

Mobile Ad hoc Networking (MANET)
Internet-Draft
Intended status: Informational
Expires: December 17, 2010

C. Dearlove
BAE Systems Advanced Technology
Centre
T. Clausen
LIX, Ecole Polytechnique, France
P. Jacquet
INRIA, France
June 15, 2010

Link Metrics for OLSRv2
draft-dearlove-olsrv2-metrics-05

Abstract

This document describes how link metrics may be added, in a relatively straightforward manner, to the specification of OLSRv2, in order to allow routing by other than minimum hop count routes. In addition to metric signaling and use, the most significant change is a separation of the routing and flooding functions of MPRs.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 17, 2010.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1.	Introduction	4
2.	Applicability	6
3.	Motivational Scenarios	7
4.	Link Metrics	9
4.1.	Link Metric Types	9
4.2.	Directional Link Metrics	11
4.3.	Reporting Link and Router Metrics	12
4.4.	Defining Incoming Link Metrics	14
4.5.	Link Metric TLVs	14
4.6.	Link Metric Values	15
5.	MPRs with Link Metrics	17
5.1.	Flooding MPRs	17
5.2.	Routing MPRs	19
5.3.	Relationship Between MPR Sets	22
6.	Implementation	24
6.1.	Parameters and Constants	25
6.2.	Local Information Base	25
6.3.	Interface Information Base	26
6.4.	Neighbor Information Base	27
6.5.	Topology Information Base	27
6.6.	Metric Representation	28
6.7.	MPR Representation	30
6.8.	HELLO Message Generation	31
6.9.	HELLO Message Processing	31
6.10.	MPR Calculation and Neighbor Set Update	32
6.11.	TC Message Generation	33
6.12.	TC Message Processing	33
6.13.	Routing Set Calculation	33
7.	IANA Considerations	35
8.	Security Considerations	36
9.	References	37
9.1.	Normative References	37
9.2.	Informative References	37
Appendix A.	MPR Routing Property	38
Appendix B.	Routing MPR Calculation	40
Appendix C.	Example Algorithm for Calculating the Routing Set	43
C.1.	Local Interfaces and Neighbors	43
C.2.	Add Neighbor Routers	44
C.3.	Add Remote Routers	44
C.4.	Add Neighbor Addresses	45
C.5.	Add Remote Routable Addresses	46
C.6.	Add Attached Networks	46
C.7.	Add 2-Hop Neighbors	47
Appendix D.	Constraints	49
Appendix E.	Acknowledgements	51

1. Introduction

The Optimized Link State Routing Protocol [OLSRv2] is a proactive routing protocol for Mobile Ad hoc NETWORKS (MANETs) [RFC2501]. In its current form, this protocol finds shortest, defined as minimum number of hops, routes from a router to all possible destinations.

However limiting to minimum hop routes may yield what are, to the user, inferior routes. Some examples are given in Section 3. This limitation is not, however, fundamental to OLSRv2. First, the extensible message format [RFC5444] used by OLSRv2 naturally permits the addition of additional information regarding links to OLSRv2 messages. Second, OLSRv2 essentially first collects topological information from the network and then forms minimum length routes. Using a definition of route length (metric) other than number of hops is a natural extension that is commonly used in link state protocols.

Addition of alternative route selection processes to OLSRv2 could be treated as a possible future extension. However in this case, legacy OLSRv2 routers, which would not recognize any additional link information, would still attempt to use minimum hop-count routes. This would mean that, in effect, routers differed over their valuation of links and routes. This can lead to the fundamental routing problem of "looping", and must be avoided. Thus if alternative route selection were to be considered only as a future extension, then routers which did, and routers which did not, implement the extension could not interoperate. This would be a significant limitation of such an extension.

This document discusses a possible improvement to OLSRv2 which could be fairly straightforwardly incorporated in a revision of [OLSRv2]. The principal suggested changes to OLSRv2 are:

- o Assigning metrics to links. This involves considering separate metrics for the two directions of a link, with the receiving router determining the metric from transmitter to receiver. Directional metrics must be signaled in HELLO messages, and are also included in TC messages. Metrics may also be:
 - * A link metric, the metric of a specific link from an OLSRv2 interface of the transmitting router to an OLSRv2 interface of the receiving router.
 - * A router metric, the minimum of the link metrics between those routers, in the indicated direction.

These metrics are necessarily the same when these routers each have a single OLSRv2 interface, but may differ when either has

more. HELLO messages will include both link metrics and router metrics. A means of inclusion that avoids unnecessary repetition (and hence minimizes message bandwidth use) will be employed. TC messages will include router metrics only.

- o Metrics to be used in OLSRv2 are dimensionless and additive. The assignment of metrics, including their relationship to real parameters such as bandwidth, loss rate and delay, is outside the scope of OLSRv2, which simply would use these metrics in a consistent manner. However by use of a registry of metric types (employing extended types of a single address block TLV type), routers can use only metrics of the physical type that they are configured to use.
- o The separation of the two functions performed by MPRs in OLSRv2, optimized flooding and reduced topology advertisement for routing, into separate sets of MPRs, denoted "flooding MPRs" and "routing MPRs". Flooding MPRs can be calculated as MPRs currently are, but can improve the selection using metrics, while routing MPRs need a metric-aware selection algorithm. Examples of each MPR selection algorithm are given in this document. The selection of routing MPRs guarantees the use of minimum distance routes using the chosen metric, while still using only two hop neighborhood information from HELLO messages and routing MPR selector information in TC messages.
- o Appropriate changes to protocol Information Bases, messages (new link metric and modified MPR TLVs) and message processing. These are described in this document.

2. Applicability

The objective of this document is to serve as a proposal for a revision of [OLSRv2]. None of the changes proposed in this document affect any of the other constituent parts of OLSRv2, in particular they do not affect [NHDP], since as some uses of that protocol will not need metrics, they should not have metrics imposed on them.

The addition of metrics in this way to OLSRv2 would form a mandatory part of the specification. An implementation that is to interwork with all other implementations of OLSRv2, subject to any administrative configuration of choice of metric type, **MUST** fully implement this use of link metrics.

3. Motivational Scenarios

The basic situation that suggests the desirability of use of routes other than minimum hop routes is shown in Figure 1.

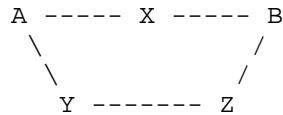


Figure 1

The minimum hop route from A to B is via X. However if the links A to X and X to B are poor (e.g., having low bandwidth or being unreliable) but the links A to Y, Y to Z and Z to B are better (e.g., having reliable high bandwidth) then the route A to B via Y and Z may be preferred.

There are other situations where, even if links do not show immediately obvious benefits to users, their use should be discouraged. Consider a network with many short range links, and a few long range links. Use of minimum hop routes will immediately lead to heavy use of the long range links. This will be particularly undesirable if those links achieve their longer range through reduced bandwidth, or through being less reliable. However, even if the long range links have the same characteristics as the short range links, it may be better to reserve usage of the long range links for when this usage is particularly valuable - for example when the use of one long range link saves several short range links, rather than the single link that is all that is needed to be saved for a minimum hop route.

A related case is that of a privileged relay. An example is an aerial router in an otherwise ground based network. The aerial router may have a link to many, or even all, other routers. That would lead to all routers attempting to send all their traffic (other than to immediate neighbors and some two hop neighbors) via the aerial router. It may however be important to reserve that capacity for cases where the aerial router is actually essential, such as if the ground based portion of the network is disconnected.

Other cases may involve attempts to avoid areas of congestion, to route around insecure routers (by preference, but prepared to use them if there is no other alternative) and routers attempting to discourage their use as relays due to, for example, limited battery power. OLSRv2 does have another mechanism to aid in this, a router's willingness to act as an MPR. However there are cases where that cannot help, but where use of non-minimum hop routes could.

Similarly note that OLSRv2's optional use of link quality (through its use of [NHDP]) is not a solution to these problems. Use of link quality as specified in [NHDP] allows a router to decline to use a link, not only on its own, but on all routers' behalf. It does not, for example, allow the use of a link otherwise determined to be too low quality to be generally useful, as part of a route where no better links exist. Note that these mechanisms (link quality and link metrics) solve different problems, and it is not suggested that use of link metrics will replace the use of link quality.

It should also be noted that the loop-free property of OLSRv2, and of this modification, apply strictly only in the static state. When the network topology is changing, and with possibly lossy messages, it is possible for transient loops to form. However with update rates appropriate to the rate of topology change, such loops will be sufficiently rare. Changing link metrics is a form of network topology change, and should be limited to a rate slower than the message information update rate (defined by the parameters HELLO_INTERVAL, HELLO_MIN_INTERVAL, TC_INTERVAL and TC_MIN_INTERVAL).

4. Link Metrics

Using the approach suggested here, link metrics will be:

- o As used by OLSRv2, dimensionless. While they may, directly or indirectly, correspond to specific physical information (such as delay, loss rate or bandwidth), this knowledge will not be used by OLSRv2. Instead, generating the metric value will be the responsibility of a mechanism external to OLSRv2.
- o Additive, so that the metric of a route is the sum of the metrics of the links forming that route. Note that this requires a metric where a low value of a link metric indicates a "good" link and a high value of a link metric indicates a "bad" link, where the former will be preferred to the latter.
- o Directional, the metric from router A to router B need not be the same as the metric from router B to router A, even when using the same OLSRv2 interfaces. At router A, a link metric from router B to router A is referred to as an incoming link metric, while a link metric from router A to router B is referred to as an outgoing link metric. (These are, of course, reversed at router B.)
- o Specific to a pair of OLSRv2 interfaces, so that if there is more than one link from router A to router B, each has its own link metric in that direction. There will also be an overall metric, a "router metric", from router A to router B. This will be the minimum value of the link metrics from router A to router B, considering symmetric links only; it is undefined if there are no such symmetric links. A router metric from one router to another is always equal to a link metric in the same direction between OLSRv2 interfaces of those routers. When referring to a specific OLSRv2 interface (for example in a Link Tuple or a HELLO message sent on that OLSRv2 interface) a link metric always refers to a link on that OLSRv2 interface, to or from the indicated 1-hop neighbor OLSRv2 interface, while a router metric may be equal to a link metric to and/or from another OLSRv2 interface.

These aspects of link and router metrics are discussed in the following sections.

4.1. Link Metric Types

There are various physical characteristics that may be used to define a link metric. Some examples, which also illustrate some characteristics of metrics that result, are:

- o Delay is a straightforward metric, as it is naturally additive, the delay of a multi-link route is the sum of the delays of the links. (This does not directly take into account delays due to routers, rather than links, but these can be divided among incoming and outgoing links.) However given a limited range of link metric value (as must be used) more than one type of delay metric may be required, representing different ranges of delay value.
- o Probability of loss on a link is, as long as probabilities of loss are small and independent, approximately additive. (A slightly more accurate approach is using a negatively scaled logarithm of the probability of not losing a packet.) If losses are not independent then this will be pessimistic. Again, more than one range of values (or, equivalently, more than one scaling of the logarithms) may be needed.
- o Bandwidth is not additive, it even has the wrong characteristic of being good when high, bad when low; thus a mapping that inverts its ordering must be applied. Such a mapping can, at best, only produce a metric that it is acceptable to treat as additive. Consider, for example, a preference for a route that maximizes the minimum bandwidth link on the route, and then prefers a route with the fewest links of each bandwidth from the lowest. If links may be of three discrete bandwidths, "high", "medium" and low", then this preference can be achieved, on the assumption that no route will have more than 10 links, with metric values of 1, 10 and 100 for the three bandwidths. If routes can have more than 10 links, the range of metrics must be increased; this indicates a preference for a wide "dynamic range" of link metric values. Depending on the ratios of the numerical values of the three bandwidths, the same effect may be achieved by using a scaling of an inverse power of the numerical values of the bandwidths. For example if the three bandwidths were 2, 5 and 10 Mbit/s, then a possible mapping would be the fourth power of 10 Mbit/s divided by the bandwidth, giving metric values of 625, 16 and 1 (good for up to 16 links in a route). This mapping can be extended to a system with more bandwidth values, for example giving a 4 Mbit/s bandwidth a metric value of about 39. This may lose the capability to produce an absolutely maximum minimum bandwidth route, but will usually produce either that, or something close (and at times maybe better, is a route of three 5 Mbit/s links really better than one of a single 4 Mbit/s link?) Specific metrics will need to define the mapping (e.g., a power and bandwidth scaling).

There are also many other possible metrics, including physical layer information (such as signal to noise ratio, and error control

statistics) and information such as packet queuing statistics.

In a well-designed network, all routers will use the same physical metric type. It will not produce good routes if, for example, some link metrics are based on bandwidth and some on path loss (except to the extent that these may be correlated). How to achieve this is an administrative matter, outside the scope of OLSRv2. In fact even the actual physical meanings of the metrics will be outside the scope of OLSRv2. This is because new metrics may be added in the future, for example as bandwidths increase, and may be based on new, possibly non-technical, considerations, for example financial cost. Each such type will have a metric type number (whose range is considered later). Initially a single link metric type zero will be defined as indicating a dimensionless metric with no predefined physical meaning.

An OLSRv2 router will then be instructed which single link metric type to use and recognize, without knowing whether it represents delay, probability of loss, bandwidth, cost or any other quantity. This recognized link metric type number will be a router parameter, and subject to change in case of reconfiguration, or possibly the use of a protocol (outside the scope of OLSRv2) permitting a process of link metric type agreement between routers.

The use of link metric type numbers also suggests the possibility of use of multiple link metric types and multiple network topologies. This is a possible future extension to OLSRv2, but is not included in this proposal. To allow for that future possibility, the sending of more than one metric, of different physical types, which should not be done for reasons of efficiency, will however not be forbidden, but other types than that configured will be ignored.

The following three sections assume a chosen single link metric type, of unspecified physical nature. The selection of that type is described in Section 4.5.

4.2. Directional Link Metrics

OLSRv2 uses only "symmetric" (bidirectional) links, which may pass traffic in either direction. A key decision is whether these links should each be assigned a single metric, used in both directions, or a metric in each direction, noting that:

- o Links can have different characteristics in each direction, use of directional link metrics recognizes this.
- o In many (possibly most) cases, the two ends of a link will naturally form different views as to what the link metric should

be. To use a single link metric requires a coordination between the two that can be avoided if using directional links. Note that if using a single metric, it would be essential that the two ends agree as to its value, otherwise it is possible for looping to occur. This problem does not occur for directional metrics.

Based on these considerations, directional metrics are preferred. Each router must thus be responsible for defining the metric in one direction only. This could be in either direction, i.e., that a router is responsible for either incoming or outgoing link metrics, as long as the choice is universal. The former (incoming) case is used because, in general, receiving routers have more information available to determine link metrics (for example received signal strength, interference levels and error control coding statistics).

Note that, using directional metrics, if router A defines the metric of the link from router B to router A, then router B must use router A's definition of that metric on that link in that direction. (Router B could, if appropriate, use a bad mismatch between directional metrics as a reason to discontinue use of this link, using the link quality mechanism in [NHDP].)

4.3. Reporting Link and Router Metrics

Links, and hence link metrics, will be reported in HELLO messages. A router must report incoming link metrics in its HELLO messages in order that these are each available at the corresponding other end of the link. This will mean that, for a symmetric link, both ends of the link will know both link metrics.

Incoming link metrics in HELLO messages are not however sufficient. In addition, when different, incoming router metrics are also required. These are used for routing MPR selection (see Section 5.2). Router metrics, not just link metrics, are required, at least for symmetric neighbors, because in general a router will not receive HELLO messages sent on all of its 1-hop neighbor's OLSRV2 interfaces.

Metrics will also be reported in TC messages. It can be shown that these need to be outgoing metrics:

- o Router A must be responsible for advertising a metric from router A to router B in TC messages. This can be seen by considering a route connecting single OLSRV2 interface routers P to Q to R to S. Router P receives its only information about the link from R to S in the TC messages transmitted by router R, which is an MPR of router S (assuming that only MPR selectors are reported in TC messages). Router S may not even transmit TC messages (if no

routers have selected it as an MPR and it has no attached networks to report). So any information about the metric of the link from R to S must also be included in the TC messages sent by router R, hence router R is responsible for reporting the metric for the link from R to S.

- o In a more general case, where there may be more than one link from R to S, the TC message must, in order that minimum metric routes can be constructed (e.g., by router P) report the minimum of these link metrics, i.e., the outgoing router metric from R to S.

In this example, router P also receives information about the existence of a link between Q and R in the HELLO messages sent by router Q. Without the use of metrics, this link may be used by OLSRv2 for two hop routing to router R using just HELLO messages sent by router Q. Assuming that this property (which accelerates local route formation) is to be retained, router P must receive the metric from Q to R in HELLO messages sent by router Q. This indicates that router Q must be responsible for reporting the metric for the outgoing link from Q to R. This is in addition to the incoming link metric information that a HELLO message must report.

This leaves two possible design choices:

- o HELLO messages can report only incoming metrics. Link metrics are required for links on this OLSRv2 interface, i.e., with a LINK_STATUS TLV, and only when indicating HEARD or SYMMETRIC. Router metrics are required when different, and when indicating SYMMETRIC using either a LINK_STATUS or OTHER_NEIGHB TLV. However this would prevent the use of two hop routes informed only by HELLO messages, and would be a change to OLSRv2.
- o HELLO messages can also report outgoing router metrics. This is required when indicating SYMMETRIC using either a LINK_STATUS or OTHER_NEIGHB TLV. This would allow the use of two hop routes using HELLO messages only.

Accelerated two hop route formation is a feature of OLSRv2 it would be unfortunate to lose, and hence the latter approach has been adopted. In addition, Section 5.1 offers an additional reason for reporting outgoing router metrics, without which metrics can properly affect only routing, not flooding.

Note that there is no need to report an outgoing link metric. The corresponding 1-hop neighbor knows that value, it specified it, and for 2-hop neighborhood use router metrics are required (as these will, in general, not use the same OLSRv2 interface).

4.4. Defining Incoming Link Metrics

When a router reports a 1-hop neighbor in a HELLO message it may do so for the first time with link status HEARD. The receiving router may then immediately consider the link to be symmetric and thus will use it.

As the router is responsible for defining and reporting incoming link metrics, it must evaluate that metric, and attach that link metric to the appropriate address (which will have link status HEARD) in the next HELLO message reporting that address on that OLSRv2 interface. There will be no outgoing link metric available to report.

Thus a router must be able to immediately decide on an incoming link metric once it has heard a neighbor on an OLSRv2 interface for the first time. This is because, on receiving a HELLO message from this router, that neighbor will (unless link quality indicates otherwise) immediately consider the link to be symmetric and use it. It may, depending on the physical nature of the link metric, be too early for an ideal decision as to that metric, however a choice must be made (even if only that a default value is used). The metric value may later be refined based on further observation of HELLO messages, other message transmissions between the routers, or other observations of the environment. It will probably be best to over-estimate the metric if initially uncertain as to its value, to discourage, rather than over-encourage, its use.

4.5. Link Metric TLVs

Metric values will naturally be reported using a new address block TLV, here named LINK_METRIC. This is used for both link metrics and router metrics. Indicating the different types of metric: physical, directional and link/router metric, will require the use of a TLV extended type to represent the type of the metric. The two least significant bits of the TLV type extension, will be allocated to manage direction and link/router metric, with options to allow common values (which may arise by accident, or due to using a bandwidth metric with a limited number of values) to be reported efficiently. The remaining most significant six bits of the type extension will be the link metric type, defined by the router parameter LINK_METRIC_TYPE, which must be in the range 0 to 63, inclusive.

Link metric types, and their physical meaning and mapping, will be allocated by IANA. Link metric types 56 to 63 will be for private/experimental use, and 1 to 55 will be allocated by expert review. Link metric type 0 will be defined by OLSRv2 as described below (thus also allowing an interoperable implementation of OLSRv2 with no further link metric type definitions and allocations).

The value field of the LINK_METRIC TLV, which may be multivalued, will be as described in Section 4.6. There will also be a default metric value, and a LINK_METRIC TLV with that value may be omitted, and if a link metric is required, but no LINK_METRIC TLV of the appropriate type is present, then that default value will be assumed.

A message TLV, of type LINK_METRIC_EXTENSION is also defined. A message may not include more than one such TLV. It takes a single octet value, which represents the default LINK_METRIC type extension. Its most significant six bits must be the router parameter LINK_METRIC_TYPE. It may also have its two least significant bits set to any value. If there is no LINK_METRIC_EXTENSION TLV, then one with value zero is assumed.

Any LINK_METRIC TLV with no type extension is treated as having a type extension equal to the value of the LINK_METRIC_EXTENSION TLV in that message. Any LINK_METRIC TLV with a type extension whose most significant six bits are all zero replaces those six bits with the most significant six bits of the value of the LINK_METRIC_EXTENSION TLV.

The use of the LINK_METRIC_EXTENSION TLV may be illustrated by assuming that physical link type N is to be used. Then in a HELLO message, where all LINK_METRIC TLVs should have type extensions 4N, 4N+1, 4N+2 or 4N+3, these TLVs can instead have type extensions 0, 1, 2 and 3, and the first of these can be omitted. In a TC message, where all LINK_METRIC TLVs should have type 4N+2, a single LINK_METRIC_EXTENSION TLV can have value 4N+2, and all LINK_METRIC TLV type extensions can be omitted.

For a network which does not use link metrics, simply omitting a LINK_METRIC_EXTENSION TLV and all LINK_METRIC TLVs uses only default values of a dimensionless metric, i.e., is equivalent to using hop count, with no additional overhead. However a router in such a network MUST still recognize and use link metrics in the event that other routers use values other than the default values.

4.6. Link Metric Values

In keeping with the requirement that OLSRV2 can be unaware of the details of metric values (which may be defined in the future) a single link metric value definition is required.

As previously noted, a reasonably wide dynamic range of link metrics is desirable. On the other hand, link metrics that occupy no more than one octet are also desirable for message size reasons. One approach that includes both requirements is already in use in OLSRV2, for time values, as described in [RFC5497]. This specifies a value

using a mantissa and exponent, together occupying 8 bits. For link metric purposes a 4 bit mantissa and a 4 bit exponent is suggested here. ([RFC5497] uses a 3 bit mantissa and a 5 bit exponent, offering increased range but reduced precision.) This would be used so that the transmitted octet $16*b + a$ represents the value $(1 + a/16) * 2^b$. This would then represent values from a minimum of 1 to a maximum of 63488. However this also allows fractional metrics, so for convenience it is suggested that the metric value range used is considered to be from a minimum of 16 to a maximum of $16 * 63488 (= 1015808)$, i.e., that $16*b + a$ represents the value $(16 + a) * 2^b$. Note that this rescaling has no effect on message contents or performance. The limiting values of the metric will be defined as the constants `MINIMUM_METRIC` (16) and `MAXIMUM_METRIC` (1015808) to allow their more convenient use. (It is recommended that all mappings from real parameters to link metric values are specified using these constants by name.)

As noted above, in order that metric use can be most efficient, a default value is needed. This also should be type-independent. It is suggested that this is in the centre of the above range logarithmically; the closest representable value is 4096 ($a = 0, b = 8$). This will be defined as the constant `DEFAULT_METRIC`. It is also suggested that route metric summation should be exact. Since a route cannot have more than 255 links, 28 bit (or more, in practice probably 32 bit) arithmetic can be used.

Which is the better flooding MPR selection by router A: B or C? If the metric represents probability of message loss, then clearly choosing B maximizes the probability of a message sent by A reaching D. This is despite that C has a lower metric in its connection to A than B does. (Similar arguments about a preference for B can be made if, for example, the metric represents bandwidth or delay rather than probability of loss.)

However, neither should only the second hop be considered. If this example is modified to that in Figure 3:

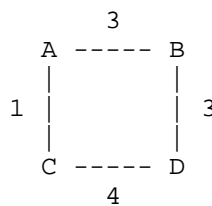


Figure 3

then it is possible that, when A is selecting flooding MPRs, selecting C is preferable to selecting B. If the metrics represent scaled values of delay, or the probability of loss, then selecting C is clearly better. This indicates that the sum of metrics is an appropriate measure to use to choose between B and C.

However, this is a particularly simple example. Usually it is not a simple choice between two routers as a flooding MPR, each only adding one router coverage. A more general process, when considering which router to next add as a flooding MPR, should incorporate the metric to that router, and the metric from that router to each symmetric strict 2-hop neighbor, as well as the number of newly covered symmetric strict 2-hop neighbors as well as the other factors used in the example algorithm in [OLSRv2].

Note that, as in [OLSRv2], each router can make its own independent choice of flooding MPRs, and flooding MPR selection algorithms, and still interoperate. A possible algorithm, representing a modification of the current algorithm in [OLSRv2] (and reducing to it when all metrics are equal) is suggested in Section 6.10.

Note that the references above to the direction of the metrics is correct: for flooding, directional metrics outward from a router are appropriate, i.e., metrics in the direction of the flooding. This is an additional reason for including outward metrics in HELLO messages, as otherwise a metric-aware MPR selection for flooding is not possible. The second hop metrics are outgoing router metrics because

the OLSRv2 interface used for a second hop transmission may not be the same as that used for the first hop reception.

5.2. Routing MPRs

The essential detail of the current MPR specification in [OLSRv2] is that a router must, per OLSRv2 interface, select a set of MPRs such that there is a two hop route from each symmetric strict 2-hop neighbor of the selecting router to the selecting router, with the intermediate router on each such route being an MPR of the selecting router.

It is sufficient, when using an additive link metric rather than a hop count, to require that these "routing MPRs" provide not just a two hop route, but a minimum distance two hop route. In addition, the concept of symmetric strict 2-hop neighbor needs an adjustment. A router is a symmetric strict 2-hop neighbor even if it is a symmetric 1-hop neighbor, as long as there is a two hop route from it that is shorter than the one hop link from it. (The property that no routes go through routers with willingness `WILL_NEVER` is retained. Examples below assume that all routers are equally willing, with none having willingness `WILL_NEVER`.)

For example, consider the network in Figure 4. Numbers are metrics of links towards router A, away from router D. Router A must pick router B as a routing MPR, whereas for minimum hop count routing it could alternatively pick router C. Note that the use of incoming router metrics in this case follows the same reasoning as for the directionality of metrics in TC messages, as described in Section 4.3.

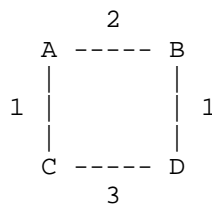


Figure 4

In Figure 5, router A must pick router B as a routing MPR, but for minimum hop count routing it would not need to pick any MPRs.

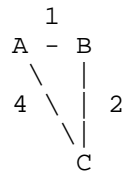


Figure 5

In Figure 6, router A must pick both routers B and C as routing MPRs, but for minimum hop count routing it could pick either.

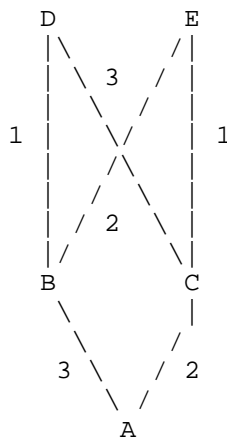


Figure 6

It is shown in Appendix A that selecting routing MPRs according to this definition, and advertising only such links (plus knowledge of local links from HELLO messages), will result in selection of shortest routes, even if all links are considered in the definition of a shortest route.

However the definition noted above as sufficient for routing MPR selection is not necessary. For example, consider the network in Figure 7. (The metrics from B to C and C to B are both assumed to be 2.)

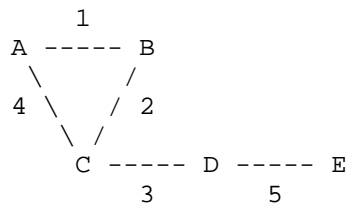


Figure 7

Using the above definition, A must pick both B and C as routing MPRs, in order to cover the symmetric strict 2-hop neighbors C and D, respectively. (C is a symmetric strict 2-hop neighbor because the route length via B is shorter than the 1-hop link.)

However, A only needs to pick B as a routing MPR, because the only reason to pick C as a routing MPR would be so that C can advertise the link to A for routing - to be used by, for example, E. But A knows that no other router should use the link C to A in a shortest route, because routing via B is shorter. So if there is no need to advertise the link from C to A, then there is no reason for A to select C as a routing MPR.

This process of "thinning out" the routing MPR selection uses only local information from HELLO messages. Using any minimum distance algorithm, the router identifies shortest routes, whether one, two or more hops, from all routers in its symmetric strict 2-hop neighborhood. It then selects as MPRs all symmetric strict 1-hop neighbors that are the last router (before the selecting router itself) on any such route. Where there is more than one shortest distance route from a router, only one such route is required. Alternative routes may be selected so as to minimize the number of last routers - this is the equivalent to the selection of a minimal set of MPRs in the non-metric case. An example of how to perform this in practice is given in Appendix B.

Note that, compared to the first proposed approach, this only removes routing MPRs whose selection can be directly seen to be unnecessary. Consequently if (as is shown in Appendix A) the first approach creates minimum distance routes, then so does this revised process.

Note that the examples in Figure 5 and Figure 6 show that use of link metrics may require a router to select more routing MPRs than when not using metrics, and even require a router to select routing MPRs when without metrics it would not need any routing MPRs. This may result in more, and larger, messages being generated, and forwarded more often. Thus the use of link metrics is not without cost, even excluding the cost of link metric signaling. There is however no

cost (in message size or number of messages) if all link metrics are default valued and no link metric TLV is used.

These examples consider only single OLSRV2 interface routers. However if routers have more than one OLSRV2 interface, then the process is unchanged, other than that if there is more than one known metric between two routers (on different OLSRV2 interfaces), then, considering symmetric links only (as only these are used for routing) the smallest link metric, i.e., the router metric, must be used. There is no need to calculate routing MPRs per OLSRV2 interface. That requirement results from the consideration of flooding and the need to avoid certain "race" conditions, which are not relevant to routing.

5.3. Relationship Between MPR Sets

It would be convenient if the two sets of flooding and routing MPRs were the same. This can be the case if all metrics are equal (whether to the default value or any other value), but in general, for "good" sets of MPRs they are not. (A reasonable definition of this is that there is no common minimal set of MPRs.) If metrics are asymmetrically valued (the two sets of MPRs use opposite direction metrics), or routers have multiple OLSRV2 interfaces (where routing MPRs can ignore this, but flooding MPRs cannot) this is particularly unlikely. However even using a symmetrically valued metric with a single OLSRV2 interface on each router, the sets are not equal, nor is one always a subset of the other. To show this, consider these examples, where all lettered routers are assumed equally willing to be MPRs, and numbers are bidirectional metrics for links.

In Figure 8, A does not require any flooding MPRs. However A must select B as a routing MPR.

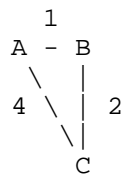


Figure 8

In Figure 9, A must select C and D as routing MPRs. However A's minimal set of flooding MPRs is just B. In this example the set of routing MPRs will serve as a set of flooding MPRs, but a non-minimal one (although one that might be better, depending on the relative importance of number of MPRs and flooding link metrics).

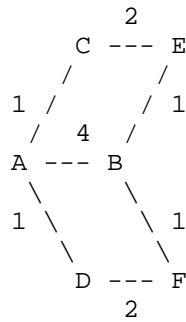


Figure 9

However, this is not always the case. In Figure 10, A's set of routing MPRs must contain B, but need not contain C. A's set of flooding MPRs need not contain B, but must contain C. (In this case, flooding with A selecting B rather than C as a flooding MPR will reach D, but in three hops rather than the minimum two that MPR flooding guarantees.)

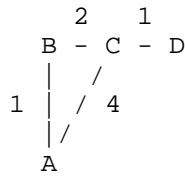


Figure 10

6. Implementation

Implementation of metrics in OLSRv2 requires the following additions to [OLSRv2]:

- o Definition of the constant minimum, maximum and default metric values `MINIMUM_METRIC`, `MAXIMUM_METRIC` and `DEFAULT_METRIC`, and the mapping between metric values and their single octet representation. In some cases a metric value is defined as "unspecified". In this case either that metric value **MUST** be excluded from all comparisons, or the unspecified value **MUST** be considered to be a value greater than or equal to `MAXIMUM_METRIC`.
- o Definition of the router parameter `LINK_METRIC_TYPE`.
- o Addition of link and router metric information to the Local Information Base, the Interface Information Base, the Neighbor Information Base and the Topology Information Base.
- o Modifications to the Interface Information Base and Neighbor Information Base to reflect the two types of MPRs to be used.
- o A `LINK_METRIC` Address Block TLV to represent metrics, to handle incoming and outgoing/agreed cases and alternative link metric types.
- o A `LINK_METRIC_EXTENSION` Message TLV to allow the simplification of the representation of the metric types in a message.
- o A modification of the TLV to represent MPRs, to report both routing and flooding MPR status.
- o HELLO message generation to add metrics and both MPR types.
- o HELLO message processing to use metrics and both MPR types.
- o Separate routing and flooding MPR calculations and update of the Neighbor Set.
- o TC message generation to add metrics.
- o TC message processing to use metrics.
- o Routing Set updates to use metrics.

These changes are summarized in the following sections. Updates to the constraints that apply to the Information Bases are summarized in Appendix D.

6.1. Parameters and Constants

The constant minimum, maximum and default metric values are defined by:

- o MINIMUM_METRIC := 16
- o MAXIMUM_METRIC := 1015808
- o DEFAULT_METRIC := 4096

The router parameter LINK_METRIC_TYPE may take any value from 0 to 63 inclusive. If this router parameter is changed, then all protocol sets which contain link metric information (i.e., all those updated in the following sections) MUST have all of their contents immediately removed, except that Link Tuples that are not pending should instead be updated by:

- o L_HEARD_time := EXPIRED
- o L_SYM_time := EXPIRED

The usual consequences of a Link Tuple no longer being symmetric, if it was, and of timeout of being heard, must be applied. The former of these will include, in all cases:

- o L_mpr_selector := false

and the latter will include, in all cases:

- o L_in_metric := unspecified
- o L_out_metric := unspecified

6.2. Local Information Base

Each Local Attached Network Tuple, defined in [OLSRv2], will need one additional element:

AL_metric is the metric of the link to the attached network with address AL_net_addr from this router;

This could replace the existing AL_dist element, however in order that the R_dist elements in a Routing Set can be set correctly (as there may be an external use for these) the AL_dist element has been retained, and hence also the hop count value in the GATEWAY TLV. Attached networks have not been discussed in this document up to this point, but they will behave very similarly to as currently defined in

[OLSRv2], with appropriate use of this metric.

6.3. Interface Information Base

Each Link Tuple, defined in [OLSRv2] by reference to [NHDP], will need three additional elements:

`L_in_metric` is the metric of the link from the OLSRv2 interface with addresses `L_neighbor_iface_addr_list` to this OLSRv2 interface;

`L_out_metric` is the metric of the link to the OLSRv2 interface with addresses `L_neighbor_iface_addr_list` from this OLSRv2 interface;

`L_mpr_selector` is a boolean flag, describing if this neighbor has selected this router as a flooding MPR, i.e., is a flooding MPR selector of this router.

`L_in_metric` will be specified by a process outside the OLSRv2 specification, similarly to `L_quality`. This MUST be done whenever a Link Tuple is created, except with `L_status = PENDING`, or the Link Tuple changes to have `L_status = HEARD` or `L_status = SYMMETRIC`. If all links are to have the same metric value then the value of `DEFAULT_METRIC SHOULD` be used. If `L_status` changes to `LOST` then `L_in_metric` MUST be set as unspecified. If `L_in_metric` is set or changed, then the corresponding `N_in_metric` MUST be updated by:

- o If there is no old `N_in_metric`, or if the new `L_in_metric` is less than the old `N_in_metric`, then set the new `N_in_metric` to the new `L_in_metric`.
- o Otherwise, if the old `L_in_metric` is equal to the old `N_in_metric`, and the new `L_in_metric` is greater than the old `N_in_metric`, then set the new `N_in_metric` to the minimum of all corresponding `L_in_metric` values, including the new `L_in_metric`.

`L_out_metric` will be defined by this protocol. When a Link Tuple is created, the default value of `L_out_metric` MUST be set as unspecified. If any `L_out_metric` is set or changed, then the corresponding `N_out_metric` MUST be updated by:

- o If there is no old `N_out_metric`, or if the new `L_out_metric` is less than the old `N_out_metric`, then set the new `N_out_metric` to the new `L_out_metric`.
- o Otherwise, if the old `L_out_metric` is equal to the old `N_out_metric`, and the new `L_out_metric` is greater than the old `N_out_metric`, then set the new `N_out_metric` to the minimum of all corresponding `L_out_metric` values, including the new `L_out_metric`.

Each 2-Hop Tuple, defined in [OLSRv2] by reference to [NHDP], will need two additional elements:

`N2_in_metric` is the router metric from the router with address `N2_2hop_iface_addr` to the router with OLSRv2 interface addresses `N2_neighbor_iface_addr_list`;

`N2_out_metric` is the router metric to the router with address `N2_2hop_iface_addr` from the router with OLSRv2 interface addresses `N2_neighbor_iface_addr_list`;

6.4. Neighbor Information Base

Each Neighbor Tuple, defined in [OLSRv2] by reference to [NHDP], will need five additional or modified elements:

`N_in_metric` is the router metric of any link from this neighbor to this router, i.e., the minimum of all corresponding `L_in_metric` with `L_status = SYMMETRIC`, unspecified if there are no such Link Tuples;

`N_out_metric` is the router metric of any link from this router to this neighbor, i.e., the minimum of all corresponding `L_out_metric` with `L_status = SYMMETRIC`, unspecified if there are no such Link Tuples;

`N_routing_mpr` is a boolean flag, describing if this neighbor is selected as a routing MPR by this router;

`N_flooding_mpr` is a boolean flag, describing if this neighbor is selected as a flooding MPR by this router;

`N_mpr_selector` is a boolean flag, describing if this neighbor has selected this router as a routing MPR, i.e., is a routing MPR selector of this router.

Note that flooding MPR selector status is recorded in the Link Sets, not in the Neighbor Set, so that the meaning of `N_mpr_selector` changes (is partly moved to `L_mpr_selector`). `N_routing_mpr` and `N_flooding_mpr` replace `N_mpr`.

6.5. Topology Information Base

Each Router Topology Tuple, defined in [OLSRv2], will need one additional element:

TR_metric is the router metric from the router with originator address TR_from_orig_addr to the router with originator address TR_to_orig_addr.

Each Routable Address Topology Tuple, defined in [OLSRv2], will need one additional element:

TA_metric is the router metric from the router with originator address TA_from_orig_addr to the router with OLSRV2 interface address TA_dest_addr.

Each Attached Network Tuple, defined in [OLSRv2], will need one additional element:

AN_metric is the metric of the link from the router with originator address AN_orig_addr to the attached network with address AN_net_addr.

The existing AN_dist element is retained, as for AL_dist in the Local Attached Network Tuple.

Each Routing Tuple, defined in [OLSRv2], will need one additional element:

R_metric is the metric of the route to the destination with address R_dest_addr.

The R_dist element has been retained as well as adding R_metric. It is outside the scope of OLSRV2 to specify how R_dist and/or R_metric may be used when the Routing Set is used to update the IP routing table or for any other purpose.

6.6. Metric Representation

Both HELLO messages and TC messages will need to associate metric values with neighbor addresses that they report. These metric values will have a type defined by router parameter LINK_METRIC_TYPE. This in turn will define the most significant six bits of a TLV type extension, where the least significant two bits define the direction of the metric, and whether this is a link metric or a router metric according to Table 1 for a HELLO message. Note that all type extensions considered here are after modification as described in Section 4.5 if a LINK_METRIC_EXTENSION message TLV is present.

Value	Interpretation
0	incoming link metric (L_in_metric) and outgoing router metric (N_out_metric)
1	incoming link metric (L_in_metric)
2	outgoing router metric (N_out_metric)
3	incoming router metric (N_in_metric)

Table 1: Interpretation of the least significant two bits of LINK_METRIC TLV type extension

Case 0 allows the representation of cases 1 and 2 in a single TLV when these values are equal. If case 3 is omitted, then the incoming router metric is assumed to equal the incoming link metric for that address. If either of cases 1 or 2 is omitted then the corresponding value is assumed to equal DEFAULT_METRIC.

For a TC message, the only metric values required are outgoing router metrics, N_out_metric values.

A HELLO message is discarded silently if any indicated metric values are inconsistent. This MUST be done for the link metric type LINK_METRIC_TYPE, it MAY be done for other link metric types. Inconsistency of values, of a single link metric type, is indicated, if for any address (including the same or different copies of that address) either of the following is indicated:

- o More than one value of a metric with the same kind (link or node) and in the same direction (incoming or outgoing).
- o A value of link metric is less than the corresponding value of router metric. As only incoming link metrics are reported, this applies only in that direction. This includes when either of these values is determined using the value DEFAULT_METRIC rather than being reported in a LINK_METRIC TLV.

A TC message MUST be silently discarded if any address is associated with more than one outgoing router metric value for link metric type LINK_METRIC_TYPE. It MAY be silently discarded if any address is associated with more than one outgoing router metric value for any other link metric type.

A metric value in a HELLO message may be represented in any way that

when following the above rules provides that value directly, or by the indicated alternatives. The expected combinations of least significant bits of type extensions used are:

- o None (with L_in_metric = N_in_metric = N_out_metric = DEFAULT_METRIC).
- o 0 (with L_in_metric = N_in_metric = N_out_metric).
- o 1 (with L_in_metric = N_in_metric and N_out_metric = DEFAULT_METRIC).
- o 2 (with L_in_metric = N_in_metric = DEFAULT_METRIC).
- o 1 and 2 (with L_in_metric = N_in_metric).
- o 0 and 3 (with L_in_metric = N_out_metric).
- o 1 and 3 (with N_out_metric = DEFAULT_METRIC).
- o 2 and 3 (with L_in_metric = DEFAULT_METRIC).
- o 1, 2 and 3.

A metric value in a TC message is expected to be represented with type extension with least significant two bits 2 (or omitted if N_out_metric = DEFAULT_METRIC).

In all cases, association with a LINK_METRIC TLV may be with a TLV covering a single or multiple addresses, and in the latter case with a single or multiple values.

6.7. MPR Representation

The current (in [OLSRv2] single TLV which reports MPR status will need to report both routing and flooding MPR status. For flooding MPRs it should do so only for addresses that have a symmetric link on the reporting OLSRV2 interface.

Rather than using separate TLVs, it is suggested that two extended types are used to represent these two types, and a third extended type is used to indicate both. The most efficient type extension, zero, could be used to represent both when a LINK_STATUS TLV with Type = SYMMETRIC is present, but to represent only routing MPR status when only an OTHER_NEIGHB TLV with Type = SYMMETRIC is present.

6.8. HELLO Message Generation

The following additional reporting by a HELLO message is required. Link metric association is as previously described (i.e., may be combined and/or omitted when default as appropriate).

- o Each included address from an `L_neighbor_iface_addr_list` with an associated `LINK_STATUS` TLV with Value = HEARD or Value = SYMMETRIC must have an associated incoming link metric for `L_in_metric`.
- o Each included address from an `N_neighbor_addr_list` with an associated `LINK_STATUS` or `OTHER_NEIGHB` TLV with Value = SYMMETRIC must have associated router metrics for `N_in_metric` and for `N_out_metric`.
- o At least one included address from each `L_neighbor_iface_addr_list` with an associated `LINK_STATUS` TLV with Value = SYMMETRIC must have an associated MPR TLV indicating flooding MPR status if and only if the corresponding `N_flooding_mpr` = true.
- o At least one included address from each `N_neighbor_addr_list` with an associated `LINK_STATUS` or `OTHER_NEIGHB` TLV with Value = SYMMETRIC must have an associated MPR TLV indicating routing MPR status if and only if the corresponding `N_routing_mpr` = true.

Routing and flooding MPR indications can be combined when appropriate.

6.9. HELLO Message Processing

Processing a HELLO message has the following extra steps:

- o When adding or updating a Link Tuple when the HELLO message includes an address of the receiving OLSRv2 interface with a `LINK_STATUS` TLV:
 - * If the reported status is HEARD or SYMMETRIC, then the appropriate `L_out_metric` must be set to the value of any incoming (to the sending router) link metric of the appropriate type associated with this address using a `LINK_METRIC` TLV. If there is no such TLV then `L_out_metric` is set to `DEFAULT_METRIC`.
 - * If the reported status is LOST then `L_out_metric` is set as unspecified.

The corresponding `N_out_metric` must also be updated if necessary.

- o All 2-Hop Tuples that are added or updated by the HELLO message also have their `N2_in_metric` updated to the value of any associated incoming (to the sending router) router metric value of the appropriate type associated with this address using a `LINK_METRIC` TLV. If there is no such TLV then `N2_in_metric` is set to `DEFAULT_METRIC`.
- o All 2-Hop Tuples that are added or updated by the HELLO message also have their `N2_out_metric` updated to the value of any associated outgoing (to the sending router) router metric value of the appropriate type associated with this address using a `LINK_METRIC` TLV. If there is no such TLV then `N2_out_metric` is set to `DEFAULT_METRIC`.
- o When adding or updating a Link Tuple, if the HELLO message includes an address of the receiving OLSRV2 interface with a `LINK_STATUS` TLV with value `SYMMETRIC`, then the presence or absence of an associated MPR TLV indicating flooding TLV status will set or clear the appropriate `L_mpr_selector`.
- o When adding or updating a Neighbor Tuple, if the HELLO message includes an address of the receiving OLSRV2 interface with a `LINK_STATUS` or `OTHER_NEIGHB` TLV with value `SYMMETRIC`, then the presence or absence of an associated MPR TLV indicating routing TLV status will set or clear the appropriate `N_mpr_selector`.

6.10. MPR Calculation and Neighbor Set Update

For routing MPRs, a possible algorithm is given in Appendix B. This sets or clears `N_routing_mpr` in all Neighbor Tuples with `N_symmetric = true`.

For flooding MPRs, the existing per OLSRV2 interface algorithm can be used unchanged. In particular its first stage (adding necessary MPRs) and third stage (removing unnecessary MPRs) are appropriate unchanged. Its second stage, which prioritizes possible added MPRs, can have link metrics (`L_out_metric + N2_out_metric`) added as a consideration in that prioritization. One suggestion is that after picking candidate new MPRs that maximize the new coverage of two hop neighbors, ties can be broken (before tie breaking based on maximizing the total coverage of two hop neighbors, new and old) by minimizing the sum of `L_out_metric + N2_out_metric` for each candidate MPR, across all newly covered two hop neighbors. Whatever algorithm is used, it sets or clears `N_flooding_mpr` instead of the current `N_mpr`.

In addition to the modified algorithms, a modification of the circumstances in which they are needed (i.e., when the neighborhood

has changed sufficiently) is also required, and is different in each case. For flooding MPRs this adds changes to `L_out_metric` and/or `N2_out_metric` values. As use of these is optional, so is the recalculation. Furthermore recalculation may be restricted to when the metrics increase for MPRs or decrease for non-MPRs. For routing MPRs this adds changes to `N_in_metric` and/or `N2_in_metric` values, and is compulsory to maintain shortest routes.

6.11. TC Message Generation

The following additional contents of a TC message are required. Link metric association is as previously described.

- o Each included `N_orig_addr` or address from an `N_neighbor_addr_list` MUST have an associated outgoing router metric of the appropriate type with value `N_out_metric`.
- o Each included `AL_net_addr` MUST have an associated outgoing router metric of the appropriate type with value `AL_metric`.

6.12. TC Message Processing

Processing a TC message has the following extra steps:

- o When adding or updating a Router Topology Tuple, set `TR_metric` to the value of any associated outgoing router `LINK_METRIC` TLV, or to `DEFAULT_METRIC` if none.
- o When adding or updating a Routable Address Topology Tuple, set `TA_metric` to the value of any associated outgoing router `LINK_METRIC` TLV, or to `DEFAULT_METRIC` if none.
- o When adding or updating an Attached Network Tuple, set `AN_metric` to the value of any associated outgoing router `LINK_METRIC` TLV, or to `DEFAULT_METRIC` if none.

6.13. Routing Set Calculation

Routing Set calculation using the Network Topology Graph is unchanged, except that when selecting a Link Tuple to add an edge `X -> S`, that Link Tuple MUST also have `L_out_metric = N_out_metric`, and that edges in the Network Topology Graph have metrics rather than hop counts:

- o For an edge `X -> Y` use `N_out_metric`.
- o For an edge `W -> U` use `TR_metric`.

- o For an edge X -> S use N_out_metric.
- o For an edge W -> V use TA_metric.
- o For an edge W -> T use AN_metric.
- o For an edge Y -> Z use N2_out_metric.

An example algorithm, modified from that in [OLSRv2], is given in Appendix C.

7. IANA Considerations

This document presents no IANA considerations. Addition of metrics to OLSRv2 will add to the IANA Considerations section of [OLSRv2].

8. Security Considerations

This document does not specify any security considerations.

9. References

9.1. Normative References

- [OLSRv2] Clausen, T., Dearlove, C., and P. Jacquet, "The Optimized Link State Routing Protocol version 2", draft-ietf-manet-olsrv2-11.txt (work in progress), April 2010.
- [RFC5444] Clausen, T., Dean, J., Dearlove, C., and C. Adjih, "Generalized MANET Packet/Message Format", RFC 5444, February 2009.

9.2. Informative References

- [NHDP] Clausen, T., Dean, J., and C. Dearlove, "MANET Neighborhood Discovery Protocol (NHDP)", work in progress draft-ietf-manet-nhdp-12.txt, March 2010.
- [RFC2501] Macker, J. and S. Corson, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations", RFC 2501, January 1999.
- [RFC5497] Clausen, T. and C. Dearlove, "Representing multi-value time in MANETs", RFC 5497, March 2009.

Appendix A. MPR Routing Property

In order that routers can find and use shortest routes in a network while using the minimum reduced topology supported by OLSRv2 (that a router only advertises its MPR selectors in TC messages), routing MPR selection must result in the property that there are shortest routes with all intermediate routers being routing MPRs.

This appendix uses the following terminology and assumptions:

- o The network is a graph of nodes connected by arcs, where nodes correspond to routers with willingness not equal to WILL_NEVER (except possibly at the ends of routes). An arc corresponds to the set of symmetric links connecting those routers; the OLSRv2 interfaces used by those links are not relevant.
- o Each arc has a metric in each direction, being the minimum of the corresponding link metrics in that direction, i.e., the corresponding router metric. This metric must be positive.
- o A sequence of arcs joining two nodes is referred to as a path.
- o Node A is an MPR of node B, if corresponding router A is a routing MPR of router B.

The required property (of using shortest routes with reduced topology) is equivalent to that for any pair of distinct nodes X and Z there is a shortest path from X to Z, $X - Y_1 - Y_2 - \dots - Y_m - Z$ such that Y_1 is an MPR of Y_2 , \dots Y_m is an MPR of Z. Call such a path a routable path, and call this property the routable path property.

The required definition for a node X selecting MPRs is that for each distinct node Z from which there is a two arc path, there is a shorter, or equally short, path which is either $Z - Y - X$ where Y is an MPR of X, or is the one arc path $Z - X$. Note that the existence of locally known, shorter, but more than two arc paths, which can be used to reduce the numbers of MPRs, is not considered here. (Such reductions are only when the remaining MPRs can be seen to retain all necessary shortest paths, and therefore retains the required property.)

Although this appendix is concerned with paths with minimum total metric, not number of arcs (hop count), it proceeds by induction on the number of arcs in a path. Although it considers minimum metric routes with a bounded number of arcs, it then allows that number of arcs to increase so that overall minimum metric paths, regardless of the number of arcs, are considered.

Specifically, the routable path property is a corollary of the property that for all positive integers n , and all distinct nodes X and Z , if there is any path from X to Z of n arcs or fewer, then there is a shortest path, from among those of n arcs or fewer, that is a routable path. This may be called the n -arc routable path property.

The n -arc routable path property is trivial for $n = 1$, and directly follows from the definition of the MPRs of Z for $n = 2$.

Proceeding by induction, assuming the n -arc routable path property is true for $n = k$, consider the case that $n = k+1$.

Suppose that $X - V_1 - V_2 - \dots - V_k - Z$ is a shortest $k+1$ arc path from X to Z . We construct a path which has no more than $k+1$ arcs, has the same or shorter length (hence has the same, shortest, length considering only paths of up to $k+1$ arcs, by assumption) and is a routable path.

First consider whether V_k is an MPR of Z . If it is not then consider the two arc path $V_{k-1} - V_k - Z$. This can be replaced either by a one arc path $V_{k-1} - Z$ or by a two arc path $V_{k-1} - W_k - Z$ where W_k is an MPR of Z , such that the metric from V_{k-1} to Z by the replacement path is no longer. In the former case (replacement one arc path) this now produces a path of length k , and the previous inductive step may be applied. In the latter case we have replaced V_k by W_k , where W_k is an MPR of Z . Thus we need only consider the case that V_k is an MPR of Z .

We now apply the previous inductive step to the path $X - V_1 - \dots - V_{k-1} - V_k$, replacing it by an equal length path $X - W_1 - \dots - W_{m-1} - V_k$, where $m \leq k$, where this path is a routable path. Then because V_k is an MPR of Z , the path $X - W_1 - \dots - W_{m-1} - V_k - Z$ is a routable path, and demonstrates the n -arc routable path property for $n = k+1$.

This thus shows that for any distinct nodes X and Z , there is a routable path using the MPR-reduced topology from X to Z , i.e., that this modification of OLSRv2 still finds minimum length paths (routes).

Appendix B. Routing MPR Calculation

This a possible algorithm for calculating routing MPRs. At the start of the calculation set `N_routing_mpr = false` in all Neighbor Tuples.

This calculation is not per OLSRv2 interface, but is for all OLSRv2 interfaces together. Thus the union of all of the router's 2-Hop Sets are considered in what follows. (The router's Link Sets are not required, only its Neighbor Set.)

Define a Local Topology Tuple (used only during routing MPR calculation) that represents a route from a final router to this router, to include:

`LT_next_orig_addr` is the `N_orig_addr` of the Neighbor Tuple corresponding to the nearest router to this router on the route;

`LT_last_orig_addr` is the `N_orig_addr` of the Neighbor Tuple corresponding to the furthest router on the route from this router, other than the final router;

`LT_final_iface_addr` is an address of the final router;

`LT_last_metric` is the metric of the part of the route from the last router to this router;

`LT_final_metric` is the metric of the link from the final router to the last router;

`LT_number_hops` is the number of hops on the route from the final router to this router.

All such final routers can reach this router in two hops, the first hop being to the last router other than the final router, but the preferred route may use more hops with a lower metric. When the route uses two hops, the last and next routers are the same (the router between this router and the final router). As for 2-Hop Tuples, a separate Local Topology Tuple is used for each address of each final router.

It is assumed here that between routes with equal metric, a route with fewest hops is preferred.

Then, for each 2-Hop Tuple for which `N_willingness` of the corresponding Neighbor Tuple is not equal to `WILL_NEVER`, create a Local Topology Tuple with:

- o `LT_next_orig_addr` := `N_orig_addr` of the corresponding Neighbor Tuple
- o `LT_last_orig_addr` := `N_orig_addr` of the corresponding Neighbor Tuple
- o `LT_final_iface_addr` := `N2_2hop_iface_addr`
- o `LT_last_metric` := `N_in_metric` of the corresponding Neighbor Tuple
- o `LT_final_metric` := `N2_in_metric`
- o `LT_number_hops` := 2

Now, while there are any two Local Topology Tuples (Tuple A and Tuple B) such that:

- o A's `LT_final_iface_addr` is in the `N_neighbor_addr_list` corresponding to B's `LT_last_orig_addr`, and
- o A's `LT_last_metric` + `LT_final_metric` < B's `LT_last_metric`

update Tuple B by:

- o `LT_next_orig_addr` := A's `LT_next_orig_addr`
- o `LT_last_metric` := A's `LT_last_metric` + `LT_final_metric`
- o `LT_number_hops` := A's `LT_number_hops` + 1

This replaces Tuple B's route from this router to its last router by Tuple A's route from this router to its final router.

Once that process is finished, remove all Local Topology Tuples such that either:

- o there is a Neighbor Tuple with `LT_final_iface_addr` in `N_neighbor_addr_list`; AND
- o `N_in_metric` <= `LT_final_metric`

or:

- o there is another Local Topology Tuple with the same `LT_final_iface_addr`; AND
 - * a smaller value of `LT_last_metric` + `LT_final_metric`; OR

- * an equal value of `LT_last_metric + LT_final_metric` and a smaller value of `LT_number_hops`.

This removes Local Topology Tuples where either there is a Neighbor Tuple offering a better one hop route, or another Local Topology Tuple offering a better route, from the final router.

For each remaining Local Topology Tuple define that the Neighbor Tuple with `N_orig_addr = LT_next_orig_addr` covers the 2-hop neighbor address `LT_final_iface_addr`.

A valid set of routing MPRs is any subset of these Neighbor Tuples which collectively cover all of these `LT_final_iface_addr`. Set the corresponding `N_routing_mpr = true`.

While any subset with this property is valid, a heuristic for a "good" subset is required. The current heuristic in [OLSRv2] has three main steps: add necessary neighbors, add additional neighbors in a prioritized order until coverage is complete, remove unneeded neighbors (possibly in order of ascending willingness). There is no reason to modify this. The middle step currently uses the following priority order: greatest willingness, maximum new coverage, maximum coverage, if an MPR selector, any. This will still work using metrics (the MPR selector step should be routing MPR selector). It may be considered that metrics could be used. However in principle this is not necessary, as metrics have already been taken into account in this construction. (This differs from flooding MPRs, where considering metrics in this step is appropriate as they are not used up to this point.)

Appendix C. Example Algorithm for Calculating the Routing Set

Note: this text (other than this paragraph) directly replaces the similarly named appendix in [OLSRv2].

The following procedure is given as an example for calculating the Routing Set using a variation of Dijkstra's algorithm. First all Routing Tuples are removed, and then, using the selections and definitions in Appendix C.1, the procedures in the following sections (each considered a "stage" of the processing) are applied in turn.

C.1. Local Interfaces and Neighbors

The following selections and definitions are made:

1. For each Local Interface Tuple, select a network address from its `I_local_iface_addr_list`, this is defined as the selected address for this Local Interface Tuple.
2. For each Link Tuple, the selected address of its corresponding Local Interface Tuple is defined as the selected local address for this Local Interface Tuple.
3. For each Neighbor Tuple with `N_symmetric = true`, select a Link Tuple with `L_status = SYMMETRIC` for which this is the corresponding Neighbor Tuple and has `L_out_metric = N_out_metric`. This is defined as the selected Link Tuple for this Neighbor Tuple.
4. For each network address (`N_orig_addr` or in `N_neighbor_addr_list`, the "neighbor address") from a Neighbor Tuple with `N_symmetric = true`, select a Link Tuple (the "selected Link Tuple") from those for which this is the corresponding Neighbor Tuple, have `L_status = SYMMETRIC`, and have `L_out_metric = N_out_metric`, by:
 1. If there is such a Link Tuple whose `L_neighbor_iface_addr_list` contains the neighbor address, select that Link Tuple.
 2. Otherwise select the selected Link Tuple for this Neighbor Tuple.

Then for this neighbor address:

3. The selected local address is defined as the selected local address for the selected Link Tuple.

4. The selected link address is defined as an address from the `L_neighbor_iface_addr_list` of the selected Link Tuple, if possible equal to this neighbor address.
5. Routing Tuple preference is decided by preference for minimum `R_dist`, and then for preference for corresponding Neighbor Tuples in this order:
 - * For greater `N_willingness`.
 - * For `N_mpr_selector = true` over `N_mpr_selector = false`.

Note that preferred Routing Tuples SHOULD be used. Routing Tuples with minimum `R_metric` MUST be used, this is specified outside the definition of preference. An implementation MAY modify this definition of preference without otherwise affecting this algorithm.

C.2. Add Neighbor Routers

The following procedure is executed once.

1. For each Neighbor Tuple with `N_symmetric = true`, add a Routing Tuple with:
 - * `R_dest_addr := N_orig_addr;`
 - * `R_next_iface_addr := selected link address for N_orig_addr;`
 - * `R_local_iface_addr := selected local address for N_orig_addr;`
 - * `R_metric := N_out_metric;`
 - * `R_dist := 1.`

C.3. Add Remote Routers

The following procedure is executed for each value of `h`, starting with `h := 1` and incrementing by 1 for each iteration. The execution MUST stop if no Routing Tuples are added or modified in an iteration.

1. For each Router Topology Tuple, if:
 - * `TR_from_orig_addr` is equal to the `R_dest_addr` of a Routing Tuple with `R_dist = h` (the "previous Routing Tuple"),then consider the candidate Routing Tuple with:

- * R_dest_addr := TR_to_orig_addr;
- * R_next_iface_addr := R_next_iface_addr of the previous Routing Tuple;
- * R_local_iface_addr := R_local_iface_addr of the previous Routing Tuple;
- * R_metric := R_metric of the previous Routing Tuple + TR_metric;
- * R_dist := h+1.

This candidate Routing Tuple MUST be added to the Routing Set if there is no existing Routing Tuple with the same R_dest_addr. Otherwise this candidate Routing Tuple MUST replace the existing Routing Tuple with the same R_dest_addr if this candidate Routing Tuple has a smaller R_metric, this candidate Routing Tuple SHOULD replace the existing Routing Tuple with the same R_dest_addr if this candidate Routing Tuple has an equal R_metric and is preferred to the existing Routing Tuple.

C.4. Add Neighbor Addresses

The following procedure is executed once.

1. For each Neighbor Tuple with N_symmetric = true:
 1. For each network address (the "neighbor address") in N_neighbor_addr_list, if the neighbor address is not equal to the R_dest_addr of any Routing Tuple, then add a new Routing Tuple, with:
 - + R_dest_addr := neighbor address;
 - + R_next_iface_addr := selected link address for the neighbor address;
 - + R_local_iface_addr := selected local address for the neighbor address;
 - + R_metric := N_out_metric;
 - + R_dist := 1.

C.5. Add Remote Routable Addresses

The following procedure is executed once.

1. For each Routable Address Topology Tuple, if:

- * TA_dest_addr is not equal to the R_dest_addr of any Routing Tuple added in an earlier stage, AND;
- * TA_from_orig_addr is equal to the R_dest_addr of a Routing Tuple (the "previous Routing Tuple"),

then add a new Routing Tuple, with:

- * R_dest_addr := TA_dest_addr;
- * R_next_iface_addr := R_next_iface_addr of the previous Routing Tuple;
- * R_local_iface_addr := R_local_iface_addr of the previous Routing Tuple;
- * R_metric := R_metric of the previous Routing Tuple + TA_metric.
- * R_dist := R_dist of the previous Routing Tuple + 1.

There may be more than one Routing Tuple that may be added for an R_dest_addr in this stage. If so, then, for each such R_dest_addr, a Routing Tuple with minimum R_metric MUST be selected, otherwise a Routing Tuple which is preferred SHOULD be added.

C.6. Add Attached Networks

The following procedure is executed once.

1. For each Attached Network Tuple, if:

- * AN_net_addr is not equal to the R_dest_addr of any Routing Tuple added in an earlier stage, AND;
- * AN_orig_addr is equal to the R_dest_addr of a Routing Tuple (the "previous Routing Tuple"),

then add a new Routing Tuple, with:

- * R_dest_addr := AN_net_addr;
- * R_next_iface_addr := R_next_iface_addr of the previous Routing Tuple;
- * R_local_iface_addr := R_local_iface_addr of the previous Routing Tuple;
- * R_metric := R_metric of the previous Routing Tuple + AN_metric;
- * R_dist := R_dist of the previous Routing Tuple + AN_dist.

There may be more than one Routing Tuple that may be added for an R_dest_addr in this stage. If so, then, for each such R_dest_addr, a Routing Tuple with minimum R_metric MUST be selected, otherwise a Routing Tuple which is preferred SHOULD be added.

C.7. Add 2-Hop Neighbors

The following procedure is executed once.

1. For each 2-Hop Tuple, if:

- * N2_2hop_addr is a routable address, AND;
- * N2_2hop_addr is not equal to the R_dest_addr of any Routing Tuple added in an earlier stage, AND;
- * the Routing Tuple with R_dest_addr = N_orig_addr of the corresponding Neighbor Tuple (the "previous Routing Tuple") has R_dist = 1,

then add a new Routing Tuple, with:

- * R_dest_addr := N2_2hop_addr;
- * R_next_iface_addr := R_next_iface_addr of the previous Routing Tuple;
- * R_local_iface_addr := R_local_iface_addr of the previous Routing Tuple;
- * R_metric := R_metric of the previous Routing Tuple + N_out_metric of the corresponding Neighbor Tuple;

* R_dist := 2.

There may be more than one Routing Tuple that may be added for an R_dest_addr in this stage. If so, then, for each such R_dest_addr, a Routing Tuple with minimum R_metric MUST be selected, otherwise a Routing Tuple which is preferred SHOULD be added.

Appendix D. Constraints

The constraints specified in [OLSRv2] must be updated to match modifications to the Information Bases. These constraint modifications are as described in this appendix.

In each Local Attached Network Tuple:

- o AL_metric MUST be representable as the value of a LINK_METRIC TLV (hence MUST NOT be less than MINIMUM_METRIC and MUST NOT be greater than MAXIMUM_METRIC).

In each Link Tuple:

- o If L_status is HEARD or SYMMETRIC then L_in_metric MUST be representable as the value of a LINK_METRIC TLV (hence MUST NOT be less than MINIMUM_METRIC and MUST NOT be greater than MAXIMUM_METRIC).
- o If L_status is PENDING or LOST then L_in_metric MUST be considered to be unspecified.
- o If L_status is SYMMETRIC then L_out_metric MUST be representable as the value of a LINK_METRIC TLV (hence MUST NOT be less than MINIMUM_METRIC and MUST NOT be greater than MAXIMUM_METRIC).
- o If L_status is not SYMMETRIC then L_out_metric MUST be considered to be unspecified.
- o If L_mpr_selector = true then the L_status MUST equal SYMMETRIC.

In each Neighbor Tuple constraints involving N_mpr apply to both N_flooding_mpr and N_routing_mpr, and:

- o If N_symmetric = true then N_in_metric MUST equal the minimum value of the L_in_metric values of all Link Tuples for which this is the corresponding Neighbor Tuple and have L_status = SYMMETRIC. N_in_metric MUST be unspecified if these are no such Link Tuples (i.e., if N_symmetric = false).
- o If N_symmetric = true then N_out_metric MUST equal the maximum value of the L_out_metric values of all Link Tuples for which this is the corresponding Neighbor Tuple and have L_status = SYMMETRIC. N_out_metric MUST be unspecified if these are no such Link Tuples (i.e., if N_symmetric = false).

In each 2-Hop Tuple:

- o N2_in_metric MUST be representable as the value of a LINK_METRIC TLV (hence MUST NOT be less than MINIMUM_METRIC and MUST NOT be greater than MAXIMUM_METRIC).
- o N2_out_metric MUST be representable as the value of a LINK_METRIC TLV (hence MUST NOT be less than MINIMUM_METRIC and MUST NOT be greater than MAXIMUM_METRIC).

In each Router Topology Tuple:

- o TR_metric MUST be representable as the value of a LINK_METRIC TLV (hence MUST NOT be less than MINIMUM_METRIC and MUST NOT be greater than MAXIMUM_METRIC).

In each Routable Address Topology Tuple:

- o TA_metric MUST be representable as the value of a LINK_METRIC TLV (hence MUST NOT be less than MINIMUM_METRIC and MUST NOT be greater than MAXIMUM_METRIC).

In each Attached Network Tuple:

- o AN_metric MUST be representable as the value of a LINK_METRIC TLV (hence MUST NOT be less than MINIMUM_METRIC and MUST NOT be greater than MAXIMUM_METRIC).

Appendix E. Acknowledgements

The authors would like to thank Alan Cullen (BAE Systems) for review and comments, and Brian Adamson (NRL), Justin Dean (NRL), Henning Rogge (FGAN), Charles Perkins (WiChorus) and Stan Ratliff (Cisco) for discussions.

Authors' Addresses

Christopher Dearlove
BAE Systems Advanced Technology Centre

Phone: +44 1245 242194
EMail: chris.dearlove@baesystems.com
URI: <http://www.baesystems.com/>

Thomas Heide Clausen
LIX, Ecole Polytechnique, France

Phone: +33 6 6058 9349
EMail: T.Clausen@computer.org
URI: <http://www.ThomasClausen.org/>

Philippe Jacquet
INRIA, France

Phone: +33 1 3963 5263
EMail: Philippe.Jacquet@inria.fr
URI: <http://hipercom.inria.fr/>

