Routing algorithms in the Internet

Paleshnikov, Nikolay
RWTH Aachen
nikolay.paleshnikov@rwth-aachen.de

Witt, Markus
RWTH Aachen
markus.witt@rwth-aachen.de

ABSTRACT

The complex structure of the Internet as a network of networks implicates that there is a vast number of possible connections between any pair of hosts. Since communication plays a central role in every network, its performance depends highly on the choice of paths interconnecting the hosts. Routing has to solve the demanding task of finding an optimal path between each source-destination pair so as to bring high network efficiency. This can only be done by means of well-thought routing algorithms suitable for each specific network environment. The subject of this paper is to place routing in the context of contemporary Internet and to outline some of the most widely used approaches applied in state-of-art routing algorithms. Their hierarchy is studied in detail concerning `OSPF` and `BGP`. Mobility and an unstable network topology cause essential problems with traditional algorithms, whose potential solutions are also discussed.

Keywords

Routing algorithms, DV, LS, OSPF, BGP, MANET

1. INTRODUCTION

One of the most challenging tasks in a network is how to provide reliable and fast interconnectivity between its multiple hosts. Most of the complexity underlying that problem is implemented in the network layer of the protocol stack. Its instances include not only the hosts, but also the intersection points (*routers*) of the network. As the Internet only provides a connectionless service at its network layer, specific paths must be chosen for each information entity (*datagram*) to be transported. Routing attempts on finding the most efficient path between source and destination of the data flow using unique identifiers such as their IP-addresses.

In a typical meshed network, where each host can be reached through multiple links, static configuration of routes proves inefficient. It also requires manual reconfiguration in case of a hardware failure. Therefore, *dynamic routing* is mostly employed. Since hosts are usually connected to a single router, the tough decisions fall on the routers in between them. These decisions must be done in a deterministic way following the steps specified by a *routing algorithm*. It should be able to respond fast to changes in topology and periodically disseminate crucial information about the network state (via *broadcast*). The routes must be stored in a *routing table* indexed by the destination host, enabling fast look-up of the next-hop router upon a datagram arrival.

With a network growing as fast as the Internet, *scalability* becomes a major factor for the success of employing a specific solution to the problems stated above. Therefore, a *hierarchy* of routing algorithms is urgent. From an architectural point of view, the fundament is based on multiple *interior gateway protocols* (`IGP`s), responsible for the datagram transportation within a single autonomous system (`AS`), on top of which an unified *exterior gateway protocol* (`EGP`) is executed so as to interconnect their functionality and form a coherent network structure. The `AS` can thus be regarded as the second large functional entity after the Internet itself. It usually represents the network maintained by a single *Internet service provider* (`ISP`). Following this hierarchical approach, `IGP`s are discussed in chapter 2, starting with an overview and an ensuing comparision of two fundamental classes of routing algorithms: distance-vector and link-state. *Open Shortest Path First* (`OSPF`) as the most common representative of `IGP`s is used to explain concepts in practice. Subsequently, chapter 3 presents a more detailed study of the delimitation of `AS`es and their integration to a connected network by use of a single `EGP` and *Border Gateway Protocol* (`BGP`) in particular. Severe problems arising from mobility and vulnerability of the network structure and their possible solutions are the topic of the last chapter of this paper.

2. INTERIOR GATEWAY PROTOCOLS

It is useful to outline some of the basic principles employed by all of the interior routing algorithms in the first place. Discussions on this issue can be found in either [1] or [2]. Firstly, the whole network is modelled as a *weighted directed graph*. The weights of the edges, also referred to as link costs, might represent the delay or bandwidth of a single link between two routers and are taken into account for the determinition of a good path for every source-destination pair. Since the weights of both edges connecting two routers in opposite directions need not be identical, the most efficient paths between a pair of nodes are rarely symmetrical. Secondly, the *optimality principle* states that if the most efficient path from router `A` to router `C` goes through router `B`, then the optimal route from `B` to `C` also falls along the same path. This principle is extensively used when we do not have a complete view of the whole network, such being the case with a *decentralized* routing algorithm.

2.1 Distance-Vector Routing

The *distance-vector algorithms*, as reviewed in [1], are a classical example of decentralized routing algorithms. There, each node maintains information about the cost of the op-

timal routes to all other nodes in the network in a data structure called a *distance vector* (DV), which is an abridged routing table consisting only of destination-cost pairs. Upon initialisation, a DV consists exclusively of entries describing the neighbours of the node. For the need of computing best routes to all nodes of the network, DVs are propagated among adjascent routers, what makes every DV algorithm a distributed one. Using this technique, a node determines peu à peu its best routes to nodes located at increasing distances. At most, it takes as many steps to build a complete DV as the diameter of the network. After receiving a new DV from some of its neighbours, each node must update both its own DV and routing table according to the *Bellmann-Ford equation*:

$$d_x(y) = min_v\{c(x,v) + d_v(y)\}$$

in which $d_x(y)$ denotes the least-cost path from x to y, v an arbitrary neighbour of x, c(x,v) the weight of the link between x and v and $d_v(y)$ the least-cost path from v to y, respectively. This equation is evidently based on the optimality principle indicated above and computes $d_x(y)$ with a time complexity of $\mathcal{O}(|v|)$, where |v| represents the number of neighbours of x.

While DV algorithms have some apparent advantages such as the space efficiency of the routing tables maintained and the little amount of information processed at the nodes of the network, a deeper analysis of their behaviour reveals a few insurmountable drawbacks, as pointed out in [3]. The so called *bouncing effect* occurs in case of a sudden link failure. Consider the scenario described in Figure 1. In the beginning, B computes its optimal route to C to cost 1 and forwards this information via its DV to A. At this point, A works out its routing table, shown in Figure 2. If the link between B and C breaks, the cost of the route from B to C will be set to infinity. Let us suppose that the DV of A, based on the routing table in Figure 2 and thus advertising a shortest path to C at the cost of 2, reaches B before the newly computed DV of B is forwarded to A. In this case, the DV of B is updated with a cost of 3 from B to C and then distributed to its neighbours. On the basis of this DV, A calculates the cost of its optimal route to C to be 4, and so on. At that point, an undetectable loop is established between A and B. Since the view of the network structure at each node is limited to its direct neighbours, it does not know through which nodes its routes go along their way to the destination. A convergence to a stable state will be achieved after the exchange of seven more DVs, as A will finally set its optimal route to C through the direct link costing 10. Meanwhile, datagrams will loop until their time-to-live expires, possibly causing congestion on the links comprising the loop. Because DV interchange relies on those links, the recovery from a link failure may be further slowed down. To secure reliability against topological changes such as a link or a router failure, the information exchange must therefore take place at well-defined time intervals in an iterative and asynchronous manner.

A special case of the bouncing effect, also known as the *count-to-infinity problem*, is the result of a break-down of a bridge in the graph, i.e. a link representing the sole connection of a node or a group of nodes to the rest of the network. Supposed that the link marked with * in Figure 3 was torn down, the bouncing effect between nodes A and B in their

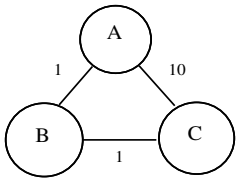

Figure 1: Network model illustrating the bouncing effect

Destination	Next Hop	Cost
B	B	1
C	B	2

Figure 2: The routing table of A

attempt to compute best-cost path to C, as well as between C, D and E in their struggle to resolve an optimal route to A, would endure infinitely. The definition of a number slighly bigger than the diameter of the network as infinity solves that issue. Beyond that number, no further computations of the cost to a specific node are undertaken.

To prevent the cause of the bouncing effect between two nodes, a technique called *split horizon* may be applied (see [3]). It states that B should not announce A a route reaching C if it has determined A to be the next-hop router to C in its routing table. Despite its simplicity, split horizon is not ground-breaking for its inablility to break loops of higher order such as the one between nodes C, D and E in Figure 2.

A distance-vector algorithm also suffers from a certain lack of robustness. Provided that a node broadcasted an incorrect link cost to its neighbours, they would not have any database to compare this cost with and would have to blindly accept it. The numerous disadvantages of DV routing have led to the development of a fundamentally different approach to intra-AS routing, called *link-state routing*.

2.2 Link-State Routing

The link-state (LS) routing algorithms, as expounded in [1], need complete information about all of the link costs in the network and are therefore often described as *global* routing algorithms. The computations they perform are based on the widely known *Dijkstra's algorithm*, solving the single-source shortest path problem for a given Graph
G = (V, E). It operates by means of a set of nodes N with an already assigned shortest path, which initially consists solely

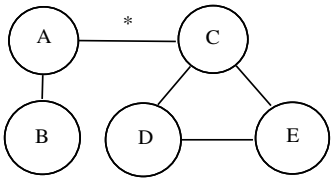

Figure 3: Scenario in which the count-to-infinity problem arises

of the source node s. During its execution it also maintains a table with the best known distances D(s,x) from s to any other node x in the network, initialized as the edge weights c(s,x) if a direct connection is present and as infinity otherwise. In each step of the following loop, it chooses the node $t \in V \setminus N$ with the minimum distance from the source D(s,t), updates the distances from the source to each of t's neighbours regarding the equation

$$D(s,v) = min\{D(s,v), D(s,t) + c(t,v)\}$$

based on the Bellmann-Ford equation discussed above, and eventually adds t to N. As both the loop and the algorithm terminate in |V| steps, in each of which at most |V|-|N| vertices are being concerned, and the cost of each directed edge is taken into account at most once, it has a running time of $\mathcal{O}(|V|^2 + |E|)$.

LS routing points out its superiority to DV in several different ways, as discussed in [3]. After a topological change, each node broadcasts the IP address of each router interface it is directly connected to and the respective link cost to reach it, both known as the *state* of the link. This way, a node gathers the complete state information of the network and is thus able to determine its actual shortest paths in a *loopless* manner. As a further concequence, the *convergence* of the algorithm at each of the nodes requires only one computation of every optimal route. Possible errors in computation are not forwarded since the LS algorithm is *independently* executed at each host. Moreover, with LS routing it is possible to maintain *multiple* least-cost paths to a particular destination so as to split the traffic over several (partly) different routes. The efficiency of using the added throughput of multiple routes is evident. Besides, it enables appropriate reaction to *congestion* at the network layer, as each datagram may be transported on alternate routes avoiding congested links.

The LS routing certainly pays the cost for its fast convergence and resilience against the broadcast of an incorrect link cost. The distribution of complete topology information to every single node of the network adds a significant *communication overhead*. The separate execution of Dijkstra's algorithm on every notification of a link cost change at every router requires a lot of *computation*. This leads to the conclusion that mere LS routing is not particularly appropriate for networks growing fast in size.

Hence the common approach is to divide a single AS into multiple *areas*, within each of which LS routing is applied, and regularly exchange aggregated routing information between the areas with a strong resemblance to DV routing. The technical details of this routine are described in section 2.3. in reference to the most widely employed intra-AS routing algorithm in the Internet, namely OSPF.

2.3 Open Shortest Paths First

The currently used OSPF version 2, as given in [4], is used within an AS to distribute the state of all links between routers to every router inside the AS. An AS may be divided into smaller areas connected through a special backbone area with the advantage of reduced traffic at area borders. The state of a link is distributed via a *link-state advertisement* (LSA) to all other routers which enables them to compile a

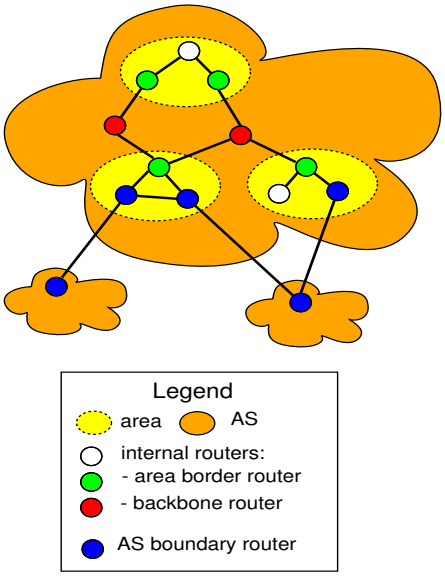

Figure 4: Router types within an AS

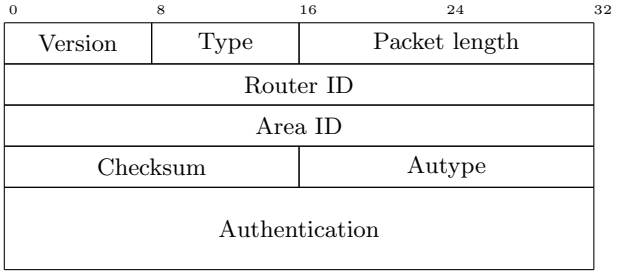

Figure 5: The header of an OSPF packet (RFC1247)

weighted and directed graph of the complete AS. This graph is later used to build a tree of shortest paths, which then yields the routing table.

Each router has a unique ID consisting of a 32-bit number and at least one interface connecting it to another router. Interfaces may be connected directly to another router or through a network connecting multiple routers. There are two distinct types of routers if areas are not used: internal routers and AS boundary routers (ASBR). The latter exchange routing information with systems outside of the AS through an EGP, as seen in chapter 3, and feed the routes into the AS. The route to an ASBR must be known by every other router. Two more types of routers exist, if areas are used in the AS: area border routers (ABR) and backbone routers (BR), displayed in figure 5. ABRs connect one or more areas to the backbone area while BRs have interfaces to the backbone area. Every ABR is a BR but not every BR is an ABR - it is then only connected to other BR. An ABR also runs the algorithm described below for each area it is connected to.

OSPF packets consist of a shared 24-byte header as seen in figure 5. The version number is always 2, the type is one of the OSPF packet types mentioned below. The checksum is calculated equally to other IP packets except that the 64-bit

authentication field is left out. The AuType field identifies the used authentication scheme and is either zero for no authentication or one for a shared key. Authentication should be used to protect the whole network against malicious routing updates.

A router periodically sends hello packets, which are OSPF packets of type one, on all interfaces to the broadcast address AllSPFRouters. They include among other data the HelloInterval, which is the number of seconds between hello packets of the router, its unique ID, and a list of neighbors from which hello packets have been received in the last RouterDeadInterval seconds. The HelloInterval must be equal in the whole AS and is used in conjunction with the RouterDeadInterval as a timeout to declare a router as down. If the router also receives hello packets from neighbors which include its ID, they are able to communicate bidirectionally. If the interface is connected to a multicast enabled network and there are no designated router (DR) and backup designated router (BDR), a DR and BDR will be elected to reduce the overhead by reducing the adjacences in a network. A router connected to a multicast enabled network may only exchange LSA with a DR which reduces the required traffic. After a bidirectional connection is established, both neighbors send database description packets (DDP) (OSPF type two) packets, containing the content of their current link-state database as link-state advertisement headers. The DDP does not contain the whole LSA which again saves traffic. The receiver saves the advertisements which are more recent than the ones in its local database or which are entirely missing. After the exchange the receiver answers with link state acknowledgement (OSPF type five) packets for each advertisement within RxmtInterval seconds or a retransmission occurs. This is required, because OSPF is not based on a reliable transport protocol like TCP and messages may be lost. The end of transmission is marked by a certain bit in the DDP. Afterwards the saved advertisements are requested through (OSPF type three) link state request (LSR) packets. These LSR packets are answered by link state update (LSU) (OSPF type four) packets.

Changes propagate in the AS through LSU packets. If the interface of a router changes its state, the router sends a new link-state advertisement in a LSU packet to all of its adjacent neighbors. The neighbor checks if the received link-state is newer than the one in its local database and updates the database if required, which triggers the repetition of the received link-state advertisement to all of its neighbors. If the received link-state is older than the local one it is ignored. This procedure is called *flooding* and is used to broadcast changes within the whole AS.

Each router sends a link-state advertisement for each interface at least every 30 minutes (specified as LSRefreshTime). There are five different types of advertisements: router link advertisements (type 1), network link advertisements (type 2), summary link advertisements (type 3 and 4) and AS external link advertisements (type 5). Every router sends advertisements of type 1, which describe the state of all interfaces of a router in a specific area, into the specific area. Type 2 advertisements are sent by the DR into an area and specify all routers of the area. Type 3 and type 4 advertisements are sent by ABR into an area and contain either the route to other networks of the AS (type 3) or the route to an ASBR (type 4). Type 5 advertisements are flooded into the whole network by the ASBR and contain a route to an external AS, but they are also used to distribute a default route into areas without an ASBR.

These link-state advertisements are used to build a local link-state database for each area the router is participating in. Entries are added if a router receives LSU packets or if the router is configured to announce a specific network. They are replaced by updates and removed after one hour (MaxAge) if they are not refreshed or updated. The local link-state database is then used to build a tree with the router as the root and the different routes with lowest costs as its paths.

OSPF version 2 was changed since 1991 through sporadic updates. RFC1583 [5] resolves a problem regarding virtual links, which are used to connect two non-continuous parts of the same area, which could cause routing loops. RFC2178 [6] enhances the autentication by adding a cryptographic authentication type while specifying the use with MD5 and enhances the flooding algorithm. If a router receives an LSU with an advertisements which is less recent than the one in its local database, it responds by flooding back the local copy. Furthermore, a detection of maximum transmission unit (MTU) mismatches was implemented. If a router receives a DDP with an MTU larger than the MTU of the interface, it drops the packet and does not form an adcacency preventing further packet loss. RFC2328 enhances the flooding of routing updates by preventing LSA-loops.

OSPF version 3 as specified in [8] changes OSPF to be compatible with IPv6. Whereas OSPFv2 has only a very few changes in the exchanged packets between RFC1247 and RFC2328, RFC5340 has significant changes to accomodate for the IPv6 addresses double in size compared to the previously used IPv4 addresses.

3. EXTERIOR GATEWAY PROTOCOL

In the previous chapter, we have seen how routing works in a single autonomous system. Its owner, usually an Internet service provider, may decide which IGP to employ within its boundaries. On the next hierarchical level, the EGP must provide interconnectivity between the multiple ASs of the network regarding their own routing policies, as described in [2].

3.1 Inter-AS routing

The simplified but accurate model of the Internet hierarchy shown in Figure 6 clarifies the relations between the ASs and the EGP, as well as between the the ASs themselves. The customer ISPs only transfer packets whose source or destination reside in the AS. All the traffic in between is carried out by transit ISPs, operating outside the boundaries of the ASs owned by the customer ISPs and charging them for the service. Peering agreements between ISPs are also common. In that case, two client ISPs advertise their address spaces to each other to enable direct routing between their ASes (and avoid paying the transit ISPs). The EGP should also manage a routing table including a vast amount of destinations, which is being kept in scope by using CIDRized IP addresses,

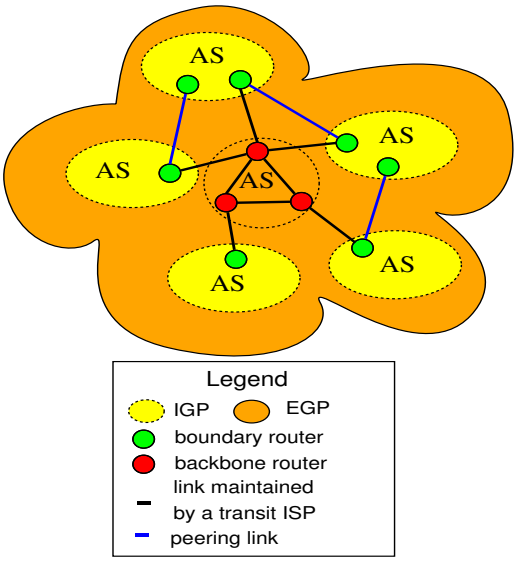

Figure 6: Abstract model of the Internet hierarchy

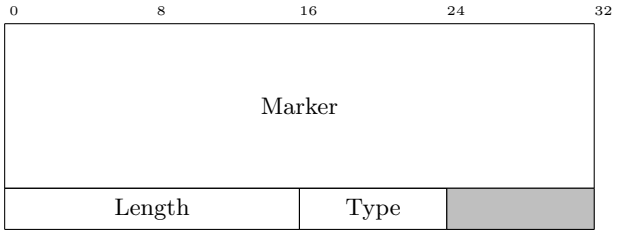

Figure 7: The header of a BGP packet (RFC4271)

explained in more detail in [1]. Following this technique, it is possible to summarize overlapping prefixes in a single entry of the routing table, while applying *longest prefix match* as a rule to make sure the route being chosen is the most specific (and appropriate) one to the destination.

It should be emphasized that a standard EGP must be run at each router of the network so that it has a coherent behaviour. At inter-AS layer, only AS are being advertised and routes between them are computed. A datagram destined for another AS leaves its AS through a boundary router and is carried through the backbone routers of the network to the boundary router of the destination AS. The transport within an AS remains still the responsibility of the IGP.

The enormous size of the Internet can only be managed by a distance-vector EGP. The standard one used in today's Internet, BGP, is analyzed in section 3.2.

3.2 Border Gateway Protocol

The Border Gateway Protocol, as specified in [9], can be used either as an IGP inside an AS or as an EGP between different AS. We will only study the EGP aspect. BGP exchanges network layer reachability information (NLRI) between two peers via TCP and a protocol tracking finite state machine (FSM) at each peer to allow for destination based forwarding. The received NLRI is filtered, saved to a local routing information base (loc-RIB) and finally distributed to other peers. The loc-RIB is used to build the routing table in a later stage. The peers of a system are predefined ASBR of other AS with which the AS peers.

To minimize the communication complexity and overhead, the protocol relies on TCP to transmit BGP datagrams on the IANA assigned port 179. The header seen in figure 7 consists of a 16 bytes marker in which every bit is always set to synchronise the begin of a packet, a two byte length field and a one-byte type field which specifies the type of the BGP message. The resulting BGP packet of header and message is between 19 and 4096 bytes long and there are four different types of messages specified: OPEN, UPDATE, NOTIFICATION and KEEPALIVE. These messages are exchanged between the peers during and after the establishment of their connectionas described in the following paragraph.

Each BGP session consists of eight mandatory attributes including the current state of the finite state machine, a ConnectRetryCounter and -Time, a HoldTime, a KeepaliveTime and an associated timer. Furthermore there are 16 mandatory state machine events as seen in table 1 from which at least four are used for a successfull peering setup. The FSM as seen in figure 8 starts in the Idle state, where no ressources are allocated and incoming connections are refused. On a ManualStart event the system initialises the ConnectRetryCounter and the ConnectRetryTimer. Then the system listens for incoming connections and initiates one to every configured peer system. After this, the system changes into the Connect state. If the ConnectRetryTimer expires, the system stays in this state and tries to reconnect to the peer system. Otherwise, if there is a confirming event, the ConnectRetryTimer stops and the BGP session setup proceeds by sending an OPEN message containing the version number 4, the number of the AS, a BGP identifier and other housekeeping information. The BGP identifier must be unique and must be an IP address which is assigned to the host. The state changes to OpenSend where the host waits for an OPEN message of its peer (Event BGPOpen), which is answered by a KEEPALIVE message which does not have data except for the header. After setting the KeepaliveTimer and HoldTimer the state changes to OpenConfirm. Now the host waits for the KEEPALIVE message of its peer (Event KeepAliveMsg), which changes the state to Established upon reception.

The other events as seen in table 1 lead to the teardown of the current connection attempt by dropping the TCP connection and resetting used timers and ressources before the state changes to Idle again.

The Established state is the only state where UpdateMsg events indicating the reception of an UPDATE message are permitted. An UPDATE message consists of the BGP packet header followed by withdrawn routes, path attributes and NLRI for these path attributes. Withdrawn routes should be deleted from the local RIB (loc-RIB) at once. There are different types of path attributes which are: mandatory and well-known, well-known, optional. Attributes which are mandatory must be included in UPDATE packets, well-known attributes must be understood by an implementation of BGP. There are three mandatory attributes: ORIGIN specifies the

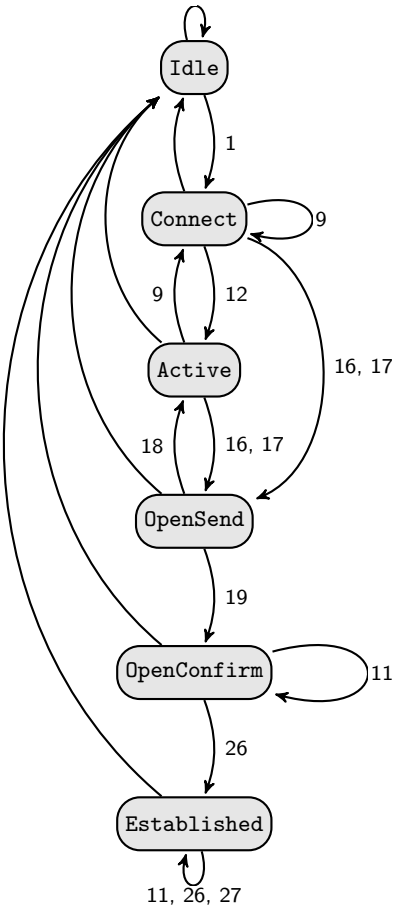

Figure 8: The FSM of a BGP connection
All non-specified events follow the unmarked transition

#	Name of the FSM event
1	ManualStart
2	ManualStop
9	ConnectRetryTimerExpires
10	HoldTimerExpires
11	KeepaliveTimerExpires
16	Tcp_CR_Acked
17	TcpConnectionConfirmed
18	TcpConnectionFails
19	BGPOpen
21	BGPHeaderError
22	BGPOpenMsgErr
24	NotifMsgVerErr
25	MotifMsg
26	KeepAliveMsg
27	UpdateMsg
28	UpdateMsgErr

Table 1: The mandatory events in a BGP FSM

origin of the path information, `AS_PATH` which contains a set of every `AS` till the `ORIGIN` and `NEXT_HOP` which specifies the unicast IP address of the next hop to reach the destination. `NLRI` consist of a network prefix and an associated prefix lenght and originate from the `ORIGIN` which announces the Network.

Two additional buffers for each peer - adjacent routing information base input and output (`Adj-RIB-In` and `Adj-RIB-Out`) - contain information about incoming and outgoing `NRLI`. When entries in `Adj-RIB-In` are added or renewed, the decision process consists of three phases. Phase one calculates a degree of preference and if the `NRLI` is used at all. The exact calculation is based on local configured costs for each peer. Phase two is started by the end of phase one and determines which route gets into the `loc-RIB`. The route is discarded if the `NEXT_HOP` is not reachable or if the `AS_PATH` contains loops. If there is more than one route for a given destination, phase two chooses the one with the highest degree of preference. Phase three is triggered by the end of phase two, if the `loc-RIB` changes or if a new `BGP` connection enters the `Established`-state. It adds routes to the `Adj-RIB-Out` and applies filters to this process so that new routes may not be announced to peers.

The advantage of `BGP` over other `DV` based protocols consists in the additional information of the path of the `NRLI`. `AS` loops can be recognised and associated paths can be ignored.

The generation of the final routing table is not the subject of `BGP`. Besides the `loc-RIB`, it may contain other routing information obtained by `IGP` or based on static configuration.

4. MOBILE HOSTS AND AD-HOC NETWORKS

The well-defined hierarchy of an `AS` running `OSPF` and being interconnected by `BGP` with the rest of the network depends highly on the assumption of *reliable* links and relatively *constant* network topology. An easy attachment can be cojoined to that network model to enable routing for *mobile hosts*, as elucidated in [2]. At the home network, which does not change over time, runs a *home agent* that tracks the current location of the host and assignes him a temporary local network address, also known as a *care of address*. The first packet arriving at the home network from a new source is tunneled to the care of address of the mobile host, which then sends a reply packet back to the source. At that point, a direct tunnel from source to mobile host is set up for the ongoing data transfer. Each of the next outbound packets is being sent directly to the care of address. The whole procedure must be run again to reset the care of address at next change of location or an unexpected connectivity loss.

The situation in a typical *mobile ad-hoc network* (`MANET`) is precicely different, for the underlying network structure changes constantly and links break down regularly. Nodes join and leave the network at any point without a warning, being highly *flexible* and *mobile*. There are *no* routers present, so that the hosts must also do routing and forwarding of packets themselves. Since one of the goals of an ad-hoc network is to be empoyable under all possible conditions, it may not rely on *any* kind of infrastructure. However, there

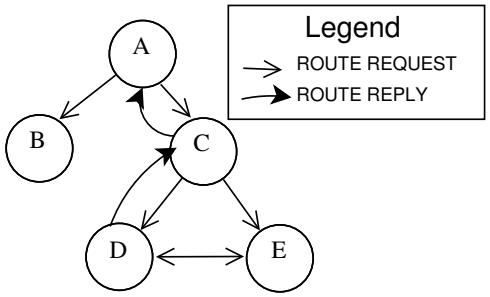

Figure 9: Route discovery in AODV

are still a few assumptions to be made, as pointed out in [10], including the equipment of each node with a portable wireless communication device and a fixed IP address. Clearly, a new and somewhat different approach should be taken to routing rather the ones discussed in chapter 2. Usually, it is more efficient to compute routes *on-demand* in a *reactive* manner instead of a *proactive* one as a response to each change in topology. In this case, elaborate routing tables necessarily become obsolete.

4.1 Ad-hoc On-Demand Distance Vector Protocol

Route discovery and maintenance in a mobile ad-hoc network can be performed using the *Ad-hoc On-Demand Distance Vector Protocol* (AODV), as outlined in [10] and [11]. Since a node does not have any information about the network topology when first having to send a datagram to a new destination, it tries to locate it by using an *expanding ring search* procedure, i.e. it sequentially broadcasts ROUTE REQUEST packets with increasing time-to-live values, as depicted in Figure 9. As one of them eventually reaches the searched node or a node that has a known path to it, a ROUTE REPLY packet is sent along the reverse path taken by the request. In our example, node A tries to find node D. One of the ROUTE REQEST packets with time-to-live value of 2 goes through C and finally reaches its destination D. Then, a ROUTE REPLY packet is sent to A through C. Furter ROUTE REQUEST packets arriving at D are simply ignored. Upon arrival of the reply, each intermediate node (and the source node itself) enters a path to the destination into its routing table. If, upon transmission, a node along the path cannot reach the next-hop router any more, it purges the route from its routing table and instructs his active neighbours by means of a ROUTE ERROR packet to also purge it from theirs in a recursive manner. The affected source needs to repeat the whole route discovery procedure if it still has some datagrams pending to be send. This strategy ensures that each node has only active, directly usable route entries in its routing table.

Probably the most important improvement of AODV over other distance-vector routing algorithms is its proved *loop-free* property. It is achieved by using a unique set of sequence numbers for each particular destination in the network. Every routing table entry and routing update message get assigned a sequence number regarding their destination. Upon arrival of an update message, the corresponding table entry for the destination is updated only if it has a smaller sequence number than the message. Even though this technique surely adds some *processing overhead*, it completely avoids the bouncing effect and the count-to-infinity problem described in section 2.1.

4.2 Optimized Link State Routing protocol

Alternatives to distance-vector routing in MANETs are to be found in [12]. The most notable one is the *Optimized Link State Routing* protocol (OLSR), which operates quite similarly to OSPF with one crucial improvement. It is the *multipoint relay mechanism* (MPR) that tackles the overhead generated by the exchange of complete state information among all nodes in classical link-state routing. The key idea behind MPR is to let each node choose a *minimal subset* of its direct neighbours that covers all nodes at most two hops away from it. The inclusion of a node in one's MPR set is a symmetric relation. Although each node broadcasts to all of its neighbours information about the hosts it is connected to and the costs of the corresponding links to them, only the nodes in one's MPR set forward this routing information in the network, which improves efficiency especially in large and dense networks. The route computations are then made locally by use of the Dijkstra's algorithm. One apparent disadvantage of OLSR is that it may determine routes never to be used, hence adding useless *communication* and *computation overhead*. On the contrary, it minimizes the *route discovery delay* since each node has up-to-date routing table entries for any destination on the network.

4.3 Hybrid solutions

Hybrid approaches are also employed in the attempt to take advantage of both proactive and reactive routing, see [10]. For that purpose, the whole network is divided into non-overlapping physical zones. Within each of them, a proactive routing algorithm such as OLSR is run, whereas a reactive one such as AODV takes the responsibility of connecting the individual zones. The resulting hierarchy should be seen as the counterpart of the classical OSPF - BGP routing architecture inapplicable in MANETs.

5. CONCLUSIONS

Routing has the uneasy task of determining optimal routes between any pair of nodes in the network. Whereas link costs are straightforwardly taken into account within an AS, policy issues also play an important role on a global level. The routing algorithms can be classified in relation to either their place in the routing hierarchy (IGPs and EGP) or the routing principle underlying their implementation(distance-vector and link-state). Regarding the strenghts and the deficiencies of the latter, LS routing is most commonly employed in IGPs, whereas the standard EGP is a DV routing protocol.

OSPF and BGP build the core of the routing algorithms employed in the Internet, offering reliable service to its customers. Nevertheless, they are by far not an ultimate solution the problem of routing in a network as complex as the Internet. Alternatives are being still developed to face the ever-emerging issues of mobility and ease of access even under poor networking conditions and volatile network topology. AODV and OLSR are representative examples of routing protocols preeminently designed for the needs of a mobile ad-hoc network.

6. REFERENCES

[1] James F. Kurose, Keith W. Ross *Computer Networking: A top-down approach* 6th edition, 2012.

[2] Andrew S. Tanenbaum, David J. Wetherall *Computer Networks* 5th edition, 2011.

[3] Christian Huitema *Routing in the Internet* 2nd edition, 1999.

[4] Internet Engineering Task Force: *OSPF Version 2* RFC 1247, 1991.

[5] Internet Engineering Task Force: *OSPF Version 2* RFC 1583, 1994.

[6] Internet Engineering Task Force: *OSPF Version 2* RFC 2178, 1997.

[7] Internet Engineering Task Force: *OSPF Version 2* RFC 2328, 1998.

[8] Internet Engineering Task Force: *OSPF for IPv6* RFC 5340, 2008.

[9] Internet Engineering Task Force: *A Border Gateway Protocol 4 (BGP-4)* RFC 4271, 2006.

[10] Zygmunt J. Haas, Jing Deng, Ben Liang, Panagiotis Papadimitratos and S. Sajama *Wireless Ad Hoc Networks* Cornell University, 2002.

[11] Charles E. Perkins, Elizabeth M. Royer *Ad-hoc On-Demand Distance Vector Routing* Proceedings of the 2nd IEEE workshop on mobile computing systems and applications, 1997.

[12] Kenneth Holter *Comparing AODV and OLSR* Technical Report, 2005.