Uni-directional links in routing

Karina Gribanova
Department of Computer Science
Helsinki University of Technology
gribkari@cc.hut.fi

Abstract

A number of network configurations, in which pairs of nodes are connected by uni-directional links (UDLs), is increasing. For the QoS related reasons, it may be also advantageous to assume that a bi-directional link (BDL) is a set of two uni-directional links. Current routing protocols are designed for networks where all links are bi-directional. They fail to perform in networks with UDLs therefore new routing schemes are needed. In this paper, several routing solutions for networks with UDLs are overviewed, focusing on the uni-cast routing.

1 Introduction

Uni-directional links (UDLs) are emerging and in future networks may be as ubiquitous as bi-directional links. Moreover, bi-directional links in reality are not symmetrical, since the bandwidth available in both directions differ, and it can be advantageous to split a bi-directional link into two uni-directional links for the Quality of Service (QoS) routing.

Since conventional routing protocols assume that pairs of routers are connected by bi-directional and symmetric links, they cannot be applied directly in the networks with UDLs, therefore alternative routing schemes are needed.

Tunnelling was researched by Uni-directional Link Routing (UDLR) Working Group (WG) of Internet Engineering Task Force (IETF). The research work was concluded by issuing the RFC 3077 in March 2001. The RFC 3077 describes a link-layer tunnelling mechanism (LLTM) for supporting UDLs in the Internet.

INRIA contributed in developing solutions based on Routing Protocol Modification (RPM), however, there is no information available about its further work on this topic.

Several new UDL routing protocols have been proposed by the research community. Some of them have inherited the design of conventional routing protocols, such as Distance-Vector UDL Routing Protocol [1], Link-State UDL Routing Protocol [2, 3], while the others were created "from scratch", such as Circuit-Based UDL routing protocol [5].

This paper gives an overview of several routing solutions for networks with UDLs. The layout of the paper is as follows. Section 2 defines uni-directional links and describes technologies in which they are experienced. In Section 3 different network configurations with UDLs are presented. Section 4 analyze routing solutions, i.e. tunnelling, RPM and new protocols. Section 5 concludes the paper.

2 Uni-directional link

Uni-directional link can be defined as one-way transmission link, i.e. a link with zero return bandwidth. There are several technologies based on UDLs, e.g. Geostationary Earth Orbit (GEO) satellite links, ad hoc networks, where routers do not have the same transmitting power, optical fiber cables, etc. Examples of systems with UDLs are Direct Broadcast Systems (DBSs) via satellite and Mobile Radio Networks. In this overview, the focus is on the uni-directional GEO satellite links. The Internet services via GEO satellite links can be provided employing the digital television technology. [4, 6]

2.1 GEO satellite links

Geostationary Earth Orbit satellite, in addition to its application in other fields, is also emerging as a technology for the Internet access.

A satellite network includes two types of stations, i.e. *feeds* and *receivers*. A feed performs transmitter functions. It incorporates a device emitting the data towards a GEO satellite (VSAT antenna), and a router connected to it via a "send-only" interface. Feeds cover nationwide areas and are installed at strategic positions in order to create shorter paths over the Internet. The main parts of a receiver are a satellite dish oriented toward a GEO satellite, and a router connected to it via a "receive-only" interface. The routers of feeds and receivers have one or more extra interfaces connected to the Internet.

Application of the GEO satellite links offers several advantages:

- *High bandwidth transmission*

- *Large geographic coverage*

- *Favorable conditions for multi-casting implementation due to the broadcast nature of the links.*

Since the cost of the feeds is too high for their application on the large scale, and the cost of the receive-only hardware is negligible, the satellite operators are willing to provide high speed receive-only access to the Internet. This approach is especially advantageous for the multi-cast applications, such as Video-on-Demand (VoD), web site mirroring, seminar multi-casting, etc.

2.1.1 Digital television

Digital television (DTV) is defined as transmission of television signals using digital methods instead of analog ones.

DVB specification	Transmission media
DVB-S	Satellite broadcasting system for the 11/12 GHz bands
DVB-C	Cable transmission system focused on 8-Mhz cable channels
DVB-CS	Transmission via (S)MATV systems
DVB-T	Terrestrial broadcasting system for the UHF bands
DVB-MS	MVDS broadcasting system (> 10 GHz)
DVB-H	Modem uplink, broadband downlink

Table 1: DVB specifications

The technical specifications for the DTV are developed by the *Digital Video Broadcasting Project (DVB)*, which is an industry-led consortium of over 300 broadcasters, manufacturers, network operators, software developers, regulatory bodies and others in over 35 countries committed [12]. Many organizations have contributed to the DVB project either by making available some results of their work (MPEG, DAVIC) or by actively co-operating with DVB in transforming specifications into standards and norms (ETSI, CENELEC).

The DVB transmission systems use the concept of data containers or data pipelines which can carry all kinds of data "quasi-error free" over all kinds of media (satellite, cable, (S)MATV, terrestrial channels, MMDS). A set of specifications has been developed to cover the aforementioned range of transmission media. Table 1 lists some of them [9].

In this overview, the main focus is on the Internet services provided via uni-directional satellite links, therefore the considered DVB standard can be one of those employing satellites for transmission, for example, DVB-S.

3   Network architecture models

Different objectives of applying satellite links result in different network architectures [8]. They are described below.

3.1   A UDL on top of BDL network

The application of satellite links for the Internet access was first proposed by satellite operators who were willing to make the satellites more profitable. Since satellites offer high bandwidth and large coverage area, they may provide a better connectivity to the Internet users. Considering that in most cases the Internet traffic is asymmetric, i.e. there are larger amounts of traffic flowing from the service provider to the receivers than opposite direction, also taking into account the high cost of feeds and negligible cost of the receivers, an option may be providing a semi-connectivity to the Internet over satellite links from the service provider to the receivers. Since the receivers need to send control data, such as requests of services and acknowledgements to the feed, a back-channel is required. It can be provided, for example, by a PPP dial up line, a cellular modem or a terrestrial Internet connection via a network with bi-directional links.

The routing in such network configuration is an issue, when receivers are not single hosts, but subnetworks. In this case, the routing requires knowledge about the subnetworks accessible via satellite link.

3.2   Bi-directional islands interconnected via a UDL

In this context, *bidirectional islands* are entities, isolated from the rest of the Internet, inside which each node can establish a bi-directional communication with the others without a help of any external link[8]. The existence of uni-directional links inside the islands is also possible. However, there are no bi-directional links connecting one island to the other. Therefore, such entities can be interconnected using one or more UDLs. In most cases, it is not likely that there are UDLs in both directions, and the traffic needs to be relayed through the third party islands.

This network configuration may involve the inter-domain routing.

3.3   All UDL links

In this case, all links are UDLs within a subset or the whole Internet. BDLs can still exist between the nodes, but may be split into two UDLs for simplicity. Since in reality the bandwidth in both BDL directions is not exactly the same, as it is assumed by current routing protocols, this approach may have an advantage in the Quality of Service (QoS) routing, as it allows calculating the link metrics more accurately, i.e. for both directions separately. An example of the network with all UDL links is an ad hoc wireless network, in which pairs of routers do not have the same transmitting power.

4   Routing solutions

The existing routing protocols assume that the nodes in the network are interconnected by BDLs, i.e. symmetric links, which have the same bandwidth and routing metrics in both directions. Therefore in the networks with BDLs the link symmetry is assumed in reachability decisions and link metric calculations. However, it is not a case in the networks with UDLs, for example, the networks, which employ GEO satellite links. Here the problem arises, when a conventional routing protocol at the feed expects routing messages advertising the subnetworks beyond the UDL to come via the same link, that is not possible. Thus, the router, connected to the "outgoing" UDL would not be able to obtain the information about the networks reachable through the link, if a conventional routing protocol was applied. The terrestrial path could be used to transmit the routing data back to the feed, but, as it comes through another interface that it was sent too, it is automatically discarded.

Proposals for the routing in the networks with UDLs can be divided into three distinct approaches:

- *Tunnelling*

- *Routing Protocol Modification (RPM)*

- *New protocols.*

Both tunnelling and RPM are considered to be *short-term* solutions, while new protocols, which take into account the presence of UDLs in the network, could serve as a *long-term solution*. [4, 8]

The more detailed overview of the approaches is given below.

4.1 Tunnelling

Tunnelling is a mean to construct virtual networks by encapsulating data. Tunnelling approach adds a layer between the network interface and a routing software at both feed and receiver sides with a purpose to emulate a bi-directional satellite link where only a UDL is available. For example, when a receiver tries to send a packet via its *"receive-only interface"*, the packet is encapsulated and sent to the feed via one of its bidirectional interfaces. When the feed receives the packet, it is decapsulated and sent to the higher layers. A UDL is totally hidden from the applications therefore conventional routing protocols can be applied.

The RFC 3077 describes a *Link-Layer Tunneling Mechanism for networks with UDLs* designed by the UDLR working group of IETF.

Generic Routing Encapsulation [RFC 2784] is suggested as the tunnelling mechanism, since it supports carrying IP, ARP datagrams, and other layer-3 protocols between nodes.

In the assumed network topology feeds and receivers are interconnected by UDLs. Two kinds of feeds are considered, *send-only feeds* and *receive capable feeds*, which have both send and receive capabilities. In the proposed scheme, it is expected, that the number of feeds is small, but according to the authors the tunnelling mechanism can be applied for the topologies with any number of feeds as well. A feed can have *send-only* or *send-and-receive interface* connected to the UDL, a receiver has *receive-only interface* to the UDL, and both a feed and a receiver must have one or more additional bidirectional communication interfaces. A feed must be a router.

It was noted that this tunnelling mechanism is not designed for the topology where pairs of nodes are connected by two uni-directional links in opposite direction.

The following features of the networks with bi-directional links are emulated by the tunnelling mechanism:

- *A receiver can send a packet to a feed (point-to-point communication between a receiver and a feed)*

- *A receiver can send a broadcast/multicast packet on the link to all nodes (point-to-multipoint)*

- *A receiver can send a packet to another receiver (point-to-point communication between two receivers).*

- *A feed can send a packet to a send-only feed (point-to-point communication between two feeds)*

- *A feed can send a broadcast/multicast packet on the link to all nodes (point-to-multipoint)*

- *A feed can send a packet to a receiver or a receive capable feed (point-to-point).*

The tunnelling mechanism works as follows:

- On the receiver, a datagram is delivered to the link-layer of the uni-directional interface for transmission and encapsulated within a MAC header corresponding to the uni-directional link. Since the packet cannot be sent through receive-only interface, the packet is encapsulated within an IP header whose destination is the IP address of a feed's bi-directional interface, also called the *tunnel end-point*. If the destination MAC address is the MAC address of a feed interface connected to the uni-directional link, the destination IP address of the encapsulated datagram is the feed's tunnel end-point. Otherwise, if the destination MAC address is a MAC broadcast/multicast address or a MAC address that belongs to the unidirectional network but is not a feed address, the destination IP address of the encapsulated datagram is the default feed tunnel end-point. After encapsulation, a datagram is passed to the network layer which forwards it according to its destination address.

- On the uni-directional interface of the feed, packets are processed in two different ways. The packets may be generated by a local application or routed by the IP layer for forwarding. In this case, if the destination MAC address is a MAC address of the receiver or receive-capable feed, the packet is sent over the uni-directional link. Otherwise, if the destination MAC address is a send-only feed, a tunnel processing is applied. Encapsulated packets, received by the feed from another receiver or feed, enter a decapsulation process, after which the original link-layer packet is retrieved.

For the forwarding of encapsulated datagrams receivers and feeds must know the feeds' tunnel end-points, i.e. the IP addresses of bi-directional links on send-only feeds. Since the number of feeds is expected to be relatively small, at every feed the list of all feeds can be configured manually. It cannot be applied at the receivers due to scalability reasons and the requirement to manage the following events:

- ***New feed detection.*** When a new feed comes up, every receiver must create a tunnel to enable bidirectional communication with it.

- ***Loss of uni-directional link detection.*** When the uni-directional link is down, receivers must disable their tunnels.

- ***Loss of feed detection.*** When a feed is down, receivers must disable their corresponding tunnel.

The *Dynamic Tunnel Configuration Protocol (DTCP)* addresses the aforementioned issues. DTCP provides a means for receivers to dynamically discover the presence of feeds and to maintain a list of operational tunnel end-points. Feeds periodically announce the tunnel-point addresses over the unidirectional link. Receivers listen to these announcements and maintain a list of tunnel end-points. [6]

The tunnelling approach solves the problem of routing in the networks with UDLs by providing a means for the application of conventional Internet routing protocols. However, it is considered to be a short-term solution, since for being

suitable for longer terms it lacks scalability and is not efficient in the networks with receivers having a high level of mobility.

4.2 Routing Protocol Modification

Routing Protocol Modification is an alternative for tunnelling, proposed by INRIA. It is based on modifying the existing routing protocols to remove the BDL assumption. In this approach, the underlying network architecture is explicitly known to the routing protocols, thus, the receivers detect the presence of feeds dynamically and send messages to them via regular links.

The modifications needed are not the same depending on the nature of the considered routing protocol and can be as follows:

- *The routing protocol must be informed about the presence of UDL*

- *Receivers must maintain the list of IP addresses of feeds*

- *Receivers must sent routing messages to the feeds from the list periodically*

In the case of *Link State (LS)* protocols, such as *Open Shortest Path First (OSPF)*, all existing routers in the network have to be updated by the new version of the protocol. Differently, for *Distant Vector (DV)* protocols, such as *Routing Information Protocol (RIP)*, only the routers located at the edges of UDL require updating.

Several Internet drafts were issued, describing the changes proposed by INRIA to the popular routing protocols, such as OSPF and RIP, needed for their application in the networks with UDLs. For some reasons, INRIA did not continue the work further. Currently, there are no new Internet documents or research papers available concerning this topic. Nevertheless, as an example of RPM, some ideas from the INRIA proposals are overviewed below:

- If the network included only feeds, OSPF could be used almost unchanged [10]. However, in the networks, where both feeds and receivers are present, the routing protocol requires some extensions. In the proposed RPM scheme all protocol packets (Hello, link state request/update, etc) sent by the feed via the UDL are authenticated. When sending "Hello" packets over the UDL, the feeds authenticate them as *"satellite packet"*. Upon reception of these "Hello packets", receivers examine the authentication code, and if they discover that a packet was sent by a feed, the IP of the source is added to a list of *"potential neighbors"*. At constant time intervals, receivers send the "Hello" packets to the potential neighbors. These packets, are not sent to the multi-cast address "all OSPF routers", but a copy is sent to the uni-cast address of each potential neighbor. These packets are also authenticated as "satellite packet". When receiving these "Hello" packets feeds process them even if they are routed by another interface. Instead of having just a list of connected routers, a list of routers, which can only send or receive packets could be supported. It should be noted, that some ideas in the overviewed

INRIA proposal are ambiguous. For example, when a receiver adds the IP addresses of the uni-directional feeds' interfaces to the potential neighbors lists, and sends "Hello" messages to each address from that list, it is not explained, how the messages reach their destination, if they are sent to the uni-cast IP address of the send-only uni-directional link.

- Exactly the same modifications, as those for the OSPF, are proposed for the RIP protocol, therefore overviewing them would not provide any useful information.

The RPM approach has the following drawbacks:

- Without a certain standardization body involved, a variety of ad-hoc versions may be created, which will differ from the original version and address a particular problem.

- Some upper level protocols and software rely on the underlying topology, therefore they might only perform well on top of BDLs and might need to be modified when the UDLs are involved, that cannot be done in the short term.

Therefore the best solution may be updating the existing routing protocols in a whole and proposing a new standard where modifications allowing uni-directional routing are merged. In this case, the research of RPM could serve in the development of the new routing protocols for the networks with UDLs.

4.3 New routing protocols for the networks with UDLs

Design of new protocols for networks with UDLs, which explicitly and optimally take into account the underlying architecture of the network is assumed to be the best long-term solution for several reasons:

- Tunnelling provides a good (not optimal) short-term solution in the networks with a small number of UDLs. However, it lacks scalability, i.e. tunnelling will not work in specific network configurations, such as all-UDL network, since it assumes the existence of a bi-directional back-channel. If the back-channel itself is uni-directional and requires to set a tunnel, a deadlock situation may arise.

- The RPM is considered to be less advantageous than tunnelling, since the design and installation of modified protocols across the Internet in a short-term would be too costly when compared with the potential benefit.

- Many routing protocols anyway require modifications or extensions to remove asymmetric link assumptions. Splitting a BDL into two UDLs solves the problem, however, it also calls for the efficient uni-directional routing.

Several new protocols designed for the networks with UDLs are proposed in the literature. Some of them adopt design features of conventional dynamic routing protocols

and introduce the mechanisms to cope with UDL routing, while the others are created from scratch for the networks with UDLs.

The overview of several new UDLR algorithms is given in the following sections.

4.3.1 Distance Vector (DV) routing protocols for the networks with UDLs

In *Distance Vector* routing protocols, every router has a routing table that shows its best route to any destination. In the basic version of DV, the information recorded in the routing table is *the destination, the cost of the best route and the next hop*. The management of routes is distributed, since each router computes shortest routes and forward them itself. In order to avoid routing loops in case of inconsistency, an upper bound called infinity is fixed for distances. The distance to the unreachable destinations is set to infinity.

The DV routing algorithm performs as follows:

- In a specific time intervals, each router sends its routing table to the neighbor routers (not to all routers). In the beginning, only the local route *"from me to me, cost=0"* is advertised.

- Based on the information in the received neighbors' routing tables, a router updates its local table. A cost to reach the neighbor router is added to the cost of each destination advertised in the neighbor's routing table for obtaining the costs to those destinations from the local router. The neighbor router is considered the next hop on the way to the destinations. The calculation results are compared with the entries in the local routing table. If the destinations do not exist in the local table, they are added to it. If there are entries, in which the destinations have higher costs than calculated, they are modified including new costs and next hop routers (see Figure 1).

If a common DV routing protocol is applied in the network with UDLs, a receiver will conclude that it can reach the destinations, advertised by the feed via the UDL, that is not true, since the feed in not the next hop for the receiver, unless there exist a direct link in both directions. Moreover, exchange of routing tables between neighboring nodes is very difficult, because information can flow only in one direction.

In [1] a DV protocol is proposed for handling dynamic changes, and node and link failures in the networks with UDLs. Its design is based on the RIP.

The all-UDLs network is considered, but, according to the authors, the routing scheme can be successfully applied in the networks where not all links are UDLs. It is assumed, that if the network is strongly interconnected, there must exist a back-channel - a series of UDLs - between a pair of nodes interconnected by a UDL. The proposed routing algorithm finds the back-channel dynamically during runtime, in contrast to other schemes, which employ a pre-arranged back channel between a pair of adjacent nodes.

For the network consisting of n nodes, the algorithm aims to generate a routing table containing an entry for each of the other $n-1$ nodes in the network. The nodes and links
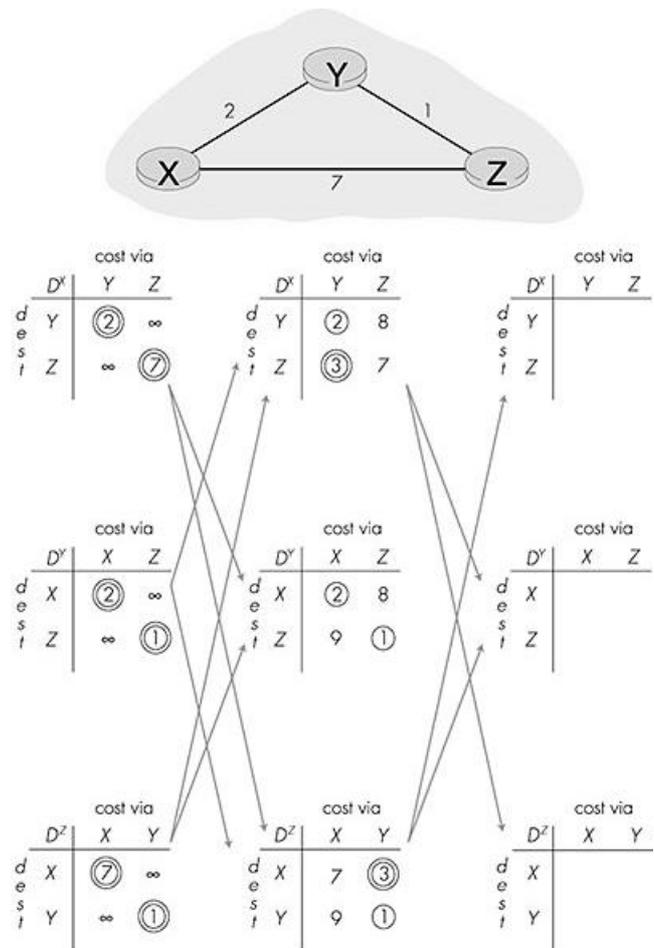

Figure 1: Distance Vector algorithm

may go down occasionally, the topology may change and link costs may vary from time to time. The major difference compared to a conventional DV routing protocol is that instead of a single routing table, each node maintains two tables, a "$FROM$" table and a "TO" table, abbreviated as FT and TT respectively. This change is introduced to eliminate the assumption of a conventional DV routing protocol that if the distant node can reach the local node via a certain link, the local node can reach the distant node via the same link. Only a TO table is used in routing.

The format of a FT entry is $\{ND, DT, NX, TTL\}$. The entry represents a path from some node, ND, to the local node, DT is the distance (the link cost) from ND to the local node, NX is the next hop node in the path from ND, and TTL is time-to-live associated with the entry. The format of a TT entry is similar, except that ND denotes the destination node, and NX is the next node following the local node. The object-oriented "dot" notation is used to represent a node's tables, table entries and fields within the entry, for example, $P.TT$ is P's TO table. Each node periodically sends its FT in a "$FROM$" packet to every TO-neighbor. Triggered by a certain event, the node may send its TT in a "TO" packet to a certain $FROM$-neighbor.

$FROM$ tables are constructed and maintained as follows:

- Initially, all FTs are empty.

- Each node periodically sends to all its TO-neighbors a $FROM$ packet containing its FT.

- When a local node Q receives a $FROM$ packet F from $FROM$-neighbor P, containing its FT, for each entry f in $F.FT$, it generates an entry $e = \{f.ND, f.DT + d(P,Q), f.NX, TTL\}$, where $d(P,Q)$ is the link cost between the two nodes. If the entry e' exists in $Q.FT$, it is replaced by e only when $(e.NX = e'.NX)$ or $(e.DT \leq e'.DT)$. If the entry does not exist and $(e.ND \neq Q)$ it is added to $Q.FT$. Finally, Q generates the entry $e = \{P, d(P,Q), Q, TTL\}$ and performs the same procedure as for other entries.

- When a timer of a $FROM$ entry expires, it is deleted.

Therefore the $FROM$ algorithm ignores entries representing alternative paths, whose DT is longer than the DT of existing matching entries, and the entry that represents a self-cycle (from Q to Q). It was shown by examples, that the $FROM$ algorithm performs well in the dynamic networks, and it can tolerate node and link failures and changes of link costs.

The TO tables are the *routing tables*, which keep the information about the shortest paths to the destination nodes. The TO tables are constructed and maintained as follows:

- Initially, all TO tables are empty.

- When a node Q receives a $FROM$ packet F from its $FROM$-neighbor P, if $F.FT$ contains an entry with $ND = Q$, Q traces the path from Q to P in $F.FT$: $(Q = N_0, N_1, N_2, ..., N_m = P)$, $m \geq 1$. $e_1, ..., e_m$ denotes the corresponding entries in $F.FT$, where $e_i = \{-, -, N_i, -\}$. For $i = 1, ..., m$, Q generates the entry $e = \{N_i, d_i, N_1, T\}$, where $d_i = d_{i-1} + [e_i.DT - e_{i+1}.DT]$ and $d_0 = d_m + 1 = 0$. If the entry e' exists in $Q.TT$, it is replaced by e only when $(e.NX = e'.NX)$ or $(e.DT \leq e'.DT)$. If the entry does not exist and $(e.ND \neq Q)$ it is added to $Q.TT$. Q sends a TO packet containing its TT to P using algorithm $SOURCE\_ROUTE$. According to this algorithm, if a $F.FT$ received from the neighbor P contains an entry with $ND = Q$, it contains a (shortest) path from Q to P. To perform source routing from Q to P, Q encodes this path in the TO packet.

- When a node P receives a TO packet from Q, for each entry t in $T.TT$, it generates an entry $e = \{t.ND, t.DT + d(P,Q), Q, TTL\}$ and may modify or include an entry to $P.TT$ following the aforementioned procedure. Finally, P generates the entry $e = \{P, d(P,Q), Q, TTL\}$ and applies the same procedure as for other entries.

- When a timer for a TO entry expires, it is deleted.

For further improvements the authors assume, that instead of regularly sending the entire $FROM$ or TO tables, they could be sent when the network topology or a link cost changes. The nodes would keep exchanging smaller control messages on a regular base. Another option could be to send only $FROM$ table and generate the TO table from it. For this purpose, a $FROM$ table would include some additional records. Both of the aforementioned improvements seek to lower the bandwidth utilization.

4.3.2 Link State (LS) routing protocols for the networks with UDLs

In **Link State** routing, each node maintains a table with the information about each link that exists in the network. Routes are computed locally as opposed to DV, therefore the changes are adapted very quickly. The decision is based on a complete knowledge of the topology obtained by flooding. Since all the nodes have a common point of view of the network from each other, they are able to forward a packet on the same best path as all other routers would do. If the result is not the same, routing loops can occur. Therefore the goal is to keep link state tables consistent, i.e. at a particular time all the nodes in the network must have the same table. However, it is not realistic due to topology changes. While a topology change is not propagated everywhere in the network, the nodes might have different link state tables, and the routing loops can occur.

In [2] a link-state routing protocol for the networks with UDLs is proposed. It adapts the *link-vector algorithm, LVA,* based on which each router maintains its own routing tree and reports to its neighbors link state updates for those links, which it uses to reach destinations, as well as for those, which it stops using.

In the analysis, a network is modelled by a directed graph $G = (V, A)$, where V is a set of nodes (routers) with a unique ID number, and $A \subseteq V \times V$ is a set of directed links. A BDL between two nodes is represented by two UDLs. Link $(u, v) \in A$ only if v can receive information from u. Here u is called the *head or upstream node* of the link and v is called the *tail or downstream node. A cycle* in G is a directed path with distinct nodes except for the starting and ending nodes. *The inclusive cycle* for a link (u, v) is the shortest path from v to u that can be defined for the link. To find the inclusive cycles for links, *cycle discovery paths* for each link are maintained and propagated in the network. *A cycle discovery path* is the shortest path from the tail of the link to the current node, denoted by $dp_i(u, v)$ for a link (u, v) at node i. Links on discovery paths at node i compose *discovery graph DG_i*. A link belongs to the discovery path, if the following condition is satisfied:

$$|dp_i(u, v)| < l_c s(u, v) \tag{1}$$

where $l_c s(u, v)$ denotes the total cost of the links for the inclusive cycle. Initially, the inclusive cycle is set to infinity and the link state propagates throughout the network. When its inclusive cycle is found at the head of the link, the link state propagates only within a radius equal to inclusive cycle from v.

The proposed UDLR algorithm consists of three parts:

- ***Neighbor Protocol (NBR)***, which provides mechanisms for a node to detect upstream neighbors, update cycle sizes of downstream links, and propagate states of the links, which satisfy Eqn. 1. NBR maintains D_i, U_i and DG_i. U_i contains statistics for detecting and maintaining incoming links. D_i monitors outgoing (down-

stream) links and recommends them for routing while they have inclusive cycles. DG_i maintains discovery graph for finding the inclusive cycles.

- **Network Routing Control Algorithm (NET)**, which calculates the *Shortest Path Tree (SPT)* based on Dijkstra's algorithm and sends changes in SPT to upstream neighbors.

- **Retransmission Protocol (RET)**, which keeps a list of packets for retransmission upon timeout, until it receives acknowledgments from their destinations or destinations become non-neighbors.

A node, supporting NBR, periodically broadcasts a HELLO packet to its neighbors. When a node i detects the HELLO packet from a neighbor, a link (u, i) is added to U_i. The cost of the active link is set to 1, while the cost of unreachable link is set to infinity.

In the protocol, both the head and tail of the link can originate link state updates for the link. However, the cycle size of the link is decided by the head of the link. The head node uses the link for routing only if it has a finite cycle size, associated with it. Therefore, when the head node decides the cycle size, it can respond appropriately in cases when the inclusive cycle of the link is broken and there is no any information about the state of the link.

In NBR, a neighbor node can be in several states. In the upstream node table U_i of node i, a neighbor node can have WAITING, NEW, CONNECTED and INVALID states. For example, if a node u in the table U_i is in the WAITING state, the link (u, i) has been detected by i but is still unnoticed by u. Updates in SPT_i are sent to an upstream node reliably only if it is in CONNECTED state. In the downstream node table D_i a neighbor node of i can be in INVALID, NEW, CONNECTED and DISCONNECTING states. When a downstream node becomes CONNECTED, the downstream link in CONNECTED state is given to NET for routing.

DG_i is propagated to find inclusive cycles of links. Dijkstra's algorithm is run on DG_i to find the shortest paths to the tails of links and to update the information about the links. Any changes in DG_i are broadcasted to downstream nodes and all CONNECTED downstream nodes are required to acknowledge the update, otherwise the update packet is retransmitted by RET.

The NET algorithm computes the Shortest Path Tree (SPT) of the network using Dijkstra's algorithm on the *topology graph*. Changes in SPT are updated reliably at its CONNECTED upstream neighbors. In the update packet, each change to SPT is identified by the operational code (OpCode). NET maintains a topological graph (TG) and a routing table (R) at each node. TG_i of a node i contains the $SPTs$ of its downstream nodes and information in DG_i. DG_i is used by node i to send its complete SPT_i to an upstream node in NEW state. $SPT_k, k \in D_i$ is used for routing to nodes unreachable by DG_i. The routing path to a destination must use a downstream neighbor as the next-hop only if the downstream neighbor reports the remaining path to the same destination. When SPT_i changes, a update packet, containing the routing changes with operational codes is sent to all upstream neighbors of i which are in CONNECTED

state. The routing table R_i at node i is a vector of entries containing information derived from SPT_i: *destination, distance and next-hop address. A link state in TG_i is updated if the link cost or the cycle size changes and if the link is deleted.*

The RET algorithm provides a reliable transmission of NET and NBR packets to CONNECTED upstream nodes and to CONNECTED downstream nodes, respectively.

According to the authors, the proposed UDL routing algorithm performs well and requires less overhead, however it needs further improvements. One of the noted drawbacks is, that the algorithm in the searching process of inclusive cycle consumes a lot of network resources by initializing the cycle of a link to infinity and flooding the link state throughout the network. Therefore, the distance of the link-state propagation in search of the inclusive cycle could be limited.

4.3.3 A Circuit-based approach

In [5] the authors propose a UDL routing protocol based on the circuit detection. It is assumed that finding the best path to get somewhere is equivalent to finding the best circuit, which is an aggregation of the best path from the source to the destination and the best path from the destination to the source. The nodes on a circuit are potential destinations and a communication in both directions can be established with all the nodes located on the circuit after the path going back to the departure place is found. The protocol does not require the full knowledge of topology, and the routing of data packets can start as soon as the circuit is found. The scope of circuits can be limited in order to adapt to the size of the network. At each node, the protocol maintains *a table with selected circuits*. The protocol does not necessary compute circuits for all the destinations or all possible circuits for a particular destination.

- The protocol is based on "one-way" **Link Advertisement Messages (LAMs)** sent by routers to announce new links to all their successors. Each node maintains *"local topology graph"* created based on the information received in LAMs. Upon the reception of a LAM with new information, a local topology graph is updated and a LAM with the new information is forwarded to all outgoing links. *The LAMs are sent by every node at pseudo regular intervals only if the local topology graph has been updated since the last LAM.*

- All nodes send periodic **Keep Alive Messages (KAMs)** to their successors. If no KAM is received during a given time period, the successors are aware of the failure of their predecessors or the link from them.

- *A decission process* running at each node computes the *"possible circuits"* based on the local topology graph. A part of the circuits are selected and stored in a *"Circuit Table"* and labelled as *"selected circuits"*. It is not necessary to use all the circuits for different reasons, for example, in the case when they lead to the same destination. However, keeping several circuits for a particular destination sometimes is advisable, as in the case of failure, one circuit can be quickly substituted

by the other. Before being used for routing, circuits selected by the decision process have to be validated by the nodes which appear on them. When the circuit is selected, a node with the smallest network address on it (called *"Circuit Origin"*) send **a Circuit Validation Message (CVM)**. The CVM contains *the network addresses for the circuit nodes, an iteration number and a time-out value based on the circuit length*. In the Circuit Origin a retransmission timer is set to the same value as time-out. Each node forwards the CVM to its successor on the circuit and sets a validation timer to the initial time-out value. If the CVM comes back to the Circuit Origin before the expiration of the retransmission timer, the circuit becomes "enabled." Otherwise the CVM is retransmitted with the iteration number increased by one. If the circuit is not validated, it is disabled and possibly deleted. *The routing algorithm* is based on the information from the enabled circuits. The shortest circuit containing a given information includes both the shortest path to the destination and the shortest path from it.

- KAMs are immediately sent on *the new links*. The new nodes are noticed by their successors, which informs the network about a new link by sending LAMs.

- In the case of failure, the immediate successors have to inform all nodes which appear on circuits involved by the failure using **Link Update Message (LUM)**. The circuits are disabled until the failure recovery and a recovery timer is set. When a link recovers from the failure, the circuits are enabled again, otherwise, if the recovery timer expires, the corresponding circuits are deleted from the table. If the link recovers after the expiry of the timer, it is processed the same way as the addition of a new node.

According to the authors, there are many open issues related to the application of the circuit-based protocol. The protocol does not take into account the point to multi-point connections. The protocol assumes the all-UDLs network. It may not perform well in the networks where there are many bi-directional links, therefore an additional mechanism to manage these links will be required. The decision process is an important research topic, since the reasons why the circuits may not be selected are not clarified.

5   Conclusions

From the overview of the present routing solutions for networks with uni-directional links, it can be implied that neither RPM-based protocols nor the new protocols have achieved an appropriate level of maturity. Moreover, since none of the new protocols is being standardized, the implementation of arbitrary protocols would result in low interoperability. Therefore tunnelling is the most attractive option for the present time. However, further development of new routing protocols is also a necessity in order to provide a long-term solution to the routing problem.

References

[1] F.C.M. Lau, G. Chen, H. Huang and L. Xie. A Distance-Vector Routing Protocol for Networks with Unidirectional Links. In *Computer Communications*, IT-23(4):418–424, February 2000.

[2] L. Bao and J.J. Garcia-Luna-Aceves. Link-State Routing in Networks with Unidirectional Links. In *Proc. 1999 8th International Conference on Computer Communications and Networks*, Boston, MA, USA, October 1999.

[3] L. Bao and J.J. Garcia-Luna-Aceves. Unidirectional Link-State Routing with Propagation Control. In *Proc. 2000 IEEE Mobile Multimedia Communications MoMuC*, Tokyo, Japan, November 2000.

[4] W. Dabbous, E. Duros and T. Ernst. Dynamic Routing in Networks with Unidirectional Links. In *Proc. The 2nd International Workshop on Satellite-Based Information Services*, Budapest, Hungary, October 1997.

[5] T. Ernst and W. Dabbous. A Circuit-Based Approach for Routing in Unidirectional Links Networks. INRIA Research Report 3292, November 1997.

[6] E. Duros, W. Dabbous, H. Izumiyama, N. Fujii and Y. Zhang. A Link-Layer Tunneling Mechanism for Unidirectional Links. RFC 3077, IETF Network Working Group, March 2001.

[7] Y. Hashimoto and B. Sarikaya. Design of IP-based Routing in a LEO Satellite Network. In *Proc. 1998 3rd International Workshop on Satellite-Based Information Services, Mobicom'98*, October 1998.

[8] T. Ernst. Dynamic Routing in Networks with Unidirectional Links. Master Thesis, University of Nice-Sophia-Antipolis, France, June 1997.

[9] D. Wood Satellites, science and success, the DVB story. EBU technical review, 1995.

[10] E. Duros and C. Huitema. Handling of Unidirectional Links with OSPF. Internet draft <draft-ietf-ospf-unidirectional-link-00.txt>, INRIA, Sophia-Antipolis, March 1996.

[11] E. Duros and C. Huitema. Handling of Unidirectional Links with RIP. Internet draft <draft-ietf-rip-unidirectional-link-00.txt>, INRIA, Sophia-Antipolis, March 1996.

[12] http://portal.etsi.org.