Steps towards Decentralized Deterministic Network Coding

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*—Recently, Li and Pan proposed a deterministic network coding resource optimization method built to overcome the issue of severe performance degradation of network coding in the presence of erasures. Considering a multicast scenario, their method relied on the use of Ant Colony Optimization (ACO) to find shortest disjoint paths from a source to each sink and identified the coding nodes as the heads of the overlapping path segments among multiple sinks. Although their technique appears to be suitable for the small artificial topologies on which it has been exemplified, there is no clear study regarding the scalability of the proposed solution for larger Internet-like networks where properties such as clustering are observed. Throughout this paper we point out and provide solutions to several problematic aspects that were not previously brought up for discussion. We conduct extensive network measurements on scale-free networks, deriving a probability function for link erasures and construct a software simulation platform to assess and enhance the practicality of their solution.**

Keywords–deterministic network coding; ant-colony optimization;

I. Introduction

As Ahlswede et al. suggested in [1], it is possible to achieve the upper bound of network flow in a multicast network scenario by applying network coding. There is, however, the problem of how to choose the coding nodes in an optimum way, such that no more than the necessary linear combinations of packets are performed at intermediate nodes in the network. While there has been some work on this issue, the problem of finding the optimum global encoding kernels in the presence of erasures, remains unsolved. Unfortunately, linear network coding, in its standard form, is not robust against packet loss (erasures) and fast topology changes.

There are recent contributions available in literature that show that the same upper bound can be achieved by the use of random linear network coding (RLNC). This, however, requires that all network nodes act as coding nodes and perform random linear combinations of the packets, by multiplying them with coefficients from a large finite field. The disadvantage of RLNC lies in the overhead introduced, since all the nodes need to append to the data packets they produce the random coefficients used in encoding. In order for the max-flow bound to be reached with high probability, it needs to be ensured that the field size chosen is large enough [2]. This significantly increases the complexity of the decoder and the memory requirements.

There has been an innovative approach to deterministic network coding based on the use of Ant Colony Optimization (ACO), as proposed in [3]. The authors describe a promising method of identifying the minimum number of coding nodes in order to achieve the multicast rate by first identifying disjoint paths from the source to each sink and recognizing the overlapping path segments. Nevertheless, there are several critical aspects left uninvestigated in [3] that need consideration before a coding solution that scales well to large wireline networks can be proposed.

Throughout the rest of this paper, our interest lies in extending the idea originating from [3] to real-life large-scale networks, subject to packet loss. Hereto, we have constructed a software simulation platform that takes into account scale-free network parameters such as the preferential attachment of nodes, and link losses which have been derived from actual Internet measurements.

The organization of this paper is as follows: Section II is concerned with the derivation of a link erasure probability function based on measurements we have conducted on the PlanetLab Internet testbed. A description of the simulation platform developed is also provided. Section III summarizes the Ant Colony Optimization (ACO) algorithm in the presence of link erasures and provides an analysis of the ACO convergence behavior for larger Internet-like networks. In Section V, we address several practical aspects previously not considered regarding the actual global encoding kernel (GEK) assignment. Concluding remarks are presented in Section VI.

II. Link Loss in Scale-Free Networks

A. Scale-Free Networks

Unlike the previous probabilistic models for network modeling, Barabasi and Albert [4] took into account the growth of real life networks such as the Internet, and the tendency to form nuclei. They have argued that the probability of a node to be connected to another one is highly dependent on the node's degree. The property of networks to form clusters was termed as preferential attachment. Based on these two

properties, it has been shown that by incrementally adding a new node at each time step and connecting the node to l different nodes already present in the system, such networks evolve asymptotically into *scale-free* networks, where the probability that a node has degree k is given by (1) for γ_{BA}=3 and β_{BA}=0.5. A *scale-free* network is a network for which the degree distribution follows a power law.

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

B. Link Loss in Scale-Free Networks

Although extensive literature is available on packet loss studies in the Internet and ISP networks, there have been only a few studies[1] publicly available on link packet loss. This is attributed to the fact that while end-nodes in a network are easily accessible and end-to-end statistics data can be easily obtained, the accessibility to intermediate nodes such as routers and switches is much more restricted. The problem of inferring link losses from packet loss measurements is quite complex and is tackled by network tomography algorithms. For an accurate evaluation of the deterministic network coding scheme detailed in Section III, we were first concerned with the derivation of a probability distribution function of the link losses, which we intended to use for the generation of erasures in B-A modeled networks.

C. Measurement Setup

We have considered the PlanetLab Europe Internet testbed [5] for our packet loss measurements. We have chosen 226 nodes from PlanetLab Europe which we have considered both as sources and sinks. We have written and deployed scripts to all the nodes for probing and centralized data gathering. The network was queried at multiple instances, using tools such as *ping* and *traceroute*.

One of the aspects that had to be considered was the fact that routers, by definition, have at least two interfaces, with some recently introduced that support up to 160 physical interfaces. Each interface can have multiple IP addresses associated, which results in the same router responding to different ICMP[2] requests under a different alias. Careful consideration had to be paid to this issue, since any inaccuracies in the topology matrix derived would propagate to our findings regarding link packet loss. For collapsing the topological graph of the network we have chosen an IP aliasing resolution technique named *ally*. *Ally* relies on the idea that consecutive packets generated by a router will have consecutive IP IDs, without regard of which source sent the datagrams and for whom they were destined. *Ally* [6] starts by considering all possible pairs of routers in the network. Taking each pair at a time, it generates probe messages that are sent to both nodes and waits for the ICMP replies. Once a reply is received, it sends another probe to the address

[1]to the knowledge of the authors

[2]Internet Control Message Protocol

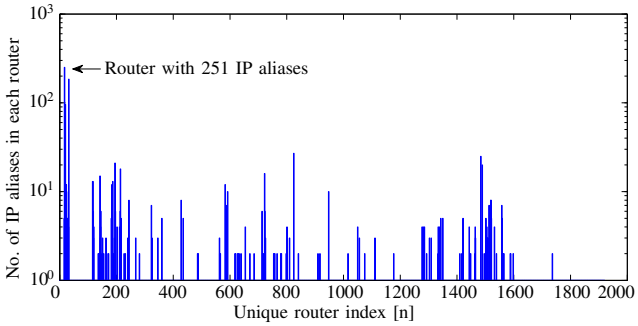

Figure 1. IP aliases

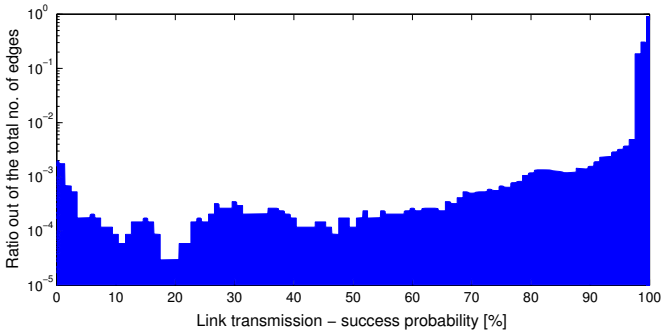

Figure 2. Link transmission – success probability

that replied first. If all three ICMP replies are received in order and close in value, there is a strong indication that the two IP addresses are aliases, i.e., they belong to the same router. For an extensive description of the IP header and the IP ID field, the reader is referred to RFC 791 [7]. The number of routers after the initial tracerouting was found to be 3049, but after the IP alias resolution performed by using the *Ally* technique, the number of unique routers was reduced to 1920. Our results show, as can be seen in Fig. 1, that quite a large fraction of routers have multiple IP aliases. The router with the highest numbers of aliases (251) in the PlanetLab Europe network was identified as being part of NORDUnet (Nordic Infrastructure for Research & Education network), located in Denmark.

The collected measurements have been processed offline, following the method described in [8], [9], namely Netscope. This allowed for the estimation of link packet loss, given path packet loss measurements (end-to-end). Our findings indicate that about 92 % of the links were completely error-free, while a small percentage of the links had 100 % packet loss.

The distribution we have obtained and shown in Fig. 2 was used for assigning link weights in a B-A network, where the network edges were modeled as binary erasure channels.

In Fig. 3, we show the theoretical degree distribution of a B-A modeled network, along with the PlanetLab degree distribution obtained from measurements.

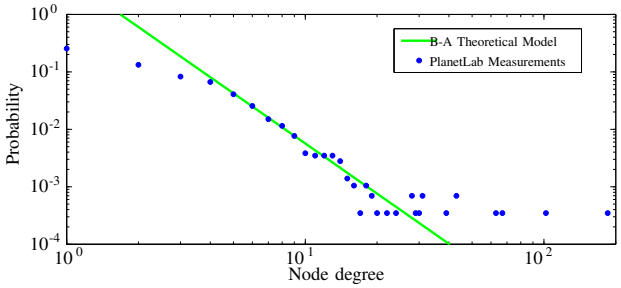

Figure 3. Node degree distribution comparison

III. Ant Colony Optimization

The Ant Colony Optimization Algorithm (ACO) 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 sources. The first ACO algorithm was proposed by Marco Dorigo in his PhD thesis [10].

A. General Description

As real ants walk through a network, looking for food sources, they lay down a pheromone trail which acts as a signal for the other ants. When more ants face a decision on which path to follow, they are more inclined to follow a path that has been previously walked and marked with pheromones, thus reinforcing the trail. In other words, once a branching is reached, the probability that an ant follows a certain edge is directly proportional to the amount of pheromone present on adjacent edges. The strength of the trail decays over time due to diffusion. After a certain amount of time, the shorter paths in the network will tend to have a higher pheromone concentration, since they will be walked more per unit of time, and the pheromone level does not have sufficient time to decay. Finally, all ants will walk the shortest path.

The difference between ACO and other well-known algorithms for finding shortest paths through a network, such as Dijkstra or Bellman-Ford, is that ACO does not require centralized topology information, and due to localized structure of the information required by the ants, it can be applied to de-centralized systems. As part of our task of designing a GEK assignment scheme with minimum overhead, in the presence of link erasures, we are first interested in finding shortest paths through the network in terms of erasure probabilities from one source to multiple destinations. We have based our description of the ACO algorithm on the notations introduced in [11], where a solution for the classical traveling salesman problem (TSP) is provided. We have tailored the proposed solution in [11] to our problem, by loosening some constraints and introducing others. A few initial assumptions are necessary. At the initialization of the algorithm, all ants leave from the colony base. Every ant k has a memory of the current tour, where the tour cost is symbolized by L_k, and every ant is able to recognize whether the node currently visited is a food source or their colony base. Every time an ant walks from one node to another, it adds the label of the node to the memory and makes a decision on which edge to follow, while being subject to the constraint that the new vertex reached by following the respective edge is not already found in its memory. In other words, an ant is not allowed to visit the same vertex twice during one tour. This constraint is necessary to prevent ants from looping. In the case of the TSP problem, the memory of an ant would reset once $L_k = n$, that is, all the vertices have been walked. In our modified version of the ACO, once an ant k reaches a food location, it goes back to the base following the same path it stored in memory, while laying down a pheromone trail that is proportional to the cost L_k of its tour. Once the base is reached, the memory of the ant is reset and the ant starts another tour.

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. Preserving the notations introduced in [10], $\tau_{i,j}(t)$ is used to denote the pheromone level associated with edge $e_{i,j} \in \mathcal{E}$ at time t and m to denote the total number of ants. Initially, all links start with equal pheromone levels. Once an ant has reached a node v_i, it makes a decision on which adjacent vertex to visit, given the pheromone levels of the incident vertices, i.e., based on

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

where $p_{i,j}$ denotes the transition probability of an ant from node v_i to v_j, in the current time instance, and $\eta_{i,j} = \frac{1}{w_{i,j}}$ is a constant representing the *visibility* of the link $e_{i,j}$. This 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. After each new time instance, the pheromone trail on all links is updated according to

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

where $\Delta\tau_{i,j}$ represents the total increment in pheromone level deposited by all the ants on link $e_{i,j}$ in time instance t, defined in (4)

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

Notation $\Delta\tau_{i,j}^{k}$ represents the pheromone quantity per unit of length laid down by the kth ant on the $e_{i,j}$ link, which

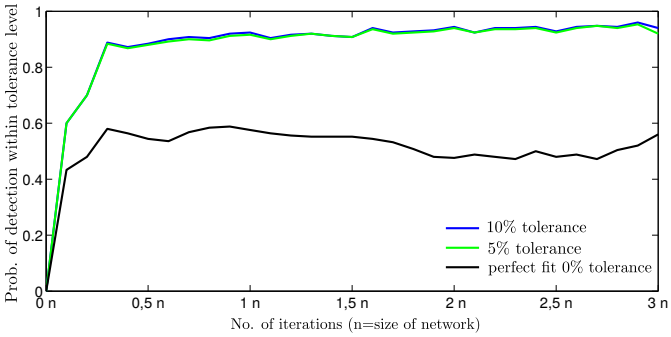

Figure 4. ACO shortest path detection

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 [11], 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 $0 < \rho < 1$.

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

We have implemented the algorithm described above, which is a slightly modified version of the one proposed in [11], and tested it on a subset of the PlanetLab network, where the erasures associated with the edges have been generated using the distribution in Fig. 2. For a parameter setting of $\alpha = 1$, $\beta = 5$, $\rho = 0.7$, and the number of ants equal to n, the size of the network, in more than 90 % of the cases, the shortest paths found by ACO are very close to the ones returned by Dijkstra's algorithm, within a 5 % tolerance limit. For the results illustrated in Fig. 4, the size of the network was kept constant, $n = 156$, and the number of iterations was varied, as a linear function of the network size.

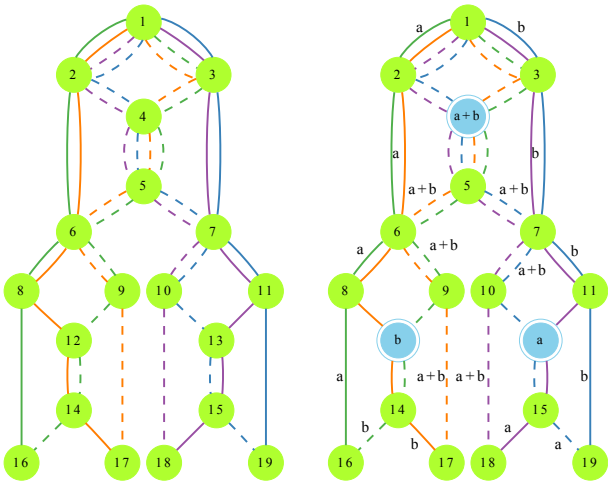

Figure 5. Finding disjoint paths

IV. Finding the Optimum Coding Nodes

Algorithm 1 Finding disjoint paths (taken from [3])

Step 1: Compute the maximum multicast rate h for the given network using the Ford-Fulkerson algorithm.
Step 2: Select a sink node which has not previously been selected and find the shortest path based on ant colony optimization.
Step 3: Block all links of the last shortest path. Find a shortest path from source to the sink based on ant colony optimization. If no paths can be found, go to Step 4, or go to Step 5.
Step 4: Abolish the last path we found, unblock all links of the last path, and block a link of the last path randomly. Define m as the times of backtracking. If still not working, unblock the previous link and block another randomly. If we cannot find $m+1$ disjoint paths, let $m = m+1$, and continue to abolish the last path, until $m+1$ disjoint paths have been found.
Step 5: If h disjoint paths have been found, go to Step 6, or go back to Step 3.
Step 6: Put all h disjoint paths into the set of paths. If all sinks have been selected, go to Step 7, or go back to Step 2.
Step 7: Analyze the set of paths and find the coding nodes.

We show, in Table I, the eight paths, $\mathcal{P}_1 - P_8$, found by the algorithm above for the network in Fig. 5. As previously mentioned, all links have a capacity of one, Node 1 is the source of the multicast, and nodes $16-19$ are the sinks. By carefully checking the table below, the first links of each of the joint path segments are found to be $4 \rightarrow 5$, $12 \rightarrow 14$, and $13 \rightarrow 15$. The coding nodes are the heads of the joint path segments, 4, 12, and 13, respectively. In Table I, the joint path segments are shown in color, the coding nodes are

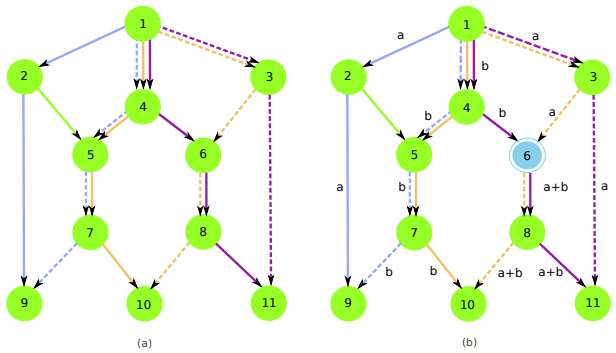

Figure 6. Finding disjoint paths

framed, and the coding links are shown in bold. If two paths, each corresponding to a different sink, start from the source, split and rejoin multiple times, then every overlapping path segment will generate a coding node. Node 1, the source of the multicast does not act as a coding node, thus links $1 \rightarrow 2$ and $1 \rightarrow 3$ are not identified as coding links.

V. Practical Aspects

The identification of the coding nodes in a network does not uniquely specify the GEK assignment for all the edges. For instance, consider the multicast scenario in Fig. 6. Node 1 is the source of the multicast, nodes 9, 10, and 11 are the sinks, and all edges have a capacity of 1. The multicast rate for the network is 2. Assume that, for illustration purposes, the paths returned by the ant colony optimization algorithm are indeed the ones in Fig. 6, two for every pair (source, sink). In this specific case, it can be easily seen that the multicast rate is achievable for all sinks simultaneously only if either Node 5 or Node 6 perform network coding. For the paths chosen in Fig. 6, Node 6 is identified as the optimum coding node. However, this is a necessary but not sufficient condition to guarantee decodability at all sinks. For instance, in Fig 6(b), the symbols a and b have been assigned to the outgoing edges of the source such that all sinks can obtain a system of equations for which the equations are not linearly dependent. If symbol b was sent instead of a on the $1 \rightarrow 3$ edge, sink 11 would be unable to recover both a and b. It follows that it is not sufficient to correctly identify the coding nodes within a network, but also to correctly choose the local encoding matrix $\mathbf{K}_s$ at the source, such that

$$\mathbf{y}_i = \mathbf{x}\mathbf{G}_i \tag{7}$$

has an unique solution for every sink t_i, where $\mathbf{y}_i$ is the vector of received symbols, $\mathbf{x}$ is the vector of transmitted symbols, and the $\omega \times \mathcal{I}n(t_i)$ matrix $\mathbf{G}_i$ holds the global encoding kernels for all edges $e \in \mathcal{I}n(t_i)$, where $\mathcal{I}n(t_i)$ represents the set of incoming edges of sink t_i and $\omega = |\mathcal{I}n(s)|$. The matrix $\mathbf{G}_i$ can be easily derived once the coding nodes have been identified.

The distinct paths found for a sink, although edge disjoint, might still intersect at a node. To understand the problem that arises from this fact, consider Fig. 5. Assume that for the first sink, Node 16, the two edge-disjoint paths cross each other at Node 6. Thus, instead of paths $\mathcal{P}_1$ and $\mathcal{P}_2$ given in Table I, $\mathcal{P}_1 = 1 \rightarrow 2 \rightarrow 6 \rightarrow 9 \rightarrow 12 \rightarrow 14 \rightarrow 16$ and $\mathcal{P}_2 = 1 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 8 \rightarrow 16$. The coding nodes in this situation remain the same, namely 4, 12, and 13. Router 6, in the previous case when the paths did not cross each other, sent noncoded symbol a to router 8 and coded symbol $a + b$ to router 9. If we now consider the situation in which the paths cross each other, we reach a dead end, since router 6 would need to send $a + b$ to router 8 and a to router 9, to reach sink 16, while at the same time sending a on link $6 \rightarrow 8$ and $a + b$ on link $6 \rightarrow 9$. Since the transmissions need to be done simultaneously, this is not possible. Thus, it must be ensured that whenever disjoint paths associated with a sink cross each other, they have to be flipped back. This is one of the critical aspects that the authors of [3] do not consider in their work.

VI. Summary and Conclusion

Network coding, despite its theoretical performance on small error-free network models, when implemented in a network subject to packet loss suffers from severe performance degradation. To overcome this effect, any deterministic code construction needs to be specifically tailored taking into account the lossy nature of the channel.

First, we were interested in developing a software simulation platform to provide aid in the performance assessment of a deterministic network coding scheme, in the presence of erasures. We conducted extensive simulations on a large-scale network, gathered statistics on packet loss and topology information and, given the path losses measured, we derived a probability function for the individual link losses. We proceeded to the simulation of large networks, that exhibited properties such as clustering, which were subject to packet loss.

Several practical aspects left unaddressed in [3] are investigated, concluding that Ant Colony Optimization may be successfully employed in the context of decentralized deterministic network coding, scaling well to larger networks prone to packet loss. The integration of this current technique in the framework of network-coding aware routing protocols is subject of future research. The ant colony optimization may actually be part of a hierarchical network coding and routing scheme, e.g., combined with constructions based on a minimum spanning tree at lower hierarchy levels. For space limitations, we could not describe these aspects in here.

Acknowledgment

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

TABLE I
DISJOINT PATHS

Sinks	First disjoint path	Second disjoint path
Sink 16	$\mathcal{P}_1$: 1→2→6→8→16	$\mathcal{P}_2$: 1→3→ **4** ⇒**5**→6→9→ **12** ⇒**14**→16
Sink 17	$\mathcal{P}_3$: 1→2→6→8→ **12** ⇒**14**→17	$\mathcal{P}_4$: 1→3→ **4** ⇒**5**→6→9→17
Sink 18	$\mathcal{P}_5$: 1→2→ **4** ⇒**5**→7→10→18	$\mathcal{P}_6$: 1→3→7→11→ **13** ⇒**15**→18
Sink 19	$\mathcal{P}_7$: 1→3→7→11→19	$\mathcal{P}_8$: 1→2→ **4** ⇒**5**→7→10 → **13** ⇒ **15**→19

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

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

[6] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP Topologies with Rocketfuel," in *Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '02, 2002, pp. 133–145.

[7] *RFC 791 Internet Protocol - DARPA Internet Programe, Protocol Specification*, Internet Engineering Task Force, September 1981. [Online]. Available: http://tools.ietf.org/html/rfc791

[8] 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.

[9] 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.

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

[11] 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.