Abstract

This document specifies an extension of OSPF for IPv6 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 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.

Table of Contents

1 Introduction ................................................. 4
1.1 Definitions of Commonly Used Terms ........................... 5
2 Overview of OSPF-MDR ........................................ 6
2.1 Selection of MDRs, BMDRs, Parents, and Adjacencies .......... 7
2.2 Flooding Procedure ........................................... 8
2.3 Link State Acknowledgments .................................. 8
2.4 Routable Neighbors .......................................... 9
2.5 Partial and Full Topology LSAs ................................ 9
2.6 Modified Hello Protocol ...................................... 10
3 Interface and Neighbor Data Structures ...................... 10
3.1 Changes to Interface Data Structure .......................... 11
3.2 New Configurable Interface Parameters ........................ 12
3.3 Changes to Neighbor Data Structure ........................... 13
4 Hello Protocol .............................................. 14
4.1 Sending Hello Packets ....................................... 14
4.2 Receiving Hello Packets ..................................... 16
4.3 Neighbor Acceptance Condition ............................... 19
5 MDR Selection Algorithm ..................................... 20
5.1 Phase 1: Creating the Neighbor Connectivity Matrix .......... 21
5.2 Phase 2: MDR Selection ...................................... 22
5.3 Phase 3: Backup MDR Selection ................................ 22
5.4 Phase 4: Selection of the (Backup) MDR Parent ............... 23
6 Interface State Machine ..................................... 24
6.1 Interface states ............................................ 24
6.2 Events that cause interface state changes ................... 24
6.3 Changes to Interface State Machine .......................... 25
7 Adjacency Maintenance ....................................... 26
7.1 Changes to Neighbor State Machine ........................... 27
7.2 Whether to Become Adjacent .................................. 28
7.3 Whether to Eliminate an Adjacency ........................... 28
7.4 Sending Database Description Packets ........................ 29
7.5 Receiving Database Description Packets ...................... 29
8 Flooding Procedure .......................................... 30
8.1 LSA Forwarding Procedure .................................... 31
8.2 Sending Link State Acknowledgments .......................... 34
8.3 Retransmitting LSAs ......................................... 35
8.4 Receiving Link State Acknowledgments ........................ 35
9 Originating LSAs ............................................ 36
9.1 Routable Neighbors .......................................... 37
9.2 Partial and Full Topology LSAs ................................ 37
10 Calculating the Routing Table ................................ 39
11 Draft Modifications .......................................... 40
1. Introduction

This document specifies an extension of OSPF for IPv6 [RFC2328, RFC2740], 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. Existing OSPF interface types do not perform adequately in such an environment, 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.

An OSPF implementation that is extended with this MANET interface type does not preclude the use of any existing interface types, and is fully compatible with a legacy OSPF implementation. MANET networks are represented externally as Point-to-Multipoint networks,
although the design borrows concepts used by the OSPF broadcast 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. By definition, all routers in the MANET either belong to the CDS or are 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 LLS TLVs, 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 what to include in router-LSAs, and Section 10 describes changes in the calculation of the routing table.

The appendix specifies packet formats, detailed pseudocode for the MDR selection algorithm, an optional algorithm for the selection of neighbors to include in router-LSAs in order to provide min-cost routing, and a proposed option that uses "non-ackable" LSAs to provide periodic flooding that reduces overhead in highly mobile networks.

1.1. Definitions of Commonly Used Terms

MANET Interface
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, which takes advantage of the broadcast capability.

**MANET Router**
An OSPF router that has at least one MANET interface.

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

**2-Hop Neighbor Information**
Information that specifies the Router IDs of each neighbor's neighbors. 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)**
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 the broadcast network.

**Backup MANET Designated Router (Backup MDR or BMDR)**
One of a set of routers responsible for providing backup flooding when neighboring MDRs fail, and for determining the set of adjacencies that must be formed. The set of MDRs and Backup MDRs forms a biconnected dominating set. The Backup MDR is a generalization of the Backup DR found in the broadcast network.

**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.

**(Backup) MDR Parent**
Each Backup MDR and MDR Other selects a Parent, which will be a neighboring MDR if one exists. If the option of biconnected adjacencies is chosen, then each MDR Other also selects a Backup Parent, which will be a neighboring MDR/BMDR if one exists that is not the Parent. Each router forms an adjacency with its Parent and its Backup Parent (if it exists).

**Bidirectional Neighbor**
A neighboring router whose neighbor state is 2-Way or greater.

**Routable Neighbor**
A bidirectional MANET neighbor becomes routable if its state is Full, or 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 MANET neighbors can be used as next hops in the SPF calculation, and can be included in LSAs originated by the router.

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 2-hop neighbor information that can be obtained from neighbors' Hellos (similar to OSPF). 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.

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 selection of MDRs can be summarized as follows. Let Rmax denote the neighbor with the lexicographically largest value of (MDR Level, RtrPri, RID), where MDR Level is 2 for an MDR, 1 for a BMDR, and 0 for an MDR Other. Then a router 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 (MDR Level, RtrPri, RID) than the router itself, where k is the parameter MDRConstraint, whose default value is 3.

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 (MDR Level, RtrPri, 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 MDR parent.

OSPF-MDR will also provide the option of full-topology adjacencies. A router that selects this option will always form an adjacency with each bidirectional neighbor that also selects this option.

Prioritizing routers according to (MDR Level, RtrPri, 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 remain 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.

Prioritizing routers according to (MDR Level, RtrPri, RID) not only increases the lifetime of MDRs and adjacencies, but also achieves consistency with the DR election algorithm, which gives highest priority to existing DRs. As a result, when applied to a fully connected MANET, the MDR selection algorithm and the DR election algorithm both select the same two routers as DR/MDR and BDR/BMDR. (The MDR section algorithm also selects a second BMDR so that the MDR/BMDR backbone is biconnected.)

2.2. Flooding Procedure

When an MDR receives a new LSA on a MANET interface, it immediately floods the LSA back out the receiving interface (unless it can be determined that such flooding is unnecessary). 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 there exists a neighbor that is not covered by another neighbor from which the LSA has been received.

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 Acks 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 Ack. Link State Acks may be delayed up to AckInterval seconds to allow coalescing multiple Acks in the same packet. The only exception is that (Backup) MDRs send a multicast Ack immediately when a duplicate LSA is received as a unicast, in order to prevent additional retransmissions. Only Acks from adjacent neighbors are processed, and retransmitted LSAs are sent (via unicast) only to adjacent neighbors.

2.4. Routable Neighbors

A bidirectional MANET neighbor becomes routable if its state is Full, or if the SPF calculation has produced a route to the neighbor and the neighbor satisfies a flexible quality condition. Once a neighbor becomes routable, it remains routable as long as it remains bidirectional. Only routable MANET 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.

Note that OSPF already allows a non-adjacent neighbor to be used as a next hop, if both routers are fully adjacent to the DR of a broadcast network. The routability condition is a generalization of this condition to MANETs. 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 paths or would require a large number of adjacencies.

2.5. Partial and Full Topology LSAs

Each router advertises a subset of its routable neighbors as point-to-point connections in its router-LSA. The choice of which neighbors to advertise is flexible, and is determined by the configurable parameter LSAFullness. As a minimum requirement, each router must advertise all of its fully adjacent neighbors in its router-LSA. This minimum choice corresponds to LSAFullness = 0. This choice results in the minimum amount of LSA flooding overhead, but does not provide routing along shortest paths.

Setting LSAFullness to 1 or 2 results in min-cost LSAs, which provide min-hop routing, and can provide min-cost routing under certain assumptions. Each router decides which neighbors to include in its router-LSA by looking at the router-LSAs originated by its neighbors, and including in its LSA the minimum set of neighbors necessary to
provide a shortest path (if LSAFullness = 1) or two shortest paths (if LSAFullness = 2) between each pair of neighbors.

Setting LSAFullness to 3 results in MDR full LSAs. Each (Backup) MDR originates a full LSA that includes all routable neighbors, while each MDR Other originates minimal LSAs. This choice provides routing along nearly min-hop paths.

If LSAFullness = 4, then each router originates a full LSA, which includes all routable neighbors.

The above LSA options are interoperable with each other, since they all require the router-LSA to include all fully adjacent neighbors.

2.6. Modified Hello Protocol

Hellos are used both for neighbor discovery and for advertising the set of bidirectional neighbors (in state 2-Way or greater), to be used by neighbors to learn 2-hop neighbor information. Differential Hellos are sent every HelloInterval seconds, except when full Hellos are sent, which happens every 2HopRefresh Hellos. The default values for HelloInterval and 2HopRefresh are 2 seconds and 3 Hellos, respectively. Differential Hellos are used to reduce overhead and to allow Hellos to be sent more frequently, for faster reaction to topology changes. Full Hellos are sent less frequently to ensure that all neighbors have current 2-hop neighbor information. The use of differential Hellos allows HelloInterval to be smaller (e.g. 1 second) while making 2HopRefresh larger (e.g. every 6th Hello), without a significant increase in overhead, allowing faster response to topology changes in a highly mobile network.

Each Hello contains a sequence number, which is incremented each time a Hello is sent on a given interface. As in OSPF, the state of a neighbor transitions to Down if no Hello is heard for RouterDeadInterval. In addition, the state of a neighbor transitions to Init if HelloRepeatCount Hellos are missed, based on the Hello sequence number.

Both differential and full Hellos may contain a list of Heard Neighbors (in state Init) and a list of Reported Neighbors (in state 2-Way or greater). In addition, differential Hellos may contain a list of Lost Neighbors (which recently transitioned to the Down state). A neighbor that transitions to a different one of these three categories is included in the appropriate list for the next HelloRepeatCount Hellos. This ensures that the neighbor will either learn the new state within HelloRepeatCount Hellos, or will declare the neighbor to be Down or Init.

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 RFC 2328.

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.

MDR Parent
Each non-MDR router selects an MDR Parent, as described in Section 5.4. The MDR Parent will be a neighboring MDR, if one exists. The MDR Parent is initialized to 0.0.0.0, indicating the lack of an MDR Parent. A non-MDR router includes the Router ID of its MDR Parent in the DR field of each Hello sent on the interface.

Backup MDR Parent
If the option of biconnected adjacencies is chosen, then each MDR Other selects a Backup MDR Parent, as described in Section 5.4. The Backup MDR Parent will be a neighboring MDR/BMDR, if one exists that is not the MDR Parent. The Backup MDR Parent is initialized to 0.0.0.0, indicating the lack of a Backup MDR Parent. An MDR Other includes the Router ID of its Backup MDR 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.

Hello Sequence Number (HSN)
The 16-bit sequence number carried by the Hello Sequence TLV. The
HSN is incremented by 1 every time a (differential or full) Hello is sent on the interface.

Lost Neighbor List (LNL)
A list of the Router IDs of neighbors whose states have recently changed to Down. These Router IDs are included in the Lost Neighbor List TLV of Hello packets sent on the interface.

3.2. New Configurable Interface Parameters

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

2HopRefresh
Full neighbor state must be included in one of every 2HopRefresh Hello packets. Other Hellos include only differential state information. Default value is 3.

HelloRepeatCount
The number of consecutive Hellos in which a neighbor must be included when its state changes. Default value is 3.

AckInterval
The maximum number of seconds that an acknowledgment may be held before it is multicast so that acknowledgments may be coalesced. Default value is 1.8 seconds.

BackupWaitInterval
The number of seconds that a Backup MDR must wait after receiving a new LSA, before it decides whether to flood the LSA. Default value is 2 seconds.

AdjConnectivity
If equal to the default value of 2, then the set of adjacencies form a biconnected graph. If equal to the optional value of 1, then the set of adjacencies form a (uni)connected graph.

MDRConstraint
A parameter of the MDR selection algorithm, which affects the number of MDRs selected. The default value of 3 results in nearly the minimum number of MDRs. The optional value 2 results in a larger number of MDRs.

LSAFullness

Determines which neighbors a router should advertise in its router-LSA. The value 0 results in minimal LSAs that include only fully adjacent neighbors. The values 1 and 2 result in partial-topology LSAs that provide min-cost routing under certain assumptions. The value 3 results in (Backup) MDRs originating full LSAs and other routers originating minimal LSAs. The value 4 results in all routers originating full LSAs. The default value is 3.
3.3. Changes to Neighbor Data Structure

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.

Reported Neighbor List (RNL)
The Reported Neighbor List for the neighbor, which is updated when a Hello is received from the neighbor that contains an RNL TLV. The Reported Neighbor Lists for all neighbors represent the 2-hop neighbor information.

Report2Hop
A single-bit variable equal to 1 if a full Hello (which contains a full Reported Neighbor List) 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 TLV in a DD packet received from the neighbor.

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

Neighbor's Backup MDR Parent
The neighbor's choice for Backup MDR Parent, obtained from the Backup DR field of the last Hello packet received from the neighbor or from the MDR 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) MDR Parent.

Dependent
A single-bit variable equal to 1 if the neighbor is a Dependent Neighbor, which is decided by the MDR selection algorithm. Dependent Neighbors become adjacent.

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

Routable
A single-bit variable equal to 1 if the neighbor is routable. A neighbor is routable if either its state is Full, or the routing table includes a route to the neighbor. Only routable neighbors are included in the router-LSA and are allowed as next hops in the
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 state or differential Hellos. 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 [RFC2740] Section 3.2.1.1 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 [RFC2740] A.3.2.

There are no changes to the Hello packet format. However, the meaning of the DR and Backup DR fields has changed. Similar to [RFC2328], 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. However, these fields are also used to advertise the router's MDR Parent and Backup MDR Parent, as specified in Appendix A.3. The Hello packet's Neighbor Router ID list is not used on the MANET interface.

Hellos are sent every HelloInterval seconds. Full state 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 contains full neighbor state information. If 2HopRefresh is set to 1, then all Hellos are full state. The first Hello sent by a neighbor should be a full state Hello.

MANET Hellos require the use of the Heard Neighbor List (HNL) TLV, Reported Neighbor List (RNL) TLV, Lost Neighbor List (LNL) TLV, Dependent Neighbor List (DNL) TLV, and Hello Sequence (HS) TLV (see Appendix A.2.2). Depending on the need, each of these TLVs are appended to the Hello packet with LLS (see Appendix A.2 for link-local signaling).

If the router has any Dependent Neighbors, then the Hello, whether full or differential, contains the DNL TLV, which is built by including a list of all Dependent Neighbors. If the router does not have any Dependent Neighbors, then the Hello does not contain the DNL TLV. Note that only (Backup) MDRs have any Dependent Neighbors.

4.1.1. Full State Hello Packets

The full state Hello requires the HS TLV and may include the HNL TLV and RNL TLV appended with LLS. The L bit is set in the Hello's...
option field to indicate LLS.

The HS TLV is built by populating the Sequence Number field with the interface's Hello Sequence Number (HSN). The HSN is then incremented.

If the router has neighbor(s) in state Init, the HNL TLV is built by including a list of all neighbors in state Init.

If the router has neighbor(s) in state 2-Way or greater, the RNL TLV is built by including a list of all neighbors in state 2-Way or greater, excluding any Dependent Neighbors (which are included in the DNL TLV).

4.1.2. Differential Hello Packets

The differential Hello requires the HS TLV and may include the HNL TLV, RNL TLV, and LNL TLV based on need. The D and L bits are set in the Hello's option field to indicate differential Hellos and link-local signaling.

The HS TLV is built by populating the Hello Sequence Number field with the interface's HSN. The HSN is then incremented.

The HNL TLV is built by including a list of all neighbors that have transitioned to state Init within the last HelloRepeatCount Hellos. If none exist, the HNL TLV is not appended.

The RNL TLV is built by including a list of all neighbors that have transitioned from Init to state 2-Way or greater within the last HelloRepeatCount Hellos, and all neighbors in state 2-Way or greater such that the router is not in the neighbor's Reported Neighbor List. If none exist, the RNL TLV is not appended.

The LNL TLV is built by including a list of all neighbors that have transitioned to state Down within the last HelloRepeatCount Hellos. These neighbors are found in the Lost Neighbor List. If none exist, the LNL TLV is not appended. Neighbors that have been in the Lost Neighbor List longer than HelloRepeatCount Hellos should be removed from the list and not included in the LNL TLV.

4.2. Receiving Hello Packets

Hello packets are received according to [RFC2740] Section 3.2.2.1 and [RFC2328] Section 10.5 with the following MANET specific specifications beginning after paragraph 3 of Section 10.5. The Hello packet format is defined in [RFC2740] A.3.2.

On a MANET interface, the source of a Hello packet is identified by the neighbor's Router ID, and the neighbor is identified by its Router ID.

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 either to be 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 is set in the options field, then there are TLVs to be processed.

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

- The Hello Sequence Number in the HS TLV should be stored in the neighbor's data structure.

- The DR and Backup DR fields should be processed as follows.
  1. If the DR field is equal to the neighbor's Router ID, set the MDR Level of the neighbor to MDR.
  2. Else if the Backup DR field is equal to the neighbor's Router ID, set the MDR Level of the neighbor to Backup MDR.
  3. Else, set the MDR Level of the neighbor to MDR Other.
  4. If the DR or Backup DR field is equal to the router's own Router ID, the neighbor's Child variable is set to 1, otherwise it is zero.

- If the router itself appears in the DNL TLV neighbor list, the neighbor's Dependent Selector variable is set to 1.

- If the router itself does not appear in the DNL TLV, or if the Hello packet does not contain a DNL TLV, the neighbor's Dependent Selector variable is set to 0.

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

4.2.1. Full State Hello Packets

- Report2Hop is set to 1.

- If the router itself appears in the HNL, RNL, or DNL TLV neighbor list, the neighbor state machine should be executed with the event 2-WayReceived. Otherwise, the neighbor state machine should be executed with the event 1-WayReceived.

- If the neighbor list in the RNL TLV differs from the Reported Neighbor List for the neighbor, the receiving interface's state machine is scheduled with the event MDRNeighborChange.
The Reported Neighbor List for the neighbor should be replaced with the union of the RNL TLV neighbor list and the DNL TLV neighbor list.

4.2.2. Differential Hello Packets

- If an LNL TLV exists, then perform the following steps.
  1. If the router itself appears in the LNL TLV neighbor list,
     a. The neighbor state machine should be executed with the event 1-WayReceived.
     b. Remove the router from the Reported Neighbor List (for the neighbor) if it is in the list.

- If an HNL TLV exists, then perform the following steps.
  1. If the router itself appears in the HNL TLV neighbor list and did not appear in the LNL TLV neighbor list,
     a. The neighbor state machine should be executed with the event 2-WayReceived.
     b. Remove the router from the Reported Neighbor List if it is in the list.

- If an RNL TLV exists, then perform the following steps.
  1. If the router itself appears in the RNL TLV neighbor list and did not appear in the LNL or HNL TLV neighbor list,
     a. The neighbor state machine should be executed with the event 2-WayReceived.
     b. Add the router itself to the Reported Neighbor List if it does not belong.

- If the router itself did not appear in any of the TLV neighbor lists, 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
should be executed with the event 2-WayReceived (the state does not change).

- If 2-WayReceived or 1-WayReceived was not executed, then the neighbor state machine should be executed with the event 1-WayReceived.

The following applies to both full state and differential Hellos.

- If a change in the neighbor's Router Priority field was noted, the receiving interface's state machine is scheduled with the event MDRNeighborChange.

- If the neighbor is bidirectional and its MDR Level has changed, then the receiving interface's state machine is scheduled with the event MDRNeighborChange, and the neighbor state machine is scheduled with the event AdjOK?.

- If the neighbor's Child status or Dependent Selector status has changed from 0 to 1, the neighbor state machine is scheduled with the event AdjOK?.

- If the neighbor's state changed from less than 2-Way to 2-Way or greater, the receiving interface's state machine is scheduled with the event MDRNeighborChange and the neighbor state machine is scheduled with the event AdjOK?. Else if the neighbor's state changed from 2-Way or greater to below 2-Way, the receiving interface's state machine is scheduled with the event MDRNeighborChange.

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 determines whether the router is an MDR, Backup MDR, or MDR Other on a given interface. The algorithm also selects the Dependent Neighbors and the (Backup) MDR Parent, which are used to decide which neighbors should become adjacent (see Section 7).

The MDR selection algorithm is invoked by the interface event MDRNeighborChange as described in Section 6. 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 (Backup) MDRs forms a biconnected CDS. Note that there may be fewer Backup MDRs than MDRs, since the MDRs themselves may already provide some redundancy.

Each MDR will become adjacent with each Dependent Neighbor that is an MDR, forming a connected backbone network. If AdjConnectivity = 2, then each (Backup) MDR will become adjacent with each Dependent Neighbor that is a (Backup) MDR, forming a biconnected backbone network. The (Backup) MDR Parents that are selected (as described below) will then connect each MDR Other router with this biconnected backbone, via two adjacencies. This ensures that the set of adjacencies forms a biconnected subgraph that spans all routers.

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 (MDR Level, Router Priority, Router ID) for each such neighbor and for the router itself, and the neighbor variables Reported Neighbor List (RNL) and Report2Hop 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 (MDR Level, RtrPri, RID). The triplet (MDR Level, RtrPri, 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 four phases. Phase 1 creates the neighbor connectivity matrix, 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. Finally, Phase 4 selects the MDR Parent and Backup MDR Parent.

The second phase 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.

For convenience, in the following description, the term "neighbor" will refer to a neighbor on the MANET interface that is bidirectional (in state 2-Way or greater).

5.1. Phase 1: Creating the Neighbor Connectivity Matrix

The neighbor connectivity matrix (NCM) assigns a value of 0 or 1 for each pair of (bidirectional) neighbors, depending on the Reported Neighbor List (RNL) and the value of Report2Hop for each neighbor. NCM is a symmetric matrix that defines a topology graph for the set of neighbors (not including the router itself). A value of 1 for a given pair of neighbors indicates that the neighbors are assumed to be neighbors of each other in the MDR selection algorithm. The value of the matrix is set as follows for each pair of neighbors j and k.

(1.1) If Report2Hop is 1 for both neighbors j and k: NCM(j,k) = NCM(k,j) is 1 only if j belongs to the RNL of neighbor k and k belongs to the RNL of neighbor j.
(1.2) If Report2Hop 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 RNL of neighbor j.
(1.3) If Report2Hop 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 neighbors of each other only if they both agree that the other router is a 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 RNL, so neighbor j is believed in deciding whether j and k are neighbors of each other. As Step 1.3 indicates, two neighbors are assumed not to be neighbors of each other if neither neighbor has reported its full RNL.
5.2. Phase 2: MDR Selection

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

(2.2) If the router has a larger value of (MDR Level, RtrPri, RID) than all of its neighbors, the router selects itself as an MDR, selects all of its MDR neighbors as Dependent Neighbors, and if AdjConnectivity = 2, selects all of its BMDR neighbors as Dependent Neighbors. Else, proceed to Step 2.3.

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

(2.4) Using NCM to determine the connectivity of neighbors, compute the minimum number of hops, denoted hops(u), from Rmax to each other neighbor u, using only intermediate nodes that are neighbors with a larger value of (MDR Level, RtrPri, 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.)

(2.5) If hops(u) is at most MDRConstraint for each neighbor u, then the router does not select itself as an MDR, and selects no Dependent Neighbors.

(2.6) Else, the router selects itself as an MDR, and selects the following neighbors as Dependent Neighbors: Rmax, each MDR neighbor u such that hops(u) is greater than MDRConstraint, and if AdjConnectivity = 2, each BMDR neighbor u such that hops(u) is greater than MDRConstraint.

Step 2.4 can be implemented using a breadth-first-search (BFS) algorithm to compute min-hop paths from node Rmax to all other neighbors, modified to allow a node as an intermediate node only if its value of (MDR Level, RtrPri, 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 the Appendix.

5.3. Phase 3: Backup MDR Selection

(3.1) The set of Dependent Neighbors initially includes the neighbors selected in Phase 2.

(3.2) Using NCM to determine the connectivity of neighbors, determine whether or not there exist two node-disjoint paths from Rmax to each other neighbor u, using only intermediate nodes that are neighbors with a larger value of (MDR Level, RtrPri, 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 neighbor u, then the router does not select itself as a Backup MDR, and selects no additional Dependent Neighbors.

(3.4) Else, the router selects itself as a Backup MDR (unless it already selected itself as an MDR in Phase 2), and if
AdjConnectivity = 2, selects each of the following neighbors as a Dependent Neighbor: Rmax, and each MDR/BMDR neighbor u such that step 3.2 did not find two node-disjoint paths from Rmax to u.

Step 3.2 can be implemented using a modification of the algorithm [Suurballe] to find the node-disjoint paths. A detailed description of this algorithm, which runs in O(d^2) time, is given in the Appendix. The Appendix also describes an alternative algorithm for Step 3.2, which is simpler but results in a larger number of Backup MDRs.

5.4. Phase 4: Selection of the (Backup) MDR Parent

Each BMDR and MDR Other selects (for each MANET interface) a Parent, which will be a neighboring MDR if one exists. If AdjConnectivity = 2, then each MDR Other also selects a Backup Parent, which will be a neighboring MDR/BMDR if one exists that is not the Parent. Each router forms an adjacency with its Parent and its Backup Parent (if it exists).

One property of the (Backup) Parent is that it always has a larger value of (MDR Level, RtrPri, RID) than the router itself. Thus, the directed graph defined by the parent relationship will not contain any cycles. All paths in this directed graph lead to an MDR that has a larger value of (MDR Level, RtrPri, RID) than all of its neighbors.

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

Otherwise (the router is a BMDR or MDR Other), the Parent is selected to be any adjacent neighbor that has 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.)

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 state 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 RFC 2328, 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 event MDRNeighborChange, defined as follows.

MDRNeighborChange
There has been a change in neighbor information that requires the MDR selection algorithm to be run. The following neighbor changes

lead to the MDRNeighborChange event:

- The state of a neighbor changes from Init or lower to 2-Way or greater, or vice versa.

- The MDR Level of a bidirectional neighbor has changed, as detected via Hello packets from the neighbor.

- The advertised Router Priority of a bidirectional neighbor has changed, as detected via Hello packets from the neighbor.

- The Router Priority of the router itself has changed.

- The Reported Neighbor List or Report2Hop has changed for a bidirectional neighbor, as detected via Hello packets from the neighbor.

6.3. Changes to Interface State Machine
This section describes the changes to the interface state machine for a MANET interface. The first two state transitions are for state-event pairs that are described in RFC 2328, but have modified action descriptions because MDRs are selected instead of DRs. The third state transition describes the action taken when the event MDRNeighborChange is invoked, and replaces the corresponding state transition in RFC 2328 for the event NeighborChange. 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 RFC 2328.

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. If the router is not eligible to become an MDR (Router Priority is 0), the state transitions to DR Other. Otherwise, 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) MDR 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.)

State(s): DR Other, Backup or DR
Event: MDRNeighborChange
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) MDR 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 one or more neighbors. (See Section 7.) To limit the amount of processing, the router may delay running the MDR selection algorithm for up to HelloInterval seconds. (For example, a router may wait until just before the next Hello is sent, allowing the updated MDR Parents to be included in the next Hello.)
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 ([OSPF] Section 10.6). The specification below relates to the MANET interface only.

Adjacencies are established with some subset of the router's neighbors. Each (Backup) MDR forms adjacencies with a subset of its (Backup) MDR neighbors to form a biconnected backbone, and each MDR forms adjacencies with two selected (Backup) MDR neighbors called "parents", thus providing a biconnected subgraph of adjacencies.

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) MDR Parent.
4. The neighbor selects the router to be its (Backup) MDR 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 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. If the interface event
MDRNeighborChange is scheduled, it should be executed before
proceeding. 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.

An adjacency is established 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 router is a (Backup) MDR and the neighbor is a child.

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

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 adjacency should be eliminated between neighbors in a state above 2-way. If the interface event MDRNeighborChange is scheduled, it should be executed before proceeding.

An adjacency is maintained if one of the following is true.

(1) The router is an MDR.
(2) The router is a Backup MDR.
(3) The neighbor is an MDR.
(4) The neighbor is a Backup MDR.

Otherwise, the adjacency is eliminated.

7.4 Sending Database Description Packets

Sending a DD packet on a MANET interface is the same as [RFC2740] Section 3.2.1.2 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 TLV appended using LLS, and the L bit is set in the DD packet's option field. The DR and Backup DR fields of the MDR TLV are set exactly the same as the DR and Backup DR fields of a Hello sent on the same interface, as specified in Appendix A.3.

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.

- If the DD packet's L bit is set in the options field and an MDR TLV is appended, then the MDR 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 Selector variable to 0.

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

   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, if the router accepts a received DD packet and processes its contents, then the following action SHOULD be performed for each LSA listed in the DD packet (whether the router is master or slave). If the router has an instance of the LSA in the Database summary list for the neighbor, which is the same or less recent than the LSA listed in the packet, then the LSA is removed from the Database summary list. This avoids including 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 RFC 2328, 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.

   o 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.

   o 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 RNL 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 RFC 2328.

8.1. LSA Forwarding Procedure

Step 1 of [RFC2328], Section 13.3 should be performed, with the following change, so that the new LSA is placed on the Link State retransmission list for each appropriate adjacent neighbor. Step 1(c) 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 an 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 is defined to be covered if it belongs to the Reported Neighbor List (RNL) for the neighbor from which the new LSA was received.

Steps 2 through 5 of [RFC2328], Section 13.3 are unchanged if the outgoing interface (on which the LSA may be forwarded) is not of type MANET. If the outgoing interface is of type MANET, then steps 2 through 5 are replaced with the following steps, to determine whether the LSA should be forwarded on each eligible MANET interface.

(2) If either of the following two conditions is satisfied for every bidirectional neighbor on the interface, examine the next
interface (the LSA is not flooded out this interface).

(a) The LSA or an ACK for the LSA has been received from the neighbor (over any interface).

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

Note that the above two 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 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 fails to satisfy both conditions (a) and (b) 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.) Examine the next interface (the LSA is not immediately flooded out this interface).

(5) If the router is an 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). The LSA is included in an LSU packet which is multicast out the interface using the destination IP address AllSPFRouters.

(6) If the LSA was received on a MANET 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 (MDR Level, RtrPri, 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.

(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 interface, but only one BackupWait Timer is associated with the LSA. Examine the next interface (the LSA is not immediately flooded out this
(7) If the optional step 6 is not performed, then the LSA is flooded out the interface. The LSA is included in an LSU packet which is multicast out the interface using the destination IP address AllSPFRouters.

8.1.1. 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 (if necessary) so that it does not occur within AckInterval plus propagation delays.

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

8.1.2. Optional Treatment of Broadcast Network as MANET

In the LSA forwarding procedure described above, a router MAY treat each of its broadcast interfaces the same as a MANET interface, with the following substitutions. A DR is treated as an MDR, a Backup DR is treated as a Backup MDR, and all neighbors on a broadcast interface are considered to be covered if the LSA was sent by the DR or Backup DR on the same interface. As in RFC 2328, Section 13.3, only the DR and Backup DR use the IP address AllSPFRouters to flood an LSA on a broadcast interface; all other routers use AllDRouters to flood an LSA on a broadcast interface.

Treating a broadcast network as a MANET can greatly reduce flooding overhead in some cases. For example, assume 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 treating the broadcast network as a MANET in 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 one of the 50 routers is an MDR).

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 RFC 2328 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, a method similar to that of [Chandra] is used for sending LS ACKs on MANET interfaces. All LS ACK packets sent on a MANET interface are multicast using the IP address AllSPFRouters.

When a router receives an LSA, it must decide whether to send a delayed ACK, an immediate ACK, or no ACK. (However, a non-ackable LSA is never acknowledged, as described in Appendix D.) A delayed ACK may be delayed for up to AckInterval seconds, and allows several LS ACKs to be grouped into a single multicast LS ACK packet. An immediate ACK is also sent in a multicast LS ACK packet, and may include other LS ACKs that were scheduled to be sent as delayed ACKs. The decision 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 a (Backup) MDR, an immediate ACK is sent out the receiving interface.
   b. If the router is an MDR Other, 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 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 RFC 2328. 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 sent 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 $\text{RxmtInterval}$, whose default is 5 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 RFC 2328, 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 which the router itself has NOT yet received. This is necessary because (unlike RFC 2328) 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 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 which is less recent than

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. Originating 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 neighbors to advertise is flexible, and is determined by the configurable parameter $\text{LSAFullness}$.

As a minimum requirement, each router must advertise all of its fully adjacent 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

A bidirectional MANET neighbor becomes routable if its state is Full, or if the SPF calculation has produced a route to the neighbor and the neighbor satisfies the routable neighbor quality condition (defined below). Since only routable neighbors are advertised in router-LSAs, 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 RNL 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 RNL includes the router itself.
The single-bit neighbor variable Routable indicates whether the neighbor is routable. This variable is initially 0, and 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 and the state of the neighbor changes to Full, Routable is set to 1 for the neighbor.
2. 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.
3. 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.

9.2. Partial and Full Topology LSAs

The choice of which MANET neighbors to include in the router-LSA is flexible, subject only to the following requirements:

1. A router MUST include all Full neighbors in its router-LSA.
2. A router MUST NOT include any non-routable neighbors in its LSA.

Thus, a minimum LSA includes only Full neighbors, corresponding to LSAFullness = 0. At the other extreme, a router may include all routable neighbors in its router-LSA, corresponding to LSAFullness = 4 (full-topology LSAs). Between these two extremes, a router may include any subset of routable neighbors in its router-LSA, as long as all Full neighbors are included. It is not necessary for different routers to make the same choice; the different choices are interoperable because each router-LSA must include all Full neighbors, which allows the SPF calculation to find routes to all reachable routers.

A new router-LSA is originated whenever an event occurs that causes the contents of the LSA to change (which depends on the choice of the LSA contents). However, as stated in RFC 2328, Section 12.4, two instances of the same LSA may not be originated within the time period MinLSInterval. This may require that the generation of the next instance be delayed by up to MinLSInterval. When a new LSA is originated, it is installed in the database as described in Section 13.2 of RFC 2328, which may cause the routing table to be recalculated. The new LSA is also flooded as described in Section 8 of this document.

This document specifies two additional choices for partial-topology LSAs, which provide shorter paths than minimal LSAs, but generate substantially less overhead than full-topology LSAs.

9.2.1. Min-Cost LSAs (LSAFullness = 1 or 2)

Each router decides which MANET neighbors to include in its router-
LSA by looking at the router-LSAs originated by its neighbors, and including in its router-LSA the minimum set of neighbors necessary to provide a shortest path between each pair of neighbors. If another neighbor is already providing such a path to a given neighbor k, then the router includes neighbor k in its LSA only if it can provide a lower cost path. If LSAFullness = 2, then the router ensures that the router-LSAs provide at least two shortest paths between each pair of neighbors, thus allowing routers to calculate multiple paths to each destination.

The min-cost LSA algorithm (described in Appendix C) may also use metric information that may be advertised in Hellos. If this option is used, each router will advertise the cost to each routable neighbor in Hello packets via an LLS TLV. (The format for advertising this information will be described in a future version of this draft.) If this option is used, then shortest paths that are calculated based on min-cost LSAs will have minimum cost in all cases, without any conditions on the metrics.

If this option is not used (i.e., metric information is not included in Hellos), then min-hop paths will be calculated if all metrics in the network are equal, and minimum-cost paths will be calculated if the metrics for all neighbors on the same interface are equal (e.g., are equal to the configured cost for the interface). For example, if the interface cost is configured to be smaller for high bandwidth routers than for low bandwidth routers, then the calculated routes will use high bandwidth routers whenever possible.

9.2.2. MDR Full LSAs (LSAFullness = 3)

Each (Backup) MDR originates a full LSA (which includes all routable neighbors), while each MDR Other originates a minimum LSA (which includes only Full neighbors). If a router has multiple MANET interfaces, its LSA includes all routable neighbors on the interfaces for which it is a (Backup) MDR, and includes only Full neighbors on its other interfaces. When a router changes its MDR Level from MDR Other to (Backup) MDR on a given interface, it originates a new LSA. This choice provides routing along nearly min-cost paths.

A variation of MDR Full LSAs is possible, in which some MDR Other routers also select themselves to originate full LSAs, based on 2-hop neighbor information. A heuristic can be used for such a selection that results in routes that are arbitrarily close to min-cost on average. Such a heuristic may be described in a future version of this draft.

10. Calculating the Routing Table

The routing table calculation is the same as specified in RFC 2328, except for the following change to Section 16.1 (Calculating the shortest-path tree for an area).

Recall from Section 9 that a router can use any routable neighbor as
a next hop to a destination. However, unless LSAFullness = 4 (full topology LSAs), the router-LSA originated by the router usually does not include all routable neighbors. Therefore, the shortest-path tree calculation described in Section 16.1 of RFC 2328 must be modified to allow any routable neighbor on a MANET interface to be used as a next hop. This is accomplished simply by modifying step 2 so that the router-LSA associated with the root vertex (i.e., the router doing the calculation) is augmented to include all routable neighbors on each MANET interface. That is, the router-LSA used in the SPF calculation is the one that the router would originate if LSAFullness were equal to 4 (even if LSAFullness is actually less than 4).

Note that, if LSAFullness is less than 4, 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 routing table must be recalculated even if the router-LSA does not change.

11. Draft Modifications

The main changes from version 06 to version 07 of this draft are as follows:

- Dependent Neighbors are now advertised in Hellos via a new LLS TLV, to allow each router to correctly update its Dependent Selectors.
- The set of Dependent Neighbors has been narrowed to include only MDR/BMDR neighbors that should become adjacent.
- The procedure for deciding whether to become adjacent with a neighbor has been simplified, and no longer depends on AdjConnectivity. (Instead, the set of Dependent Neighbors now depends on AdjConnectivity.)
- The parent selection algorithm has been simplified. The (Backup) Parent is now selected to be an adjacent (Backup) MDR neighbor whenever such a neighbor exists, to avoid forming a new adjacency when possible.
- The min-cost LSA algorithm has been extended to support multiple interfaces, including non-MANET interfaces.

The main changes from version 05 to version 06 of this draft are as follows:

- The min-cost LSA algorithm has been modified to improve its efficiency, by requiring each router to include in its router-LSA only neighbors to which (and not from which) a path is required. This reduces the number of neighbors that are included in each router-LSA by about 50%.
- The min-cost LSA algorithm has been extended to allow the option of providing redundant paths.
- The min-cost LSA algorithm has been extended to allow the option...
of using metric information advertised in Hellos, allowing min-
cost paths to be calculated in all cases, without any conditions

The main changes from version 04 to version 05 of this draft are as follows:

- The flooding procedure has been simplified so that the decision to forward a new LSA does not depend on which neighbors are (backup) dependent.
- To avoid accepting poor quality neighbors, and to employ hysteresis, a router may require that a stricter quality condition be satisfied before changing the state of a MANET neighbor from Down to Init or greater.
- To avoid selecting poor quality neighbors as routable neighbors, a router may require that a stricter quality condition be satisfied before declaring a neighbor to be routable.
- Subsection 1.1 has been added, which defines commonly used terms.

The main changes from version 03 to version 04 of this draft are as follows:

- The draft has been rewritten to specify complete details.
- Packet formats are now specified.
- The term MANET Designated Router (MDR) is now used instead of Designated Router (DR) for MANET interfaces.
- Only a single parametrized MDR selection algorithm is now specified (previously called the MPN CDS algorithm), which includes the Essential CDS algorithm as a special case. This algorithm runs in \(O(d^2)\) time, where \(d\) is the number of neighbors.
- The optional ANP CDS algorithm has been omitted from the draft.
- A procedure for selecting the MDR Parent and Backup MDR Parent has been added as Phase 4 of the MDR selection algorithm.
- The term "synchronized neighbor" has been changed to "routable neighbor", to reflect that such a neighbor is not perfectly synchronized, but is sufficiently synchronized to be advertised in router-LSAs and used as a next hop.
- A new option for partial-topology LSAs, called min-cost LSAs, has been added, which provides minimum cost routes under certain assumptions.

References


[LLS] Zinin, A., Friedman, B., Roy, A., Nguyen, L., and D. Yeung,
"OSPF Link-local Signaling", draft-nguyen-ospf-lls-05.txt (work in progress), March 2005.


A. Packet Formats

A.1. Options Field

A new bit, called L (for LLS) is introduced to OSPFv3 Options field (see Figure A.1). The mask for the bit is 0x200. Routers set the L bit in Hello and DD packets to indicate that the packet contains LLS data block. Routers set the L bit in a self-originated router-LSA to indicate that the LSA is non-ackable.

A new D bit is defined in the OSPFv3 option field. The bit is defined for Hello packets and indicates that only differential information is present. The mask for the bit is 0x400.

```
0                   1                   2
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |D|L|AF|*|*|DC| R| N|MC| E|V6|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```

Figure A.1: The Options field

A.2. Link-Local Signaling

Link-local signaling (LLS) describes a modification to [OSPF] which allows the exchange of arbitrary data using existing, standard [OSPF] packet types.

The proposal for extending [OSPF] can be found in [LLS]. Here we use the LLS method in [OSPFv3], as is done in [Chandra].

LLS is accomplished by adding an LLS data block at the end of the OSPFv3 packet.

The IPv6 header length includes the total length of the OSPFv3 header, OSPFv3 data, and LLS data, but the OSPFv3 header does not contain the LLS data length in its length field. The IPv6 packet
The LLS data block may be attached to OSPFv3 Hello and Database Description (DD) packets. The data included in the LLS block attached to a Hello packet may be used for dynamic signaling, since Hello packets may be sent at any moment in time. However, delivery of LLS data in Hello packets is not guaranteed. The data sent with DD packets is guaranteed to be delivered as part of the adjacency forming process.

A.2.1 LLS Data Block

The data block used for link-local signaling is formatted as described below (see Figure A.3 for illustration).

![Figure A.2: Attaching 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 TLVs

The contents of LLS data block is constructed using TLVs. See Figure A.4 for the TLV format.

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

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

A.2.2.1 Heard Neighbor List TLV

A new TLV is defined in this document which indicates neighbor(s) that are in state Init (or recently changed to Init if the Hello is differential). This TLV is used in conjunction with a Hello packet.
A.2.2.2 Reported Neighbor List TLV

A new TLV is defined in this document which indicates neighbor(s) that are in state 2-Way or greater (or recently changed to 2-Way or greater if the Hello is differential). This TLV is used in conjunction with a Hello packet.

```
<table>
<thead>
<tr>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reported Neighbor(s)</td>
<td></td>
</tr>
<tr>
<td>....</td>
<td></td>
</tr>
</tbody>
</table>
```

- Type: Type, set to 12.
- Length: Set to the number of reported neighbors included in the TLV multiplied by 4.
- Reported Neighbor(s) - Router ID of the reported neighbor.

A.2.2.3 Lost Neighbor List TLV

A new TLV is defined in this document which indicates neighbor(s) that have recently been lost by the sender. This TLV is used in conjunction with a Hello packet.

```
<table>
<thead>
<tr>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lost Neighbor(s)</td>
<td></td>
</tr>
<tr>
<td>....</td>
<td></td>
</tr>
</tbody>
</table>
```

- Type: Type, set to 13.
- Length: Set to the number of lost neighbors included in the TLV multiplied by 4.
- Lost Neighbor(s) - Router ID of the reported neighbor.

A.2.2.4 Hello Sequence TLV
A new TLV is defined that indicates the current Hello sequence number (HSN) for the transmitting interface. This TLV is used in conjunction with a Hello packet.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|            Type               |           Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| Hello Sequence Number     |           Reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```

- **Type**: Type, set to 14.
- **Length**: Set to 4.
- **Hello Sequence Number**: A circular two octet unsigned integer indicating the current HSN for the transmitting interface. The HSN for the interface MUST be incremented by 1 every time a (differential or full) Hello is sent on the interface.
- **Reserved**: Set to 0. Reserved for future use.

### A.2.2.5 MDR TLV

A new TLV is defined which includes the same two Router IDs that are included in the DR and Backup DR fields of a Hello sent by the router. This TLV is used in conjunction with a Database Description packet, and is used to indicate the router's MDR Level and selected 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|            Type               |           Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|             DR                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| Backup DR                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```

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

### A.2.2.6 Dependent Neighbor List TLV

A new TLV is defined which indicates neighbor(s) that are currently selected as Dependent Neighbors. This TLV is used in conjunction with a Hello packet.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```

- **Type**: 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 Appendix A.3).
- **Backup DR**: The same Router ID that is included in the Backup DR field of a Hello sent by the router (see Appendix A.3).
<table>
<thead>
<tr>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dependent Neighbor(s)</td>
<td></td>
</tr>
<tr>
<td>....</td>
<td></td>
</tr>
</tbody>
</table>

- **Type**: Type, set to 16.
- **Length**: Set to the number of reported neighbors included in the TLV multiplied by 4.
- **Dependent Neighbor(s)** - Router ID of the reported neighbor.

### 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:

- **DR**: If the router is an MDR, this field is the router's own Router ID. Otherwise, this field is the router's MDR Parent, or is 0.0.0.0 if the MDR Parent is null.
- **Backup DR**: If the router is a BMDR, this field is the router's own Router ID. If the router is an MDR, this field is the router's MDR Parent. Otherwise, this field is the router's Backup MDR Parent, or is 0.0.0.0 if the Backup MDR Parent is null.

### A.4. LSA Formats and Examples

LSA formats are specified in [OSPFv3] Section 3.4.3. Figure A.5 below gives an example network map for a MANET in a single area.

- Four MANET nodes RT1, RT2, RT3, and RT4 are in area 1.
- RT1's MANET interface has links to RT2 and RT3's MANET interfaces.
- RT2's MANET interface has links to RT1 and RT3's MANET interfaces.
- RT3's MANET interface has links to RT1, RT2, and RT3's MANET interfaces.
- RT4's MANET interface has a link to RT3's MANET interface.
- RT1 and RT2 have stub networks attached on broadcast interfaces.
- RT3 has a transit network attached on a broadcast interface.
Table 1: IPv6 link prefixes for sample network

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

Table 2: IPv6 link prefixes for sample network

<table>
<thead>
<tr>
<th>Router</th>
<th>interface</th>
<th>Interface ID</th>
<th>link-local address</th>
</tr>
</thead>
<tbody>
<tr>
<td>RT1</td>
<td>LOOPBACK</td>
<td>0</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td>to N1</td>
<td>1</td>
<td>fe80:0001::RT1</td>
</tr>
<tr>
<td></td>
<td>to N3</td>
<td>2</td>
<td>fe80:0002::RT1</td>
</tr>
<tr>
<td>RT2</td>
<td>LOOPBACK</td>
<td>0</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td>to N2</td>
<td>1</td>
<td>fe80:0001::RT2</td>
</tr>
<tr>
<td></td>
<td>to N3</td>
<td>2</td>
<td>fe80:0002::RT2</td>
</tr>
<tr>
<td>RT3</td>
<td>LOOPBACK</td>
<td>0</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td>to N3</td>
<td>1</td>
<td>fe80:0001::RT3</td>
</tr>
<tr>
<td></td>
<td>to N4</td>
<td>2</td>
<td>fe80:0002::RT3</td>
</tr>
<tr>
<td>RT4</td>
<td>LOOPBACK</td>
<td>0</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td>to N3</td>
<td>1</td>
<td>fe80:0001::RT4</td>
</tr>
</tbody>
</table>
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 = 11 ;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 = 12 ;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 = 13 ;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

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

B. Pseudocode for MDR Selection Algorithm

This section gives detailed pseudocode for Phase 2 (MDR Selection) and Phase 3 (Backup MDR Selection) of the MDR selection algorithm described in Section 5. The pseudocode uses a breadth-first search (BFS) algorithm for Step 2.4 of Phase 2, and uses a variation of the Suurballe-Tarjan algorithm [Suurballe] for finding pairs of node-disjoint paths in Step 3.2 of Phase 3. Both algorithms run in O(d^2) time, where d is the number of neighbors. An alternative algorithm for Phase 3, which is simpler but results in a larger number of Backup MDRs, is given at the end of this section.

For convenience, in the following description, the term "neighbor" will refer to a neighbor on the MANET interface that is bidirectional (in state 2-Way or greater). Also, node i denotes the router performing the calculation.

The following pseudocode 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 neighbor with the (lexicographically) largest value of (MDR Level, RtrPri, 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 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 neighbors of node i and that have a larger value of (MDR Level, RtrPri, RID) than node i. Also, parent(u) will be equal to the parent of node u on the BFS tree, which is used in Step 3.2.

B.1. Pseudocode for Step 2.4 of the MDR Selection Algorithm

(a) Compute a matrix of link costs c(u,v) for each pair of neighbors u and v as follows: If node u has a larger value of (MDR Level, RtrPri, 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 neighbors u other than Rmax, and set hops(Rmax) = 0. Initially, parent(u) is undefined for each neighbor u. 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 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 parent(v) = u, and add node v to the tail of the queue.

The following pseudocode performs Step 3.2 of the MDR selection algorithm, and assumes that Phases 1 and 2 have been performed. When the BFS algorithm terminates, hops2(u), for each neighbor node u of node i, will be finite if and only if there exist two node-disjoint paths from Rmax to node u, using only intermediate nodes that are neighbors of node i and that have a larger value of (MDR Level, RtrPri, RID) than node i.

B.2. Pseudocode for Step 3.2 of the MDR Selection Algorithm

(a) Compute a matrix of link costs c2(u,v) for each pair of neighbors u and v as follows: If c(u,v) is infinity, then set c2(u,v) to infinity. Otherwise set c2(u,v) = 1 + hops(u) - hops(v).

(b) Set hops2(u) = infinity for all neighbors u other than Rmax, and set hop2(Rmax) = 0. Initially, all neighbors u are unlabeled.

(c) Label node Rmax. This divides the BFS tree (defined by the parents selected in Phase 1) into smaller unlabeled subtrees, one for each child of node Rmax. For each pair
u, v of nodes belonging to different subtrees:
If hops2(v) > c2(u,v), then set hops2(v) = c2(u,v).

(d) While there exists an unlabeled node with a finite value of hops2:

- Let node k be the unlabeled node with the minimum value of hops2, and label node k. This divides the unlabeled subtree containing k into smaller unlabeled subtrees, one subtree (called the parent subtree) containing the parent of k if it exists and is unlabeled, and one subtree (called a child subtree) for each unlabeled child of node k. If the parent of k does not exist or is labeled, then continue with the next iteration of step (d).

- For each node u in the parent subtree:
  If hops2(u) > hops2(k) + c2(k,u), set
  \[ \text{hop2}(u) = \text{hops2}(k) + c2(k,u). \]

- For each node v in one of the child subtrees:
  If hop2(v) > hops2(k) + c2(u,v), set
  \[ \text{hop2}(v) = \text{hops2}(k) + c2(u,v). \]

- If hop2(u) > hops2(k) + c2(v,u), set
  \[ \text{hop2}(u) = \text{hops2}(k) + c2(v,u). \]

When the above algorithm terminates, hops2(u), if finite, will be equal to the total number of hops in both disjoint paths from Rmax to u, minus 2 * hops(u). Thus, if hops2(u) = 0, then both disjoint paths have the same length, hops(u). We do not give the procedure for constructing the disjoint paths themselves, since this is not required for the MDR selection algorithm.

We note that in step (d), the nodes of each unlabeled subtree can be found using a depth-first search (DFS), starting from the root of the subtree, and using labeled nodes to define the boundary of the subtree. The tree structure is defined by the values of parent(u) computed in Step 2.4, which can be used to define a list of children for each node. The algorithm runs in \( O(d^2) \) time, since each pair of nodes \((u,v)\) is considered only once in step (d).

We next describe an alternative algorithm for Step 3.2 of Phase 3, which is simpler but typically results in a larger number of Backup MDRs, since it imposes a more restrictive condition on the disjoint paths, i.e., the second path is not allowed to use any intermediate nodes of the BFS tree computed in Phase 2.

B.3. Alternative Algorithm for Step 3.2

(a) Compute a matrix of link costs \( c2(u,v) \) for each pair of neighbors u and v as follows: If \( c(u,v) \) is infinity, or if u is an intermediate node of the BFS tree computed in Phase 2 (i.e., is not Rmax and is the parent of some other node), then set \( c2(u,v) \) to infinity. Otherwise set \( c2(u,v) = 1. \)

(b) Run BFS to compute min-hop paths from node Rmax to the other
neighbors of node i, using the link costs c2(u,v). Let
hops2(u) equal the number of hops in the resulting min-hop
path from Rmax to u, or infinity if no finite cost path exists.

(c) Note that step (b) does not compute disjoint paths to
neighbors of node Rmax. For each neighbor u of node i that is
a neighbor of node Rmax: If there exists another neighbor v of
node i that is a neighbor of both nodes Rmax and u, and has a
larger value of (MDR Level, RtrPri, RID) than node i, then set
hops2(u) = 2; else set hops2(u) = infinity.

If hops2(u) is finite for all neighbors u, then in Step 3.3 of Phase
3, node i does not select itself as a Backup MDR, and does not select
any additional Dependent Neighbors.

Otherwise, in Step 3.4, node i selects itself as a Backup MDR (unless
it already selected itself as an MDR in Phase 2), and if
AdjConnectivity = 2, selects each of the following neighbors as a
Dependent Neighbor: Rmax, and each MDR/BMDR neighbor u such that
hops2(u) equals infinity.

C. Min-Cost LSA Algorithm

This section describes the algorithm for determining which neighbors
to include in the router-LSA when LSAFullness is 1 or 2 (min-cost
LSAs). The algorithm assumes that a router may have multiple
interfaces, at least one of which is a MANET interface. The input to
this algorithm includes, for each MANET interface, the set of
routable neighbors and the the Reported Neighbor List (RNL) for each
bidirectional neighbor on the interface. The input also includes the
router-LSA originated by each bidirectional neighbor (on any
interface), and the network-LSA for each transit broadcast or NBMA
network to which any such neighbor is attached.

The output of the algorithm is the set of advertised neighbors to be
included in the router-LSA, for each MANET interface. The min-cost
LSA algorithm must be run to possibly originate a new router-LSA
whenever any of the following events occurs:

- The set of routable neighbors changes.
- The Reported Neighbor List or Report2Hop changes for a neighbor.

- A new router-LSA originated by a neighbor is received.

If LSAFullness = 1, then the min-cost LSA algorithm ensures that the
router-LSAs (of the router and its neighbors) provide at least one
shortest path between each pair of neighbors (including neighbors on
non-MANET interfaces). If LSAFullness = 2, then the algorithm
ensures that the router-LSAs provide at least two shortest paths
between each pair of neighbors. Although it is straightforward to
extend the algorithm to provide three or more paths, the algorithm is
described only for one or two paths. If more than two paths are
desired, then it is probably better to use full LSAs.
The min-cost LSA algorithm may also use metric information that may be advertised in Hellos. If this option is used, then each router will advertise the cost to each routable neighbor in Hello packets via an LLS TLV. (The format for advertising this information will be described in a future version of this draft.) If this option is used, then shortest paths that are calculated based on min-cost LSAs will have minimum cost in all cases, without any conditions on the metrics.

If this option is not used (i.e., metric information is not included in Hellos), then minimum-cost paths will still be calculated if the metrics for all neighbors on the same interface are equal, and min-hop paths will be calculated if all metrics in the network are equal.

For convenience, in the following description, the term "neighbor" will refer to a neighbor on any interface that is bidirectional (in state 2-Way or greater). Also, node 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. MNCM(j,k) is set to 1 if NCM(j,k) equals 1 for any MANET interface. MNCM(j,k) is also set to 1 if j appears in k's router-LSA and k appears in j's router-LSA. MNCM(j,k) is also set to 1 if j and k are both attached to the same transit broadcast or NBMA network, and the network-LSA for that transit network exists and includes both j and k. Otherwise, MNCM(j,k) is set to 0.

(2) Create the Hello cost matrix HCM as follows. For each pair j, k of routers such that j is a neighbor, and k is either a neighbor or node i itself:
   (a) If j and k are neighbors of each other (based on MNCM): If

router j is reporting neighbor metrics in its Hellos (this is optional), then set HCM(j,k) to the metric reported by router j to its neighbor k; otherwise set HCM(j,k) = 0. We assume that routers with multiple interfaces do not report metrics in Hellos. (The reporting of metrics in Hellos may be extended to multiple interfaces in a future version of this draft.)

(b) If j and k are not neighbors of each other (MNCM(j,k) = 0), set HCM(j,k) to LSInfinity.

(3) Create the LSA cost matrix LCM as follows. Initialize LCM(j,k) to LSInfinity for each pair of neighbors j and k. For each neighbor j:
   (a) Find the router-LSA originated by neighbor j. If the LSA does not exist in the database, examine the next neighbor.
For each point-to-point connection described in the router-LSA, set $LCM(j,k)$ to the metric for the connection, where $k$ is the neighbor advertised for the connection. If the router-LSA contains multiple connections to $k$ via different interfaces, set $LCM(j,k)$ to the smallest metric for these connections.

If the router-LSA for $j$ indicates that $j$ is attached to a transit broadcast or NBMA network, with metric $M$, and the network-LSA for this network exists, set $LCM(j,k)$ to $M$ for each neighbor $k$ that is included in the network-LSA.

Initialize the set of advertised neighbors to include all MANET neighbors in the Full state. Let $metric(i,k)$ denote the router's own metric to each MANET neighbor $k$. If $k$ is a neighbor on multiple interfaces, let $metric(i,k)$ be the smallest such metric.

For each pair $j$, $k$ of neighbors such that $k$ is a routable MANET neighbor and $j$ is either a routable MANET neighbor or a non-MANET neighbor in the Full state:

(a) Find the neighbor $u$ with the minimum value of $HCM(j,u) + LCM(u,k)$, if such a neighbor exists such that this sum is less than $HCM(j,k)$. (Note that $HCM(j,k) = LSInfinity$ if $j$ and $k$ are not neighbors of each other.) If multiple neighbors achieve this minimum value, choose the one that maximizes $(MDR \text{ Level}, \text{ RtrPri}, \text{ RID})$.

(b) If the router itself (node $i$) is currently advertising neighbor $k$ in its router-LSA: If either $HCM(j,i) + metric(i,k) < HCM(j,u) + LCM(u,k)$, or $HCM(j,i) + metric(i,k) = HCM(j,u) + LCM(u,k)$ and the router itself has a larger value of $(MDR \text{ Level}, \text{ RtrPri}, \text{ RID})$ than neighbor $u$, or if $u$ does not exist and $HCM(j,i) + metric(i,k) < HCM(j,k)$, add $k$ to the set of advertised neighbors ($k$ will continue to be advertised).

(c) Else (the router is not currently advertising neighbor $k$): If $HCM(j,i) + metric(i,k) < HCM(j,u) + LCM(u,k)$, or if $u$ does not exist and $HCM(j,i) + metric(i,k) < HCM(j,k)$, add $k$ to the set of advertised neighbors.

If the option of providing two paths between each pair of neighbors is elected ($LSAFullness = 2$), then the following step 6 is performed instead of step 5 above.

For each pair $j$, $k$ of neighbors such that $k$ is a routable MANET neighbor and $j$ is either a routable MANET neighbor or a non-MANET neighbor in the Full state:

(a) Choose neighbor $u$ as in step 5(a). Let $v$ be the neighbor, excluding node $u$ (if it exists), with the minimum value of $HCM(j,v) + LCM(v,k)$, if such a neighbor exists such that this sum is less than $LSInfinity$. If multiple neighbors achieve
this minimum value, choose the one that maximizes (MDR Level, RtrPri, RID).

(b) If the router itself (node i) is currently advertising neighbor k in its router-LSA: If either HCM(j,i) + metric(i,k) < HCM(j,v) + LCM(v,k), or HCM(j,i) + metric(i,k) = HCM(j,v) + LCM(v,k) and the router itself has a larger value of (MDR Level, RtrPri, RID) than neighbor v, or if v does not exist and HCM(j,i) + metric(i,k) < LSInfinity, add k to the set of advertised neighbors (k will continue to be advertised).

(c) Else (the router is not currently advertising neighbor k): If HCM(j,i) + metric(i,k) < HCM(j,v) + LCM(v,k), or if v does not exist and HCM(j,i) + metric(i,k) < LSInfinity, add k to the set of advertised neighbors.

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. 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 are needed 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). The retransmission interval should be slightly larger than MinLSInterval (e.g., MinLSInterval + 1) 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.

Authors' Addresses

Richard G. Ogier
SRI International
Email: rich.ogier@earthlink.net, richard.ogier@sri.com

Phil Spagnolo
Boeing Phantom Works
Email: phillip.a.spagnolo@boeing.com

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Ogier & Spagnolo Expires September 6, 2006 [Page 58]
Internet-Draft MANET Extension of OSPF March 2006

Copyright Statement

Copyright (C) The Internet Society (2006). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.