Counting to Infinity

Janne Lindqvist
Helsinki University of Technology
Telecommunications Software and Multimedia Laboratory
janne.lindqvist@iki.fi

Abstract

Routing protocols have to deal with a known problem: the count-to-infinity. Counting to infinity can occur when a link breaks in the network, and the algorithm in the routing protocol tries to calculate new shortest paths. In this paper I survey the past and present solutions to the this problem.

KEYWORDS: count-to-infinity, loop freedom

1 Introduction

Routing protocols can be roughly divided to distance vector and link-state protocols, depending on the underlying algorithm to count the routes and how the messages are propagated in the network.. The main emphasis on this paper is on distance vector protocols, but we will also visit the link-state routing.

Distance vector routing is based on algorithmic solutions to the graph theoretic all-pairs shortest path problem: Given a graph G, we want to find the shortest paths from every node to all other nodes. An interesting detail is that when textbooks today speak of all-pairs shortest path, the original non-distributed algorithm for finding shortest paths (routes in networks of cities) was presented in a journal article named "On a routing problem" [1]. This was before computer networks, as we know them, were even introduced.

The count-to-infinity problem is a result of how routing protocols, especially distance vector, work: "Good news travels quickly, bad news travels slowly" [5]. If a certain kind of link failure occurs in a routed network, the result is that the algorithm and hence the protocol tries to count the shortest paths to infinity. An example of this behaviour is given later.

Despite of count-to-infinity and few other problems, distance vector routing is still interesting, since new protocols for ad hoc networks are developed based on that kind of solution. Link breakages occur often in ad hoc networks, so the routing protocol has to somehow ensure loop freedom. According to Huitema, Routing Information Protocol (RIP), which is a distance vector protocol, was the most used interior gateway protocol in 2000 [3]. An interior gateway protocol handles routing information within an autonomous system. Thus, interior gateway protocols do not participate routing between autonomous systems.

The rest of this paper is organized as follows. First section 2 introduces distance vector routing in fixed networks and gives an example of the count-to-infinity problem. Then section 3 explains past solutions to the problem, mainly derived from the Routing Information Protocol. After that section 4 briefly explains link-state routing. Following that section 5 tells how the problem is solved in ad hoc networks while using distance vector protocols. Next in section 6 I compare the different solutions, and finally in section 7 are the conclusions.

2 Distance vector routing in fixed networks

To describe formally how distance vector routing works, we need some definitions first.

D_j current estimate of the minimum cost from node j to the destination node
C_{ij} link cost from node i to node j
$C_{ii} = 0$
$C_{ik} = \infty$, if nodes i and k are not directly connected

2.1 The distributed Bellman-Ford Algorithm

The distributed Bellman-Ford algorithm is based on the idea that if node C is in the shortest path between A and B, then the path from the node A to C must be a shortest path. And the path from the node to B to node C must also be a shortest path. The following formal presentation is adopted from the textbook Communication Networks by Leon-Garcia and Widjaja. [5]

Now, the actual algorithm:
1. Initialization
$D_i = \infty, \forall i \neq d$
$D_d = 0$

Intuitively, we mark the distances to every other node to infinity and the distance to the node itself (d) to zero.

2. Updating: For each $i \neq d$,
$D_i = \min_j \{C_{ij} + D_j\}, \forall j \neq i$
Repeat step 2 until no more changes occur in the iteration.

Intuitively, the algorithm compares the advertised routing information from the other nodes, and selects the shortest path to be updated to the routing table.

2.2 A distance vector routing protocol

The protocol uses the above algorithm for counting of the shortest paths. This is done distributedly, every node counts all paths to every other node. The shortest path is based on a metric. Sometimes the metric is referred in the routing literature as a count or a distance.

The protocol uses a particular property of the network as the metric. The metric could be, for example 1/capacity, packet delay, or congestion. To understand why "1/capacity" could be used is that the more capacity the link has, the smaller the cost is in the graph. Thus, the link with more capacity has a shorter path in the graph than a link with less capacity. Delay or congestion can be treated straightforwardly, the more congested the network is, more There are other metrics too, for example Routing Information Protocol uses hop count as the metric.

The protocol keeps a routing table, that consists of 3-tuples (N,D,DI). The N is the node from which node D is available and DI is the overall (shortest) distance to the node D. For example (3,2,5) means that node 2 is available through node 3 and the total distance to the node 2 is 5. In other words, the node 3 is the next hop towards node 2.

When the protocol starts, the tuples are marked as $(-1, -1, \infty)$ to indicate that no routes are available.

The protocol works by sending periodic updates of the tuples to neighbor nodes, and after that, recounting the shortest paths using the algorithm defined in the section 2.1.

2.3 The count-to-infinity problem

The following example is based on an example presented in the textbook Communication Networks by Leon-Garcia and Widjaja [5].

Consider a graph with four nodes, where each node is connected to another node in an ascending order: 1-2-3-4. All nodes have routes to all nodes counted with the algorithm. The link cost is "1".

Now, the link between nodes three and four brakes. The algorithm now tries to recompute the route to node 4. Result is that the algorithm tries to count to infinity.

The result can be seen from the algorithm and protocol description. When the link breaks, node 3 counts that it's shortest path to node 4 is through node 2 and sends this data to node 2. When node 2 receives the data, it counts it's shortest path to node 4 and finds out that it is through node 1. Node 2 then sends the data to both node 1 and node 3 which compute the shortest paths. In other words, all the nodes believe that one or all of the neighbors have a shortest path to the node 4 and update their routing tables accordingly.

The counting would stop no later than the end of the world or when the allocated memory for the routes runs out, whichever comes first. In the following section are few solutions to this kind of link breakage. Unfortunately, they do not cover all the possible cases.

3 Past solutions

This section presents and briefly analyses the past solutions to count-to-infinity problem regarding distance vector protocols based on the distributed Bellman-Ford algorithm. Each subsection discusses a different solution

3.1 Defining the maximum count

For example, the Routing Information Protocol defines the maximum count as 16 in RFC 1058 [2]. This means that if all else fails, the counting to infinity stops at the 16th iteration.

RIP uses hop count as a metric, the maximum count also means that it cannot support networks consisting of subnetworks more than 15 hops away.

3.2 Split Horizon

RFC 1058 introduces also another method for Routing Information Protocol: the use of Split Horizon [2]. Split Horizon means that if node A has learned a route to node C through node B, then node A does not send the distance vector of C to node B during a routing update.

3.3 Poisoned Reverse

Poisoned Reverse is an additional technique to be used with Split Horizon, it is also explained in RFC 1058 [2]. Split Horizon is modified so that instead of not sending the routing data, the node sends the data but puts the cost of the link to infinity.

Split horizon with poisoned reserve prevents routing loops involving only two routers. For loops involving more routers on the same link, split horizon with poisoned reverse will not suffice. [2] A detailed example of this behaviour is available in the book by Huitema [3].

3.4 Triggered updates

RFC 1058 describes also triggered updates and states that "counting to infinity is still possible", even with these mechanisms [2]. When a node receives a update, it counts the shortest path again. If the counting results in as a change of the metric, that is, a new shortest path is found, the node sends updates to the network. The problem with this approach is that routes can change also while the triggered updates are being sent.

3.5 Hold Down timer

An example of a Hold Down timer is given in RFC 2091 Triggered Extensions to RIP to Support Demand Circuits [6]. The hold down timer is started, when

- a timer for a particular route expires or
- a formerly reachable route changes to unreachable in an incoming response.

When the timer reaches zero, the route is removed since the link or path the node is considered dead. The "Trigger RIP" has also other functionalities which differ from the distance vector protocol explained earlier, thus a complete description of the hold down timer is not given.

4 Link-state routing

In this section, we will briefly visit a different routing scheme called link-state routing. Link-state routing is inherently loop free according to Huitema, that is, link-state routing protocols don't suffer from the count-to-infinity problem [3]. The following is mainly based on the textbook Computer Networks by Tanenbaum [9].

Link-state protocols have 4 different functions: Neighbor discovery, Cost (metric) measurement to each of the neighbors, Sending a packet *to all other routers* of the data it just learned. Compute the shortest paths.

The key differences between link-state and distance vector routing is that link-state uses algorithms derivated from the Dijkstra's shortest path algorithm while distance vector uses distributed Bellman-Ford and distance vector protocols start counting from the updates even if the entire topology is not known, while link-state protocols find out the entire topology and then start counting.

Link-state routing uses more memory for keeping the whole topology and the protocols are more complex than distance vector.

5 Present and future solutions

Destination Sequenced Distance Vector (DSDV) routing protocol [8] is explained in this section. In the first subsection I present briefly how the protocol works, and in the second subsection I explain the use of the sequence numbers to solve the count-to-infinity problem.

5.1 Destination Sequenced Distance Vector

Destination Sequenced Distance Vector routing protocol is designed by Charles E. Perkins and Pravin Bhagwat for routing in ad hoc networks. Ad hoc networks are mobile networks without a fixed infrastructure. A fixed network can be thougth as a special case of a mobile network.

DSDV is a proactive protocol which means that it derives routing information before the routes are needed. Alternative to proactive routing is reactive or on-demand routing. Reactive routing protocols establish routes when needed. An example of a reactive routing protocol is the Ad hoc On-Demand Distance Vector (AODV) Routing protocol [7].

The DSDV can be considered as an extension to the imaginary distance vector protocol presented in section 2.2. The routing table consist of 4-tuples (N,D,DI,SEQ), where the "SEQ" denotes the sequence number. Simplified, the DSDV works almost like the Routing Information Protocol. The metric is hop, but because of the sequence numbers, techniques similar to the maximum hop count or split horizon or poisoned reverse are not needed.

5.2 Sequence numbers

DSDV uses so called sequence numbers in routing messages for a number of purposes. One of the results of using them is that DSDV is provably loop-free, that is, it solves the count-to-infinity problem. [8]

The sequence numbers are used to decide whether the routing table is updated or not. When a routing message is received, the message is compared to the routing table. If a route does not exist the advertised route from the message is added to the routing table. If a route *exists*, then the sequence numbers from the routing table and the message are compared.

If the sequence numbers are *equal* the routing message is discarded unless the metric is better, that is, a smaller hop count. Then the old route is replaced with the new route.

If the sequence number is *smaller* than the one in the routing table, the message is discarded.

If the sequence number is *greater* than the number in the routing table, the old route is replaced with the new route.

Jaffe and Moss have presented a preliminary result in their work [4] for necessary requirements for loop freedom in distance vector routing. Perkins and Bhagwat build on that to prove that DSDV is loop-free. The proof is available in the book Ad hoc networking [8].

6 Comparison of past and present

The problems with the past solutions are that they do not cover all possible situations where a link brakes. The protocol developed by Perkins provably solve all possible kind of link breakages. Therefore, it can be concluded that the use sequence numbers are indeed a preferred way to solve the count-to-infinity problem.

7 Conclusion

We have now surveyed the distance vector routing and solutions for count-to-infinity problem in fixed networks. The Routing Information Protocol had many mechanism trying to solve the count-to-infinity problem. We have also peeked at the link-state routing, which is inherently loop-free. Finally, we presented the sequence number solution, which is used in the routing protocols for ad hoc networks developed by Charles E. Perkins. Based on the survey, I conclude that sequence numbers will be used in all future routing protocols, if the distance vector paradigm prevails.

References

[1] Richard Bellman. On a routing problem. Quarterly of Applied Mathematics 16, 1958. William Byrd Press, INC., Richmond, Virginia.

[2] C. Hendrick. Routing Information Protocol. RFC 1058, IETF Network Working Group, June 1988. Category: Historical.

[3] Christian Huitema. Routing in the Internet, 2nd Edition. Prentice Hall, Upper Saddle River, 2000. ISBN 0-13-022647-5

[4] J.M. Jaffe and F. Moss. A Responsive Distributed Routing Alorithm for Computer Networks. IEEE Transactions on Communications COM-30:1758-1762, July 1982.

[5] Alberto Leon-Garcia and Indra Widjaja. Communication Networks, Fundamental Concepts and Key Architectures. McGraw≠Hill Higher Education, Singapore, International Editions 2000. ISBN 0-07-022839-6.

[6] G. Meyer and S. Sherry. Triggered Extensions to RIP to Support Demand Circuits. RFC 2091, IETF Network Working Group, January 1997. Category: Proposed Standard.

[7] C. Perkins, E. Belding-Royer and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, IETF Network Working Group, July 2003. Category: Experimental.

[8] Charles E. Perkins. Ad hoc networking. Addison-Wesley, 2001. ISBN 0-201-30976-9.

[9] Andrew S. Tanenbaum. Computer Networks, Third Edition. Prentice Hall International, Upper Saddle River, 1996. ISBN 0-13-394248-1.