello packet received on a MANET interface is processed as described in [RFC5340], Section 4.2.2.1, and the first two paragraphs of [RFC2328], Section 10.5, followed by the processing specified below.

The source of a received Hello packet is identified by the Router ID found in the Hello's OSPF packet header. If a matching neighbor cannot be found in the interface's data structure, one is created with the Neighbor ID set to the Router ID found in the OSPF packet header, the state initialized to Down, all MANET-specific neighbor variables (specified in Section 3.3) initialized to zero, and the neighbor’s DNS, SANS, and BNS initialized to empty sets.

The neighbor structure’s Router Priority is set to the value of the corresponding field in the received Hello packet. The Neighbor’s Parent is set to the value of the DR field, and the Neighbor’s Backup Parent is set to the value of the Backup DR field.

Now the rest of the Hello Packet is examined, generating events to be given to the neighbor and interface state machines. These state machines are specified to be either executed or scheduled (see [RFC2328], Section 4.4, "Tasking support"). For example, by specifying below that the neighbor state machine be executed in line, several neighbor state transitions may be affected by a single received Hello.

- If the L bit in the options field is not set, then an error has occurred and the Hello is discarded.

- If the LLS contains an MDR-Hello TLV, the neighbor state machine is executed with the event HelloReceived. Otherwise, an error has occurred and the Hello is discarded.

- The Hello Sequence Number and the A-bit in the MDR-Hello TLV are copied to the neighbor’s data structure.

- The DR and Backup DR fields are processed as follows.

  (1) If the DR field is equal to the neighbor’s Router ID, set the neighbor’s MDR Level to MDR.

  (2) Else if the Backup DR field is equal to the neighbor’s Router ID, set the neighbor’s MDR Level to Backup MDR.

  (3) Else, set the neighbor’s MDR Level to MDR Other and set the neighbor’s Dependent Neighbor variable to 0. (Only MDR/BMDR neighbors can be Dependent.)

  (4) If the DR or Backup DR field is equal to the router’s own Router ID, set the neighbor’s Child variable to 1; otherwise, set it to 0.

The neighbor ID list of the Hello is divided as follows into the five lists defined in Section 4.1, where N1, N2, N3, and N4 are obtained from the corresponding fields of the MDR-Hello TLV. List 1 is defined to be the first N1 neighbor IDs, List 2 is defined to be the next N2 neighbor IDs, List 3 is defined to be the next N3 neighbor IDs, List 4 is defined to be the next N4 neighbor IDs, and List 5 is defined to be the remaining neighbor IDs in the Hello.

Further processing of the Hello depends on whether it is full or differential, which is indicated by the value of the D-bit of the MDR-Hello TLV.

4.2.1. Full Hello Packet

If the received Hello is full (the D-bit of the MDR-Hello TLV is 0), the following steps are performed:

- o If the N1 field of the MDR-Hello TLV is not zero, then an error has occurred and the Hello is discarded. Otherwise, set FullHelloRcvd to 1.
- o In the neighbor structure, modify the neighbor’s DNS to equal the set of neighbor IDs in the Hello’s List 3, modify the neighbor’s SANS to equal the set of neighbor IDs in the Hello’s List 4, and modify the neighbor’s BNS to equal the set of neighbor IDs in the union of Lists 3, 4, and 5.
- o If the router itself appears in the Hello’s neighbor ID list, the neighbor state machine is executed with the event 2-WayReceived after the Hello is processed. Otherwise, the neighbor state machine is executed with the event 1-WayReceived after the Hello is processed.

4.2.2. Differential Hello Packet

If the received Hello is differential (the D-bit of the MDR-Hello TLV is 1), the following steps are performed:

(1) For each neighbor ID in List 1 or List 2 of the Hello:

- o Remove the neighbor ID from the neighbor’s DNS, SANS, and BNS, if it belongs to the neighbor set.

(2) For each neighbor ID in List 3 of the Hello:

- o Add the neighbor ID to the neighbor’s DNS and BNS, if it does not belong to the neighbor set.
- o Remove the neighbor ID from the neighbor’s SANS, if it belongs to the neighbor set.

(3) For each neighbor ID in List 4 of the Hello:

- o Add the neighbor ID to the neighbor’s SANS and BNS, if it does not belong to the neighbor set.

- o Remove the neighbor ID from the neighbor’s DNS, if it belongs to the neighbor set.

(4) For each neighbor ID in List 5 of the Hello:

- o Add the neighbor ID to the neighbor’s BNS, if it does not belong to the neighbor set.

- o Remove the neighbor ID from the neighbor’s DNS and SANS, if it belongs to the neighbor set.

(5) If the router’s own RID appears in List 1, execute the neighbor state machine with the event 1-WayReceived after the Hello is processed.

(6) If the router’s own RID appears in List 2, 3, 4, or 5, execute the neighbor state machine with the event 2-WayReceived after the Hello is processed.

(7) If the router’s own RID does not appear in the Hello’s neighbor ID list, and the neighbor state is 2-Way or greater, and the Hello Sequence Number is less than or equal to the previous sequence number plus HelloRepeatCount, then the neighbor state machine is executed with the event 2-WayReceived after the Hello is processed (the state does not change).

(8) If 2-WayReceived is not executed, then 1-WayReceived is executed after the Hello is processed.

4.2.3. Additional Processing for Both Hello Types

The following applies to both full and differential Hellos.

If the router itself belongs to the neighbor’s DNS, the neighbor’s Dependent Selector variable is set to 1; otherwise, it is set to 0.

The receiving interface’s MDRNeighborChange variable is set to 1 if any of the following changes occurred as a result of processing the Hello:

- o The neighbor’s state changed from less than 2-Way to 2-Way or greater, or vice versa.

- The neighbor is bidirectional and any of the following neighbor variables has changed: MDR Level, Router Priority, FullHelloRcvd, and Bidirectional Neighbor Set (BNS).

The neighbor state machine is scheduled with the event AdjOK? if any of the following changes occurred as a result of processing the Hello:

- The neighbor's state changed from less than 2-Way to 2-Way or greater.

- The neighbor is bidirectional and its MDR Level has changed, or its Child variable or Dependent Selector variable has changed from 0 to 1.

If the LLS contains a Metric TLV, it is processed by updating the neighbor's link metrics according to the format of the Metric TLV specified in Section A.2.5. If the LLS does not contain a Metric TLV and LSAFullness is 1 or 2, the metric for each of the neighbor's links is set to 1.

4.3. Neighbor Acceptance Condition

In wireless networks, a single Hello can be received from a neighbor with which a poor connection exists, e.g., because the neighbor is almost out of range. To avoid accepting poor-quality neighbors, and to employ hysteresis, a router may require that a stricter condition be satisfied before changing the state of a MANET neighbor from Down to Init or greater. This condition is called the "neighbor acceptance condition", which by default is the reception of a single Hello or DD packet. For example, the neighbor acceptance condition may require that 2 consecutive Hellos be received from a neighbor before changing the neighbor's state from Down to Init. Other possible conditions include the reception of 3 consecutive Hellos, or the reception of 2 of the last 3 Hellos. The neighbor acceptance condition may also impose thresholds on other measurements such as received signal strength.

The neighbor state transition for state Down and event HelloReceived is thus modified (see Section 7.1) to depend on the neighbor acceptance condition.

5. MDR Selection Algorithm

This section describes the MDR selection algorithm, which is run for each MANET interface to determine whether the router is an MDR, Backup MDR, or MDR Other for that interface. The algorithm also selects the Dependent Neighbors and the (Backup) Parent, which are used to decide which neighbors should become adjacent (see Section 7.2).

The MDR selection algorithm must be executed just before sending a Hello if the MDRNeighborChange bit is set for the interface. The algorithm SHOULD also be executed whenever a bidirectional neighbor transitions to less than 2-Way, and MAY be executed at other times when the MDRNeighborChange bit is set. The bit is cleared after the algorithm is executed.

To simplify the implementation, the MDR selection algorithm MAY be executed periodically just before sending each Hello, to avoid having to determine when the MDRNeighborChange bit should be set. After running the MDR selection algorithm, the AdjOK? event may be invoked for some or all neighbors as specified in Section 7.

The purpose of the MDRs is to provide a minimal set of relays for flooding LSAs, and the purpose of the Backup MDRs is to provide backup relays to flood LSAs when flooding by MDRs does not succeed. The set of MDRs forms a CDS, and the set of MDRs and Backup MDRs forms a biconnected CDS (if the network itself is biconnected).

Each MDR selects and becomes adjacent with a subset of its MDR neighbors, called Dependent Neighbors, forming a connected backbone. Each non-MDR router connects to this backbone by selecting and becoming adjacent with an MDR neighbor called its Parent. Each MDR selects itself as Parent, to inform neighbors that it is an MDR.

If AdjConnectivity = 2, then each (Backup) MDR selects and becomes adjacent with additional (Backup) MDR neighbors to form a biconnected backbone, and each MDR Other selects and becomes adjacent with a second (Backup) MDR neighbor called its Backup Parent, thus becoming connected to the backbone via two adjacencies. Each BMDR selects itself as Backup Parent, to inform neighbors that it is a BMDR.

The MDR selection algorithm is a distributed CDS algorithm that uses 2-hop neighbor information obtained from Hellos. More specifically, it uses as inputs the set of bidirectional neighbors (in state 2-Way or greater), the triplet (Router Priority, MDR Level, Router ID) for each such neighbor and for the router itself, and the neighbor variables Bidirectional Neighbor Set (BNS) and FullHelloRcvd for each such neighbor. The MDR selection algorithm can be implemented in O(d^2) time, where d is the number of neighbors.

The above triplet will be abbreviated as (RtrPri, MDR Level, RID). The triplet (RtrPri, MDR Level, RID) is said to be larger for Router A than for Router B if the triplet for Router A is lexicographically greater than the triplet for Router B. Routers that have larger values of this triplet are preferred for selection as an MDR. The algorithm therefore prefers routers that are already MDRs, resulting in a longer average MDR lifetime.

The MDR selection algorithm consists of five phases, the last of which is optional. Phase 1 creates the neighbor connectivity matrix for the interface, which determines which pairs of neighbors are neighbors of each other. Phase 2 decides whether the calculating router is an MDR, and which MDR neighbors are Dependent. Phase 3 decides whether the calculating router is a Backup MDR and, if AdjConnectivity = 2, which additional MDR/BMDR neighbors are Dependent. Phase 4 selects the Parent and Backup Parent.

The algorithm simplifies considerably if AdjConnectivity is 0 (full-topology adjacencies). In this case, the set of Dependent Neighbors is empty and MDR Other routers need not select Parents. Also, Phase 3 (BMDR selection) is not required if AdjConnectivity is 0 or 1. However, Phase 3 MUST be executed if AdjConnectivity is 2, and SHOULD be executed if AdjConnectivity is 0 or 1, since BMDRs improve robustness by providing backup flooding.

A router that has selected itself as an MDR in Phase 2 MAY execute Phase 5 to possibly declare itself a non-flooding MDR. A non-flooding MDR is the same as a flooding MDR except that it does not automatically flood received LSAs back out the receiving interface, because it has determined that neighboring MDRs are sufficient to flood the LSA to all neighbors. Instead, a non-flooding MDR performs backup flooding just like a BMDR. A non-flooding MDR maintains its MDR level (rather than being demoted to a BMDR) in order to maximize the stability of adjacencies. (The decision to form an adjacency does not depend on whether an MDR is non-flooding.) By having MDRs declare themselves to be non-flooding when possible, flooding overhead is reduced. The resulting reduction in flooding overhead can be dramatic for certain regular topologies, but has been found to be less than 15% for random topologies.

The following subsections describe the MDR selection algorithm, which is applied independently to each MANET interface. For convenience, the term "bi-neighbor" will be used as an abbreviation for "bidirectional neighbor".

5.1. Phase 1: Creating the Neighbor Connectivity Matrix

Phase 1 creates the neighbor connectivity matrix (NCM) for the interface. The NCM is a symmetric matrix that defines a topology graph for the set of bi-neighbors on the interface. The NCM assigns a value of 0 or 1 for each pair of bi-neighbors; a value of 1 indicates that the neighbors are assumed to be bi-neighbors of each other in the MDR selection algorithm. Letting i denote the router itself, NCM(i,j) and NCM(j,i) are set to 1 for each bi-neighbor j. The value of the matrix is set as follows for each pair of bi-neighbors j and k on the interface.

(1.1) If FullHelloRcvd is 1 for both neighbors j and k: NCM(j,k) = NCM(k,j) is 1 only if j belongs to the BNS of neighbor k and k belongs to the BNS of neighbor j.

(1.2) If FullHelloRcvd is 1 for neighbor j and is 0 for neighbor k: NCM(j,k) = NCM(k,j) is 1 only if k belongs to the BNS of neighbor j.

(1.3) If FullHelloRcvd is 0 for both neighbors j and k: NCM(j,k) = NCM(k,j) = 0.

In Step 1.1 above, two neighbors are considered to be bi-neighbors of each other only if they both agree that the other router is a bi-neighbor. This provides faster response to the failure of a link between two neighbors, since it is likely that one router will detect the failure before the other router. In Step 1.2 above, only neighbor j has reported its full BNS, so neighbor j is believed in deciding whether j and k are bi-neighbors of each other. As Step 1.3 indicates, two neighbors are assumed not to be bi-neighbors of each other if neither neighbor has reported its full BNS.

5.2. Phase 2: MDR Selection

Phase 2 depends on the parameter MDRConstraint, which affects the number of MDRs selected. The default value of 3 results in nearly the minimum number of MDRs, while the value 2 results in a larger number of MDRs. If AdjConnectivity = 0 (full-topology adjacencies), then the following steps are modified in that Dependent Neighbors are not selected.

(2.1) The set of Dependent Neighbors is initialized to be empty.

(2.2) If the router has a larger value of (RtrPri, MDR Level, RID) than all of its bi-neighbors, the router selects itself as an MDR; selects all of its MDR bi-neighbors as Dependent Neighbors; if AdjConnectivity = 2, selects all of its BMDR bi-neighbors as Dependent Neighbors; then proceeds to Phase 4.

(2.3) Let Rmax be the bi-neighbor with the largest value of (RtrPri, MDR Level, RID).

(2.4) Using NCM to determine the connectivity of bi-neighbors, compute the minimum number of hops, denoted hops(u), from Rmax to each other bi-neighbor u, using only intermediate nodes that are bi-neighbors with a larger value of (RtrPri, MDR Level, RID) than the router itself. If no such path from Rmax to u exists, then hops(u) equals infinity. (See Appendix B for a detailed algorithm using breadth-first search.)

(2.5) If hops(u) is at most MDRConstraint for each bi-neighbor u, the router selects no Dependent Neighbors, and sets its MDR Level as follows: If the MDR Level is currently MDR, then it is changed to BMDR if Phase 3 will be executed and to MDR Other if Phase 3 will not be executed. Otherwise, the MDR Level is not changed.

(2.6) Else, the router sets its MDR Level to MDR and selects the following neighbors as Dependent Neighbors: Rmax if it is an MDR or BMDR; each MDR bi-neighbor u such that hops(u) is greater than MDRConstraint; and if AdjConnectivity = 2, each BMDR bi-neighbor u such that hops(u) is greater than MDRConstraint.

(2.7) If steps 2.1 through 2.6 resulted in the MDR Level changing to BMDR, or to MDR with AdjConnectivity equal to 1 or 2, then execute steps 2.1 through 2.6 again. (This is necessary because the change in MDR Level can cause the set of Dependent Neighbors and the BFS tree to change.) This step is not required if the MDR selection algorithm is executed periodically.

Step 2.4 can be implemented using a breadth-first search (BFS) algorithm to compute min-hop paths from Rmax to all other bi-neighbors, modified to allow a bi-neighbor to be an intermediate node only if its value of (RtrPri, MDR Level, RID) is larger than that of the router itself. A detailed description of this algorithm, which runs in O(d^2) time, is given in Appendix B.

5.3. Phase 3: Backup MDR Selection

(3.1) If the MDR Level is MDR (after running Phase 2) and AdjConnectivity is not 2, then proceed to Phase 4. (If the MDR Level is MDR and AdjConnectivity = 2, then Phase 3 may select additional Dependent Neighbors to create a biconnected backbone.)

(3.2) Using NCM to determine the connectivity of bi-neighbors, determine whether or not there exist two node-disjoint paths from Rmax to each other bi-neighbor u, using only intermediate nodes that are bi-neighbors with a larger value of (RtrPri, MDR Level, RID) than the router itself. (See Appendix B for a detailed algorithm.)

(3.3) If there exist two such node-disjoint paths from Rmax to each other bi-neighbor u, then the router selects no additional Dependent Neighbors and sets its MDR Level to MDR Other.

(3.4) Else, the router sets its MDR Level to Backup MDR unless it already selected itself as an MDR in Phase 2, and if AdjConnectivity = 2, adds each of the following neighbors to the set of Dependent Neighbors: Rmax if it is an MDR or BMDR, and each MDR/BMDR bi-neighbor u such that Step 3.2 did not find two node-disjoint paths from Rmax to u.

(3.5) If steps 3.1 through 3.4 resulted in the MDR Level changing from MDR Other to BMDR, then run Phases 2 and 3 again. (This is necessary because running Phase 2 again can cause the MDR Level to change to MDR.) This step is not required if the MDR selection algorithm is executed periodically.

Step 3.2 can be implemented in O(d^2) time using the algorithm given in Appendix B. A simplified version of the algorithm is also specified, which results in a larger number of BMDRs.

5.4. Phase 4: Parent Selection

Each router selects a Parent for each MANET interface. The Parent of a non-MDR router will be a neighboring MDR if one exists. If the option of biconnected adjacencies is chosen, then each MDR Other selects a Backup Parent, which will be a neighboring MDR/BMDR if one exists that is not the Parent. The Parent of an MDR is always the router itself, and the Backup Parent of a BMDR is always the router itself.

The (Backup) Parent is advertised in the (Backup) DR field of each Hello sent on the interface. As specified in Section 7.2, each router forms an adjacency with its Parent and Backup Parent if it exists and is a neighboring MDR/BMDR.

For a given MANET interface, let Rmax denote the router with the largest value of (RtrPri, MDR Level, RID) among all bidirectional neighbors, if such a neighbor exists that has a larger value of (RtrPri, MDR Level, RID) than the router itself. Otherwise, Rmax is null.

If the calculating router has selected itself as an MDR, then the Parent is equal to the router itself, and the Backup Parent is Rmax. (The latter design choice was made because it results in slightly better performance than choosing no Backup Parent.) If the router has selected itself as a BMDR, then the Backup Parent is equal to the router itself.

If the calculating router is a BMDR or MDR Other, the Parent is selected to be any adjacent neighbor that is an MDR, if such a neighbor exists. If no adjacent MDR neighbor exists, then the Parent is selected to be Rmax. By giving preference to neighbors that are already adjacent, the formation of a new adjacency is avoided when possible. Note that the Parent can be a non-MDR neighbor temporarily when no MDR neighbor exists. (This design choice was also made for performance reasons.)

If AdjConnectivity = 2 and the calculating router is an MDR Other, then the Backup Parent is selected to be any adjacent neighbor that is an MDR or BMDR, other than the Parent selected in the previous paragraph, if such a neighbor exists. If no such adjacent neighbor exists, then the Backup Parent is selected to be the bidirectional neighbor, excluding the selected Parent, with the largest value of (RtrPri, MDR Level, RID), if such a neighbor exists. Otherwise, the Backup Parent is null.

5.5. Phase 5: Optional Selection of Non-Flooding MDRs

A router that has selected itself as an MDR MAY execute the following steps to possibly declare itself a non-flooding MDR. An MDR that does not execute the following steps is by default a flooding MDR.

(5.1) If the router has a larger value of (RtrPri, MDR Level, RID) than all of its bi-neighbors, the router is a flooding MDR. Else, proceed to Step 5.2.

(5.2) Let Rmax be the bi-neighbor that has the largest value of (RtrPri, MDR Level, RID).

(5.3) Using NCM to determine the connectivity of bi-neighbors, compute the minimum number of hops, denoted hops(u), from Rmax to each other bi-neighbor u, using only intermediate nodes that are MDR bi-neighbors with a smaller value of (RtrPri, RID) than the router itself. (This can be done using BFS as in Step 2.4).

(5.4) If hops(u) is at most MDRConstraint for each bi-neighbor u, then the router is a non-flooding MDR. Else, it is a flooding MDR.

6. Interface State Machine

6.1. Interface States

No new states are defined for a MANET interface. However, the DR and Backup states now imply that the router is an MDR or Backup MDR, respectively. The following modified definitions apply to MANET interfaces:

Waiting
In this state, the router learns neighbor information from the Hello packets it receives, but is not allowed to run the MDR selection algorithm until it transitions out of the Waiting state (when the Wait Timer expires). This prevents unnecessary changes in the MDR selection resulting from incomplete neighbor information. The length of the Wait Timer is 2HopRefresh * HelloInterval seconds (the interval between full Hellos).

DR Other
The router has run the MDR selection algorithm and determined that it is not an MDR or a Backup MDR.

Backup
The router has selected itself as a Backup MDR.

DR
The router has selected itself as an MDR.

6.2. Events that Cause Interface State Changes

All interface events defined in [RFC2328], Section 9.2, apply to MANET interfaces, except for BackupSeen and NeighborChange. BackupSeen is never invoked for a MANET interface (since seeing a Backup MDR does not imply that the router itself cannot also be an MDR or Backup MDR).

The event NeighborChange is replaced with the new interface variable MDRNeighborChange, which indicates that the MDR selection algorithm must be executed due to a change in neighbor information (see Section 4.2.3).

6.3. Changes to Interface State Machine

This section describes the changes to the interface state machine for a MANET interface. The two state transitions specified below are for state-event pairs that are described in [RFC2328], but have modified action descriptions because MDRs are selected instead of DRs. The state transition in [RFC2328] for the event NeighborChange is omitted; instead, the new interface variable MDRNeighborChange is used to indicate when the MDR selection algorithm needs to be executed. The state transition for the event BackupSeen does not apply to MANET interfaces, since this event is never invoked for a MANET interface. The interface state transitions for the events Loopback and UnloopInd are unchanged from [RFC2328].

```
     State:  Down
     Event:  InterfaceUp
 New state:  Depends on action routine.

    Action:  Start the interval Hello Timer, enabling the periodic
             sending of Hello packets out the interface.  The state
             transitions to Waiting and the single shot Wait Timer
             is started.


     State:  Waiting
     Event:  WaitTimer
 New state:  Depends on action routine.

    Action:  Run the MDR selection algorithm, which may result in a
             change to the router's MDR Level, Dependent Neighbors,
             and (Backup) Parent.  As a result of this calculation,
             the new interface state will be DR Other, Backup, or DR.

             As a result of these changes, the AdjOK? neighbor event
             may be invoked for some or all neighbors.  (See
             Section 7.)
```

7. Adjacency Maintenance

Adjacency forming and eliminating on non-MANET interfaces remain unchanged. Adjacency maintenance on a MANET interface requires changes to transitions in the neighbor state machine ([RFC2328], Section 10.3), to deciding whether to become adjacent ([RFC2328], Section 10.4), sending of DD packets ([RFC2328], Section 10.8), and receiving of DD packets ([RFC2328], Section 10.6). The specification below relates to the MANET interface only.

If full-topology adjacencies are used (AdjConnectivity = 0), the router forms an adjacency with each bidirectional neighbor. If adjacency reduction is used (AdjConnectivity is 1 or 2), the router forms adjacencies with a subset of its neighbors, according to the rules specified in Section 7.2.

An adjacency maintenance decision is made when any of the following four events occur between a router and its neighbor. The decision is made by executing the neighbor event AdjOK?.

(1) The neighbor state changes from Init to 2-Way.

(2) The MDR Level changes for the neighbor or for the router itself.

(3) The neighbor is selected to be the (Backup) Parent.

(4) The neighbor selects the router to be its (Backup) Parent.

7.1. Changes to Neighbor State Machine

The following specifies new transitions in the neighbor state machine.

```
 State(s):  Down
    Event:  HelloReceived
New state:  Depends on action routine.

   Action:  If the neighbor acceptance condition is satisfied (see
            Section 4.3), the neighbor state transitions to Init and
            the Inactivity Timer is started.  Otherwise, the neighbor
            remains in the Down state.


 State(s):  Init
    Event:  2-WayReceived
New state:  2-Way

   Action:  Transition to neighbor state 2-Way.

 State(s):  2-Way
    Event:  AdjOK?
New state:  Depends on action routine.
```

Action: Determine whether an adjacency should be formed with the neighboring router (see Section 7.2). If not, the neighbor state remains at 2-Way and no further action is taken.

Otherwise, the neighbor state changes to ExStart, and the following actions are performed. If the neighbor has a larger Router ID than the router’s own ID, and the received packet is a DD packet with the initialize (I), more (M), and master (MS) bits set, then execute the event NegotiationDone, which causes the state to transition to Exchange.

Otherwise (negotiation is not complete), the router increments the DD sequence number in the neighbor data structure. If this is the first time that an adjacency has been attempted, the DD sequence number should be assigned a unique value (like the time of day clock). It then declares itself master (sets the master/slave bit to master), and starts sending Database Description packets, with the initialize (I), more (M), and master (MS) bits set, the MDR-DD TLV included in an LLS, and the L bit set. This Database Description packet should be otherwise empty. This Database Description packet should be retransmitted at intervals of RxmtInterval until the next state is entered (see [RFC2328], Section 10.8).

State(s): ExStart or greater
Event: AdjOK?
New state: Depends on action routine.

Action: Determine whether the neighboring router should still be adjacent (see Section 7.3). If yes, there is no state change and no further action is necessary. Otherwise, the (possibly partially formed) adjacency must be destroyed. The neighbor state transitions to 2-Way. The Link state retransmission list, Database summary list, and Link state request list are cleared of LSAs.

7.2. Whether to Become Adjacent

The following defines the method to determine if an adjacency should be formed between neighbors in state 2-Way. The following procedure does not depend on whether AdjConnectivity is 1 or 2, but the selection of Dependent Neighbors (by the MDR selection algorithm) depends on AdjConnectivity.

If adjacency reduction is not used (AdjConnectivity = 0), then an adjacency is formed with each neighbor in state 2-Way. Otherwise, an adjacency is formed with a neighbor in state 2-Way if any of the following conditions is true:

(1) The router is a (Backup) MDR and the neighbor is a (Backup) MDR and is either a Dependent Neighbor or a Dependent Selector.

(2) The neighbor is a (Backup) MDR and is the router's (Backup) Parent.

(3) The router is a (Backup) MDR and the neighbor is a child.

(4) The neighbor's A-bit is 1, indicating that the neighbor is using full-topology adjacencies.

Otherwise, an adjacency is not established and the neighbor remains in state 2-Way.

7.3. Whether to Eliminate an Adjacency

The following defines the method to determine if an existing adjacency should be eliminated. An existing adjacency is maintained if any of the following is true:

(1) The router is an MDR or Backup MDR.

(2) The neighbor is an MDR or Backup MDR.

(3) The neighbor's A-bit is 1, indicating that the neighbor is using full-topology adjacencies.

Otherwise, the adjacency MAY be eliminated.

7.4. Sending Database Description Packets

Sending a DD packet on a MANET interface is the same as [RFC5340], Section 4.2.1.2, and [RFC2328], Section 10.8, with the following additions to paragraph 3 of Section 10.8.

If the neighbor state is ExStart, the standard initialization packet is sent with an MDR-DD TLV appended using LLS, and the L bit is set in the DD packet's option field. The format for the MDR-DD TLV is specified in Section A.2.4. The DR and Backup DR fields of the MDR-DD TLV are set exactly the same as the DR and Backup DR fields of a Hello sent on the same interface.

7.5. Receiving Database Description Packets

Processing a DD packet received on a MANET interface is the same as [RFC2328], Section 10.6, except for the changes described in this section. The following additional steps are performed before processing the packet based on neighbor state in paragraph 3 of Section 10.6.

o If the DD packet’s L bit is set in the options field and an MDR-DD TLV is appended, then the MDR-DD TLV is processed as follows.

(1) If the DR field is equal to the neighbor’s Router ID:

(a) Set the MDR Level of the neighbor to MDR.

(b) Set the neighbor’s Dependent Selector variable to 1.

(2) Else if the Backup DR field is equal to the neighbor’s Router ID:

(a) Set the MDR Level of the neighbor to Backup MDR.

(b) Set the neighbor’s Dependent Selector variable to 1.

(3) Else:

(a) Set the MDR Level of the neighbor to MDR Other.

(b) Set the neighbor’s Dependent Neighbor variable to 0.

(4) If the DR or Backup DR field is equal to the router’s own Router ID, set the neighbor’s Child variable to 1; otherwise, set it to 0.

o If the neighbor state is Init, the neighbor event 2-WayReceived is executed.

o If the MDR Level of the neighbor changed, the neighbor state machine is scheduled with the event AdjOK?.

o If the neighbor’s Child status has changed from 0 to 1, the neighbor state machine is scheduled with the event AdjOK?.

o If the neighbor’s neighbor state changed from less than 2-Way to 2-Way or greater, the neighbor state machine is scheduled with the event AdjOK?.

In addition, the Database Exchange optimization described in [RFC5243] SHOULD be performed as follows. If the router accepts a received DD packet as the next in sequence, the following additional step should be performed for each LSA listed in the DD packet (whether the router is master or slave). If the Database summary list contains an instance of the LSA that is the same as or less recent than the listed LSA, the LSA is removed from the Database summary list. This avoids listing the LSA in a DD packet sent to the neighbor, when the neighbor already has an instance of the LSA that is the same or more recent. This optimization reduces overhead due to DD packets by approximately 50% in large networks.

8. Flooding Procedure

This section specifies the changes to [RFC2328], Section 13, for routers that support OSPF-MDR. The first part of Section 13 (before Section 13.1) is the same except for the following three changes.

- To exploit the broadcast nature of MANETs, if the Link State Update (LSU) packet was received on a MANET interface, then the packet is dropped without further processing only if the sending neighbor is in a lesser state than 2-Way. Otherwise, the LSU packet is processed as described in this section.

- If the received LSA is the same instance as the database copy, the following actions are performed in addition to Step 7. For each MANET interface for which a BackupWait Neighbor List exists for the LSA (see Section 8.1):

  (a) Remove the sending neighbor from the BackupWait Neighbor List if it belongs to the list.

  (b) For each neighbor on the receiving interface that belongs to the BNS for the sending neighbor, remove the neighbor from the BackupWait Neighbor List if it belongs to the list.

- Step 8, which handles the case in which the database copy of the LSA is more recent than the received LSA, is modified as follows. If the sending neighbor is in a lesser state than Exchange, then the router does not send the LSA back to the sending neighbor.

There are no changes to Sections 13.1, 13.2, or 13.4. The following subsections describe the changes to Sections 13.3 (Next step in the flooding procedure), 13.5 (Sending Link State Acknowledgments), 13.6 (Retransmitting LSAs), and 13.7 (Receiving Link State Acknowledgments) of [RFC2328].

8.1. LSA Forwarding Procedure

When a new LSA is received, Steps 1 through 5 of [RFC2328], Section 13.3, are performed without modification for each eligible (outgoing) interface that is not of type MANET. This section specifies the modified steps that must be performed for each eligible MANET interface. The eligible interfaces depend on the LSA's flooding scope as described in [RFC5340], Section 4.5.2. Whenever an LSA is flooded out a MANET interface, it is included in an LSU packet that is sent to the multicast address AllSPFRouters. (Retransmitted LSAs are always unicast, as specified in Section 8.3.)

Step 1 of [RFC2328], Section 13.3, is performed for each eligible MANET interface with the following modification, so that the new LSA is placed on the Link State retransmission list for each appropriate adjacent neighbor. Step 1c is replaced with the following action, so that the LSA is not placed on the retransmission list for a neighbor that has already acknowledged the LSA.

- If the new LSA was received from this neighbor, or a Link State Acknowledgment (LS Ack) for the new LSA has already been received from this neighbor, examine the next neighbor.

To determine whether an Ack for the new LSA has been received from the neighbor, the router maintains an Acked LSA List for each adjacent neighbor, as described in Section 8.4. When a new LSA is received, the Acked LSA List for each neighbor, on each MANET interface, should be updated by removing any LS Ack that is for an older instance of the LSA than the one received.

The following description will use the notion of a "covered" neighbor. A neighbor k is defined to be covered if the LSA was sent as a multicast by a MANET neighbor j, and neighbor k belongs to the Bidirectional Neighbor Set (BNS) for neighbor j. A neighbor k is also defined to be covered if the LSA was sent to the multicast address AllSPFRouters by a neighbor j on a broadcast interface on which both j and k are neighbors. (Note that j must be the DR or Backup DR for the broadcast network, since only these routers may send LSAs to AllSPFRouters on a broadcast network.)

The following steps must be performed for each eligible MANET interface, to determine whether the new LSA should be forwarded on the interface.

(2) If every bidirectional neighbor on the interface satisfies at least one of the following three conditions, examine the next interface (the LSA is not flooded out this interface).

(a) The LSA was received from the neighbor.

(b) The LSA was received on a MANET or broadcast interface and the neighbor is covered (defined above).

(c) An Ack for the LSA has been received from the neighbor.

Condition (c) MAY be omitted (thus ignoring Acks) in order to simplify this step. Note that the above conditions do not assume the outgoing interface is the same as the receiving interface.

(3) If the LSA was received on this interface, and the router is an MDR Other for this interface, examine the next interface (the LSA is not flooded out this interface).

(4) If the LSA was received on this interface, and the router is a Backup MDR or a non-flooding MDR for this interface, then the router waits BackupWaitInterval before deciding whether to flood the LSA. To accomplish this, the router creates a BackupWait Neighbor List for the LSA, which initially includes every bidirectional neighbor on this interface that does not satisfy any of the conditions in Step 2. A single-shot BackupWait Timer associated with the LSA is started, which is set to expire after BackupWaitInterval seconds plus a small amount of random jitter. (The actions performed when the BackupWait Timer expires are described below in Section 8.1.2.) Examine the next interface (the LSA is not yet flooded out this interface).

(5) If the router is a flooding MDR for this interface, or if the LSA was originated by the router itself, then the LSA is flooded out the interface (whether or not the LSA was received on this interface) and the next interface is examined.

(6) If the LSA was received on a MANET or broadcast interface that is different from this (outgoing) interface, then the following two steps SHOULD be performed to avoid redundant flooding.

(a) If the router has a larger value of (RtrPri, MDR Level, RID) on the outgoing interface than every covered neighbor (defined above) that is a neighbor on BOTH the receiving and outgoing interfaces (or if no such neighbor exists), then the LSA is flooded out the interface and the next interface is examined.

(b) Else, the router waits BackupWaitInterval before deciding whether to flood the LSA on the interface, by performing the actions in Step 4 for a Backup MDR (whether or not the router is a Backup MDR on this interface). A separate BackupWait

Neighbor List is created for each MANET interface, but only one BackupWait Timer is associated with the LSA. Examine the next interface (the LSA is not yet flooded out this interface).

(7) If this step is reached, the LSA is flooded out the interface.

8.1.1. Note on Step 6 of LSA Forwarding Procedure

Performing the optional Step 6 can greatly reduce flooding overhead if the LSA was received on a MANET or broadcast interface. For example, assume that the LSA was received from the DR of a broadcast network that includes 100 routers, and 50 of the routers (not including the DR) are also attached to a MANET. Assume that these 50 routers are neighbors of each other in the MANET and that each has a neighbor in the MANET that is not attached to the broadcast network (and is therefore not covered). Then by performing Step 6 of the LSA forwarding procedure, the number of routers that forward the LSA from the broadcast network to the MANET is reduced from 50 to just 1 (assuming that at most 1 of the 50 routers is an MDR).

8.1.2. BackupWait Timer Expiration

If the BackupWait Timer for an LSA expires, then the following steps are performed for each (MANET) interface for which a BackupWait Neighbor List exists for the LSA.

(1) If the BackupWait Neighbor List for the interface contains at least one router that is currently a bidirectional neighbor, the following actions are performed.

   (a) The LSA is flooded out the interface.

   (b) If the LSA is on the Ack List for the interface (i.e., is scheduled to be included in a delayed Link State Acknowledgment packet), then the router SHOULD remove the LSA from the Ack List, since the flooded LSA will be treated as an implicit Ack.

   (c) If the LSA is on the Link State retransmission list for any neighbor, the retransmission SHOULD be rescheduled to occur after RxmtInterval seconds.

(2) The BackupWait Neighbor List is then deleted (whether or not the LSA is flooded).

8.2. Sending Link State Acknowledgments

This section describes the procedure for sending Link State Acknowledgments (LS Acks) on MANET interfaces. Section 13.5 of [RFC2328] remains unchanged for non-MANET interfaces, but does not apply to MANET interfaces. To minimize overhead due to LS Acks, and to take advantage of the broadcast nature of MANETs, all LS Ack packets sent on a MANET interface are multicast using the IP address AllSPFRouters. In addition, duplicate LSAs received as a multicast are not acknowledged.

When a router receives an LSA, it must decide whether to send a delayed Ack, an immediate Ack, or no Ack. The interface parameter AckInterval is the interval between LS Ack packets when only delayed Acks need to be sent. A delayed Ack SHOULD be delayed by at least (RxmtInterval - AckInterval - 0.5) seconds and at most (RxmtInterval - 0.5) seconds after the LSA instance being acknowledged was first received. If AckInterval and RxmtInterval are equal to their default values of 1 and 7 seconds, respectively, this reduces Ack traffic by increasing the chance that a new instance of the LSA will be received before the delayed Ack is sent. An immediate Ack is sent immediately in a multicast LS Ack packet, which may also include delayed Acks that were scheduled to be sent.

The decision whether to send a delayed or immediate Ack depends on whether the received LSA is new (i.e., is more recent than the database copy) or a duplicate (the same instance as the database copy), and on whether the LSA was received as a multicast or a unicast (which indicates a retransmitted LSA). The following rules are used to make this decision.

(1) If the received LSA is new, a delayed Ack is sent on each MANET interface associated with the area, unless the LSA is flooded out the interface.

(2) If the LSA is a duplicate and was received as a multicast, the LSA is not acknowledged.

(3) If the LSA is a duplicate and was received as a unicast:

   (a) If the router is an MDR, or AdjConnectivity = 2 and the router is a Backup MDR, or AdjConnectivity = 0, then an immediate Ack is sent out the receiving interface.

   (b) Otherwise, a delayed Ack is sent out the receiving interface.

The reason that (Backup) MDRs send an immediate Ack when a retransmitted LSA is received is to try to prevent other adjacent neighbors from retransmitting the LSA, since (Backup) MDRs usually have a large number of adjacent neighbors. MDR Other routers do not send an immediate Ack (unless AdjConnectivity = 0) because they have fewer adjacent neighbors, and so the potential benefit does not justify the additional overhead resulting from sending immediate Acks.

8.3. Retransmitting LSAs

LSAs are retransmitted according to Section 13.6 of [RFC2328]. Thus, LSAs are retransmitted only to adjacent routers. Therefore, since OSPF-MDR does not allow an adjacency to be formed between two MDR Other routers, an MDR Other never retransmits an LSA to another MDR Other, only to its Parents, which are (Backup) MDRs.

Retransmitted LSAs are included in LSU packets that are unicast directly to an adjacent neighbor that did not acknowledge the LSA (explicitly or implicitly). The length of time between retransmissions is given by the configurable interface parameter RxmtInterval, whose default is 7 seconds for a MANET interface. To reduce overhead, several retransmitted LSAs should be included in a single LSU packet whenever possible.

8.4. Receiving Link State Acknowledgments

A Link State Acknowledgment (LS Ack) packet that is received from an adjacent neighbor (in state Exchange or greater) is processed as described in Section 13.7 of [RFC2328], with the additional steps described in this section. An LS Ack packet that is received from a neighbor in a lesser state than Exchange is discarded.

Each router maintains an Acked LSA List for each adjacent neighbor, to keep track of any LSA instances the neighbor has acknowledged but that the router itself has NOT yet received. This is necessary because (unlike [RFC2328]) each router acknowledges an LSA only the first time it is received as a multicast.

If the neighbor from which the LS Ack packet was received is in state Exchange or greater, then the following steps are performed for each LS Ack in the received LS Ack packet:

(1) If the router does not have a database copy of the LSA being acknowledged, or has a database copy that is less recent than the one being acknowledged, the LS Ack is added to the Acked LSA List for the sending neighbor.

(2) If the router has a database copy of the LSA being acknowledged, which is the same as the instance being acknowledged, then the following action is performed. For each MANET interface for which a BackupWait Neighbor List exists for the LSA (see Section 8.1), remove the sending neighbor from the BackupWait Neighbor List if it belongs to the list.

9. Router-LSAs

Unlike the DR of an OSPF broadcast network, an MDR does not originate a network-LSA, since a network-LSA cannot be used to describe the general topology of a MANET. Instead, each router advertises a subset of its MANET neighbors as point-to-point links in its router-LSA. The choice of which MANET neighbors to include in the router-LSA is flexible. Whether or not adjacency reduction is used, the router can originate either partial-topology or full-topology LSAs.

If adjacency reduction is used (AdjConnectivity is 1 or 2), then as a minimum requirement each router must advertise a minimum set of "backbone" neighbors in its router-LSA. This minimum choice corresponds to LSAFullness = 0, and results in the minimum amount of LSA flooding overhead, but does not provide routing along shortest paths.

Therefore, to allow routers to calculate shortest paths, without requiring every pair of neighboring routers along the shortest paths to be adjacent (which would be inefficient due to requiring a large number of adjacencies), a router-LSA may also advertise non-adjacent neighbors that satisfy a synchronization condition described below.

To motivate this, we note that OSPF already allows a non-adjacent neighbor to be a next hop, if both the router and the neighbor belong to the same broadcast network (and are both adjacent to the DR). A network-LSA for a broadcast network (which includes all routers attached to the network) implies that any router attached to the network can forward packets directly to any other router attached to the network (which is why the distance from the network to all attached routers is zero in the graph representing the link-state database).

Since a network-LSA cannot be used to describe the general topology of a MANET, the only way to advertise non-adjacent neighbors that can be used as next hops is to include them in the router-LSA. However, to ensure that such neighbors are sufficiently synchronized, only "routable" neighbors are allowed to be included in LSAs, and to be used as next hops in the SPF calculation.

9.1. Routable Neighbors

If adjacency reduction is used, a bidirectional MANET neighbor becomes routable if the SPF calculation has found a route to the neighbor and the neighbor satisfies the routable neighbor quality condition (defined below). Since only routable and Full neighbors are advertised in router-LSAs, and since adjacencies are selected to form a connected spanning subgraph, this definition implies that there exists, or recently existed, a path of full adjacencies from the router to the routable neighbor. The idea is that, since a routable neighbor can be reached through an acceptable path, it makes sense to take a "shortcut" and forward packets directly to the routable neighbor.

This requirement does not guarantee perfect synchronization, but simulations have shown that it performs well in mobile networks. This requirement avoids, for example, forwarding packets to a new neighbor that is poorly synchronized because it was not reachable before it became a neighbor.

To avoid selecting poor-quality neighbors as routable neighbors, a neighbor that is selected as a routable neighbor must satisfy the routable neighbor quality condition. By default, this condition is that the neighbor’s BNS must include the router itself (indicating that the neighbor agrees the connection is bidirectional). Optionally, a router may impose a stricter condition. For example, a router may require that two Hellos have been received from the neighbor that (explicitly or implicitly) indicate that the neighbor’s BNS includes the router itself.

The single-bit neighbor variable Routable indicates whether the neighbor is routable, and is initially set to 0. If adjacency reduction is used, Routable is updated as follows when the state of the neighbor changes, or the SPF calculation finds a route to the neighbor, or a Hello is received that affects the routable neighbor quality condition.

(1) If Routable is 0 for the neighbor, the state of the neighbor is 2-Way or greater, there exists a route to the neighbor, and the routable neighbor quality condition (defined above) is satisfied, then Routable is set to 1 for the neighbor.

(2) If Routable is 1 for the neighbor and the state of the neighbor is less than 2-Way, Routable is set to 0 for the neighbor.

If adjacency reduction is not used (AdjConnectivity = 0), then routable neighbors are not computed and the set of routable neighbors remains empty.

9.2. Backbone Neighbors

The flexible choice for the router-LSA is made possible by defining two types of neighbors that are included in the router-LSA: backbone neighbors and Selected Advertised Neighbors.

If adjacency reduction is used, a bidirectional neighbor is defined to be a backbone neighbor if and only if it satisfies the condition for becoming adjacent (see Section 7.2). If adjacency reduction is not used (AdjConnectivity = 0), a bidirectional neighbor is a backbone neighbor if and only if the neighbor’s A-bit is 0 (indicating that the neighbor is using adjacency reduction). This definition allows the interoperation between routers that use adjacency reduction and routers that do not.

If adjacency reduction is used, then a router MUST include in its router-LSA all Full neighbors and all routable backbone neighbors. A minimal LSA, corresponding to LSAFullness = 0, includes only these neighbors. This choice guarantees connectivity, but does not ensure shortest paths. However, this choice is useful in large networks to achieve maximum scalability.

9.3. Selected Advertised Neighbors

To allow flexibility while ensuring that router-LSAs are symmetric (i.e., router i advertises a link to router j if and only if router j advertises a link to router i), each router maintains a Selected Advertised Neighbor set (SANS), which consists of MANET neighbors that the router has selected to advertise in its router-LSA, not including backbone neighbors. Since the SANS does not include backbone neighbors (and thus Dependent Neighbors), the SANS and DNS are disjoint. Both of these neighbor sets are advertised in Hellos.

If LSAFullness is 0 (minimal LSAs), then the SANS is empty. At the other extreme, if LSAFullness is 4 (full-topology LSAs), the SANS includes all bidirectional MANET neighbors except backbone neighbors. In between these two extremes, a router that is using adjacency reduction may select any subset of bidirectional non-backbone neighbors as its SANS. The resulting router-LSA is constructed as specified in Section 9.4.

Since a router that is not using adjacency reduction typically has no backbone neighbors (unless it has neighbors that are using adjacency reduction), to ensure connectivity, such a router must choose its SANS to contain the SANS corresponding to LSAFullness = 1. Thus, if AdjConnectivity is 0 (no adjacency reduction), then LSAFullness must be 1, 2, or 4.

If LSAFullness is 1, the router originates min-cost LSAs, which are partial-topology LSAs that (when flooded) provide each router with sufficient information to calculate a shortest (minimum-cost) path to each destination. Appendix C describes the algorithm for selecting the neighbors to include in the SANS that results in min-cost LSAs. The input to this algorithm includes information obtained from Hellos received from each MANET neighbor, including the neighbor's Bidirectional Neighbor Set (BNS), Dependent Neighbor Set (DNS), Selected Advertised Neighbor Set (SANS), and the Metric TLV. The Metric TLV, specified in Section A.2.5, is appended to each Hello (unless all link costs are 1) to advertise the link cost to each bidirectional neighbor.

If LSAFullness is 2, the SANS must be selected to be a superset of the SANS corresponding to LSAFullness = 1. This choice provides shortest-path routing while allowing the router to advertise additional neighbors to provide redundant routes.

If LSAFullness is 3, each MDR originates a full-topology LSA (which includes all Full and routable neighbors), while each non-MDR router originates a minimal LSA. If the router has multiple MANET interfaces, the router-LSA includes all Full and routable neighbors on each interface for which it is an MDR, and advertises only Full neighbors and routable backbone neighbors on its other interfaces. This choice provides routing along nearly shortest paths with relatively low overhead.

Although this document specifies a few choices of the SANS, which correspond to different values of LSAFullness, it is important to note that other choices are possible. In addition, it is not necessary for different routers to choose the same value of LSAFullness. The different choices are interoperable because they all require the router-LSA to include a minimum set of neighbors, and because the construction of the router-LSA (described below) ensures that the router-LSAs originated by different routers are consistent.

9.4. Originating Router-LSAs

When a new router-LSA is originated, it includes a point-to-point (type 1) link for each MANET neighbor that is advertised. The set of neighbors to be advertised is determined as follows. If adjacency reduction is used, the router advertises all Full neighbors, and advertises each routable neighbor j that satisfies any of the following three conditions. If adjacency reduction is not used (AdjConnectivity = 0), the router advertises each Full neighbor j that satisfies any of the following three conditions.

(1) The router's SANS (for any interface) includes j.

(2) Neighbor j’s SANS includes the router (to ensure symmetry).

(3) Neighbor j is a backbone neighbor.

Note that backbone neighbors and neighbors in the SANS need not be routable or Full, but only routable and Full neighbors may be included in the router-LSA. This is done so that the SANS, which is advertised in Hellos, does not depend on routability.

The events that cause a new router-LSA to be originated are the same as in [RFC2328] and [RFC5340] except that a MANET neighbor changing to/from the Full state does not always cause a new router-LSA to be originated. Instead, a new router-LSA is originated whenever a change occurs that causes any of the following three conditions to occur:

o There exists a MANET neighbor j that satisfies the above conditions for inclusion in the router-LSA, but is not included in the current router-LSA.

o The current router-LSA includes a MANET neighbor that is no longer bidirectional.

o The link metric has changed for a MANET neighbor that is included in the current router-LSA.

The above conditions may be checked periodically just before sending each Hello, instead of checking them every time one of the neighbor sets changes. The following implementation was found to work well. Just before sending each Hello, and whenever a bidirectional neighbor transitions to less than 2-Way, the router runs the MDR selection algorithm; updates its adjacencies, routable neighbors, and Selected Advertised Neighbors; then checks the above conditions to see if a new router-LSA should be originated. In addition, if a neighbor becomes bidirectional or Full, the router updates its routable neighbors and checks the above conditions.

10. Calculating the Routing Table

The routing table calculation is the same as specified in [RFC2328], except for the following changes to Section 16.1 (Calculating the shortest-path tree for an area). If full-topology adjacencies and full-topology LSAs are used (AdjConnectivity = 0 and LSAFullness = 4), there is no change to Section 16.1.

If adjacency reduction is used (AdjConnectivity is 1 or 2), then Section 16.1 is modified as follows. Recall from Section 9 that a router can use any routable neighbor as a next hop to a destination, whether or not the neighbor is advertised in the router-LSA. This is accomplished by modifying Step 2 so that the router-LSA associated with the root vertex is replaced with a dummy router-LSA that includes links to all Full neighbors and all routable MANET neighbors. In addition, Step 2b (checking for a link from W back to V) MUST be skipped when V is the root vertex and W is a routable MANET neighbor. However, Step 2b must still be executed when V is not the root vertex, to ensure compatibility with OSPFv3.

If LSAFullness is 0 (minimal LSAs), then the calculated paths need not be shortest paths. In this case, the path actually taken by a packet can be shorter than the calculated path, since intermediate routers may have routable neighbors that are not advertised in any router-LSA.

If full-topology adjacencies and partial-topology LSAs are used, then Section 16.1 is modified as follows. Step 2 is modified so that the router-LSA associated with the root vertex is replaced with a dummy router-LSA that includes links to all Full neighbors. In addition, Step 2b MUST be skipped when V is the root vertex and W is a Full MANET neighbor. (This is necessary since the neighbor’s router-LSA need not contain a link back to the router.)

If adjacency reduction is used with partial-topology LSAs, then the set of routable neighbors can change without causing the contents of the router-LSA to change. This could happen, for example, if a routable neighbor that was not included in the router-LSA transitions to the Down or Init state. Therefore, if the set of routable neighbors changes, the shortest-path tree must be recalculated, even if the router-LSA does not change.

After the shortest-path tree and routing table are calculated, the set of routable neighbors must be updated, since a route to a non-routable neighbor may have been discovered. If the set of routable neighbors changes, then the shortest-path tree and routing table must be calculated a second time. The second calculation will not change the set of routable neighbors again, so two calculations are sufficient. If the set of routable neighbors is updated periodically every HelloInterval seconds, then it is not necessary to update the set of routable neighbors immediately after the routing table is updated.

11. Security Considerations

As with OSPFv3 [RFC5340], OSPF-MDR can use the IPv6 Authentication Header (AH) [RFC4302] and/or the IPv6 Encapsulation Security Payload (ESP) [RFC4303] to provide authentication, integrity, and/or confidentiality. The use of AH and ESP for OSPFv3 is described in [RFC4552].

Generic threats to routing protocols are described and categorized in [RFC4593]. The mechanisms described in [RFC4552] provide protection against many of these threats, but not all of them. In particular, as mentioned in [RFC5340], these mechanisms do not provide protection against compromised, malfunctioning, or misconfigured routers (also called Byzantine routers); this is true for both OSPFv3 and OSPF-MDR.

The extension of OSPFv3 to include MANET routers does not introduce any new security threats. However, the use of a wireless medium and lack of infrastructure, inherent with MANET routers, may render some of the attacks described in [RFC4593] easier to mount. Depending on the network context, these increased vulnerabilities may increase the need to provide authentication, integrity, and/or confidentiality, as well as anti-replay service.

For example, sniffing of routing information and traffic analysis are easier tasks with wireless routers than with wired routers, since the attacker only needs to be within the radio range of a router. The use of confidentiality (encryption) provides protection against sniffing but not traffic analysis.

Similarly, interference attacks are also easier to mount against MANET routers due to their wireless nature. Such attacks can be mounted even if OSPF packets are protected by authentication and confidentiality, e.g., by transmitting noise or replaying outdated OSPF packets. As discussed below, an anti-replay service (provided by both ESP and AH) can be used to protect against the latter attack.

The following threat actions are also easier with MANET routers: spoofing (assuming the identify of a legitimate router), falsification (sending false routing information), and overloading (sending or triggering an excessive amount of routing updates). These attacks are only possible if authentication is not used, or the attacker takes control of a router or is able to forge legitimacy (e.g., by discovering the cryptographic key).

[RFC4552] mandates the use of manual keying when current IPsec protocols are used with OSPFv3. Routers are required to use manually configured keys with the same security association (SA) parameters for both inbound and outbound traffic. For MANET routers, this implies that all routers attached to the same MANET must use the same key for multicasting packets. This is required in order to achieve scalability and feasibility, as explained in [RFC4552]. Future specifications can explore the use of automated key management protocols that may be suitable for MANETs.

As discussed in [RFC4552], the use of manual keys can increase vulnerability. For example, manual keys are usually long lived, thus giving an attacker more time to discover the keys. In addition, the use of the same key on all routers attached to the same MANET leaves all routers insecure against impersonation attacks if any one of the routers is compromised.

Although [RFC4302] and [RFC4303] state that implementations of AH and ESP SHOULD NOT provide anti-replay service in conjunction with SAs that are manually keyed, it is important to note that such service is allowed if the sequence number counter at the sender is correctly maintained across local reboots until the key is replaced. Therefore, it may be possible for MANET routers to make use of the anti-replay service provided by AH and ESP.

When an OSPF routing domain includes both MANET networks and fixed networks, the frequency of OSPF updates either due to actual topology changes or malfeasance could result in instability in the fixed networks. In situations where this is a concern, it is recommended that the border routers segregate the MANET networks from the fixed networks with either separate OSPF areas or, in cases where legacy routers are very sensitive to OSPF update frequency, separate OSPF instances. With separate OSPF areas, the 5-second MinLSInterval will dampen the frequency of changes originated in the MANET networks. Additionally, OSPF ranges can be configured to aggregate prefixes for the areas supporting MANET networks. With separate OSPF instances, more conservative local policies can be employed to limit the volume of updates emanating from the MANET networks.

12. IANA Considerations

This document defines three new LLS TLV types: MDR-Hello TLV (14), MDR-Metric TLV (16), and MDR-DD TLV (15) (see Section A.2).

13. Acknowledgments

Thanks to Aniket Desai for helpful discussions and comments, including the suggestion that Router Priority should come before MDR Level in the lexicographical comparison of (RtrPri, MDR Level, RID) when selecting MDRs and BMDRs, and that the MDR calculation should be repeated if it causes the MDR Level to change. Thanks also to Tom Henderson, Acee Lindem, and Emmanuel Baccelli for helpful discussions and comments.

14. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, December 2005.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC4552] Gupta, M. and N. Melam, "Authentication/Confidentiality for OSPFv3", RFC 4552, June 2006.

[RFC5243] Ogier, R., "OSPF Database Exchange Summary List Optimization", RFC 5243, May 2008.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, July 2008.

[RFC5613] Zinin, A., Roy, A., Nguyen, L., Friedman, B., and D. Yeung, "OSPF Link-Local Signaling", RFC 5613, August 2009.

15. Informative References

[Lawler] Lawler, E., "Combinatorial Optimization: Networks and Matroids", Holt, Rinehart, and Winston, New York, 1976.

[Suurballe] Suurballe, J.W. and R.E. Tarjan, "A Quick Method for Finding Shortest Pairs of Disjoint Paths", Networks, Vol. 14, pp. 325-336, 1984.

[RFC4593] Barbir, A., Murphy, S., and Y. Yang, "Generic Threats to Routing Protocols", RFC 4593, October 2006.

Appendix A. Packet Formats

A.1. Options Field

The L bit of the OSPF options field is used for link-local signaling, as described in [RFC5613]. Routers set the L bit in Hello and DD packets to indicate that the packet contains an LLS data block. Routers set the L bit in a self-originated router-LSA to indicate that the LSA is non-ackable.

A.2. Link-Local Signaling

OSPF-MDR uses link-local signaling [RFC5613] to append the MDR-Hello TLV and MDR-Metric TLV to Hello packets, and to append the MDR-DD TLV to Database Description packets. Link-local signaling is an extension of OSPFv2 and OSPFv3 that allows the exchange of arbitrary data using existing OSPF packet types. Here we use LLS for OSPFv3, which is accomplished by adding an LLS data block at the end of the OSPFv3 packet. The OSPF packet length field does not include the length of the LLS data block, but the IPv6 packet length does include this length.

A.2.1. LLS Data Block

The data block used for link-local signaling is formatted as described below in Figure A.1.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |            Checksum           |       LLS Data Length         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                           LLS TLVs                            |
    .                                                               .
    .                                                               .
    .                                                               .
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure A.1: Format of LLS Data Block

The Checksum field contains the standard IP checksum of the entire contents of the LLS block.

The 16-bit LLS Data Length field contains the length (in 32-bit words) of the LLS block including the header and payload. Implementations should not use the Length field in the IPv6 packet header to determine the length of the LLS data block.

The rest of the block contains a set of Type/Length/Value (TLV) triplets as described in the following section. All TLVs must be 32-bit aligned (with padding if necessary).

A.2.2. LLS TLV Format

The contents of the LLS data block are constructed using TLVs. See Figure A.2 for the TLV format.

The Type field contains the TLV ID, which is unique for each type of TLV. The Length field contains the length of the Value field (in bytes) that is variable and contains arbitrary data.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |            Type               |           Length              |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    .                                                               .
    .                             Value                             .
    .                                                               .
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure A.2: Format of LLS TLVs

Note that TLVs are always padded to a 32-bit boundary, but padding bytes are not included in the TLV Length field (though they are included in the LLS Data Length field of the LLS block header). All unknown TLVs MUST be silently ignored.

A.2.3. MDR-Hello TLV

The MDR-Hello TLV is appended to each MANET Hello using LLS. It includes the current Hello sequence number (HSN) for the transmitting interface and the number of neighbors of each type that are listed in the body of the Hello (see Section 4.1). It also indicates whether the Hello is differential (via the D-bit), and whether the router is using full-topology adjacencies (via the A-bit).

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |            Type               |           Length              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |    Hello Sequence Number      |          Reserved         |A|D|
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |      N1       |      N2       |      N3       |      N4       |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Type: Set to 14.

- Length: Set to 8.

- Hello Sequence Number: A circular two-octet unsigned integer indicating the current HSN for the transmitting interface. The HSN for the interface is incremented by 1 (modulo 2^16) every time a (differential or full) Hello is sent on the interface.

- Reserved: Set to 0. Reserved for future use.

- A (1 bit): Set to 1 if AdjConnectivity is 0; otherwise, set to 0.

- D (1 bit): Set to 1 for a differential Hello and 0 for a full Hello.

- N1 (8 bits): The number of neighbors listed in the Hello that are in state Down. N1 is zero if the Hello is not differential.

- N2 (8 bits): The number of neighbors listed in the Hello that are in state Init.

- N3 (8 bits): The number of neighbors listed in the Hello that are Dependent.

- N4 (8 bits): The number of neighbors listed in the Hello that are Selected Advertised Neighbors.

A.2.4. MDR-DD TLV

When a Database Description packet is sent to a neighbor in state ExStart, an MDR-DD TLV is appended to the packet using LLS. It includes the same two Router IDs that are included in the DR and Backup DR fields of a Hello sent by the router, and is used to indicate the router’s MDR Level and Parent(s).

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |             Type              |           Length              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                               DR                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                            Backup DR                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Type: Set to 15.

- Length: Set to 8.

- DR: The same Router ID that is included in the DR field of a Hello sent by the router (see Section A.3).

- Backup DR: The same Router ID that is included in the Backup DR field of a Hello sent by the router (see Section A.3).

A.2.5. MDR-Metric TLV

If LSAFullness is 1 or 2, an MDR-Metric TLV must be appended to each MANET Hello packet using LLS, unless all link metrics are 1. This TLV advertises the link metric for each bidirectional neighbor listed in the body of the Hello. At a minimum, this TLV advertises a single default metric. If the I bit is set, the Router ID and link metric are included for each bidirectional neighbor listed in the body of the Hello whose link metric is not equal to the default metric. This option reduces overhead when all neighbors have the same link metric, or only a few neighbors have a link metric that differs from the default metric. If the I bit is zero, the link metric is included for each bidirectional neighbor that is listed in the body of the Hello and the neighbor RIDs are omitted from the TLV.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |             Type              |             Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |       Default Metric          |          Reserved           |I|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                         Neighbor ID (1)                       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                         Neighbor ID (2)                       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                              ...                              |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          Metric (1)           |          Metric (2)           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |           ...
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Type: Set to 16.

- Length: Set to 4 + 6*N if the I bit is 1, and to 4 + 2*N if the I bit is 0, where N is the number of neighbors included in the TLV.

- Default Metric: If the I bit is 1, this is the link metric that applies to every bidirectional neighbor listed in the body of the Hello whose RID is not listed in the Metric TLV.

- Neighbor ID: If the I bit is 1, the RID is listed for each bidirectional neighbor (Lists 3 through 5 as defined in Section 4.1) in the body of the Hello whose link metric is not equal to the default metric. Omitted if the I bit is 0.

- Metric: Link metric for each bidirectional neighbor, listed in the same order as the Neighbor IDs in the TLV if the I bit is 1, and in the same order as the Neighbor IDs of bidirectional neighbors (Lists 3 through 5 as defined in Section 4.1) in the body of the Hello if the I bit is 0.

A.3. Hello Packet DR and Backup DR Fields

The Designated Router (DR) and Backup DR fields of a Hello packet are set as follows:

- o DR: This field is the router’s Parent, or is 0.0.0.0 if the Parent is null. The Parent of an MDR is always the router’s own RID.

- o Backup DR: This field is the router’s Backup Parent, or is 0.0.0.0 if the Backup Parent is null. The Backup Parent of a BMDR is always the router’s own RID.

A.4. LSA Formats and Examples

LSA formats are specified in [RFC5340], Section 4.4. Figure A.3 below gives an example network map for a MANET in a single area.

- o Four MANET routers RT1, RT2, RT3, and RT4 are in area 1.

- o RT1’s MANET interface has links to RT2 and RT3’s MANET interfaces.

- o RT2’s MANET interface has links to RT1 and RT3’s MANET interfaces.

- o RT3’s MANET interface has links to RT1, RT2, and RT3’s MANET interfaces.

- o RT4’s MANET interface has a link to RT3’s MANET interface.

- o RT1 and RT2 have stub networks attached on broadcast interfaces.

- o RT3 has a transit network attached on a broadcast interface.

```
       ..........................................
       .                                  Area 1.
       .     +                                  .
       .     |                                  .
       .     |  2+---+1                      1+---+
       .  N1 |---|RT1|----+               +---|RT4|----
       .     |   +---+    |\             /    +---+
       .     |            | \           /       .
       .     +            |  \   N3    /        .
       .                  |   \       /         .
       .     +            |    \     /          .
       .     |            |     \   /           .
       .     |  2+---+1   |      \ /            .
       .  N2 |---|RT2|----+-------+             .
       .     |   +---+            |1            .
       .     |                  +---+           .
       .     |                  |RT3|----------------
       .     +                  +---+           .
       .                          |2            .
       .                   +------------+       .
       .                      |1   N4           .
       .                    +---+               .
       .                    |RT5|               .
       .                    +---+               .
       ..........................................
```

Figure A.3: Area 1 with IP Addresses Shown

Network	IPv6 prefix
N1	5f00:0000:c001:0200::/56
N2	5f00:0000:c001:0300::/56
N4	5f00:0000:c001:0400::/56

Table 1: IPv6 link prefixes for sample network

Router	interface	Interface ID	IPv6 global unicast prefix
RT1	LOOPBACK	0	5f00:0001::/64
	to N3	1	n/a
	to N1	2	5f00:0000:c001:0200::RT1/56
RT2	LOOPBACK	0	5f00:0002::/64
	to N3	1	n/a
	to N2	2	5f00:0000:c001:0300::RT2/56
RT3	LOOPBACK	0	5f00:0003::/64
	to N3	1	n/a
	to N4	2	5f00:0000:c001:0400::RT3/56
RT4	LOOPBACK	0	5f00:0004::/64
	to N3	1	n/a
RT5	to N4	1	5f00:0000:c001:0400::RT5/56

Table 2: IPv6 link prefixes for sample network

Router	interface	Interface ID	link-local address
RT1	LOOPBACK	0	n/a
	to N1	1	fe80:0001::RT1
	to N3	2	fe80:0002::RT1
RT2	LOOPBACK	0	n/a
	to N2	1	fe80:0001::RT2
	to N3	2	fe80:0002::RT2
RT3	LOOPBACK	0	n/a
	to N3	1	fe80:0001::RT3
	to N4	2	fe80:0002::RT3
RT4	LOOPBACK	0	n/a
	to N3	1	fe80:0001::RT4
RT5	to N4	1	fe80:0002::RT5

Table 3: OSPF interface IDs and link-local addresses

A.4.1. Router-LSAs

As an example, consider the router-LSA that node RT3 would originate. The node consists of one MANET, one broadcast, and one loopback interface.

RT3’s router-LSA

```
LS age = DoNotAge+0               ;newly originated
LS type = 0x2001                  ;router-LSA
Link State ID = 0                 ;first fragment
Advertising Router = 192.1.1.3    ;RT3’s Router ID
bit E = 0                         ;not an AS boundary router
bit B = 1                         ;area border router
Options = (V6-bit|E-bit|R-bit)
  Type = 1                         ;p2p link to RT1
  Metric = 1                       ;cost to RT1
  Interface ID = 1                 ;Interface ID
  Neighbor Interface ID = 1        ;Interface ID
  Neighbor Router ID = 192.1.1.1   ;RT1’s Router ID
  Type = 1                         ;p2p link to RT2
  Metric = 1                       ;cost to RT2
  Interface ID = 1                 ;Interface ID
  Neighbor Interface ID = 1        ;Interface ID
  Neighbor Router ID = 192.1.1.2   ;RT2’s Router ID
  Type = 1                         ;p2p link to RT4
  Metric = 1                       ;cost to RT4
  Interface ID = 1                 ;Interface ID
  Neighbor Interface ID = 1        ;Interface ID
  Neighbor Router ID = 192.1.1.4   ;RT4’s Router ID
  Type = 2                         ;connects to N4
  Metric = 1                       ;cost to N4
  Interface ID = 2                 ;RT3’s Interface ID
  Neighbor Interface ID = 1        ;RT5’s Interface ID (elected DR)
  Neighbor Router ID = 192.1.1.5   ;RT5’s Router ID  (elected DR)
```

A.4.2. Link-LSAs

Consider the link-LSA that RT3 would originate for its MANET interface.

RT3’s link-LSA for its MANET interface

```
LS age = DoNotAge+0              ;newly originated
LS type = 0x0008                 ;Link-LSA
Link State ID = 1                ;Interface ID
Advertising Router = 192.1.1.3   ;RT3’s Router ID
RtrPri = 1                       ;default priority
Options = (V6-bit|E-bit|R-bit)
Link-local Interface Address = fe80:0001::RT3
# prefixes = 0                   ;no global unicast address
```

A.4.3. Intra-Area-Prefix-LSAs

A MANET node originates an intra-area-prefix-LSA to advertise its own prefixes, and those of its attached networks or stub links. As an example, consider the intra-area-prefix-LSA that RT3 will build.

RT2’s intra-area-prefix-LSA for its own prefixes

```
LS age = DoNotAge+0              ;newly originated
LS type = 0x2009                 ;intra-area-prefix-LSA
Link State ID = 177              ;or something
Advertising Router = 192.1.1.3   ;RT3’s Router ID
# prefixes = 2
Referenced LS type = 0x2001      ;router-LSA reference
Referenced Link State ID = 0     ;always 0 for router-LSA reference
Referenced Advertising Router = 192.1.1.3 ;RT2’s Router ID
  PrefixLength = 64                ;prefix on RT3’s LOOPBACK
  PrefixOptions = 0
  Metric = 0                       ;cost of RT3’s LOOPBACK
  Address Prefix = 5f00:0003::/64
  PrefixLength = 56                ;prefix on RT3’s interface 2
  PrefixOptions = 0
  Metric = 1                       ;cost of RT3’s interface 2
  Address Prefix = 5f00:0000:c001:0400::RT3/56    ;pad
```

Appendix B. Detailed Algorithms for MDR/BMDR Selection

This section provides detailed algorithms for Step 2.4 of Phase 2 (MDR selection) and Step 3.2 of Phase 3 (BMDR selection) of the MDR selection algorithm described in Section 5. Step 2.4 uses a breadth-first search (BFS) algorithm, and Step 3.2 uses an efficient algorithm for finding pairs of node-disjoint paths from Rmax to all other neighbors. Both algorithms run in O(d^2) time, where d is the number of neighbors.

For convenience, in the following description, the term "bi-neighbor" will be used as an abbreviation for "bidirectional neighbor". Also, node i denotes the router performing the calculation.

B.1. Detailed Algorithm for Step 2.4 (MDR Selection)

The following algorithm performs Step 2.4 of the MDR selection algorithm, and assumes that Phase 1 and Steps 2.1 through 2.3 have been performed, so that the neighbor connectivity matrix NCM has been computed and Rmax is the bi-neighbor with the (lexicographically) largest value of (RtrPri, MDR Level, RID). The BFS algorithm uses a FIFO queue so that all nodes 1 hop from node Rmax are processed first, then 2 hops, etc. When the BFS algorithm terminates, hops(u), for each bi-neighbor node u of node i, will be equal to the minimum number of hops from node Rmax to node u, using only intermediate nodes that are bi-neighbors of node i and that have a larger value of (RtrPri, MDR Level, RID) than node i. The algorithm also computes, for each node u, the tree parent p(u) and the second node r(u) on the tree path from Rmax to u, which will be used in Step 3.2.

(a) Compute a matrix of link costs c(u,v) for each pair of bi-neighbors u and v as follows: If node u has a larger value of (RtrPri, MDR Level, RID) than node i, and NCM(u,v) = 1, then set c(u,v) to 1. Otherwise, set c(u,v) to infinity. (Note that the matrix NCM(u,v) is symmetric, but the matrix c(u,v) is not.)

(b) Set hops(u) = infinity for all bi-neighbors u other than Rmax, and set hops(Rmax) = 0. Initially, p(u) is undefined for each neighbor u. For each bi-neighbor u such that c(Rmax,u) = 1, set r(u) = u; for all other u, r(u) is initially undefined. Add node Rmax to the FIFO queue.

(c) While the FIFO queue is nonempty: Remove the node at the head of the queue; call it node u. For each bi-neighbor v of node i such that c(u,v) = 1:

```
  If hops(v) > hops(u) + 1, then set hops(v) = hops(u) + 1, set
  p(v) = u, set r(v) = r(u) if hops(v) > 1, and add node v to
  the tail of the queue.
```

B.2. Detailed Algorithm for Step 3.2 (BMDR Selection)

Step 3.2 of the MDR selection algorithm requires the router to determine whether there exist two node-disjoint paths from Rmax to each other bi-neighbor u, via bi-neighbors that have a larger value of (RtrPri, MDR Level, RID) than the router itself. This information is needed to determine whether the router should select itself as a BMDR.

It is possible to determine separately for each bi-neighbor u whether there exist two node-disjoint paths from Rmax to u, using the well-known augmenting path algorithm [Lawler] that runs in O(n^2) time, but this must be done for all bi-neighbors u, thus requiring a total run time of O(n^3). The algorithm described below makes the same determination simultaneously for all bi-neighbors u, achieving a much faster total run time of O(n^2). The algorithm is a simplified variation of the Suurballe-Tarjan algorithm [Suurballe] for finding pairs of disjoint paths.

The algorithm described below uses the following output of Phase 2: the tree parent p(u) of each node (which defines the BFS tree computed in Phase 2), and the second node r(u) on the tree path from Rmax to u.

The algorithm uses the following concepts. For any node u on the BFS tree other than Rmax, we define g(u) to be the first labeled node on the reverse tree path from u to Rmax, if such a labeled node exists other than Rmax. (The reverse tree path consists of u, p(u), p(p(u)), ..., Rmax.) If no such labeled node exists, then g(u) is defined to be r(u). In particular, if u is labeled then g(u) = u. Note that g(u) either must be labeled or must be a neighbor of Rmax.

For any node k that either is labeled or is a neighbor of Rmax, we define the unlabeled subtree rooted at k, denoted S(k), to be the set of nodes u such that g(u) = k. Thus, S(k) includes node k itself and the set of unlabeled nodes downstream of k on the BFS tree that can be reached without going through any labeled nodes. This set can be obtained in linear time using a depth-first search starting at node k, and using labeled nodes to indicate the boundaries of the search. Note that g(u) and S(k) are not maintained as variables in the algorithm given below, but simply refer to the definitions given above.

The BMDR algorithm maintains a set B, which is initially empty. A node u is added to B when it is known that two node-disjoint paths exist from Rmax to u via nodes that have a larger value of (RtrPri, MDR Level, RID) than the router itself. When the algorithm terminates, B consists of all nodes that have this property.

The algorithm consists of the following two steps.

(a) Mark Rmax as labeled. For each pair of nodes u, v on the BFS tree other than Rmax such that r(u) is not equal to r(v) (i.e., u and v have different second nodes), NCM(u,v) = 1, and node u has a greater value of (RtrPri, MDR level, RID) than the router itself, add v to B. (Clearly there are two disjoint paths from Rmax to v.)

(b) While there exists a node in B that is not labeled, do the following. Choose any node k in B that is not labeled, and let j = g(k). Now mark k as labeled. (This creates a new unlabeled subtree S(k), and makes S(j) smaller by removing S(k) from it.) For each pair of nodes u, v such that u is in S(k), v is in S(j), and NCM(u,v) = 1:

- If u has a larger value of (RtrPri, MDR level, RID) than the router itself, and v is not in B, then add v to B.

- If v has a larger value of (RtrPri, MDR level, RID) than the router itself, and u is not in B, then add u to B.

A simplified version of the algorithm MAY be performed by omitting step (b). However, the simplified algorithm will result in more BMDRs, and is not recommended if AdjConnectivity = 2 since it will result in more adjacencies.

The above algorithm can be executed in O(n^2) time, where n is the number of neighbors. Step (a) clearly requires O(n^2) time since it considers all pairs of nodes u and v. Step (b) also requires O(n^2) time because each pair of nodes is considered at most once. This is because labeling nodes divides unlabeled subtrees into smaller unlabeled subtrees, and a given pair u, v is considered only the first time u and v belong to different unlabeled subtrees.

Appendix C. Min-Cost LSA Algorithm

This section describes the algorithm for determining which MANET neighbors to include in the router-LSA when LSAFullness is 1. The min-cost LSA algorithm ensures that the link-state database provides sufficient information to calculate at least one shortest (minimum-cost) path to each destination. The algorithm assumes that a router may have multiple interfaces, at least one of which is a MANET interface. The algorithm becomes significantly simpler if the router has only a single (MANET) interface.

The input to this algorithm includes information obtained from Hellos received from each neighbor on each MANET interface, including the neighbor’s Bidirectional Neighbor Set (BNS), Dependent Neighbor Set (DNS), Selected Advertised Neighbor Set (SANS), and link metrics. The input also includes the link-state database if the router has a non-MANET interface.

The output of the algorithm is the router’s SANS for each MANET interface. The SANS is used to construct the router-LSA as described in Section 9.4. The min-cost LSA algorithm must be run to update the SANS (and possibly originate a new router-LSA) either periodically just before sending each Hello, or whenever any of the following events occurs:

o The state or routability of a neighbor changes.

o A Hello received from a neighbor indicates a change in its MDR Level, Router Priority, FullHelloRcvd, BNS, DNS, SANS, Parent(s), or link metrics.

o An LSA originated by a non-MANET neighbor is received.

Although the algorithm described below runs in O(d^3) time, where d is the number of neighbors, an incremental version for a single topology change runs in O(d^2) time, as discussed following the algorithm description.

For convenience, in the following description, the term "bi-neighbor" will be used as an abbreviation for "bidirectional neighbor". Also, router i will denote the router doing the calculation. To perform the min-cost LSA algorithm, the following steps are performed.

(1) Create the neighbor connectivity matrix (NCM) for each MANET interface, as described in Section 5.1. Create the multiple-interface neighbor connectivity matrix MNCM as follows. For each bi-neighbor j, set MNCM(i,j) = MNCM(j,i) = 1. For each pair j, k of MANET bi-neighbors, set MNCM(j,k) = 1 if NCM(j,k) equals 1 for

any MANET interface. For each pair j, k of non-MANET bi-neighbors, set MNCM(j,k) = 1 if the link-state database indicates that a direct link exists between j and k. Otherwise, set MNCM(j,k) = 0. (Note that a given router can be a neighbor on both a MANET interface and a non-MANET interface.)

(2) Create the inter-neighbor cost matrix (COST) as follows. For each pair j, k of routers such that each of j and k is a bi-neighbor or router i itself:

(a) If MNCM(j,k) = 1, set COST(j,k) to the metric of the link from j to k obtained from j’s Hellos (for a MANET interface), or from the link-state database (for a non-MANET interface). If there are multiple links from j to k (via multiple interfaces), COST(j,k) is set to the minimum cost of these links.

(b) Otherwise, set COST(j,k) to LSInfinity.

(3) Create the backbone neighbor matrix (BNM) as follows. BNM indicates which pairs of MANET bi-neighbors are backbone neighbors of each other, as defined in Section 9.2.1. If adjacency reduction is not used (AdjConnectivity = 0), set all entries of BNM to zero and proceed to Step 4.

In the following, if a link exists from router j to router k on more than one interface, we consider only interfaces for which the cost from j to k equals COST(j,k); such interfaces will be called "candidate" interfaces.

For each pair j, k of MANET bi-neighbors, BNM(j,k) is set to 1 if j and k are backbone neighbors of each other on a candidate MANET interface. That is, BNM(j,k) is set to 1 if, for any candidate MANET interface, NCM(j,k) = 1 and either of the following conditions is satisfied:

(a) Router k is included in j’s DNS or router j is included in k’s DNS.

(b) Router j is the (Backup) Parent of router k or router k is the (Backup) Parent of router j.

Otherwise, BNM(j,k) is set to 0.

(4) Create the Selected Advertised Neighbor Matrix (SANM) as follows. For each pair j, k of routers such that each of j and k is a bi-neighbor or router i itself, SANM(j,k) is set to 1 if, for any

candidate MANET interface, NCM(j,k) = 1 and k is included in j’s SANS. Otherwise, SANM(j,k) is set to 0. Note that SANM(i,k) is set to 1 if k is currently a Selected Advertised Neighbor.

(5) Compute the new set of Selected Advertised Neighbors as follows. For each MANET bi-neighbor j, initialize the bit variable new_sel_adv(j) to 0. (This bit will be set to 1 if j is selected.) For each MANET bi-neighbor j:

(a) If j is a bi-neighbor on more than one interface, consider only candidate interfaces (for which the cost to j is minimum). If one of the candidate interfaces is a non-MANET interface, examine the next neighbor (j is not selected since it will be advertised anyway).

(b) If adjacency reduction is used, and one of the candidate interfaces is a MANET interface on which j is a backbone neighbor (see Section 9.2), examine the next neighbor (j is not selected since it will be advertised anyway).

(c) Otherwise, if there is more than one candidate MANET interface, select the "preferred" interface by using the following preference rules in the given order: an interface is preferred if (1) router i’s SANS for that interface already includes j, (2) router i’s Router Priority is larger on that interface, and (3) router i’s MDR Level is larger on that interface.

(d) For each bi-neighbor k (on any interface) such that COST(k,j) > COST(k,i) + COST(i,j), determine whether there exists another bi-neighbor u such that either COST(k,u) + COST(u,j) < COST(k,i) + COST(i,j), or COST(k,u) + COST(u,j) = COST(k,i) + COST(i,j) and either of the following conditions is true:

- BNM(u,j) = 1, or

- (SANM(j,u), SANM(u,j), RtrPri(u), RID(u)) is lexicographically greater than (SANM(j,i), SANM(i,j), RtrPri(i), RID(i)).

If for some such bi-neighbor k, there does not exist such a bi-neighbor u, then set new_sel_adv(j) = 1.

(6) For each MANET interface I, update the SANS to equal the set of all bi-neighbors j such that new_sel_adv(j) = 1 and I is the preferred interface for j.

(7) With the SANS updated, a new router-LSA may need to be originated as described in Section 9.4.

The lexicographical comparison of Step 5d gives preference to links that are already advertised, in order to improve LSA stability.

The above algorithm can be run in O(d^2) time if a single link change occurs. For example, if link (x,y) fails where x and y are neighbors of router i, and either SANS(x,y) = 1 or BNM(x,y) = 1, then Step 5 need only be performed for pairs j, k such that either j or k is equal to x or y.

Appendix D. Non-Ackable LSAs for Periodic Flooding

In a highly mobile network, it is possible that a router almost always originates a new router-LSA every MinLSInterval seconds. In this case, it should not be necessary to send Acks for such an LSA, or to retransmit such an LSA as a unicast, or to describe such an LSA in a DD packet. In this case, the originator of an LSA MAY indicate that the router-LSA is "non-ackable" by setting the L bit in the options field of the LSA (see Section A.1). For example, a router can originate non-ackable LSAs if it determines (e.g., based on an exponential moving average) that a new LSA is originated every MinLSInterval seconds at least 90 percent of the time. (Simulations can be used to determine the best threshold.)

A non-ackable LSA is never acknowledged, nor is it ever retransmitted as a unicast or described in a DD packet, thus saving substantial overhead. However, the originating router must periodically retransmit the current instance of its router-LSA as a multicast (until it originates a new LSA, which will usually happen before the previous instance is retransmitted), and each MDR must periodically retransmit each non-ackable LSA as a multicast (until it receives a new instance of the LSA, which will usually happen before the previous instance is retransmitted). For this option to work, RxmtInterval must be larger than MinLSInterval so that a new instance of the LSA is usually received before the previous one is retransmitted. Note that the reception of a retransmitted (duplicate) LSA does not result in immediate forwarding of the LSA; only a new LSA (with a larger sequence number) may be forwarded immediately, according to the flooding procedure of Section 8.

Appendix E. Simulation Results

This section presents simulation results that predict the performance of OSPF-MDR for up to 160 nodes with min-cost LSAs and up to 200 nodes with minimal LSAs. The results were obtained using the GTNetS simulator with OSPF-MDR version 1.01, available at http://hipserver.mct.phantomworks.org/ietf/ospf.

The following scenario parameter values were used: radio range = 200 m and 250 m, grid length = 500 m, wireless alpha = 0.5, (maximum) velocity = 10 m/s, pause time = 0, packet rate = 10 pkts/s, packet size = 40 bytes, random seed = 8, start time (for gathering statistics) = 1800 s. The stop time was 3600 s for up to 80 nodes and 2700 s for more than 80 nodes. The source and destination are selected randomly for each generated UDP packet. The simulated MAC protocol is 802.11b.

Tables 4 and 6 show the results for the default configuration of OSPF-MDR, except that differential Hellos were used (2HopRefresh = 3) since they are recommended when the number of neighbors is large. Tables 5 and 7 show the results for the same configuration except that minimal LSAs were used instead of min-cost LSAs. The tables show the results for total OSPF overhead in kb/s, the total number of OSPF packets per second, the delivery ratio for UDP packets, and the average number of hops traveled by UDP packets that reach their destination.

Tables 5 and 7 for minimal LSAs also show the following statistics: the average number of bidirectional neighbors per node, the average number of fully adjacent neighbors per node, the number of changes in the set of bidirectional neighbors per node per second, and the number of changes in the set of fully adjacent neighbors per node per second. These statistics do not change significantly when min-cost LSAs are used instead of minimal LSAs.

The results show that OSPF-MDR achieves good performance for up to at least 160 nodes when min-cost LSAs are used, and up to at least 200 nodes when minimal LSAs are used. Also, the results for the number of hops show that the routes obtained with minimal LSAs are only 2.3% to 4.5% longer than with min-cost LSAs when the range is 250 m, and 3.5% to 7.4% longer when the range is 200 m.

The results also show that the number of adjacencies per node and the number of adjacency changes per node per second do not increase as the number of nodes increases, and are dramatically smaller than the number of neighbors per node and the number of neighbor changes per node per second, respectively. These factors contribute to the low overhead achieved by OSPF-MDR. For example, the results in Table 5 imply that with 200 nodes and range 250 m, there are 2.136/.039 = 55 times as many adjacency formations with full-topology adjacencies as with uniconnected adjacencies. Additional simulation results for OSPF-MDR can be found at http://www.manet-routing.org.

	Number of nodes						
	20	40	60	80	100	120	160
OSPF kb/s	27.1	74.2	175.3	248.6	354.6	479.2	795.7
OSPF pkts/s	29.9	69.2	122.9	163.7	210.3	257.2	357.7
Delivery ratio	.970	.968	.954	.958	.957	.956	.953
Avg no. hops	1.433	1.348	1.389	1.368	1.411	1.361	1.386

Table 4: Results for range 250 m with min-cost LSAs

	Number of nodes						
	20	40	60	80	120	160	200
OSPF kb/s	15.5	41.6	91.0	132.9	246.3	419.0	637.4
OSPF pkts/sec	18.8	42.5	78.6	102.8	166.8	245.6	321.0
Delivery ratio	.968	.968	.951	.953	.962	.956	.951
Avg no. hops	1.466	1.387	1.433	1.412	1.407	1.430	1.411
Avg no. nbrs/node	11.38	25.82	36.30	50.13	75.87	98.65	125.59
Avg no. adjs/node	2.60	2.32	2.38	2.26	2.25	2.32	2.13
Nbr changes/node/s	.173	.372	.575	.752	1.223	1.654	2.136
Adj changes/node/s	.035	.036	.046	.040	.032	.035	.039

Table 5: Results for range 250 m with minimal LSAs

	Number of nodes						
	20	40	60	80	100	120	160
OSPF kb/s	40.5	123.4	286.5	415.7	597.5	788.9	1309.8
OSPF pkts/s	37.6	83.9	135.1	168.6	205.4	247.7	352.3
Delivery ratio	.926	.919	.897	.900	.898	.895	.892
Avg no. hops	1.790	1.628	1.666	1.632	1.683	1.608	1.641

Table 6: Results for range 200 m with min-cost LSAs

	Number of nodes						
	20	40	60	80	120	160	200
OSPF kb/s	24.0	63.6	140.6	195.2	346.9	573.2	824.6
OSPF pkts/sec	26.4	58.8	108.3	138.8	215.2	311.3	401.3
Delivery ratio	.930	.927	.897	.907	.907	.904	.902
Avg no. hops	1.853	1.714	1.771	1.743	1.727	1.758	1.747
Avg no. nbrs/node	7.64	18.12	25.27	35.29	52.99	68.13	86.74
Avg no. adjs/node	2.78	2.60	2.70	2.50	2.39	2.36	2.24
Nbr changes/node/s	.199	.482	.702	.959	1.525	2.017	2.611
Adj changes/node/s	.068	.069	.081	.068	.055	.058	.057

Table 7: Results for range 200 m with minimal LSAs

Authors' Addresses

Richard G. Ogier
SRI International

EMail: rich.ogier@earthlink.net or rich.ogier@gmail.com

Phil Spagnolo
Boeing Phantom Works

EMail: phillipspagnolo@gmail.com