Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 22, 2018

H. Chen
D. Cheng
Huawei Technologies
M. Toy
Verizon
Y. Yang
IBM

April 20, 2018

OSPF Flooding Reduction
draft-cc-ospf-flooding-reduction-01

Abstract

This document proposes an approach to flood OSPF link state advertisements on a topology that is a subgraph of the complete OSPF topology per underline physical network, so that the amount of flooding traffic in the network is greatly reduced, and it would reduce convergence time with a more stable and optimized routing environment. The approach can be applied to any network topology in a single OSPF area, and can be used in both OSPFv2 ([RFC2328]) network and OSPFv3 ([RFC5340]) network.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 October 22, 2018.

Copyright Notice

Table of Contents

1. Introduction

For some networks such as dense Data Center (DC) networks, the existing OSPF Link State Advertisement (LSA) flooding mechanism is not efficient and may have some issues. The extra LSA flooding consumes network bandwidth. Processing the extra LSA flooding, including receiving, buffering and decoding the extra LSAs, wastes memory space and processor time. This may cause scalability issues and affect the network convergence negatively.

A flooding reduction method between spines and leaves is proposed in [I-D.shen-isis-spine-leaf-ext]. The problem on flooding reduction and an architectural solution are discussed in [I-D.li-dynamic-flooding]. This document proposes an approach to flood OSPF LSAs on a topology that is a subgraph of the entire OSPF topology per underline physical network, so that the amount of flooding traffic in the network is greatly reduced. The workload for processing the extra LSA flooding is decreased significantly. This would improve the scalability and speed up the network convergence, stable and optimize the routing environment.

The approach proposed is applicable to any network topology in a single OSPF area. It can be used in both a OSPFv2 network and a OSPFv3 network. The approach is backward compatible.

2. Problem Statement

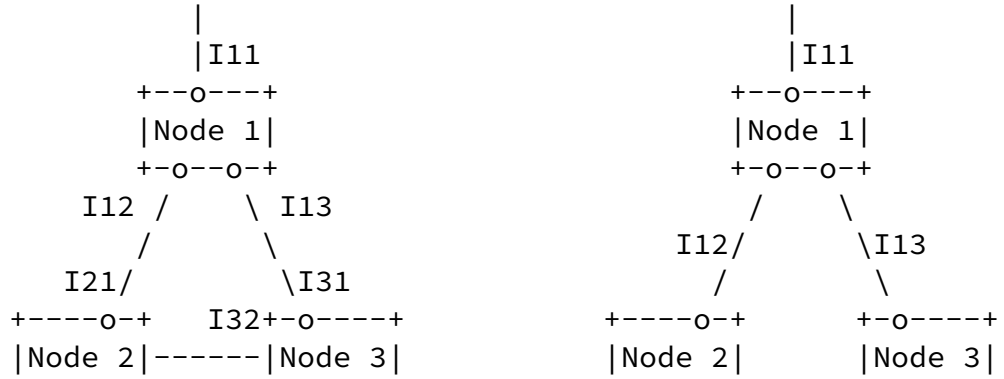
OSPF, like other link-state routing protocols, deploys a so-called reliable flooding mechanism, where a node must transmit a received or self-originated LSA to all its OSPF interfaces (except the interface where a LSA is received) in the defined context. While this mechanism assures each LSA being distributed to every OSPF node in the relevant routing area or domain, the side-effect is that the mechanism often causes redundant LSAs in individual network segments (e.g., on an OSPF point-to-point link or a broadcast subnet), which

in turn forces OSPF nodes to process identical LSAs more than once. This results waste of OSPF link bandwidth and OSPF nodes' computing resources, and the delay of OSPF topology convergence.

The problem explained above becomes more serious in OSPF networks with large number of nodes and links, and in particular, higher degree of interconnection (e.g., meshed topology, spine-leaf topology, etc,). In some environment such as in data centers, the drawback of the existing flooding mechanism has already caused operational problems, including repeated and waves of flooding storms, chock of computing resources, slow convergence, oscillating topology changes, instability of routing environment.

One example is as shown in Figure 1 (a), where Node 1, Node 2 and Node 3 are interconnected in a mesh. When Node 1 receives a new or updated OSPF LSA on its interface I11, it by default would forward to its interface Il2 and I13 towards Node 2 and Node 3, respectively, after processing. Node 2 and Node 3 upon reception of the LSA and after processing, would potentially flood the same LSA over their respective interface I23 and I32 toward each other, which is obviously not necessary and at the cost of link bandwidth as well as both nodes' computing resource.

In example Figure 1 (b), Node 2 and Node 3 both connect to a LAN where Node 4, Node 5 and Node 6 also connect to. When Node 1 receives a LSA as in (a) and floods it to Node 2 and Node 3 respectively, the two nodes would in turn both (instead of one) flood to the LAN, which is unnecessary and at the cost of link bandwidth as well as computing resource of all nodes connected to the LAN.



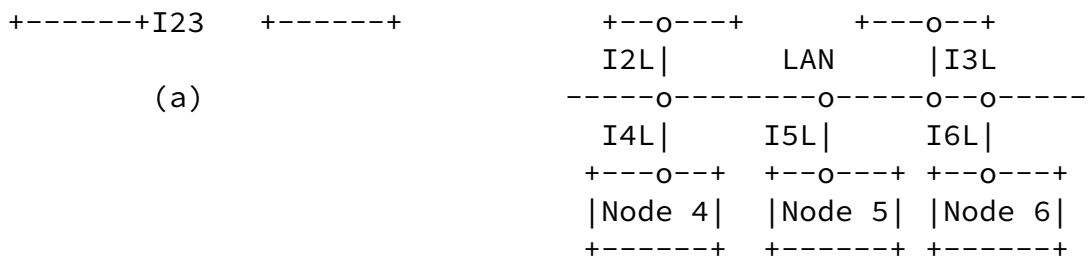


Figure 1

3. Flooding Topology

It is a norm that an OSPF node sending a received LSA and self-originated LSA to all its OSPF interfaces (except that where a LSA is received), as the reliable-flooding mechanism requires, i.e., any

OSPF LSA would potentially traverses on each OSPF link in a given OSPF network topology, sometimes both directions. As demonstrated in Section 2, dissemination over the entire OSPF network topology has drawbacks.

To change OSPF's aggressive flooding behavior, a flooding topology is introduced. For a given OSPF network topology, a flooding topology is a sub-graph or sub-network of the given network topology that has the same reachability to every node as the given network topology. Thus all the nodes in the given network topology MUST be in the flooding topology. All the nodes MUST be inter-connected directly or indirectly. As a result, OSPF flooding will in most cases occur only on the flooding topology, that includes all OSPF nodes but a subset of OSPF links. Note even the flooding topology is a sub-graph of the original OSPF topology, any single LSA MUST still be disseminated in the entire OSPF network.

There are many different flooding topologies for a given OSPF network topology. A chain connecting all the nodes in the given network topology is a flooding topology. A circle connecting all the nodes is another flooding topology. A tree connecting all the nodes is a flooding topology. In addition, the tree plus the connections

between some leaves of the tree and branch nodes of the tree is a flooding topology.

There are many different ways to construct a flooding topology for a given OSPF network topology. A few of them are listed below:

- One node in the network builds a flooding topology and floods the flooding topology to all the other nodes in the network (This seems not very good. Flooding the flooding topology may increase the flooding.);
- Each node in the network automatically calculates a flooding topology by using the same algorithm (No flooding for flooding topology);
- Links on the flooding topology are configured statically.

The minimum requirement for a flooding topology is all OSPF nodes are interconnected (directly or indirectly), but there is only one path from any node to any other node. While this lean-and-mean type of flooding topology degrades OSPF flooding traffic volume to the least, it may introduce some delay of topology convergence in the network with some network topologies. To compensate convergence efficiency, additional OSPF links may be added as part of the flooding topology. There is a trade-off between the density of the flooding topology and the convergence efficiency.

Note that the flooding topology constructed by an OSPF node is dynamic in nature, that means when the OSPF's base topology (the entire topology graph) changes, the flooding topology (the sub-graph) MUST be re-computed/re-constructed to ensure that any node that is reachable on the base topology MUST also be reachable on the flooding topology.

For reference purpose, some algorithms that allow OSPF nodes to automatically compute flooding topology are elaborated in Appendix A. However, this document does not attempt to standardize how a flooding topology is established.

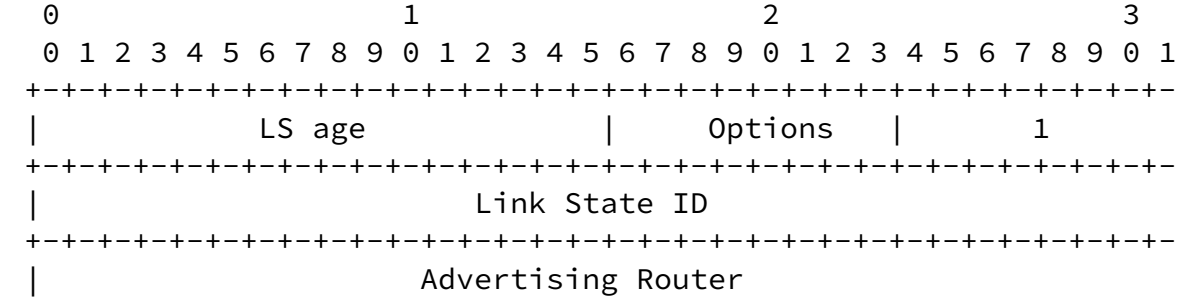
4. Extensions to OSPF

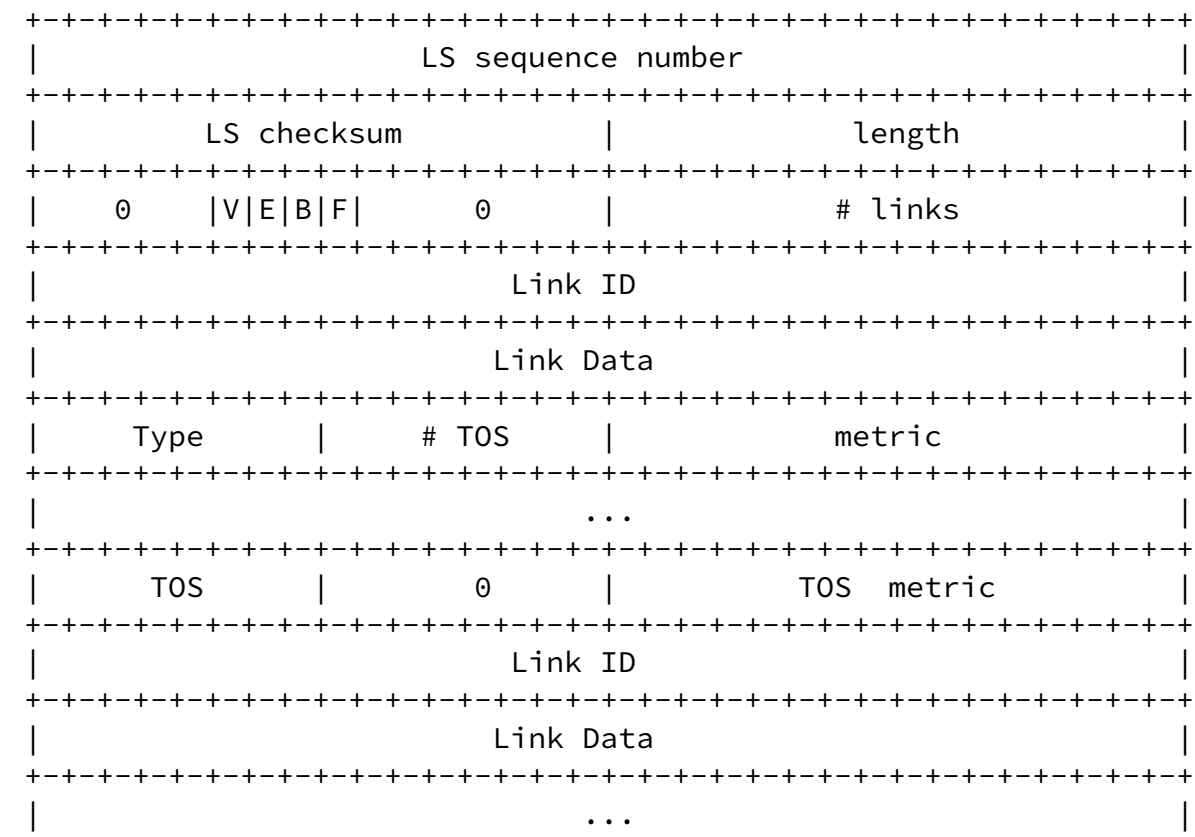
A 1-bit flag F is defined in an OSPF Router LSA. Flag F set to 1 indicates that the router supports OSPF LSA flood reduction described in this document; and Flag F set to 0 indicates that the router does not do so.

This flag is used for an OSPF node during the process of computing a flooding topology. An OSPF node that advertises its Router LSA with "F" bit set to 1 MUST always be included in the flooding topology computed by other OSPF nodes; but in contrast, the node with "F" bit set to zero may or may not be included in the flooding topology by other nodes, depending on how other nodes construct their flooding topology.

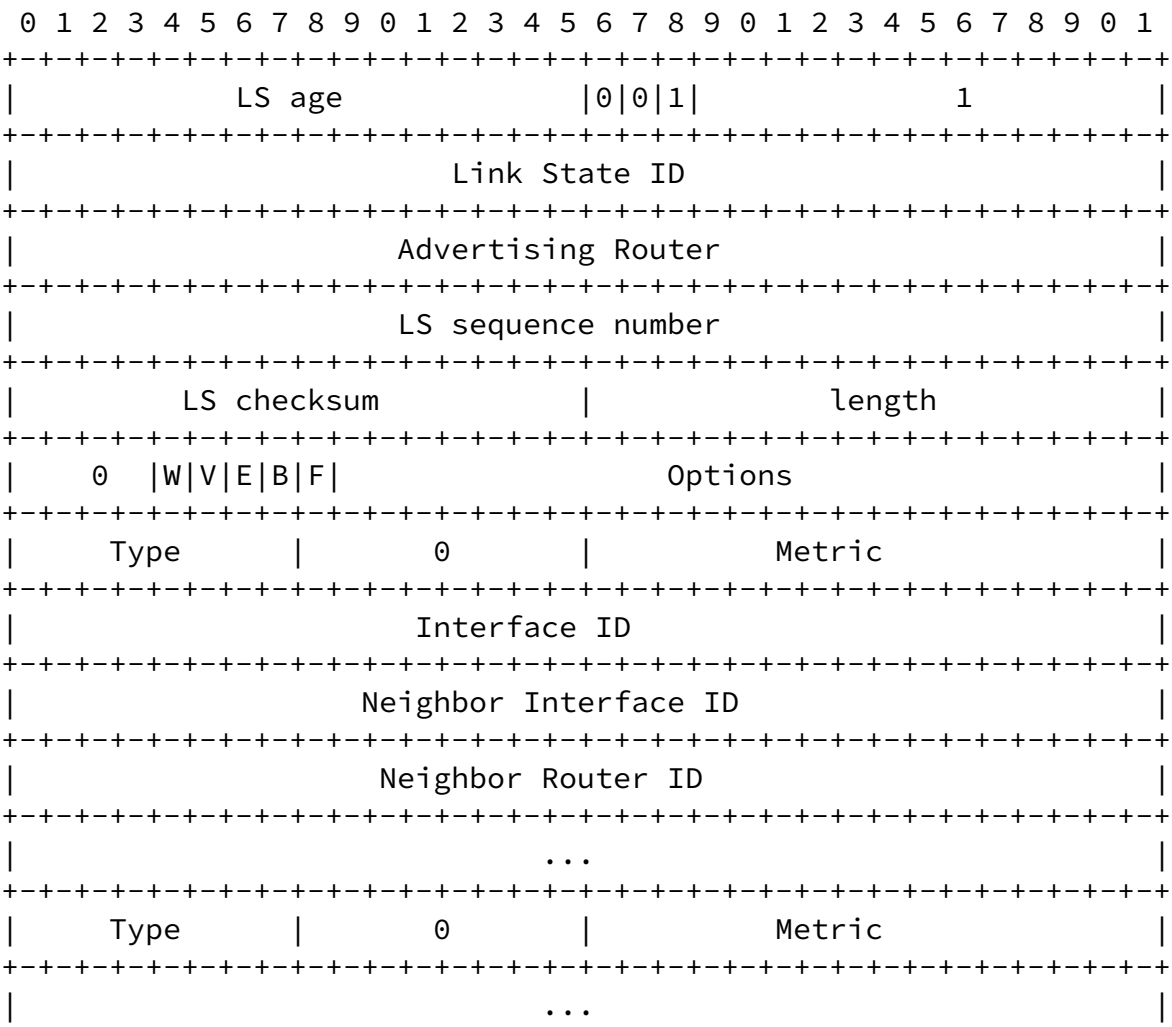
This flag can also be used for an OSPF node to trigger a decision whether it wants to perform LSA flooding to its neighbor.

The format of an OSPFv2 Router LSA with flag F is illustrated below.





The format of an OSPFv3 Router LSA with flag F is shown below.



5. Flooding Behavior

5.1. Nodes Support Flooding Reduction

This section describes OSPF flooding behavior for OSPF nodes that support flooding reduction described in this document. For these nodes, they MUST set "F" bit to 1 in their Router LSA (see Section 4). The flooding behavior for these nodes differs from that as specified in OSPFv2 ([RFC2328]) and OSPFv3 ([RFC5340]). Section 5.1.1 describes the flooding behavior when an OSPF node receives an OSPF LSA from one of its interface, and Section 5.1.2 describes the flooding behavior for LSA originated by itself.

The revised flooding procedure MUST flood LSAs to every node in the network in any case, as the standard OSPF flooding procedure does.

It assumes that the OSPF node of which the flooding behavior is described below is on the flooding topology, i.e., the node and at

least one of its OSPF interface are on the flooding topology, where:

1. When the node has only one interface on the flooding topology, the node is a leaf on the topology.

2. When the node has two interfaces on the flooding topology, the node is a transit node on the topology.

3. A flooding topology with nodes having one or two interfaces on the topology is a lean graph, i.e., there is only one path from any node to any other node on the graph. For flooding efficiency, there could be extra OSPF interfaces that are on the flooding topology, i.e., a node may have more than two interfaces that belong to the flooding topology.

5.1.1. Receiving an OSPF LSA

The flooding behavior when an OSPF node receives a newer OSPF LSA that is not originated by itself from one of its OSPF interface is as follows:

1. The LSA is received on a link that is on the flooding topology. The LSA is flooded only to all the other interfaces that are on the flooding topology.

2. The LSA is received on a link that is not on the flooding topology. This situation can happen when a neighboring node on a point-to-point link newly forms adjacency with the receiving node, or is not currently on the flooding topology; it can happen when the LSA sending neighbor does not support the OSPF flooding reduction (i.e., with "F" bit set to zero); it can also happen as the receiving link is a broadcast-type interface. The LSA is flooded only to all other interfaces that are on the flooding topology.

3. In both cases above, if there is any neighboring node that is advertising its Router LSA with "F" bit set to zero (see Section 4) but it is not on the flooding topology, the received LSA MUST also be sent to this neighboring node.

In any case, the LSA must not be transmitted back to the receiving interface.

Note before forwarding a received LSA, the OSPF node would do the normal processing as usual.

5.1.2. Originating an OSPF LSA

The flooding behavior when an OSPF node originates an OSPF LSA is as follows:

1. If it is a refresh LSA, i.e., there is no significant change contained in the LSA comparing to the previous LSA, the LSA is transmitted over links on the flooding topology. In addition, if there is any neighboring node that is advertising its Router LSA with "F" bit set to zero (see Section 4) but it is not on the flooding topology, the LSA MUST also be sent to this neighboring node.

2. Otherwise, the LSA is transmitted to all OSPF interfaces. Choosing this action instead of limiting to links on flooding topology would speed up the synchronization around the advertising node's neighbors, which could then disseminate the new LSA quickly.

5.1.3. An Exception Case

In Section 5.1.1 and Section 5.1.2, there are times when an OSPF node sending out a LSA to an interface on the flooding topology detects a critical interface or node failure. A critical interface is an interface on the flooding topology and is the only connection among some nodes on the flooding topology. When this interface goes down, the flooding topology will be split. Note the flooding topology was pre-computed/pre-constructed; but if at the time a critical interface or a node goes down before a re-newed flooding topology can be computed/constructed, the OSPF node MUST send out the LSA to all interfaces (except where it is received from) as a traditional OSPF node would do. This handling is also taking place if there are more than one interfaces or nodes on the existing flooding topology fail, i.e., if more than one interfaces or nodes on the flooding topology fail, the OSPF node does traditional flooding before the flooding topology is re-built.

5.1.4. One More Note

The destination address that is used when an OSPF node sends out a LSA on an interface on its flooding topology follows the specification in OSPFv2 ([RFC2328]) and OSPFv3 ([RFC5340]). This means on a local LAN, all other OSPF nodes will receive the LSA.

5.2. Nodes Not Support Flooding Reduction

For OSPF nodes that do not support flooding reduction as described in this document, they MUST set "F" bit to 0 in their Router LSA (see

Section 4); note this is also a default setting. These nodes may or may not be on the flooding topology constructed by other nodes that support flooding reduction in the same OSPF area, however that is not a business these nodes need to concern.

The LSA flooding behavior of OSPF nodes that do not support reduction as described in this document MUST follow that as specified in OSPFv2 ([RFC2328]) and OSPFv3 ([RFC5340]).

6. Security Considerations

This document does not introduce any security issue.

7. IANA Considerations

This document has no request to IANA.

8. Acknowledgements

The authors would like to thank Acee Lindem, Zhibo Hu, Robin Li, Stephane Litkowski and Alvaro Retana for their valuable suggestions and comments on this draft.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <https://www.rfc-editor.org/info/rfc2328>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

9.2. Informative References

[I-D.li-dynamic-flooding]
Li, T., "Dynamic Flooding on Dense Graphs",

draft-li-dynamic-flooding-04 (work in progress), March 2018.

[I-D.shen-isis-spine-leaf-ext]
Shen, N., Ginsberg, L., and S. Thyamagundalu, "IS-IS Routing for Spine-Leaf Topology", draft-shen-isis-spine-leaf-ext-05 (work in progress), January 2018.

Appendix A. Algorithms to Build Flooding Topology

There are many algorithms to build a flooding topology. A simple and efficient one is briefed below.

- Select a node R according to a rule such as the node with the biggest/smallest node ID;
- Build a tree using R as root of the tree (details below); and then
- Connect k (k>=0) leaves to the tree to have a flooding topology (details follow).

A.1. Algorithms to Build Tree without Considering Flag F

An algorithm for building a tree from node R as root starts with a candidate queue Cq containing R and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft

2. If Cq is empty, then return with Ft

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and X1, X2, ..., Xn are in a special order. For example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than the cost of the link between A and Xj (j = i + 1). If two costs are the same, Xi's ID is less than Xj's ID. In another example, X1, X2, ..., Xn are ordered by their IDs. If they are not ordered, then make them in the order.

4. Add Xi (i = 1, 2, ..., n) into the end of Cq, goto step 1.

Another algorithm for building a tree from node R as root starts with a candidate queue Cq containing R and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft

2. If Cq is empty, then return with Ft

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and X1, X2, ..., Xn are in a special order. For example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than the cost of the link between A and Xj (j = i + 1). If two costs are the same, Xi's ID is less than Xj's ID. In another example, X1, X2, ..., Xn are ordered by their IDs. If they are not ordered, then make them in the order.

4. Add Xi (i = 1, 2, ..., n) into the front of Cq and goto step 1.

A third algorithm for building a tree from node R as root starts with a candidate list Cq containing R associated with cost 0 and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft

2. If all the nodes are on Ft, then return with Ft

3. Suppose that node A is associated with a cost Ca which is the cost from root R to node A, node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and the cost of the link between A and Xi is LCi (i=1, 2, ..., n). Compute Ci = Ca + LCi, check if Xi is in Cq and if Cxi (cost from R to Xi) < Ci. If Xi is not in Cq, then add Xi with cost Ci into Cq; If Xi is in Cq, then If Cxi > Ci then replace Xi with cost Cxi by Xi with Ci in Cq; If Cxi == Ci then add Xi with cost Ci into Cq.

4. Make sure Cq is in a special order. Suppose that Ai (i=1, 2, ..., m) are the nodes in Cq, Cai is the cost associated with Ai, and IDi is the ID of Ai. One order is that for any k = 1, 2, ..., m-1, Cak < Caj (j = k+1) or Cak = Caj and IDk < IDj. Goto step 1.

A.2. Algorithms to Build Tree Considering Flag F

An algorithm for building a tree from node R as root with consideration of flag F starts with a candidate queue Cq containing R associated with previous hop PH=0 and an empty flooding topology Ft:

1. Remove the first node A with its flag F set to one from the candidate queue Cq if there is such a node A; otherwise (i.e., if there is not such node A in Cq), then remove the first node A from Cq. Add A into the flooding topology Ft.

2. If Cq is empty or all nodes are on Ft, then return with Ft

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in the flooding topology Ft and X1, X2, ..., Xn are in a special order considering whether some of them with flag F = 1. For example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than that of the link between A and Xj (j = i + 1). If two costs are the same, Xi's ID is less than Xj's ID. The cost of a link is redefined such that 1) the cost of a link between A and Xi both with F = 1 is much less than the cost of any link between A

and Xk where Xk with F=0; 2) the real metric of a link between A and Xi and the real metric of a link between A and Xk are used as their costs for determining the order of Xi and Xk if they all (i.e., A, Xi and Xk) with F = 1 or none of Xi and Xk with F = 1.

4. Add Xi (i = 1, 2, ..., n) associated with previous hop PH=A into the end of the candidate queue Cq, and goto step 1.

Another algorithm for building a tree from node R as root with consideration of flag F starts with a candidate queue Cq containing R associated with previous hop PH=0 and an empty flooding topology Ft:

1. Remove the first node A with its flag F set to one from the candidate queue Cq if there is such a node A; otherwise (i.e., if there is not such node A in Cq), then remove the first node A from Cq. Add A into the flooding topology Ft.

2. If Cq is empty or all nodes are on Ft, then return with Ft.

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in the flooding topology Ft and X1, X2, ..., Xn are in a special order considering whether some of them with F = 1. For example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than the cost of the link between A and Xj (j = i + 1). If two costs are the same, Xi's ID is less than Xj's ID. The cost of a link is redefined such that 1) the cost of a link between A and Xi both with F = 1 is much less than the cost of any link between A and Xk where Xk with F = 0; 2) the real metric of a link between A and Xi and the real metric of a link between A and Xk are used as their costs for determining the order of Xi and Xk if they all (i.e., A, Xi and Xk) have F = 1 or none of Xi and Xk has F = 1.

4. Add Xi (i = 1, 2, ..., n) associated with previous hop PH=A into the front of the candidate queue Cq, and goto step 1.

A third algorithm for building a tree from node R as root with consideration of flag F starts with a candidate list Cq containing R associated with low order cost Lc=0, high order cost Hc=0 and previous hop ID PH=0, and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft.

2. If all the nodes are on Ft, then return with Ft

3. Suppose that node A is associated with a cost Ca which is the cost from root R to node A, node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and the cost of the link between A and Xi is LCi (i=1, 2, ..., n). Compute Ci = Ca + LCi, check if Xi is in Cq and if Cxi (cost from R to Xi) < Ci. If Xi is not in Cq, then add Xi with cost Ci into Cq; If Xi is in Cq, then If Cxi > Ci then replace Xi with cost Cxi by Xi with Ci in Cq; If Cxi == Ci then add Xi with cost Ci into Cq.

4. Suppose that node A is associated with a low order cost LCa which is the low order cost from root R to node A and a high order cost HCa which is the high order cost from R to A, node Xi (i = 1, 2, ..., n) is connected to node A and not in the flooding topology Ft and the real cost of the link between A and Xi is Ci (i=1, 2, ..., n). Compute LCxi and HCxi: LCxi = LCa + Ci if both A and Xi have flag F set to one, otherwise LCxi = LCa HCxi = HCa + Ci if A or Xi does not have flag F set to one, otherwise HCxi = HCa If Xi is not in Cq, then add Xi associated with LCxi, HCxi and PH = A into Cq; If Xi associated with LCxi' and HCxi' and PHxi' is in Cq, then If HCxi' > HCxi then replace Xi with HCxi', LCxi' and PHxi' by Xi with HCxi, LCxi and PH=A in Cq; otherwise (i.e., HCxi' == HCxi) if LCxi' > LCxi , then replace Xi with HCxi', LCxi' and PHxi' by Xi with HCxi, LCxi and PH=A in Cq; otherwise (i.e., HCxi' == HCxi and LCxi' == LCxi) if PHxi' > PH, then replace Xi with HCxi', LCxi' and PHxi' by Xi with HCxi, LCxi and PH=A in Cq.

5. Make sure Cq is in a special order. Suppose that Ai (i=1, 2, ..., m) are the nodes in Cq, HCai and LCai are low order cost and high order cost associated with Ai, and IDi is the ID of Ai. One order is that for any k = 1, 2, ..., m-1, HCak < HCaj (j = k+1) or HCak = HCaj and LCak < LCaj or HCak = HCaj and LCak = LCaj and IDk < IDj. Goto step 1.

A.3. Connecting Leaves

Suppose that we have a flooding topology Ft built by one of the algorithms described above. Ft is like a tree. We may connect k (k >=0) leaves to the tree to have a enhanced flooding topology with

more connectivity.

Suppose that there are m (0 < m) leaves directly connected to a node X on the flooding topology Ft. Select k (k <= m) leaves through using a deterministic algorithm or rule. One algorithm or rule is to select k leaves that have smaller or larger IDs (i.e., the IDs of these k leaves are smaller/bigger than the IDs of the other leaves directly connected to node X). Since every node has a unique ID, selecting k leaves with smaller or larger IDs is deterministic.

If k = 1, the leaf selected has the smallest/largest node ID among the IDs of all the leaves directly connected to node X.

For a selected leaf L directly connected to a node N in the flooding topology Ft, select a connection/adjacency to another node from node L in Ft through using a deterministic algorithm or rule.

Suppose that leaf node L is directly connected to nodes Ni (i = 1,2,...,s) in the flooding topology Ft via adjacencies and node Ni is not node N, IDi is the ID of node Ni, and Hi (i = 1,2,...,s) is the number of hops from node L to node Ni in the flooding topology Ft.

One Algorithm or rule is to select the connection to node Nj (1 <= j <= s) such that Hj is the largest among H1, H2, ..., Hs. If there is another node Na (1 <= a <= s) and Hj = Ha, then select the one with smaller (or larger) node ID. That is that if Hj == Ha and IDj < IDa then select the connection to Nj for selecting the one with smaller node ID (or if Hj == Ha and IDj < IDa then select the connection to Na for selecting the one with larger node ID).

Suppose that the number of connections in total between leaves selected and the nodes in the flooding topology Ft to be added is NLc. We may have a limit to NLc.

Authors' Addresses

Huaimo Chen
Huawei Technologies

Email: huaimo.chen@huawei.com

Dean Cheng
Huawei Technologies

Email: dean.cheng@huawei.com

Mehmet Toy
Verizon
USA

Email: mehmet.toy@verizon.com

Yi Yang
IBM
Cary, NC
United States of America

Email: yyietf@gmail.com