Towards Deterministic Network Coding in Hierarchical Networks

Oana Graur
Transmission Systems Group (TrSys)
Jacobs University
Bremen, Germany
Email: o.graur@jacobs-university.de

Werner Henkel
Transmission Systems Group (TrSys)
Jacobs University
Bremen, Germany
Email: w.henkel@jacobs-university.de

***Abstract*—Starting from a recently proposed network coding resource optimization method, we discuss the scalability of this solution to larger scale-free topologies that are also characterized by a realistic link loss distribution which we derive from measurements. We sketch a possibility of designing a network coding aware multicast routing scheme for hierarchical networks.**

Keywords–deterministic network coding; hierarchical networks;

I. Introduction

The purpose of this work was to address the concept of deterministic linear network coding in the context of scale-free hierarchical networks, while considering practical aspects such as link loss and node degree distribution, which are typically ignored in literature. Network coding has been already shown to provide high throughput gains, achieving the maximum flow in error-free networks [1]. It is also known that when packet loss is present in the network, the transmission performance of networks using deterministic network coding can be degraded beyond the scenario in which no network coding is employed, if the global kernel (GEK) assignment is poorly chosen. Random linear network coding (RLNC), on the other hand, is more resilient to packet loss and topology changes but it introduces a significant overhead. In the case of RLNC, in order to ensure decodability at the sink, with a high probability, the size of the finite field from which the coefficients are chosen has to be large. This fact also arises some practical aspects that need special consideration. The multiplication and division operations require look-up tables according to the size of the finite field. Nevertheless, for easy retrieval, these tables should reside in the CPU cache, being subjected to size constraints [2]. In order to overcome the performance degradation of deterministic linear network coding when confronted with erasures, we analyze a solution for a better identification of the optimum coding nodes. The idea was originally exemplified by Li and Pan [3] and makes use of the Ant Colony Optimization algorithm. Unfortunately, in the original work, the algorithm is discussed only on a small artificial topology and the authors do not provide any insight towards whether it is applicable to practical network scenarios, that exhibit link loss and other characteristics. We hereby propose to extend this idea and study its scalability to much larger scale-free networks, such as the Internet, that exhibit specific properties like preferential attachment and growth over time. Although several methods for inferring link loss statistics from path loss measurements have been previously presented in literature [4], [5], to the knowledge of the authors, no probability distribution regarding the actual link losses in large scale-free networks was publicly available. For this reason, we conduct extensive network measurements and present our findings in Section III.

Since the focus of our ongoing research aims at extending current routing algorithms to allow for network coding, we summarize, in Section II, the hierarchical structure of the Internet and two of the most widely employed routing protocols: an interior gateway protocol, namely Open Shortest Path First (OSPF), and Border Gateway Protocol (BGP). Throughout Section III, some previous theoretical results regarding node degree distribution in the Internet are compared to measurements we performed in the PlanetLab testbed. A link loss distribution is also presented, which we have obtained based on path loss measurements. Section IV is concerned with a method of performing network coding in non-hierarchical networks, based on the identification of optimum coding nodes through the use of Ant Colony Optimization. In Section V, we study the possibility to extend the network coding scheme to hierarchical scale-free networks. Section VI summarizes and concludes our findings.

II. Hierarchical Networks

The Internet, as it is today, can be seen as a large hierarchical aggregation of *Autonomous Systems (AS)*, connected through backbone links. An *Autonomous Systems (AS)*, as defined in RFC 1930 [6] and RFC 4271 [7], is a set of routers of one or more IP prefixes, under a single technical administration and with a clearly defined routing policy. Within this collection of routers, interior gateway routing (IGP) protocols are used, while exterior gateway protocols (EGP), such as the Border Gateway Protocol (BGP), are used to route traffic between multiple ASs. Each AS is identified by an unique Autonomous System Number (ASN) and falls within one of the three categories: stub, multihomed, or

transit AS. A stub AS carries only local traffic, having only one connection to another AS. A multihomed AS has multiple connections to other ASs but it does not carry transit traffic, while a transit AS has multiple connections and by default it carries both local and transit traffic. A reasonable assumption is that a high gain in throughput would be achieved by employing network coding for significantly large traffic flows in the Internet. This assumption is based on recent increasing trends in multimedia applications such as videoconferencing and IPTV, as well as on studies showing that approximately 9 % of flows between ASs account for 90 % of the total number of bytes transmitted [8].

A. OSPF - Open Shortest Path First

Two widely employed interior gateway protocols are the *Open Shortest Path First (OSPF)* protocol and IS-IS (Intermediate-System to Intermediate-System). OSPF is a link state protocol described in RFC 2328 [9]. As opposed to previous intraAS routing protocols, such as RIP, OSPF builds shortest paths within a network based on multiple cost metrics, such as link bandwidth, delay, distance, etc.. In OSPF, larger ASs are split into areas and each node (router) actively probes the status and costs of the incident edges and builds a table that is sent to its neighbors and further propagated through the entire area. Withing an area, each router shares the same link state information and computes a Dijkstra tree with itself as root in order to determine the shortest paths to all the other nodes. This subdivisioning into hierarchical structures was necessary to limit the size of routing tables. Areas are connected through area border routers, which summarize reachability and cost information for every area and propagate it to the other adjacent areas. This ensures that internal routers of a certain area are able to determine the best area border router that will route their traffic destined to a foreign area in terms of minimum cost. The internal routers of an area need not be aware of the internal topology of adjacent areas. In the case of multiaccess networks, a Designated Router (DR) is elected with the purpose of avoiding excessive flooding of link state information and to represent the network to the rest of the internetwork.

For the rest of the paper, we will refer to the term 'logical node' to refer to a simplified abstractization of a network area, as defined in the context of OSPF.

B. BGP - Border Gateway Protocol

While the choice of a protocol for intradomain routing might vary, in between ASs the same exterior gateway protocol needs to be used. The most popular interdomain routing protocol is the Border Gateway Protocol (BGP), defined in RFC 1267 [10]. BGP was first introduced to account for the need of different routing policies based on economic, political, and security aspects. Unlike OSPF, BGP not only maintains the costs to each destination but also enumerates the routes to those destinations, as a sequence of ASs being traversed. While doing so, it hides all the details of the networks crossed, such as topology and individual link costs.

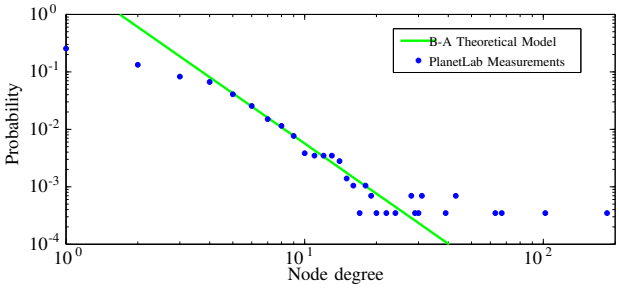

Figure 1. Node degree distribution comparison

III. Packet Loss in Scale Free Networks

A. Degree Distribution

In the past decade, a lot of research has been conducted on modeling large network structures such as the Internet. Barabasi and Albert have shown in [11] that previous probabilistic network models from graph theory, such as the Erdős-Rényi models, were obsolete, since they did not account for the tendency of networks to form clusters and grow over time. Barabasi and Albert have argued that given a network such as the Internet, the probability of a new node introduced to be connected to another one is highly dependent on the other nodes' degree. The term coined to describe this property was preferential attachment. Taking into account the two new characteristics, Barabasi and Albert proposed a new degree distribution for scale-free networks, given in (1).

$$P(k) = 2l^{1/\beta_{BA}} k^{-\gamma_{BA}} \tag{1}$$

By incrementally adding a node at each time step and connecting the node to l different nodes already present in the system, the node degree distribution evolves asymptotically into a power law function. The two constants in (1) were found to be $\gamma_{BA} = 3$ and $\beta_{BA} = 0.5$.

We have conducted several sets of measurements in the PlanetLab Europe [12] network. In Fig. 1, we show the theoretical degree distributions of a B-A modeled network, along with the PlanetLab degree distribution obtained from measurements.

B. Path loss and link loss

One of the tasks of the more general research problem investigated here was link loss estimation. Due to various constraints such as restricted router accessibility and ISP policies, link loss data is not freely available in the Internet.

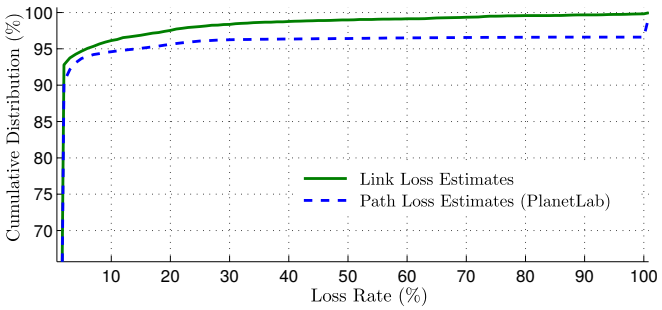

Figure 2. Cumulative distribution functions for link loss and path loss

End-to-end nodes, on the other hand, are easily accessible and can provide accurate measurements of path packet loss. Several network tomography techniques that infer link loss from path packet loss measurements are discussed in the literature [4], [5].

We have chosen 226 nodes from the PlanetLab Europe network and performed network measurements between each pair using tools such as *ping* and *traceroute*. The total number of investigated paths was 50850. One of the practical aspects that had to be considered was the fact that routers have at least two interfaces and multiple IP addresses can be associated with each physical interface. IP aliasing resolution techniques are utilized to identify multiple aliases belonging to the same router. During our measurements we have employed Ally, an active probing IP aliasing resolution technique, and discovered that the 3049 IPs initially discovered belonged to a number of only 1920 distinct routers.

We have performed an estimation of the link loss probability distribution function after processing multiple sets of measurements, following Netscope, an inferrence method described in [4], [5]. Our findings indicate that more than 95 % of the links had an average packet loss rate smaller than 5 %. Both link and path loss cumulative distributions are shown in Fig. 2. The link loss distribution was used for assigning link weights in a B-A network, where the network edges were modeled as binary erasure channels.

IV. Finding shortest paths

A. Ant Colony Optimization (ACO) for finding shortest paths

Although many other algorithms such as Dijkstra and Bellman-Ford can be successfully used in solving shortest paths problems, we discuss here a decentralized algorithm. Ant Colony Optimization (ACO), developed by Marco Dorigo in his PhD thesis [13], was developed based on the analogy that agents of a decentralized system with a collective behavior, such as an ant colony, manage to find shortest paths in an environment, in their search for food locations.

The idea behind ACO originates from the fact that ants, while walking through a network looking for food, deposit a pheromone trail. Once at a node, when an ant is faced with the choice of selecting one of the several incident edges to follow, it is more inclined to choose the one with the highest pheromone concentration. A concentrated pheromone trail is an indicator that the edge has been walked before. Once an ant reaches food, it goes back to its base and starts the search all over again. Over time, the shorter paths will be walked more, thus exhibiting a higher pheromone level. After a certain number of iterations, the algorithm converges and all ants walk the same shortest path.

Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ vertices, $|\mathcal{E}|$ edges, and edge mappings $w : \mathcal{E} \to \mathbb{R}$, the shortest path between a source and a destination can be found by the simple ant algorithm described below. Following the notations introduced by Dorigo in [13], the probability that a certain ant moves from node i to node j in the current step is given in (2), where $\tau_{i,j}(t)$ is used to denote the pheromone level of edge $e_{i,j} \in \mathcal{E}$ at time t and m is total number of ants. Since at the start of the algorithm none of the links have been previously walked, the pheromone level is initialized to the same value throughout the entire graph.

$$p_{i,j}(t) = \frac{\tau_{i,j}^{\alpha}(t)\eta_{i,j}^{\beta}}{\sum\limits_{l\ allowed} \tau_{i,l}^{\alpha}(t)\eta_{i,l}^{\beta}} , \tag{2}$$

$\eta_{i,j} = \frac{1}{w_{i,j}}$ is a constant representing the *visibility* of the link $e_{i,j}$ which is introduced to account for the effect of the edge weights $w_{i,j}$. α and β are two parameters that control the trade-off between the pheromone trail importance and the visibility. The pheromone level of every link is updated after every time instance as given by

$$\tau_{i,j}(t+1) = \rho\tau_{i,j}(t) + \Delta\tau_{i,j} , \tag{3}$$

where $\Delta\tau_{i,j}$ represents the total increment in pheromone level deposited by all the ants on link $e_{i,j}$ at current time t, obtained as

$$\Delta\tau_{i,j} = \sum_{k=1}^{m} \Delta\tau_{i,j}^{k} , \tag{4}$$

with $\Delta\tau_{i,j}^{k}$ denoting the pheromone deposited per unit of length by the kth ant on the $e_{i,j}$ link, which is given by

$$\Delta\tau_{i,j}^{k} = \begin{cases} \frac{Q}{L_k} , & \text{if ant } k \text{ deposited pheromones on } e_{i,j} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Following the notations in [14], Q is a constant and L_k was previously defined as the current tour cost of ant k. Given that each edge has been modeled as a binary erasure channel, L_k is equivalent to the overall erasure probability of the current tour. For an arbitrary path $\mathcal{P}_k$ in a network, walked by ant k, with edges represented by e_i, and individual link erasure probabilities given by p_i, the overall erasure probability can be computed as in (6). Parameter ρ in (3) is introduced to control the pheromone *evaporation rate*, such

that the trail does not build up indefinitely, and is defined within the range $0 < \rho < 1$.

$$L_k = 1 - \prod_{i=1}^{|\mathcal{P}_k|}(1 - p_i) \tag{6}$$

After the convergence time is over, the shortest paths in terms of an arbitrary metric, in our case the path packet loss, can be identified based on the increased pheromone concentration. Although we only use the packet loss metric in our simulations to determine optimum paths through the network, ACO can be easily modified to account for a mixture of other metrics.

B. Identifying coding nodes

Given a network modeled by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and links with an equal capacity of one, we define a source s and multiple sinks t_i. It has been shown in [15] that if maxflow$(t_i) \geq h$ for a sink t_i, then there are at least h disjoint paths between the source and the sink, where h is defined as the minimum cut between the source and any of the receivers. Although h could be achieved for any of the sinks independently, it might not be achievable for a store-and-forward multicast, depending on the network topology. If the network has bottlenecks, the only possibility to achieve the multicast rate for all the sinks simultaneously is to employ network coding at the bottleneck nodes, e.g., combining multiple incoming packets into an outgoing packet.

Since all the paths start at the source and end at various sinks, the bottlenecks in the network will consist of overlapping path segments, in particular, we search for paths that start at the source, diverge, then overlap again later. The length of the overlap should be of at least one link in length, it is not sufficient that the paths cross each other. Once we have identified the path overlaps among different sinks, the coding nodes are found to be the heads of the overlapping segments. This idea was initially briefly presented in [3]. We discuss some practical aspects left uninvestigated and offer a more clear description in [16].

For our simulations, we have chosen a number of ants m equal to the number of nodes. Ants are prevented from looping by having a memory that is reset at the beginning of each new cycle. A new cycle for a certain ant begins after it has managed to find a food source and has returned to the base, depositing a pheromone trail. After a certain number of iterations, 'sniffing ants' are released from the source. The sniffing ants follow the edges with the highest pheromone trail, in a greedy fashion, one ant per disjoint path. In doing so, they lay down another type of pheromone marker that prevents any other sniffing ants running through the network during that cycle to walk the same edge. Unfortunately, there is a possibility that some sniffing ants will block the paths of the others and although there are theoretically at least h disjoint paths between the source and each of the sinks, the sniffing ants could get stuck. A backtracking procedure is necessary to ensure that all the disjoint paths are found. Assuming that for a certain source-sink pair, only $h_f \leq h$ paths have been found, a certain percentage of the edges belonging to the longest among the h_f paths can have the pheromone level decreased before resetting the sniffing ants back to the source. Although it might introduce a small compromise regarding the overall cost of the final disjoint paths, by reducing the pheromone level of some of the edges belonging to the longest of the h_f paths, the desirability of the sniffing ants to follow those paths will be reduced. This ensures a faster convergence on average, since once a dead-end is reached, the region of the search space is changed. The longer a path is, in terms of number of edges, the higher the chances of blocking other disjoint paths.

The fact that nodes only store the pheromone of incident edges for every possible pair sink-source considered was the main factor in choosing ACO for finding shortest paths. The typical assumption is that full network topology information is not available at the nodes. Our findings indicate that the minimum number of required iterations scales with the size of the network. When this condition is satisfied, for a parameter setting of $\alpha = 1$, $\beta = 5$ and $\rho = 0.7$, in more than 90 % of the cases, the paths returned by ACO are within 5 % tolerance of the ideal Dijkstra paths. Considering the overhead ACO introduces and given the fact that for each source-sink pair a new search needs to be performed, the initial flooding of link state information that OSPF requires in order to obtain full topology information for constructing Dijkstra shortest path trees might still be preferred.

V. Network-Coding aware Routing

The problem we face when trying to design a hierarchical network coding scheme is illustrated in Fig. 3. Starting from the canonical butterfly network configuration, we substitute nodes with *logical nodes*, or pseudonodes. We introduce the term logical node to refer to a larger network, similar in concept to an AS. Consider Fig. 3 (a), where all nodes have been replaced with larger subnetworks, namely logical nodes. Node 4 (in blue) can be viewed as the autonomous system in Fig. 3 (b) that needs to be traversed by two traffic flows, a and b.

Depending on the type of traffic, whether it is local traffic or transit, two different situations emerge. In the case of local traffic, network coding can be performed as previously described, once the optimum coding nodes have been identified by analyzing the overlapping path segments between sets of disjoint paths. Here a reasonable assumption is the fact that if the traffic is destined for a local sink, it will not be routed on paths outside of the local area. In the case of transit traffic, a top-down approach is necessary. Starting with the top-most level, the networks (logical nodes) within which network coding needs to be done are first identified. Once the GEK assignment has been determined at the top-most level, it is now clear which areas will simply

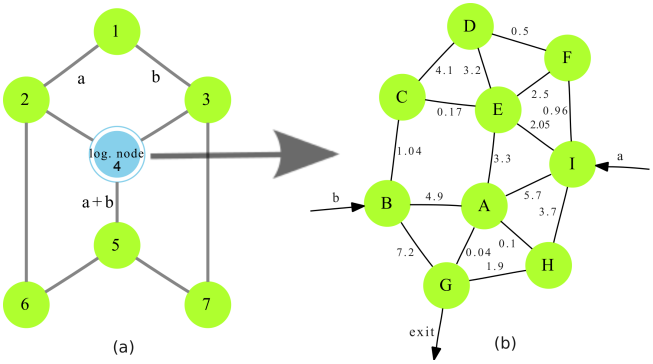

Figure 3. Hierarchical network structure

forward the traffic flows and which will have to combine them. Considering the two-layer example illustrated in Fig. 3 nodes B, I, and G can be viewed as border routers which route traffic in and out of the logical node. Unlike in the previous example in Section IV, where only edges in a graph were assigned a weight, it is now necessary to introduce costs for the vertices. In a typical routing scenario, node B would have to advertise the overall cost of reaching logical nodes 5, 6, and 7 to its neighboring border router in logical node 3, while node I would advertise the overall costs to its corresponding peer in logical node 2, for reaching the same destinations 5, 6, and 7. The cost for reaching a destination propagates through the network in a cumulative fashion, where each area adds its own cost and passes the information to the adjacent areas. The cost advertised by Node B to its adjacent area are different for reaching destinations 5, 6, and 7, from the ones advertised by Node I for the same destinations. This is easily understood, since a packet routed from logical Node 3 to Node 5 through B would follow a different path through logical Node 4 than another packet traveling from 2 to 5 through I.

The overall cost from a sink to destination, whether the metric is bandwidth, delay, link reliability, etc., as defined in the store-and-forward scenario turns to be wrong for the case when packets are combined within the network. Considering the logical node 4 in Fig. 3, with two incoming flows a and b, the optimum coding node to perform the linear combination is found by an examination of the pruned minimum spanning tree rooted at G. Figure 4 illustrates this idea. Node E is easily identified as the node of interest. In the case more input flows need to be combined within an area, the optimum coding nodes are the ones in the pruned minimum spanning tree that have multiple children.

The choice of a MST instead of a Dijkstra tree comes from the fact that our interest was to minimize the overall packet loss for the combined packet outgoing from Node I. Every area border router needs to be aware of the topology of the area it resides in. The overall cost of the MST rooted at the exit of the logical node and pruned at all the other incoming

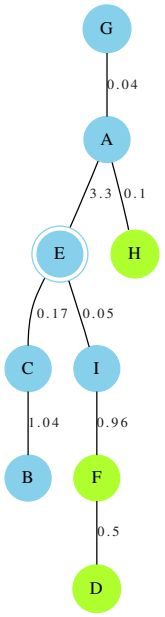

Figure 4. Hierarchical network structure

points into the logical node will always be smaller than the cost of the pruned Dijkstra tree rooted at the same exit node and pruned at all the other incoming points. In our example, if the two flows a and b, specified by a source ID and the same multicast group, are marked to be coded in a specific logical node, the first router in the logical node receiving both flows simultaneously can generate a new combined flow, mark it as coded and output it on the corresponding interface. Due to the MST structure, no loops can occur, thus the distinct flows cannot be combined more than once. If a router receives a flow marked for coding, the MST has to be checked, otherwise the flow should be forwarded according to the Dijkstra tree. The concept described above can be extended recursively to networks with more that 2 layers, either using ACO or MSTs.

VI. Summary and Conclusions

Measurements have been carried out and statistics data has been gathered and processed concerning packet loss and degree distribution in the Internet. We have used Netscope to infer link loss from packet loss and shown the resulting cumulative distribution functions for both path and link losses. We describe an idea of finding coding nodes within a network by identifying the segments of overlapping paths between a source and each of the multicast sinks.

We conclude that although the idea can be extended to larger hierarchical networks, other shortest path algorithms might be preferred over ACO. We sketched a possibility of designing a network aware routing scheme which makes use of a mechanism for exchanging link state information and computing Dijkstra shortest paths trees similar to the one used in OSPF, but also incorporates the construction of

minimum spanning trees.

ACKNOWLEDGMENT

This work is supported by the German Research Foundation (Deutsche Forschungsgemeinschaft – DFG).

REFERENCES

[1] R. Ahlswede, N. Cai, S.-y. R. Li, and R. W. Yeung, "Network Information Flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[2] M. Medard and A. Sprinston, *Network Coding: Fundamentals and Applications*. Academic Press, 2011.

[3] J. Li and Y. Pan, "Research of Network Coding Resources Optimization Based on Ant Colony Optimization ," in *International Conference on Computer Application and System Modeling (ICCASM 2010)*, October 2010.

[4] D. Ghita, H. Nguyen, M. Kurant, K. Argyraki, and P. Thiran, "Netscope: Practical Network Loss Tomography," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2010.

[5] H. X. Nguyen and P. Thiran, "Network Loss Inference with Second Order Statistics of End-to-End Flows," in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '07, 2007, pp. 227–240.

[6] J. Hawkinson and T. Bates, "Guidelines for Creation, Selection, and Registration of an Autonomous System (AS)," RFC 1930 (Best Current Practice), Internet Engineering Task Force, Mar. 1996. [Online]. Available: http://www.ietf.org/rfc/rfc1930.txt

[7] Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4)," RFC 4271 (Draft Standard), Internet Engineering Task Force, Jan. 2006, updated by RFCs 6286, 6608, 6793. [Online]. Available: http://www.ietf.org/rfc/rfc4271.txt

[8] W. Fang and L. Peterson, "Inter-AS Traffic Patterns and Their Implications," vol. 3, 1999, pp. 1859–1868 vol.3.

[9] J. Moy, "OSPF Version 2," RFC 2328, Internet Engineering Task Force, Apr. 1998, updated by RFCs 5709, 6549, 6845, 6860. [Online]. Available: http://www.ietf.org/rfc/rfc2328.txt

[10] K. Lougheed and Y. Rekhter, "Border Gateway Protocol 3 (BGP-3)," RFC 1267, Internet Engineering Task Force, Oct. 1991. [Online]. Available: http://www.ietf.org/rfc/rfc1267.txt

[11] A. Reka and Barabási, "Statistical Mechanics of Complex Networks," *Rev. Mod. Phys.*, vol. 74, pp. 47–97, Jun. 2002.

[12] PlanetLab Europe. [Online]. Available: http://www.planetlab.eu

[13] M. Dorigo, "Optimization, Learning, and Natural Algorithms (in Italian)," Ph.D. dissertation, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy, 1992.

[14] M. Dorigo, V. Maniezzo, and A. Colorni, "The Ant System: Optimization by a Colony of Cooperating Agents," *IEEE Transactions on Systems, MAN, and Cybernetics-Part B*, vol. 26, no. 1, pp. 29–41, 1996.

[15] R. W. Yeung, S.-Y. R. Li, N. Cai, and Z. Zhang, "Network Coding Theory," *Foundation and Trends in Communications and Information Theory*, vol. 2, no. 4 and 5, pp. 241–381, 2005.

[16] O. Graur and W. Henkel, "Steps towards Decentralized Deterministic Network Coding," in *Second International Conference on Artificial Intelligence, Modelling and Simulation (AIMS2014)*, 2014.