Internet Engineering Task Force
INTERNET-DRAFT
draft-ietf-ospf-omp-00

Curtis Villamizar
ANS

March 13, 1998

# OSPF Optimized Multipath (OSPF-OMP)

## Status of this Memo



## Abstract

OSPF may form multiple equal cost paths between points. This is true of any link state protocol. In the absense of any explicit support to take advantage of this, a path may be chosen arbitrarily. Techniques have been utilized to divide traffic somewhat evenly among the available paths. These techniques have been referred to as Equal Cost Multipath (ECMP). An unequal division of traffic among the available paths is generally preferable. Routers generally have no knowledge of traffic loading on distant links and therefore have no basis to optimize the allocation of traffic.

Optimized Mulitpath is a compatible extension to OSPF, utilizing the Opaque LSA to distribute loading information, proposing a means to adjust forwarding, and providing an algorithm to make the adjustments gradually enough to insure stability yet provide reasonably fast adjustment when needed.

## 1 Overview

Networks running OSPF are often heavily loaded. Topologies often evolve to include multiple paths. Multiple paths may be initially designed to provide redundancy but also result from incremental addition of circuits to accomodate traffic growth. The redundant paths provide a potential to distribute traffic loading and reduce congestion. Optimized Mulitpath (OMP) provides a means for OSPF to make better use of this potential to distribute loading.

Early attempts to provide load sensitive routing involved changing link costs according to loading. These attempts were doomed to failure because the adjustment increment was grossly course and oscillation was inevitable [?].

A widely utilized technique to improve loading is known as Equal Cost Multipath (ECMP). If the topology is such that equal cost paths exist, then an attempt is made to divide traffic equally among the paths. Three methods of splitting traffic have been used.

1. Per packet round robin forwarding.
2. Dividing destination prefixes among available next hops in the forwarding entries.
3. Dividing traffic according to a hash function applied to the source and desination pair.

The "per packet round robin forwarding" technique is only applicable if the delays on the paths are almost equal. The delay difference must be small relative to packet serialization time. Delay differences greater than three times the packet serialization time can cause terrible TCP performance. for example, packet 2, 4, and 6 may arrive before packet 1, triggering TCP fast retransmit. The result will be limiting TCP to a very small window and very poor performance over long delay paths.

The delay differences must be quite small. A 532 byte packet is serialized onto a DS1 link in under 2.8 msec. At DS3 speed, serialization is accomplished in under 100 usec. At OC12 it is under 7 usec. For this reason "per packet round robin forwarding" is not applicable to a high speed WAN.

Dividing destination prefixes among available next hops provides a very course and unpredictable load split. Long prefixes are problematic. In reaching an end node, the majority of traffic is often destined to a single prefix. This technique is applicable to a high speed WAN but with the drawbacks just mentioned better techniques are needed.

The "source/destination hash" based technique was used as far back as the T3-NSFNET in the NSS routers. A hash function, such as CRC-16, is applied over the source address and destination address. The hash space is then split evenly among the available paths by either setting threshholds or performing a modulo operation. Traffic between any given source and destination remain on the same path. Because the technique is based on host addresses, and uses both the source and destination address, it does not suffer the course granularity problem of the prefix based technique, even when forwarding to a single prefix. Source/destination hash is the best technique available for a high speed WAN.

The forwarding decision for the "source/destination hash" based technique is quite simple. When a packet arrives, look up the forwarding entry in the radix tree. The next hop entry can be an array index into a set of structures, each containing one or more actual next hops. If more than one next hop is present, compute a CRC16 value based on the source and destination addresses. The CRC16 can be implemented in hardware and computed in parallel to the radix tree lookup in high speed implementations, and discarded if not needed. Each next hop entry in the structure must contain a boundary value and the next hop itself. An integer "less than" comparison is made against the boundary value determining whether to use this next hop or move to the next a comparison. In hardware the full set of comparisons can be made simultaneously for up to some number of next hops. This yields the next hop to use.

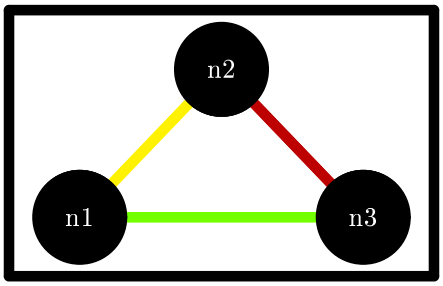


Figure 1: A very simple application of ECMP

| Nodes | Traffic | Node Names |
|---|---|---|
| n3-n1 | 60.000 | Node 3 -> Node 1 |
| n1-n3 | 60.000 | Node 1 -> Node 3 |
| n3-n2 | 20.000 | Node 3 -> Node 2 |
| n2-n3 | 20.000 | Node 2 -> Node 3 |
| n2-n1 | 10.000 | Node 2 -> Node 1 |
| n1-n2 | 10.000 | Node 1 -> Node 2 |

Table 1: Traffic loading for the example in Figure 1

For ECMP, the boundary values are set by first dividing one more than the maximum value that the hash computation can return (65536 for CRC16) by the number of available next hops and then setting the Nth boundary to N times that number (with the Nth value fixed at one more than the maximum value regardless of underflow caused by trucating during division, 65536 for CRC16).

An equal load split is not always optimal. Consider the example in Figure 1 with the offered traffic in Table 1. If all of the link costs are set equally, then the link N1—N3 is significantly overloaded (135.75%) while the path N1—N2—N3 is lightly loaded (45.25% and 22.62%). If the cost on the N1—N3 link is equal to the cost of the N1—N2—N3 path, then N1 will try to split the load destined toward N3 across the two paths.

Given the offered traffic in Table 1 the loading on N1—N3 is reduced to 67.87% but the link loading on the path N2—N3 becomes 113.12%. Ideally node N1 should put 1/3 of the traffic toward N3 on the path N1—N2—N3 and 2/3 on the path N1—N3. To know to do this N1 must know the loading on N2–N3.

This is where Optimized Multipath (OMP) provides additional benefit over ECMP. Ignoring for the moment how node N1 knows to put 1/3 of the traffic toward N3 on the path N1—N2—N3, the way this is accomplished from a forwarding standpoint is to move the boundary in the forwarding structure from the default value of 1/2 of 65536 to about 1/3 of 65536. If there are a very large set of source and destination host addresses pairs, then the traffic will be split among the 65536 possible hash values. This provides the means for a very fine granularity of adjustment.

Having explained how a fine granularity of forwarding adjustment can be accomplished, what remains is to define how nodes in a large topology can know what the loading levels are elsewhere in the topology and defining an algorithm which can allow autonomous unsyncronized decisions on the parts of many routers in a topology to quickly converge on a near optimal loading without the risk of oscillation.

# 2 Flooding Loading Information

Loading information is flooded within an OSPF area using Opaque LSAs [1]. Area local scope (link-state type 10) link state attributes are flooded containing an "Opaque Type" of LSA_OMP_LINK_LOAD or LSA_OMP_PATH_LOAD. The type LSA_OMP_LINK_LOAD Opaque LSA is used to flood link loading information within an area. The type LSA_OMP_PATH_LOAD Opaque LSA is used to flood loading information for use with inter-area routes. Loading information obtained from an exterior routing protocol may also be considered if available. The means of passing loading information in an exterior routing protocol is beyond the scope of this document.

## 2.1 Link Loading Information

Within an area link loading is flooded using the type LSA_OMP_LINK_LOAD Opaque LSA. The "Opaque ID" must contain a 24 bit integer that is unique to the advertising router and link or virtual link. The method of assignment of these 24 bit integers is a local matter. A router must be capable of being configured to put a fixed value in N of the bits and use the remainin 24-N bits to uniquely identify an interface.

The "Opaque Information" in the type LSA_OMP_LINK_LOAD Opaque LSA contains the following.

1. a measure of link loading in each direction as a fraction of link capacity,
2. a measure of packets dropped due to queue overflow in each direction (if known) expressed as a fraction,
3. the link capacity in killobits per second (or unity if less than 1000 bits per second).

Generally the number of ouput packets dropped will be known. In designs where drops occur on the input (generally a bad design practice), the rate of input queue drops should be recorded. These measures of loading and drop are computed using the interface counters generally maintained for SNMP purposes, plus a running count of output queue drops if available. The counters are sampled every OMP_SAMPLE_INTERVAL seconds.

The previous value of each of the counters is substracted from the current value. The counters to be sampled are the following.

1. bytes out,
2. bytes in,
3. packets out,
4. packets in,

5. output queue drops,
6. input queue drops.

A value of instantaneous load in each direction is based on byte count and link capacity. An instantaneous output queue drop rate is based on queue drops and packet count. Each of these is combined with a running filtered value according to the following method. The running total is shifted down by OMP_SHIFT_BITS bits and subtracted from the running total. The instantaneous value is shifted down by OMP_SHIFT_BITS bits and added to the running total.

The last time that a type LSA_OMP_LINK_LOAD Opaque LSA with the same Opaque ID was sent is recorded and the values sent are recorded. For the purpose of determining when to reflood, an equivalent loading figure is used. The computation of equivalent loading is described in Section 2.3.

The higher of the current equivalent loading computation and the previous is used when determining whether to send the type LSA_OMP_LINK_LOAD Opaque LSA. The type LSA_OMP_LINK_LOAD Opaque LSA is sent if any of the following is true.

1. The equivalent load is over 100% and the change in equivalent load since last resent is over 5% and 30 seconds has elapsed since last sent.
2. The equivalent load is over 100% and the change in equivalent load since last resent is over 2% and 90 seconds has elapsed since last sent.
3. The equivalent load is over 100% and 3 minutes has elapsed since last sent.
4. The equivalent load is over 90% and and the change in equivalent load since last resent is over 5% and 1 minute has elapsed since last sent.
5. The equivalent load is over 90% and the change in equivalent load since last resent is over 2% and 4 minutes has elapsed since last sent.
6. The equivalent load is over 90% and 10 minutes has elapsed since last sent.
7. The equivalent load is over 70% and the change in equivalent load since last resent is over 10% and 1 minute has elapsed since last sent.
8. The equivalent load is over 70% and the change in equivalent load since last resent is over 5% and 2 minutes has elapsed since last sent.
9. The equivalent load is over 70% and the change in equivalent load since last resent is over 2% and 8 minutes has elapsed since last sent.
10. The equivalent load is over 70% and 15 minutes has elapsed since last sent.
11. The equivalent load is over 50% and the change in equivalent load since last resent is over 10% and 1 minute has elapsed since last sent.
12. The equivalent load is over 50% and the change in equivalent load since last resent is over 5% and 5 minutes has elapsed since last sent.
13. The equivalent load is over 25% and the change in equivalent load since last resent is over 25% and 2 minutes has elapsed since last sent.
14. The equivalent load is over 25% and 20 minutes has elapsed since last sent.

15. The type LSA_OMP_LINK_LOAD Opaque LSA has never been sent.

The point of this graduated scale is to reduce the amount of flooding that is occurring unless links are in trouble or undergoing a significant traffic shift. Change may occur in a quiescent network due to failure external to the network that causes traffic to take alternate paths. In this case, the more frequent flooding will trigger a faster convergence. Traffic shift may also occur due to shedding of loading by the OMP algortihm itself as the algorithm converges in response to an external change.

## 2.2 Path Loading Information

Path loading information regarding an adjacent area is flooded by an Area Border Router (ABR) using the type LSA_OMP_PATH_LOAD Opaque LSA. The "Opaque ID" must contain a 24 bit integer that is unique to the router and an advertised summary LSA. The method of assignment of these 24 bit integers is a local matter. A router must be capable of being configured to put a fixed value in N of the bits and use the remainin 24-N bits to uniquely identify the summary LSA.

The "Opaque Information" in the type LSA_OMP_PATH_LOAD Opaque LSA contains the following.

1. the highest loading in the direction toward the destination as a fraction of link capacity,
2. a measure of total packet drop due to queue overflow in the direction toward the destination expressed as a fraction,
3. the smallest link capacity on the path to the destination.

These values are taken from the link on the path from the ABR to the destination of the summary LSA. The link with the highest loading may not be the link with the lowest capacity. The queue drop value is one minus the product of fraction of packets that are not dropped at each measurement point on the path (input and output in the direction of the path). The following computation is used.

$$Loss_{path} \;=\; 1 - \prod_{links} (1 - Loss_{in})(1 - Loss_{out})$$

The path loading and path loss rate are filtered according to the same algorithm defined in the prior section. Rather than polling a set of counters the current value of the path loading and path loss rate is used. An equivalent load is calculated for each path to a summary LSA destination as described in Section 2.3. A type LSA_OMP_PATH_LOAD Opaque LSA is flooded according to the same rate schedule as described in the prior section.

An ABR may be configured to not send type LSA_OMP_PATH_LOAD Opaque LSA into any given area.

### 2.3 Computing equivalent loading

The equivalent load is the actual percent loading multiplied by a factor that provides an estimate of the extent to which TCP would be slowing down to avoid congestion. This estimate is based on the link bandwidth and loss rate, knowledge of TCP dynamics, and some assumption about the characteristics of the TCP flows being passed through the link. Some of the assumptions must be configured.

If loss is low, the equivalent load will be equal to the actual percent loading. If loss is high and loading is at or near 100%, then the equivalent load calculation provides a means of deciding which links are more heavily overloaded. The equivalent load figure is not intended to be an accurate pridiction of offerred load, simply a metric for use in deciding which link to offload.

Mathis and Mahdavi provide the following estimate of loss given TCP window size and round trip time [2].

$$p \; < \; (\frac{MSS}{BW\ RTT})^2$$

The basis for the estimate is that TCP slows down roughly in proportion to the inverse of the square root of loss. There is no way to know how fast TCP would be going if no loss were present if there are other bottlenecks. A somewhat arbitrary assumption is made that TCP would go no faster than if loss were at 0.5%. If loss is greater than 0.5% then TCP performance would be reduced. The equivalent loading is estimated using the following computation.

$$Load_{equiv} \; = \; Load_{actual} \; K \; \sqrt{1 - ((1 - Loss_{in})(1 - Loss_{out}))}$$

The inverse of the square root of 0.1% is 10 so 10 may be used for the value of "K". A square root is somewhat time consuming to compute, so a table lookup can be done to avoid this computation. Increments of 0.5% would yield a 200 entry table. The computation could then be done with a table lookup, a shift, and an integer multiply. At most this needs to be done on links with loss once every OMP_SAMPLE_INTERVAL seconds.

The conversion of loss to estimated loading is not at all accurate. The non-linearity does affect the time to converge though convergence still occurs as long as loss is positively correlated to loading. This is discussed further in Section D.1.

## 3 Adjusting Equal Cost Path Loadings

Adjustments are made to a next hop structure to reflect differences in loading on the paths as reported by the type LSA_OMP_LINK_LOAD Opaque LSA and type LSA_OMP_PATH_LOAD Opaque LSA. Section 3.2 describes the selection of a "critically loaded segment" which is used to determine when to make adjustments and the size of the adjustments. An adjustment to loading of a given set of equal cost paths is made when one of the following conditions are true.

1. The LSA for the "critically loaded segment" has been reflooded.
2. The difference between the equivalent load of the "critically loaded segment" and the lightest loaded path is greater than 5% and the equivalent load of the "critically loaded segment" is greater than 100% and 90 seconds has elapsed since the last adjustment.
3. The difference between the equivalent load of the "critically loaded segment" and the lightest loaded path is greater than 3% and the equivalent load of the "critically loaded segment" is greater than 100% and 9 180 seconds has elapsed since the last adjustment.
4. The difference between the equivalent load of the "critically loaded segment" and the lightest loaded path is greater than 5% and the equivalent load of the "critically loaded segment" is greater than 90% and 120 seconds has elapsed since the last adjustment.
5. The difference between the equivalent load of the "critically loaded segment" and the lightest loaded path is greater than 3% and the equivalent load of the "critically loaded segment" is greater than 90% and 240 seconds has elapsed since the last adjustment.
6. 300 seconds has elapsed since the last adjustment.

The reflooding algorithm is designed to be slightly less aggressive than the adjustment algorithm. This reduces the need to continuously flood small changes except in conditions of overload or substantial change in loading. Some overshoot may occur due to adjustments made in the absence of accurate knowledge of loading.

## 3.1 Next hop structures

For intra-AS routes, a separate next hop structure must exist for each destination router or network. For inter-AS routes, a separate structure must exist for each intra-AS route if a type LSA_OMP_PATH_LOAD Opaque LSA exists for the summary LSA and more than one ABR is advertising the summary route and the equivalent load for the summary LSA is greater than 50% and the equivalent load is not sufficiently smaller than the intra-AS loading. If a separate structure is not used for the intra-AS route, then the next hop structure associated with the reaching the ABR or set of ABRs is used. For external routes, if an equivalent loading exists, and more than one ASBR is advertising the route, and the equivalent load is greater than 50% and the equivalent load is not sufficiently smaller than the internal route loading associated with the external next hop, then a separate structure is used. If a separate structure is not used for an external route, then the next hop structure associated with the reaching external next hop is used.

Hysterysis must be used in the algorithm for determining if an equivalent load on a summary LSA or external route is considered sufficiently smaller than the intra-AS equivalent load or if an external route is considered sufficiently smaller than the inter-AS equivalent load. For for the purpose of describing this algorithm one euivalent load is referred to as the more external, and the other as the more internal equivalent load.

If the more external equivalent load exceeds the more internal equivalent load by 5% and the more internal equivalent load is under 90%, then a separate next hop structure is created. If the more external equivalent load falls below 20% of the more internal equivalent load or the more internal equivalent load exceeds 95%, then an existing separate next hop structure is marked for removal and combined with the more internal next hop structure (see Section 3.4). The more external equivalent load should not fall significantly below the more internal unless either the traffic toward the more external destination increases or the loading on the more internal increases, since the more internal equivalent load will become the critical segment on the separate next hop structure if the load is sufficiently shifted but is unlikely to overshoot by 20%. These threshholds should be configurable at least per type of routes (inter-AS or external).

### 3.2 Critcally loaded segment

For every set of intra-AS paths, one link or part of the path is identified as the "critcally loaded" segment. This is the part of the path with the highest equivalent load as defined in Section 2.3. For an inter-AS route with a separate next hop structure, the critcally loaded segment is the critcally loaded segment for the intra-AS set of paths, or the summary LSA if the equivalent load on the summary LSA is greater. For an external route with a separate next hop structure, the critcally loaded segment is the critcally loaded segment for the internal route or the external route if the equivalent load on the external route is greater.

Each next hop structure has exactly one "critcally loaded" segment. An Opaque LSA may be the critcally loaded segment for no next hop structures if it is lightly loaded. An Opaque LSA may be the critcally loaded segment for many next hop structures if it is heavily loaded.

### 3.3 Load Adjustment Rate

In order to assure stability the rate of adjustment must be sufficiently limited. An adaptive adjustment rate method is used.

When the SPF is recalculated paths may disappear and new paths may appear. If a new equal cost path is added to a set of existing set of paths or a single path, the new path would have previously not carried any traffic of the traffic. The next hop structure is initialized or modified if there had previously been equal cost paths such that the new path is unused. The procedures for handling links coming up or going down is covered in Section 4.

A "critcally loaded" segment for a next hop structure is determined as described in Section 3.2. When the type LSA_OMP_LINK_LOAD Opaque LSA or type LSA_OMP_PATH_LOAD Opaque LSA for this segment is updated, load is shed toward all equal cost paths that do not involve that segment. A separate set of variables controlling rate of adjustment is kept for each alternate path.

The number of paths may exceed the number of next hops. The distinction between paths which share a next hop is important if one of the paths sharing a next hop goes down (see Section 4). This distinction is only needed in making the computations. When moving the next hop structure into the data structures used for forwarding, paths which share a common next hop may be combined. The following variables are kept for each path in a next hop structure.

1. The current "traffic share" (an integer, the range is 0 to 65355 for a CRC16 hash),
2. The current "move increment" (an integer, the range is 0 to 65355 for a CRC16 hash),
3. The number of moves in the same direction, referred to as the "move count".

If there is no prior history for a path, then the move increment is initialized to about 1% or 650. The number of moves in the same direction is initialized to 3. No loading adjustment is made on the first iteration.

If critcally loaded segment has not changed, or if the current path did not contain the previous critcally loaded segment, then the adjustment is continuing in the same direction. If the critcally loaded segment has just changed and the path being shed load toward contains the prior critcally loaded segment, then the adjustment direction has reversed for this path.

If the adjustment direction has reversed, the number of moves in the same direction is set to zero and the move increment is reduced by half. This move increment is then used.

If the adjustment is continuing in the same direction, the number of moves in the same direction is considered before increasing the move increment. This value is here referred to as the move count. The move increment is updated according to the following conditions.

1. If the move count is greater than 4, the move increment is increased by its current value divided by the number of equal cost paths in the next hop structure.
2. If the move count is less than or equal to 4, the move increment is increased by 1/2 its current value divided by the number of equal cost paths in the next hop structure.

The move increment is never less than one and the increase in move increment is never less than one. The move increment is never allowed to exceed the size of the hash space divided by the number of equal cost paths in the next hop structure.

The dramatic decrease in move increment when move direction is reversed and the slow increase in move increment when it remains in the same direction keeps the algorithm stable. The exponential nature of the increase allows the algorithm to track externally caused changes in traffic loading.

The amount of hash space allocated to a path is incremented by the move amount and the amount of hash space allocated to the critcally loaded path or paths are decremented by this amount.

This process is repeated for each alternate path. The new hash space boundaries are then moved to the forwarding engine.

## 3.4 Creating and destroying next hop structures

Section 3.1 describes the conditions under which a next hop structure would be needed for an inter-AS route or AS external route. Briefly, a separate next hop structure is needed if the loading indicated by the type LSA_OMP_PATH_LOAD Opaque LSA or exterior routing protocol is sufficiently high to require separate balancing for traffic to the summary-LSA or exterior route and the intra-AS loading is sufficiently low.

When a separate next hop structure is created, the same available paths appear in the structure, leading to the same set of ABR or ASBR. The balance on these available paths should be copied from the existing more internal next hop structure. By initializing the new next hop structure this way, a sudden change in loading is avoided if the summary route or external route sinks a great deal of traffic.

When a separate next hop structure can be destroyed, the traffic should be transitioned gradually. The next hop structure must be marked for deletion. The traffic share in this separate next hop structure should be gradually changed so that it exactly matches the traffic share in the more internal next hop structure. The gradual change should follow the adjustment rate schedule described in Section 3.3 where the move increment is increased gradually as moves continue in the same direction. Once the separate next hop structure marked for deletion matches the more internal next hop structure, the summary route or external route can be changed to point to the more internal next hop structure and the deletion can be made.

# 4 Dealing with Link Failures

Link failures do occur for various reasons. OSPF routing will converge to a new set of paths. Whatever load balance had previously existed will be upset and the load balancing will have to converge to a new load balanced state.

Links which are intermitent may be the most harmful. The OSPF "Hello" protocol is inadequate for handling intermitent links. When such a link is up it may draw traffic during periods of high loss, even brief periods of complete loss.

The inadequacies of the OSPF "Hello" protocol is well known and many implementations provide lower level protocol state information to OSPF to indicate a link in the "down" state. For example, indications may include carrier loss, excessive framing errors, unavailable seconds, or loss indications from PPP LQM.

Even where the use of a link is avoided by providing indication of lower level link availability, intermitent links are still problematic. During a brief period immediately after a link state attribute is initially flooded OSPF state can be inconsistent among routers within the AS. This inconsistency can cause intermittent routing loops and have a severe short term impact on link loading. An oscillating link can cause high levels of loss and is generally better off held in the neighbor adjacency "down" state. The algorithm described in the [4] can be used when advertising OSPF type 1 or type 2 LSA (router and network LSAs).

Regardless as to whether router and network LSAs are damped, neighbor adjacency state changes will occur and router and network LSAs will have to be handled. The LSA may indicate an up transition or a down transition. In either an up or down transition, when the SPF algorithm is applied, existing paths from the router doing the SPF to specific destinations may no longer be usable and new paths may become usable. In the case of an up transition, some paths may no longer be usable because their cost is no longer among those tied for the best. In the case of down transitions, new paths may become usable because they are now the best path still available.

When a path becomes unusable, paths which previously had the same cost may remain. This can only occur on an LSA down transition. A new next hop entry should be created in which the proportion of source/destination hash space allocated to the now infeasible path is distributed to the remaining paths proportionally to their prior allocation. Very high loading percentages should result, triggering an increase in LSA_OMP_LINK_LOAD Opaque LSA flooding rate until convergence is approached.

When a new path becomes usable it may be tied for best with paths carrying existing traffic. This can only occur on an LSA up transition. A new next hop entry should be created in which the loading on the new path is zero. If such a path were to oscillate, little or no load would be affected. If the path remains usable, the shift of load to this path will accellerate until a balance is reached.

If a completely new set of best paths becomes available, the load should be split across the available paths, proportional to 10% of link capacity plus the remaining link bandwidth as determined by prior LSA_OMP_LINK_LOAD Opaque LSA values.

# A Data Formats

@@ This is obviously a very important piece and is missing.

# B Concise Statement of the Algorithms

@@ MIB counters -> pseudo code – pull code from the simulation and past it here.

# C Changes to Existing OSPF Implementations

@@ BSD and gated would make a nice reference implementation

## C.1 Forwarding

@@ ip_output.c or equiv

## C.2 Routing process interface to forwarding

@@ route socket or equiv

## C.3 The routing process

@@ gated or equiv

# D Algorithm Performance

@@ see http://engr.ans.net/ospf-omp

### D.1 Conversion from Loss to Equivalent Load

@@ black art - consult local TCP guru - very few exist

### D.2 Performance when tracking traffic load change

@@ see http://engr.ans.net/ospf-omp/ramp.html

### D.3 Performance when traffic loading is constant

@@ it converges and oscillates a tiny bit - about 0.5% in simulations.

### D.4 Convergence after a major perturbation

@@ see simulations at http://engr.ans.net/ospf-omp - we're going kill a link and watch the thing converge at some later date.

## References

[1] Rob Coltun. The ospf opaque lsa option. Internet Draft (Work in Progress) draft-ietf-ospf-opaque-04, Internet Engineering Task Force, 2 1998. ftp://ds.internic.net/internet-drafts/draft-ietf-ospf-opaque-04.txt.

[2] M. Mathis, J. Semke, J. Mahdavi, and T. Ott. The macroscopic behavior of the TCP congestion avoidance algorithm. *ACM Computer Communication Review*, 27(3), July 1997.

[3] J. Moy. Ospf version 2. Technical Report RFC 2178, Internet Engineering Task Force, 1997. ftp://ds.internic.net/rfc/rfc2178.txt.

[4] C. Villamizar, R. Govindan, and R. Chandra. Bgp route flap damping. Internet Draft (Work in Progress) draft-ietf-idr-route-damp-02, Internet Engineering Task Force, 2 1998. ftp://ds.internic.net/internet-drafts/draft-ietf-idr-route-damp-02.txt.

## Security Considerations

@@ You can't be sure if loading the *.doc version of this draft into your word processor may infect you with a dangerous virus so a *.doc version has not been made available. [Something vaguely serious to go in here at a later date, subject to IESG whim de jour.] @@

## Author's Addresses

Curtis Villamizar
ANS Communications
<curtis@ans.net>