A Distributed Approach for Virtual Network Discovery

João Nogueira[1,2], Márcio Melo[1,2], Jorge Carapinha[2], Susana Sargento[1]
[1] Instituto de Telecomunicações, University of Aveiro, Portugal
[2] Portugal Telecom Inovação, Aveiro, Portugal
joaonogueira@ua.pt, marciomelo@av.it.pt, jorgec@ptinovacao.pt, susana@ua.pt

***Abstract*—The ever-increasing requirements for performance, flexibility, and robustness are imposing a severe strain on the Internet's stagnated architecture. Network virtualization arises as a potential solution for improving the current situation by letting multiple networks with different requirements, architectures and protocols to coexist and share the same infrastructure in an independent way. These advantages are of great interest for network operators.**
This paper presents a virtualization platform that enables the creation, discovery, monitoring and management of virtual networks. It then concentrates on the proposal of a distributed discovery algorithm that is able to discover both physical and virtual network topologies. This algorithm uses the concepts of level 2 multicast and spanning tree to decrease the overhead related to the exchange of discovery messages between the nodes. The simulation results show that, when compared to gossip based discovery algorithms, the performance of our approach is significantly increased, up to an improvement in exchanged messages of three orders of magnitude, when considering physical networks with 500 nodes. Experimental results have shown that distributed approaches also introduce significant improvements when compared with centralized approaches.

***Index Terms*—Virtualization, Virtual Networks, Embedding, Mapping, Distributed Discovery**

I. Introduction & Motivation

Network virtualization is, nowadays, considered as an enabler technology that will allow future, radically different and innovative network protocols and architectures. Substantial efforts have been put into research through several Future Internet initiatives from Europe, Japan and the USA. Even though, initially, the main purpose of network virtualization was to experiment and evaluate new protocols and network architectures, there has been an increasing interest from the network's operators on this technology [1], [2].

The main advantages for operators rely on the ability to improve the revenues through consolidation of resources, and the added business advantages of being able to provide their customers with custom-tailored networks. With network virtualization in place, new business roles will arise [3] and the decoupling of the physical infrastructure from the services running on top of it will be fostered, hence increasing the adaptability to the ever-changing business needs. It can also be seen as a non-disruptive path to introduce disruptive technologies and business models.

Although very attractive from several standpoints, network virtualization contains several challenges [4]. Virtualizing the networks' resources is one of them; research on efficient virtual routers that can compete with the current hardware routers is underway and promising results have been attained [5], [6]; still, they cannot compete on the high-performance routing arena.

Other fundamental issues are related to the management, monitoring and embedding of virtual networks in the physical infrastructure. For instance, to our best knowledge, no discovery algorithms focusing on virtual network topologies have been developed so far. Most of the focus has been on overlay networks [7], [8], [9].

This paper presents an architecture developed to facilitate the creation and management of virtual networks. It then focuses on a distributed topology discovery algorithm that performs the discovery of all nodes in a virtual and physical topology. The proposed algorithm for full network discovery relies on the neighbourhood and multicast concepts to optimize the process of messages exchange between nodes, and on the spanning tree algorithm to reduce the information exchange of the distributed approach by electing a node to exchange information. The obtained results, both through simulation and real experimental scenarios, show that the proposed algorithm is able to discover the virtual topology with low overhead and discovery times, with significant improvements when compared to both centralized approaches and with state of the art distributed approches applied to virtual networks.

The paper is organized as follows. Section II discusses the related work on discovery algorithms. Section III presents the architecture of the virtualization platform and its functionalities. Section IV presents the proposed discovery algorithm and its main functionalities and characteristics. After analyzing the performance of the discovery algorithm on section V, both in simulation and real experiment environments, the paper concludes on section VI.

II. Related Work on Discovery Algorithms

A fundamental requirement to support embedding algorithms is to know exactly the existing physical and virtual topologies and resources' characteristics, and the status of all network elements and links. This knowledge can be provided by a centralized or distributed approach. A centralized approach requires a central network element to gather information about topologies updates in order to guarantee the consistency of the topology databases and the detection of failures. In a large network, this approach can be prone to scalability problems. A distributed approach requires the discovery of the network nodes to be performed by the nodes themselves. A distributed discovery mechanism should be robust, fast to converge and efficient in gathering and disseminating network information with a reduced footprint in the substrate network.

Regarding physical topology discovery, there are multiple commercial applications that rely on Layer 3 information to build the physical networks' topology, showing the logical connections between the resources. Although the discovery of the physical network topology is essential, the discovery of virtual networks' topologies is also required and presents several unaddressed challenges. Since virtual networks are a relatively new concept, and no complete network virtualization platform has been developed so far, there is a general lack of scientific studies regarding virtual topology discovery, although guidelines have been provided by some authors. Some initiatives, like CABO [10] advocate the use of a separate independent discovery plane, and an implementation using distributed algorithms was suggested by [11].

Virtual networks are made of virtual resources laying upon physical resources, therefore the information regarding their topologies is spread-out throughout the physical network. If we consider overlay networks, one will quickly realize the immense similarities between them. Since overlay networks have already been studied extensively, in part due to the popularity of Peer-to-Peer (P2P) communities, their topology discovery mechanisms are a good starting point for developing a virtual network discovery algorithm.

Due to the distributed nature of P2P, the focus has been on distributed discovery mechanisms. Gossip-based broadcast algorithms [7], also known as probabilistic broadcast algorithms, are known for trading reliability guarantees for scalability properties, since they impose a smaller overhead on the network than uncontrolled flooding methods.

T-Man [7], [8] is one of such algorithms. It is gossip based and targets large scale and highly dynamic systems. Assuming random overlay networks with nodes connected through a routed network, the algorithm tries to find each node's neighbours, based on ranking functions that take into account the properties of each node, such as ID and geographic location. By sharing neighbourhood knowledge each node will build its relevant neighbour table, i.e. its target topology.

The algorithm in [9], in the context of P2P networks, proposes a hybrid approach to peer discovery, using a central cache for peers not in the local network, i.e. behind some gateway with Network Address Translation (NAT), and multicast for discovering peers within the same local network. This dual approach combines the benefits of both the centralized and distributed model.

Although none of the studied algorithms is specifically targeted at virtual networks, most of the information sharing, propagation and topology building concepts can be applied and will thus lay the foundations for the virtual topology discovery algorithm.

III. Platform Design

The goal of the developed virtualization platform is to provide the operators with a network virtualization solution that is easy to use, versatile, and efficient in virtual network discovery. The resulting platform provides the necessary functionalities to discover, monitor, deploy and manage virtual networks running on top of a substrate network. It is designed to run on Fedora Core 8 and Debian Lenny Linux distributions with the Xen kernel.

A. Architecture

This virtualization platform is composed by three modules: the Agent module, the Manager module and the Control Centre module; their hierarchical decomposition can be analysed on figure 1. The Control Centre module is the user's front-end,

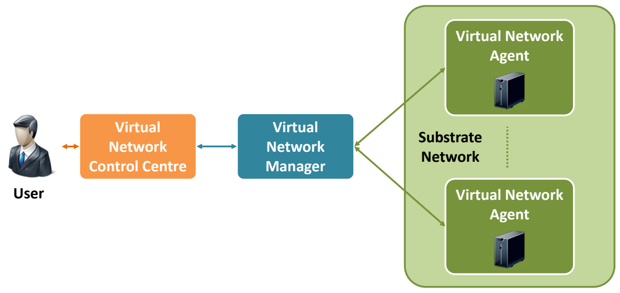


Figure 1. Network Virtualization System Suite - Existing modules

i.e. the Graphical User Interface. It is used in order to perform actions on the network and analyse the gathered data. It provides the user with the virtual network creation, management and monitoring features. The Manager module's functions are

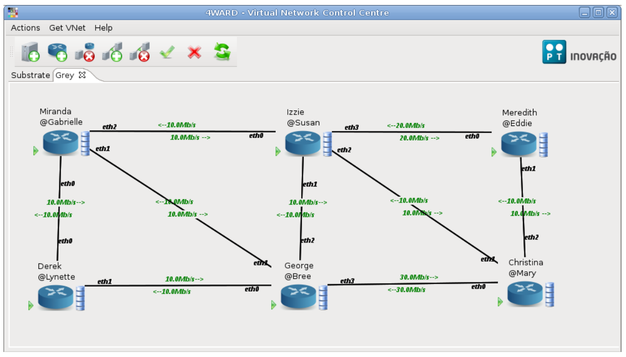


Figure 2. Virtual Network Control Centre

many-fold: (1) it gathers information from the Agents and sends commands to them; (2) it aggregates their information to build the substrate and virtual network's topologies and to maintain an up-to-date database containing the resources' static and dynamic information; and (3) it keeps the Control Centre with up-to-date information about its requested virtual networks (this information is used to change the state of a resource, or map and deploy a virtual network request).

Finally, the Agent module is designed to run on every substrate node. The Agents send their local resources' information to the Manager, provide discovery functions through a distributed algorithm, and execute resource creation and network configuration requests.

B. Functionalities

This section briefly describes the main functionalities of the virtual network platform.

1) Physical and Virtual Resource and Topology Discovery: By providing means of automatic discovery of both the network resources and links, the network administrator can have a global view of the running networks and respective topologies at a glance, in a simplified manner. The nodes exchange messages to discover each other, thus every node, virtual or physical, knows exactly its neighbours. By combining the knowledge acquired by every node, it is possible to build a map of the full topology, both physical and virtual.

2) Substrate and Virtual Network Monitoring: Resource monitoring is fundamental if one wants to have an accurate view of the virtual and physical networks at a given point in time. The monitoring functions periodically update the resources' information, therefore it is possible to identify diverse situations, such as failures and high resource usage, which may require immediate action. Monitoring for both physical and virtual link information is also provided.

To provide proper updated information, every Agent periodically checks its local resources' configuration and status, and reports back to the Manager if any change occurs. These triggered, event-driven updates reduce the overhead traffic on the network. Several parameters are monitored: Central Processing Unit (CPU) load, Random Access Memory (RAM), Hard Disk Drive (HDD) usage, interface and link status, interface bridge attachment and configuration, number of running virtual machines and their state. If a resource crashes or becomes misconfigured, the network administrator will have this information and will be able to take proper actions.

3) Virtual Network Mapping: The virtual network embedding problem is a complex one, where the placement of resources and links must be optimized. This dual optimization can be computationally heavy; therefore, efficient mechanisms must be developed in order to provide a "good enough" mapping that is not excessively time-consuming.

The virtual network deployment feature requires user interaction. The user shall place the resources via drag-and-drop functions and connect them with links. The user may specify the nodes' CPU capabilities, RAM amount, location, number of interfaces, and also perform network addressing configurations. The final step in creating a new virtual network is to commit it to the Manager. The Manager will then evaluate the specified virtual network and either accept it or refuse it.

4) Virtual Network Management: Just like the previously described monitoring ability, the management feature is also a fundamental one. To that end, some functionalities are provided, such as changing the virtual resource state, i.e.: rebooting, shutting down, suspending or powering the resource up, the amount of assigned RAM memory, deleting the resource or even the full virtual network. The assigned RAM memory may be changed in runtime.

IV. Distributed Topology Discovery

In order to be able to properly map new virtual networks and allow the user to monitor the physical and existing virtual networks, mechanisms for discovering their topology are required. The distributed topology discovery, on the one hand, intends to reduce the required processing power on the Manager, while on the other hand, it is a step forward towards in-network management mechanisms.

The algorithm for full network discovery relies on the neighbourhood concept, where each physical node knows exactly who its neighbours are, and also, who are the virtual neighbours of its local virtual nodes. The Agents register themselves in a predefined multicast group and, afterwards, exchange messages with each other. The multicast group used is link-local, i.e. Time-To-Live (TTL) of 1, in order to avoid sending the discovery messages to nodes that do not want it and would otherwise need to process the packet before discarding it. By aggregating each Agent's knowledge, the full topology map may be built.

In a given network segment, one of the Agents has a special function, the Designated Root (DR) function. This approach has its roots on the spanning tree algorithm. It aims to reduce the information exchange of the distributed approach by electing a node to exchange information. This node is responsible for transmitting all the information about its network segment to a new Agent arriving at the network.

A. Physical Topology Discovery

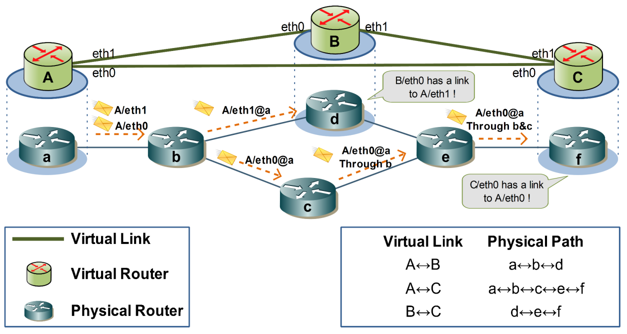
Upon start-up and periodically, each Agent sends a multicast Hello message through its interfaces. The messages are specific to the interface and indicate their origin interface. This Hello message exchange allows each Agent to know its directly connected physical neighbours.

B. Virtual Topology Discovery

The virtual network topology is not simple to achieve, since a virtual link may span through several physical links, thus message forwarding mechanisms had to be designed. The Agents exchange information about two kinds of resources: their local resources and the resources advertised by their neighbours that utilize the local physical node as a network hop.

In figure 3, an example of virtual resource advertisement, in this case regarding the virtual resource A, can be seen. This resource is running on the physical node a that realizes that the virtual node A has two virtual interfaces bridged to different Virtual Local Area Networks (VLANs) on one of its physical interfaces. The physical node will thus advertise A through its physical interface. The advertisement messages will be received by the physical node b which will then register A, with its interfaces, as a potential neighbour. b will then check if any local virtual resource is connected to A, either through its *eth0* or *eth1* interface. Since this is not the case, *b* will then check if it is a hop to any possible link of *A* and will identify two bridge entries: one that forwards *A's eth0* interface and another one that forwards the *A's eth1* interface, through different physical interfaces, in this case. After identifying itself as a hop, *b* will advertise *A* and its interfaces to the proper physical links. *d* will receive the resource advertisement, and checks for a connection with a local virtual resource. This verification succeeds, and *d* becomes aware that its virtual resource *B* is connected to *A's eth1* interface through *B's* virtual interface *eth0*.

The described "verify-and-forward" mechanism is repeated on the lower network branch until *f* becomes aware that its virtual resource *C* is also connected to *A*.



Virtual Link	Physical Path
A↔B	a↔b↔d
A↔C	a↔b↔c↔e↔f
B↔C	d↔e↔f

Figure 3. Distributed Discovery - Resource Advertisement

C. Designated Root

The DR is elected based on the Agents' ID, which is an integer attributed from the Manager when an Agent starts-up; the Agent with the lowest ID on a network segment is elected the DR and is responsible for sending the networks' information to newcomers. At start-up, every Agent 's DR is itself; after receiving a message from an Agent with a lower ID, the DR will be updated to reflect the new ID. Each neighbour has an expiration timer that will trigger a new DR election if the current DR fails to communicate within a given time period, i.e. if no message is received from it.

D. Bootstraping

In order to begin the discovery mechanism, the Agent must have a unique ID given by the Manager and the initial local resource discovery must be completed.

After these initial conditions are met, the discovery algorithm 1 starts and the periodic sending of Hello messages begins. Upon receiving an Hello message, the current DR will identify a new physical neighbour and, since it is the DR, a full update regarding the network segments' associated virtual resources is sent to the arriving Agent. On the new Agent's side, after receiving any message of a previously existing physical neighbour, an update containing its full knowledge, i.e. only local resources, will be sent. The DR will be afterwards updated based on the IDs of the discovered physical neighbours. This is represented in the first half of the algorithm.

Algorithm 1: Per-Interface Discovery Algorithm

```
input : IFC_i
DR = My_ID ;
repeat Msg_Type = Multicast_Receive(IFC_i,Msg_Buffer)
    NG = GetNeighbour(Msg_Buffer);
    if NewNeighbour(NG) then
        AddToNeighbourList(IFC_i,NG) ;
        if DR == My_ID then
            SendAllKnowledge(IFC_i) ;
        end
        else
            ExclusiveUpdateDR(IFC_i,NG) ;
            if DR == My_ID then
                SendAllKnowledge(IFC_i) ;
            end
            UpdateDR(IFC_i);
        end
    end
    switch Msg_Type do
        case Resource Message
            VNG = GetVirtualNeighbour(Msg_Buffer);
            if NewVirtualNeighbour(VNG) then
                AddtoVirtualNeighbourList(IFC_i,VNG);
            end
            if size (IFC_List = LinkToOtherInterfaces(IFC_i,VNG))>0 then
                AddToVirtualNeighbourLists((IFC_List,VNG) ;
                SendResourceMessage((IFC_List,VNG) ;
            end
            CheckForVirtualLinksWithLocalResources(VNG);
        endsw
        case Delete Message
            VNG = GetVirtualNeighbour(Msg_Buffer);
            IFC_List = LocateResourceEntriesOnAllInterfaces(VNG);
            SendDeleteMessageThroughRelevantInterfaces(IFC_List,VNG) ;
            RemoveEntriesOnAllInterfaces(IFC_List,VNG);
        endsw
    endsw
    UpdateLastContactTime(NG);
until Terminate Signal;
```

E. Resource Update Mechanism

There are two main situations where an Agent advertises a resource. The first one is when a new local resource is created; the Agent where the resource resides will send updates through the interfaces related to that particular resource. The second possible situation happens when a resource is advertised as a consequence of a received resource advertisement, i.e. there is an advertisement forwarding.

The Agents receiving the new resource advertisement will place an entry on the receiving interface's knowledge database. They will then locate potential output interfaces for that resource, i.e. the Agents will verify if they are or not a hop for any virtual link belonging to the advertised virtual resource. If they are, they will forward the resource information through the relevant interfaces; otherwise, they will just keep the resource information stored, since it may be needed later. A local verification will also be performed in order to asses if any of the local virtual resources is connected through a virtual link to the received resource. This is represented in the second half of the algorithm by 'Resource Message'.

F. Resource Removal Mechanism

Another fundamental part of topology discovery is to be able to delete virtual resources and maintain the consistency in the existing databases. To that end, a mechanism for virtual resource removal also exists. The forwarding mechanisms are

Number of physical nodes	Increment	Number of simulation runs
4 to 50	2	100
60 to 150	10	100
200 to 250	50	50
300 to 350	50	20
400 to 500	100	10

Table I
DISTRIBUTED DISCOVERY - 1ST SIMULATION PARAMETERS.

similar to the ones of new virtual resource advertisement. This is represented in the second half of the algorithm by 'Delete Message'.

V. PERFORMANCE RESULTS

A. Simulation Results

In order to assess the scalability of the proposed distributed discovery algorithm, two different simulation tests were performed in Matlab. The first one assesses the scalability of the algorithm with the increase of physical nodes, in the presence of a single virtual network spanning half of these nodes. The second one assesses the scalability with the increase of the number of virtual networks. The physical topology was generated using the Waxman random topology generation [12] method using the recommended parameters, $\alpha = 0.4$ and $\beta = 0.4$, while the virtual networks were generated by selecting randomly half of the physical nodes, and generating virtual links using the same Waxman model. Full connectivity was assured on every topology.

Since the virtual links did not match existing physical links most of the times, the Dijkstra algorithm was run in order to get a physical path for each virtual link, where the link costs were the actual link sizes.

For comparison purposes, three discovery algorithms were considered: the proposed one, and two others based on uncontrolled and probabilistic flooding [7], with a flooding probability of 50%.

For the first simulation scenario, a single random virtual network was generated on top of a random physical network. The number of physical nodes varied between 4 and 500 in a non-uniform way. Due to the time required to complete the simulation when in the presence of many nodes, the number of simulations runs for each number of physical nodes varied according to table I.

When running the proposed algorithm, the simulation stopped when the discovery message reached the intended destination, and was not propagated afterwards. When running the flooding algorithms, the simulation has stopped when all nodes had been visited.

For the second simulation scenario, random physical networks were generated with 100 nodes and an increasing number of random virtual networks, from [1 to 21] were generated on top of them. The simulations were repeated 10 times and the presented values correspond to the mean values and the 95% confidence values (although very small and not visible in the figures).

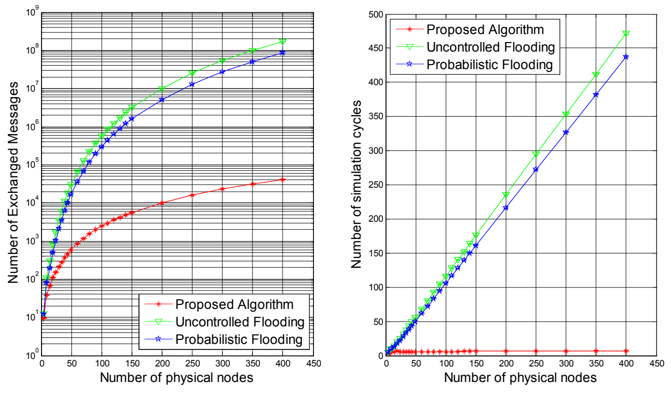


Figure 4. Discovery Algorithm Scalability Tests – Number of Physical Nodes.

Considering the graphics relating the first simulation scenario, it is clear that the proposed algorithm imposes a much smaller overhead than flooding and probabilistic flooding techniques. Regarding figure 4, exhibiting the number of exchanged messages, in the case of 500 physical nodes, the difference between the probabilistic flooding algorithm and the proposed one is of about three orders of magnitude.

With respect to the required simulation cycles, the proposed algorithm shows a behaviour similar to that of the flooding algorithms for less than 10 physical nodes. However, for a significant number of nodes, the number of required simulation cycles starts to stabilize on our approach, while in the other cases it continues growing.

The behaviour of the flooding algorithms in respect to simulation cycles is to be expected. Since the network size keeps on growing, so will the number of hops that forward the discovery messages; thus, the number of cycles required for the messages to reach every node is proportional to the number of physical nodes.

In the proposed algorithm, the path crossed by the discovery messages is a previously optimized one; therefore, the number of simulation cycles required for convergence is much smaller. The stagnation in the required number of cycles observed is due to the number of physical hops utilized by the virtual links, being kept approximately constant with the increase of substrate nodes, since the virtual networks are always created with half of the physical nodes.

Figure 5 shows the results obtained with the increase in the number of virtual networks on top of a single substrate network. It is possible to observe that its behaviour follows a linear trend, i.e. that the number of exchanged messages and required simulation cycles are much reduced when compared to the other approaches.

B. Experimental Results

In this section, we determine the time required to perform the discovery of the network nodes, in a real scenario of 0 to 40 running virtual networks. These results are obtained with the real virtualization platform. For simplicity purposes, all the existing VNets were equal and their topology was an exact replica of the underlying physical network. Each

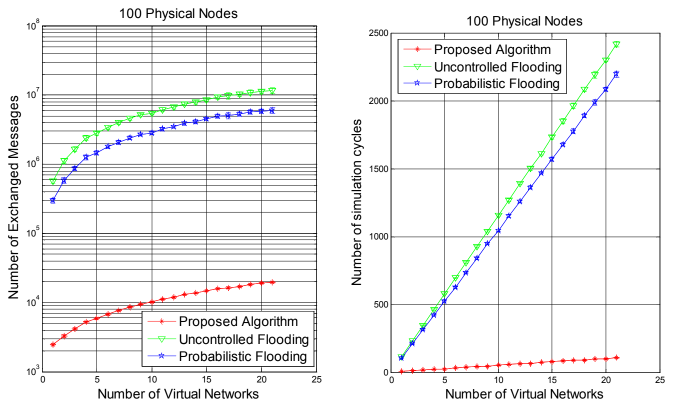


Figure 5. Discovery Algorithm Scalability Tests – Number of Virtual Networks.

virtual node was configured with 1 CPU, 64MB of RAM and all virtual links were setup with a 1.0Mbps bandwidth. The *Manager* was running on an external computer, connected directly to the testbed (Intel Core 2 Duo P8600; 4GB of RAM; 100Mbps link). In an attempt to determine the amount of time required for an updated network view, two virtual link discovery approaches were tested:

- Centralised: The *Manager* has full knowledge of existing Physical and Virtual Nodes, interfaces and bridges, and it performs computations in order the determine the existing Virtual and Physical topologies.
- Distributed: The Agents communicate with each other in other in order to determine existing virtual links, as explained in section IV in our distributed discovery approach. The agents then report back to the *Manager*, which then only has to perform the simple task of appending nodes and links to the virtual network.

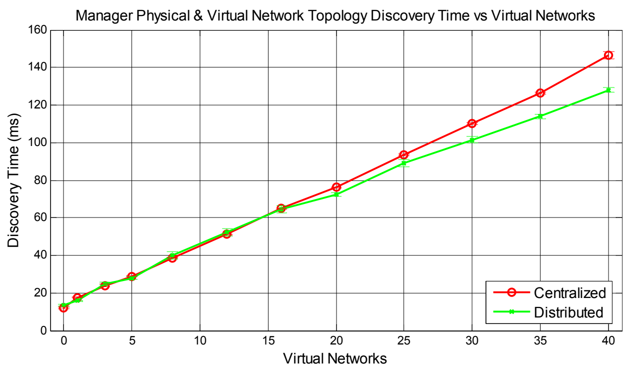


Figure 6. Experimental Results - Distributed vs Centralized Discovery

Considering the obtained experimental results of figure 6, it is possible to observe the linear trend also seen on the simulation results. For a reduced amount of virtual networks, the convergence time for both the centralized and the distributed algorithms are similar, but, as the number of virtual networks begins to grow, the distributed approach provides better results. Taking into consideration the size and complexity of the testbed, which is small, we can conclude that distributed approaches need to be supported in future and complex networks.

VI. Conclusions

Virtual network discovery algorithms, despite being fundamental for a future virtualized network, have not been a major research target in the past few years.

The developed distributed discovery algorithm succeeded on providing a simple, fast and low overhead virtual and physical topology discovery algorithm, as proven both by the obtained simulation results and the experimental tests conducted.

In addition to the distributed discovery algorithm, the developed software suite provides embedding, monitoring and management functionalities and aggregates them in a convenient and easy to use graphical user interface. Therefore, future work will be devoted to the proposal of new mechanisms for the mapping of virtual networks on the physical infrastructure, and on the support of monitoring and management functionalities. Multi-provider environments will also be investigated.

References

[1] J. Carapinha and J. Jiménez, “Network virtualization: a view from the bottom,” in *Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures*. Barcelona, Spain: ACM, 2009, pp. 73–80. [Online]. Available: http://portal.acm.org/citation.cfm?id=1592648.1592660

[2] M. Melo, S. Sargento, and J. Carapinha, “Network Virtualisation from an Operator Perspective,” *Proc Conf. sobre Redes de Computadores - CRC*, October, 2009.

[3] G. Schaffrath, C. Werle, P. Papadimitriou, A. Feldmann, R. Bless, A. Greenhalgh, A. Wundsam, M. Kind, O. Maennel, and L. Mathy, “Network virtualization architecture: proposal and initial prototype,” in *VISA ’09: Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures*. New York, NY, USA: ACM, 2009, pp. 63–72.

[4] N. M. K. Chowdhury and R. Boutaba, “Network virtualization: State of the art and research challenges,” *IEEE Communications Magazine*, vol. 47, no. 7, pp. 20–26, July 2009. [Online]. Available: http://www.mosharaf.com/wp-content/uploads/nv-overview-commag09.pdf

[5] N. Egi, A. Greenhalgh, M. Handley, M. Hoerdt, F. Huici, L. Mathy, and P. Papadimitriou, “A platform for high performance and flexible virtual routers on commodity hardware,” *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 127–128, 2010.

[6] N. Egi, A. Greenhalgh, M. Handley, M. Hoerdt, F. Huici, and L. Mathy, “Towards high performance virtual routers on commodity hardware,” in *CoNEXT ’08: Proceedings of the 2008 ACM CoNEXT Conference*. New York, NY, USA: ACM, 2008, pp. 1–12.

[7] M. Jelasity, A. Montresor, and O. Babaoglu, “T-Man: Gossip-based fast overlay topology construction,” *Comput. Netw.*, vol. 53, no. 13, pp. 2321–2339, 2009.

[8] M. Jelasity and O. Babaoglu, “T-Man: Gossip-based overlay topology management,” in *In 3rd Int. Workshop on Engineering Self-Organising Applications (ESOA’05)*. Springer-Verlag, 2005, pp. 1–15.

[9] Y. Liu, G. Zhu, and H. Yin, “A practical hybrid mechanism for peer discovery,” in *Intelligent Signal Processing and Communication Systems, 2007. ISPACS 2007. International Symposium on*, nov. 2007, pp. 706 –709.

[10] N. Feamster, L. Gao, and J. Rexford, “How to lease the internet in your spare time,” *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 1, pp. 61–64, 2007.

[11] N. M. K. Chowdhury and R. Boutaba, “A survey of network virtualization,” *Computer Networks*, vol. 54, no. 5, pp. 862–876, April 2010. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2009.10.017

[12] B. Waxman, “Routing of multipoint connections,” *Selected Areas in Communications, IEEE Journal on*, vol. 6, no. 9, pp. 1617 –1622, dec 1988.