A Virtual Network Allocation Algorithm for Reliability Negotiation

Rafael L. Gomes, Luiz F. Bittencourt, Edmundo R. M. Madeira
Institute of Computing (IC), University of Campinas (UNICAMP), Brazil
Email: {rafaellgom,bit,edmundo}@ic.unicamp.br

***Abstract*—The Internet has become a primary means of communication, even though it currently does not guarantee Quality of Service (QoS). In order to tackle with this problem, companies establish Service Level Agreements (SLA) with their providers, including the reliability of services. This paper shows an algorithm for virtual network allocation in a reliability context, where some techniques of disjoint path and reliability calculation are used to deploy the virtual network according to the client's specification. Simulations using real network topologies show the capacity of the algorithm to deploy the virtual network according to the client requirements.**

I. Introduction

Currently, the Internet has become the base for many company services, however with no guarantee of service level. One way to pursue quality of service (QoS) with providers on the Internet is to establish Service Level Agreements, where companies negotiate and specify quality parameters to be achieved. One important parameter that can be negotiated is the network reliability, sometimes called network protection or survivability. Network reliability refers to the reliability of the overall network to provide communication in the event of failure of a component or components in the network.

Usually, Network reliability is based on the establishment of link-disjoint path pairs: the working path (WP) and the backup path (BP). When a link failure occurs the WPs affected by the link failure switch over the traffic to their respective BPS [1]. The disjoint-path based protection effectively addresses the case of a single point of failure [2].

There is a consensus that the Internet needs to be updated, creating the so-called Future Internet [3]. Along with this, the Network Virtualization (NV) approach has arisen as one of the most prominent technologies for the Future Internet. In a nutshell, NV is a technology that enables the deployment of multiple network environments over the same physical infrastructure. The flexibility of virtualization allows the customization of several virtual services, including the virtual topology and virtual network resources.

Each client/company has different needs, and therefore distinct infrastructure requirements. In the same way, providers have different infrastructure levels, each one with its particular features and resources. Tied to this fact, the provider charges more to offer a high quality infrastructure. So, the companies may want to pay the price according to the required infrastructure, where the network reliability is an important characteristic to be considered.

Based on the Future Internet world, many proposals work under this private network infrastructure of each provider to perform the negotiation between providers and clients, according to the client's specification [4]–[6]. In this context, this paper proposes an algorithm for virtual network allocation in a network reliability negotiation context. We want to adapt the virtual network allocation according to the reliability defined by the client, where this reliability can be achieved through a partial disjoint path approach. We restrict the redundancy percentage of the network according to the reliability desired by the client: high redundancy for high reliability, and low redundancy for low reliability. The algorithm generates a virtual network topology that encompasses all the border gateways in the ISP, and calculates the network reliability of this topology. The calculation of network reliability is based on the method proposed by Li et al. [7].

To generate the virtual network topology according to the redundancy restriction, we developed a variation of the method for finding shortest pairs of disjoint paths proposed by Suurballe et al. [8]. The shortest path approach enables the ISP to deploy a virtual network but also saving network resources (mainly bandwidth) for future negotiations.

This paper is organized as follows. Section II details related work regarding to network reliability. Section III describes the proposed algorithm, as well as the steps to define the virtual topology and to calculate its reliability. Section IV presents and explains some results of the developed algorithm, and Section V concludes the paper and presents future work.

II. Related Work

In this section we describe some important works related to disjoint paths and the virtualization approach in a network reliability context.

Urra et al. [1] proposed a protection routing algorithm to minimize the resource consumption taking into account the multi-layer scenario of GMPLS-based networks. In order to share more backup bandwidth (minimizing the resource consumption) a definition of link-disjoint path based on Shared Risk Link Group (SRLG) was made.

Gomes et al. [9] aimed to find the most reliable pair of link disjoint paths. To perform this task, the authors used an algorithm to sequentially obtain the k-shortest length constrained paths (with a defined maximum number of arcs per path). The proposal was evaluated through experiments for randomly generated networks with low connectivity to adapt the experiments to the WDM optical networks context.

Lee et al. [2] studied the path protection issue in a network with multiple, possibly correlated, failures. To address

this problem, the authors developed an algorithm for finding diverse routes (two paths) with minimum joint failure probability.

Pu et al. [10] proposed a variation of Open Shortest Path First (OSPF), called Reliable OSPF (ROSPF) routing protocol, where one primary path and two alternate backup paths are deployed. The authors analyze the reliability of three partially disjoint paths between a pair of nodes, and a 3-D Venn diagram is used to model the overall probability of success of a connection between two nodes.

Rahman et al. [11] formulated the Survivable Virtual Network Embedding (SVNE) problem to incorporate substrate failures in the virtual network embedding problem. In general, the authors remove the assumption that the network is operational all the time and develop a hybrid policy heuristic. The policy is based on a fast re-routing strategy and utilizes a pre-reserved quota for backup on each physical link.

Sun et al. [12] designed a framework for solving the Survivable Virtual Network Mapping (SVNM) problem with resource constraints in a region failure context. The framework addresses two key issues of the SVNM problem: reduce VN mapping cost and minimize the computational complexity.

None of the papers found in the literature focuses on the development of an algorithm to enable the negotiation of the virtual network reliability giving flexibility to the client to pay according to the desired reliability, which is the proposal of this work.

III. Proposal

In the current context of Future Internet, based on network virtualization, the possibility to customize the networks and services provided by the ISP for the clients arises. Many works [4]–[6] use this flexibility of virtualization to bring to the client the desired service, adapting its quality according to the client's definition. Consequently, the client pays a price relative to the desired quality of the service.

In this paper, we propose an algorithm to generate a virtual network topology according to the reliability requested by the client. The network reliability is the capacity of the network to continue operating even in case of arbitrary failure of any component.

When a client wants a virtual network from an ISP, usually he/she wants to pass through the ISP to reach another domain or specific nodes inside the domain. Thus, the ISP needs to allocate the virtual network between the client's gateway node and the desired destination nodes, which could be the border gateways or the final destination. To generate the topology, we propose an algorithm to define the topology according to redundancy restrictions by applying a variation of the method for finding shortest pairs of disjoint paths proposed by Suurballe et al. [8]. With the topology already defined, we compute the network reliability from the method proposed by Li et al. [7], where few adaptations are made to the virtual network deployment in the ISP context.

The virtual network topology generation algorithm and the network reliability calculation approach are explained in Sections III-A and III-B, respectively.

A. Algorithm to Generate the Virtual Network

In this section we describe the proposed algorithm, where Algorithm 1 overviews the idea applied to generate the relative disjoint paths between a node and a set of destinations.

Let $G = (V, E)$ be a weighted directed graph representing the topology of the ISP, where $V = \{v_1, ..., v_n\}$ is a set of n vertices (or nodes) and $E = \{e_1, ..., e_m\}$ is a set of m directed edges (or links). Let v_s be the vertex/node in G where the client connects to the ISP network, and D be a set of k designated destination nodes. Let $w_{u,v}$ be the cost of edge/link $e_{u,v} \in E$, so $e_{u,v}$ is the link between node u and node v, where u and v are distinct nodes. Let $\wp$ represent a quantity close to infinity. And, let $d(s, u)$ be the cost of the shortest path from node v_s to node v_u in the shortest path tree rooted at v_s.

In the proposed algorithm, the weight of each link (u, v) is defined as in Equation (1), where $R_{(u,v)}$ is the current available resource capacity of the link and R_r is the amount of bandwidth requested by the client. For example, if the client requests 10 Mbps and the link has only 8Mbps available, its weight will be 3.

$$w_{u,v} = 1 + 10\left(1 - \frac{R_{(u,v)}}{R_r}\right) \tag{1}$$

This strategy aims to prioritize the use of links that meet the demand requested by the client, where links without available capacity ($R_{(u,v)} = 0$) are disregarded in the algorithm.

Initially, we find the shortest tree T_1 with node s as root by running Dijkstra's algorithm ($Dijkstra(Graph,\ Node)$ function), counting the number of times that each link is used to be part of a path. This enables identification of links that, if kept in the topology, will save bandwidth. The T_1 tree contains, for every node u, a shortest path from s to each node in D.

In Algorithm 1, line 2 makes P_1 to contain the links belonging to the paths from s to nodes D in T_1, which are extracted by $Edges(Graph,\ Set\ of\ Nodes)$ function. Line 3 assigns to e the number of links to be updated to generate the redundancy in the topology. e is calculated as a percentage from the number of links in P_1 according to the desired redundancy, called p ($0 \leqslant p \leqslant 1$).

After running the Dijkstra's algorithm, Algorithm 1 updates the cost of each link in the graph, creating a new graph $G' = (V', E')$ (lines 5 to 20). First, the algorithm orders the links according to their usage to reach the nodes in D, i.e., the number of times each link is part of all found paths ($OrderedSet(Set\ of\ Edges)$, line 3).

To create G' the algorithm replaces the cost $w_{u,v}$ of the first e links of P_1 by $w'_{u,v} = \wp$, and the remaining links by $w'_{u,v} = 0$. The other links in G' are updated with $w'_{u,v} = w_{u,v} - d(s, v) + d(s, u)$. This process is performed by lines 5 to 20 of Algorithm 1.

In the next step, the algorithm finds the shortest path T_2 in the graph with the updated cost of the links by running Dijkstra's algorithm, and it assigns to P_2 the set of links belonging to the paths from s to nodes D in T_2 (lines 21 and 22, respectively).

Finishing that process, the algorithm merges the paths of P_1 and P_2 (line 23), i.e., the algorithm creates a graph adding the links and nodes that exist in both sets. This process is performed by the $Merge(Set\ of\ Edge, Set\ of\ Edge)$ function. The resulting graph is the final topology G_f that contains relative disjoint paths between the node s and the nodes of D.

Algorithm 1 Generate a Disjoint Virtual Network Topology

```
1: Tree T_1 = Dijkstra(G, s);
2: Path P_1 = Edges(T_1, D);
3: int e = p * |P_1|;
4: Path P_1 = OrderedSet(P_1);
5: for all edge i ∈ G do
6:     for all edge j ∈ P_1 do
7:         node u = j.from;
8:         node v = j.to;
9:         if (i == j) then
10:            if (e > 0) then
11:                w'_{u,v} = ℘ ;
12:                e = e − 1 ;
13:            else
14:                w'_{u,v} = 0;
15:            end if
16:        else
17:            w'_{u,v} = w_{u,v} − d(s, v) + d(s, u);
18:        end if
19:    end for
20: end for
21: Tree T_2 = Dijkstra(G', s);
22: Path P_2 = Edges(T_2, D);
23: Graph G_f = MergePaths(P_1, P_2)
```

Basically, the algorithm constructs an initial spanning tree in the first steps, and after that, with the update of link cost, it seeks alternative paths by adding new edges to the initial spanning tree. This addition of links is limited by the redundancy factor defined by the client. However, there may exist cases where there is no alternative path to reach the desired node. Here, the $\wp$ is used to avoid the usage of edges already being used, but without discarding them as an option, thus forcing the algorithm to seek for alternative paths to reach the desired nodes in the second call of Dijkstra algorithm (line 21). If no alternative path exists, the algorithm uses the path already known due to non existence of a disjoint path. Therefore, there may be cases where the desired redundancy can not be met due to the lack of alternative paths, where different redundancy factors can generate similar topologies.

By running the proposed algorithm of relative disjoint paths, a virtual topology according to the redundancy desired by the client is constructed, taking advantage of the flexibility offered by the network virtualization. This flexibility enables the bandwidth usage reduction (since it can use part of the edges in the network) or/and the provision of reliability (since it enhances the routing possibilities) according to the client requirement.

B. The Network Reliability Calculation

After the topology definition, a network reliability method is applied, which evaluates the topology considering that each component (nodes and edges) has a reliability value (i.e., a probability of the component to be operational). The applied method is based on the reference [7] to model the network states and the most probable states of the network. We modified the method in order the final reliability to include only the communication between the client's node and all the destinations.

It is assumed that, at any time, elements fail randomly and independently of one another, according to certain known probabilities. Therefore, each component can be in one of two states: operational or failed. The failure of a link means it is removed from the network, while the failure of a node means that the node and all the links related to it are removed. So, a network with n unreliable components can be in 2^n states.

Let p_i be the probability of a component i to be operational, and q_i the probability to be failed, with $q_i = 1 - p_i$. We denote the possible states of the network by S_k, where $k = 1, ..., 2^n$. The probability of the system to be on state S_k is given by Equation (2), where $T_i(S_k)$ is 0 if the component i is operational in state S_k, and 1 otherwise.

$$P(S_k) = \prod_{i=1}^{n} p_i (q_i/p_i)^{T_i(S_k)} \tag{2}$$

Assuming that $p_i > q_i \forall i$, i.e., the probability of a component to be operational is higher than the probability to be failed, the most probable state is S_1, which corresponds to no failures.

1) Most Probable Network States: When the number of unreliable components is large, this method requires exponential time to run, since there are 2^n network states. So, we need a faster method to evaluate the reliability of large networks.

An alternative approach is to enumerate the most probable states of the network, providing a good approximation of the network reliability without the space explosion of 2^n states. We need to estimate the number of most probable states in a network with n equally unreliable components, i.e., $p = p_i \forall i$. Let f_L be the probability associated with the states when we consider up to L failures, as shown in Equation 3.

$$f_L = \sum_{k=0}^{L} \binom{n}{k} p^{n-k} q^k \tag{3}$$

With the values of f_L, n, and p, we use Algorithm 2 to calculate L', which is in turn used in Equation (4) to define the number of most probable states m.

Algorithm 2 Most Probable States

```
1: k = 1
2: sum = p^n
3: while (f_L > sum) do
4:     sum = sum + C(n,k) p^{n−k} q^k
5:     k++
6: end while
7: L' = k − 1
```

$$m = \begin{cases} \binom{n}{L'} & if\ f_L = f_{L'} \\ \frac{f_L - f_{L'-1}}{p^{n-L'}q^{L'}} + \sum_{k=0}^{L'-1}\binom{n}{k} & if\ f_L < f_{L'} \end{cases} \quad (4)$$

After that, we apply the Order algorithm [7] to define the set of most probable states. The idea behind this algorithm is that the power set of failed components is a subset of the all possible network states.

Let $A = S_a, S_b, ..., S_c$ denote a partially ordered set (*poset*) of failure states such that $P(S_a) \geq P(S_b) \geq ... \geq P(S_l)$. The Order algorithm uses the following operations:

1) Append (denoted $||$): $A \,||\, i = \bigcup_{a \in A}\{a \cup \{i\}\}$
2) Insert (denoted $A \rightarrow B$): is the *poset* $A \cup B$
3) Select (denoted $T_m(A)$): is the *poset* that contains the first m elements of A.

First we find i', which is the smallest integer such that $2^{i'} \geq m$, i. e., $i' \geq \frac{\log n}{\log 2}$. After that, we use i' in Algorithm 3 to generate the set of most probable states. At the end, the A_n *poset* contains, in decreasing order, the m most probable states.

Algorithm 3 Order Algorithm

```
S_1 = ∅
A_0 = {S_1}
for i = 1 → i' do
    B_{i-1} = A_{i-1} || i
    A_i = B_{i-1} → A_{i-1}
end for
A_{i'+1} = T_m(A_i)
for i = i' + 1 → m do
    B_{i-1} = A_{i-1} || i
    A_i = B_{i-1} → A_{i-1}
    A_i = T_m(A_i)
end for
```

2) Reliability Bounds: In this paper, we consider the probability/capacity of the network link from the client's gateway (root node) to all destination nodes, called C. So, the reliability of the network can be measured according to Equation (5), where $C(S_k)$ is 1 if the network can link the root node to all the destinations in state S_k, and 0 otherwise.

$$C = \sum_{k=1}^{2_n} P(S_k)C(S_k) \quad (5)$$

Applying the most probable states approach, we can find the Upper ($Comm_{Up}$) and Lower ($Comm_{Low}$) bounds of the network performance, using the Equations 6 and 7 respectively [7].

$$Comm_{Low} = \sum_{k=1}^{m} P(S_k)C(S_k) + \left(1 - \sum_{k=1}^{m} P(S_k)\right) C(S_{2^n}) \quad (6)$$

$$Comm_{Up} = \sum_{k=1}^{m} P(S_k)C(S_k) + \left(1 - \sum_{k=1}^{m} P(S_k)\right) C(S_1) \quad (7)$$

The Upper and Lower bounds assume that the performance for S_{m+1} to S_{2^n} is the same as in S_1 (no failures state) and S_{2^n} (failure of all components state), respectively. We define the probability $Comm$, as the mean value of $Comm_{Low}$ and $Comm_{Up}$.

IV. Experiments

To evaluate the proposal we applied the algorithm in two distinct real network topologies with different sizes and link options. The experiments on each scenario are detailed in Sections IV-A and IV-B. The goal was to evaluate the capacity of the proposal to generate the virtual topology according to the client's needs.

We work in the context of deploying a virtual network inside an ISP, so we want to generate a topology that connects the border gateway of the client (the root node) with specific border nodes in the ISP.

The metrics considered in the experiments are: the total bandwidth (Bw) allocated for the virtual network, the probability of the network be connected ($Comm$), the time to generate the virtual topology ($tTopology$), the time to run the Order algorithm ($tOrderAlg$), and the time to calculate the probability of the network be connected ($tComm$).

In all scenarios the virtual network bandwidth requested was 10 Mbps, while the reliability of each component was set to 99%, where we seek the number of probable states for one failure in the network, i.e., $L = 1$ according to the process described in Section III-B1. The results regarding running times are presented in milliseconds (ms) with a 95% confidence interval.

A. Internet2 Scenario

This scenario uses the Internet2 Network Topology[1], illustrated in the Figure 1. The Internet2 Network is a community of USA and international leaders in research, academia, industry, and government who collaborate via innovative technologies. The Internet2 Network offers a range of network services tailored to the needs of research and education.

In this scenario, we defined the root node as Chicago and the destination nodes as New York, Atlanta, Seattle, and Los Angeles. In the topology we utilized the nodes that are IP and/or redundant OADM to represent the nodes that can be used to deploy a virtual network, since the remaining ones are basically optical regeneration nodes. Table I presents results for all metrics regarding to the Internet2 network.

A 0% of redundancy basically generates a spanning tree between the root and the destination nodes. It uses less bandwidth but decreases the probability of communication between the nodes (the reliability of the network). In this trade-off, a higher reliability generates a higher Bw allocation and price.

[1] http://www.internet2.edu/

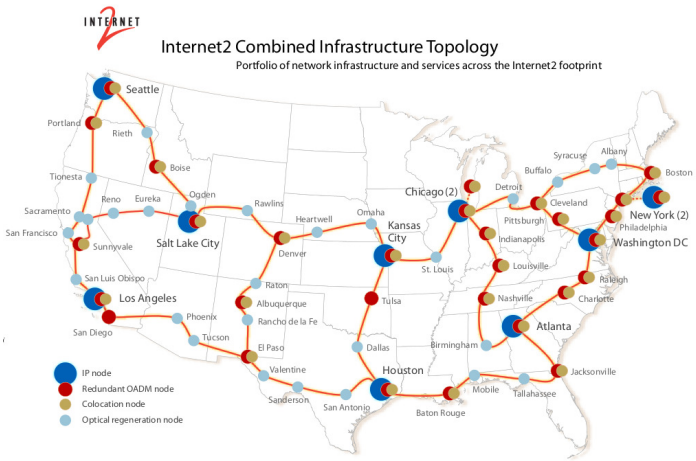

Fig. 1. Topology of Internet2 Network

TABLE I. RESULTS FOR THE INTERNET2 NETWORK

Metric	0%	25%	50%	75%	100%
Bw	150	210	260	280	280
$Comm$	0.74	0.85	0.86	0.86	0.86
$tTopology$	1.74	3.59	3.61	3.63	3.65
$tOrderAlg$	773.58	2399.47	5143.05	6737.19	6725.84
$tComm$	64.22	220.03	493.58	747.83	748.67
Total Time	839.54	2623.09	5640.23	7488.65	7478.17

The $Comm$ value increases as larger is the number of links allocated (which reflects in the Bw metric), also increasing the total time. So, the algorithm enables the client to chose the level of reliability, which is tied to a price, where higher levels of reliability will result in higher prices.

Figure 2 shows a graphic representing the total running time of the algorithm. The time spent to generate the most probable states of the network comes mostly from the Order algorithm (described in Section III-B1). Since it is based on the number of unreliable components, its running time increases exponentially with the network size.

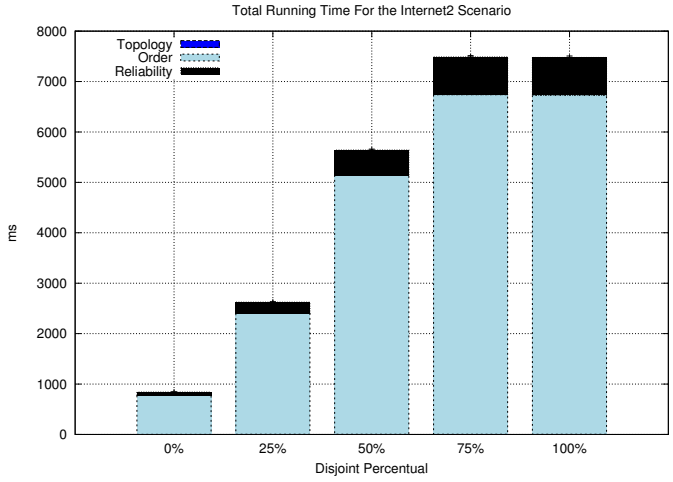

Fig. 2. Total runtime in Internet2 network

In Figure 2, we observe that the time to generate the topology is negligible. Following the information presented in Table I, the time for the case without redundancy (0%) is smaller. This occurs because when redundancy is higher than zero, the Dijkstra's algorithm is run twice to generate the alternative paths.

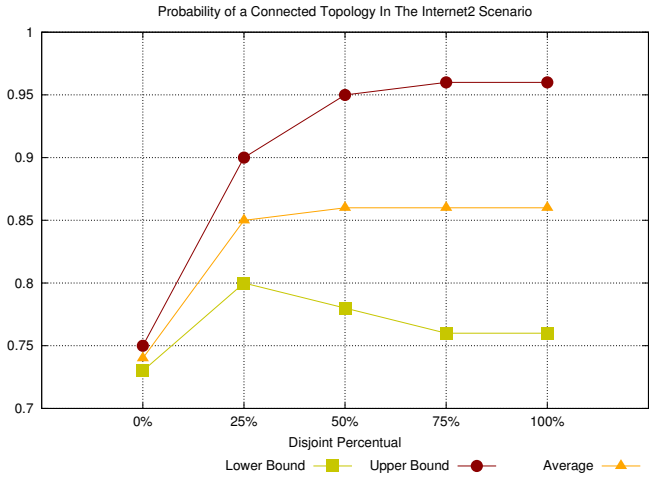

Fig. 3. $Comm$ in Internet2 network

Figure 3 shows the Mean, Lower, and Upper bound of communication probability ($Comm$) between the root node and the destination nodes in the Internet2 scenario. The $Comm$ value for no redundancy is much lower when compared with full redundancy applying the proposed algorithm, where even the lower bound for 100% redundancy is higher than the upper bound for 0%.

In the Internet2 scenario we can see the situation mentioned in Section III-A, where there is no other path to reach the desired node and different redundancy factor generates similar topologies. In this case the 75% and 100% factors generated similar networks.

Based on the results, we can see in cases where the ISP wants a 0.85 of $Comm$ in a network with desired destinations, it needs to allocate a topology with only 50% of redundancy, saving bandwidth.

The links chosen to be part of the network in the cases of no redundancy (0%), half redundancy (50%) and full redundancy (100%) are shown in Figure 4. In Figure 4 the orange circle represents the root node, where the orange squares illustrate the destination nodes. Regarding to the links, the black lines represent the links allocated in the no redundancy case, the blue lines symbolize the additional links allocated in the half redundancy situation and the yellow lines depict the supplementary links for the full redundancy case.

Due to the greedy behavior of the proposed algorithm, the links allocated in the lower redundancy cases are also allocated in higher allocation situations, i.e, all the links used in no redundancy case are also used in the half/full redundancy case.

B. GEANT Scenario

This scenario uses the GEANT Network Topology[2], illustrated in Figure 5. GEANT is the high bandwidth pan-European research and education backbone that interconnects

[2]http://www.geant.net/

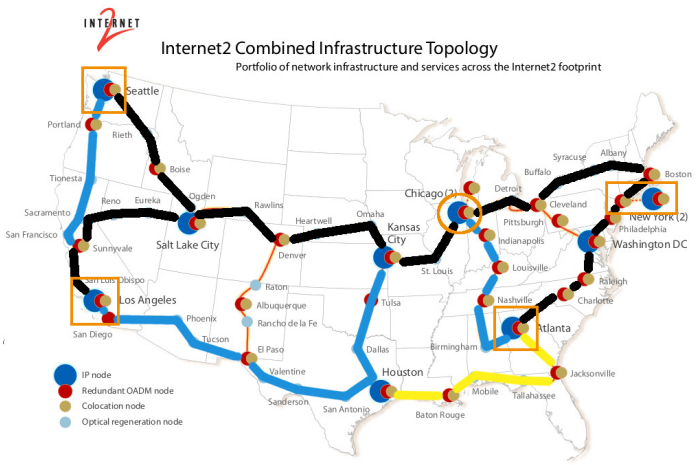

Fig. 4. Links allocated in Internet2 network

National Research and Education Networks (NRENs) across Europe and provides worldwide connectivity through links with other regional networks.

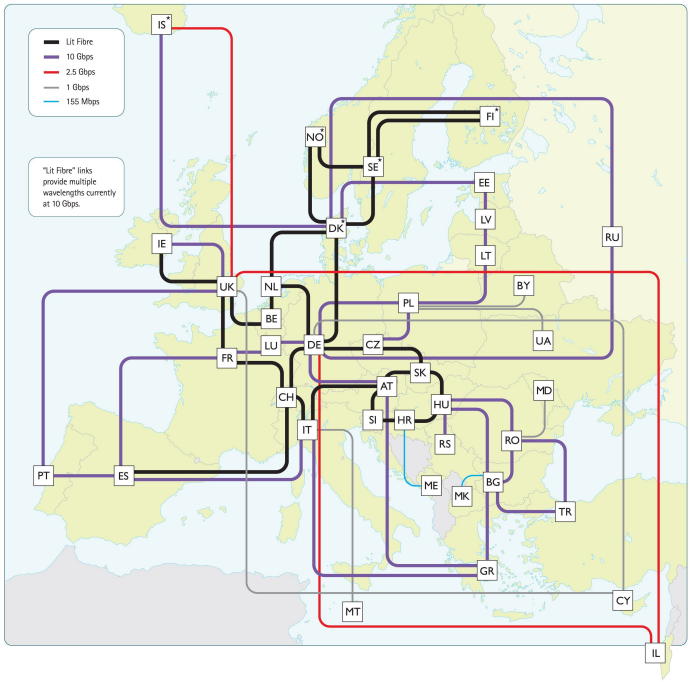

Fig. 5. Topology of GEANT network

In this scenario, we set the root node as Germany (DE) and the destination nodes as Spain (ES), United Kingdom (UK), Bulgaria (BG), Netherlands (NL) and France (FR). This set of nodes were chosen because they encompass the main global connectivity border gateways of GEANT network.

Regarding the network size, and hence the number of link options, the GEANT network is larger than Internet2 network. So, in this scenario situations of lack of options hardly happen, fact that improves the capacity of the algorithm to find alternative paths. Table II summarizes the results for the GEANT scenario.

TABLE II. RESULTS FOR THE GEANT NETWORK

Metric	0%	25%	50%	75%	100%
Bw	100	130	170	190	200
$Comm$	0.82	0.87	0.91	0.90	0.93
$tTopology$	5.22	10.50	10.58	10.60	10.62
$tOrderAlg$	106.26	288.09	801.26	1286.37	1543.32
$tComm$	15.45	33.42	85.79	121.66	159.53
Total Time	126.93	332.01	897.63	1418.64	1713.47

The information presented in Table II follows the same behavior from the Internet2 scenario: a trade-off between network reliability and bandwidth allocation (used links). However, the higher number of links/paths options makes clearer the difference between the redundancy levels, as shown in Figure 6.

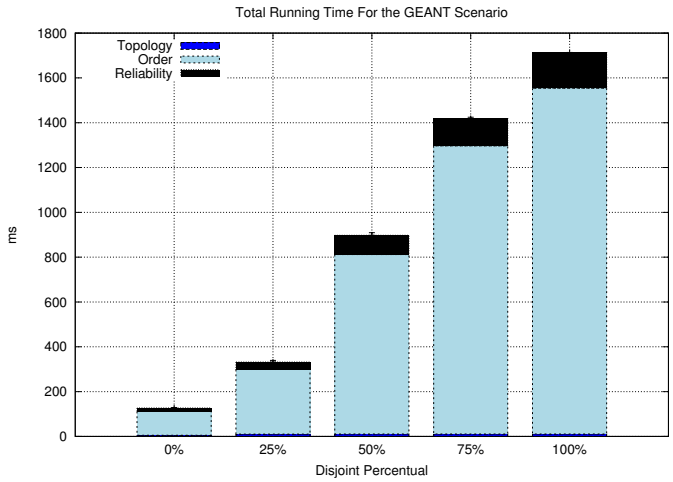

Fig. 6. Total runtime in GEANT Network

As in the previous scenario, the time for topology generation is very small, where the more expensive processing is the Order algorithm, following the network size increase.

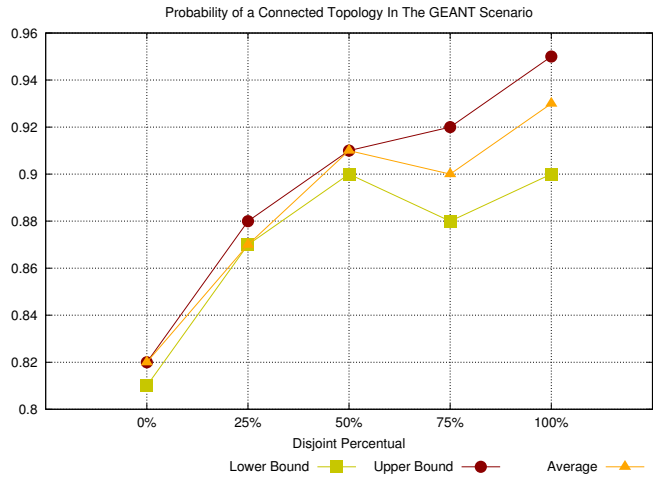

Fig. 7. $Comm$ in GEANT Network

Regarding to the communication probability values, presented in Figure 7, the no redundancy approach has the smallest values of $Comm$, where even the worst case (lower bound) of the algorithm usage is better than its best case (upper bound).

The rounded values for $Comm$ with 50% and 75% are the same, however the difference between upper and lower bounds in 75% is larger, which represents a bigger instability and variation when compared with 50%. The full redundancy approach (100%) has the highest $Comm$ values due to many alternative paths in the GEANT topology.

The links chosen to be part of the network in the cases of no redundancy (0%), half redundancy (50%) and full redundancy (100%) are shown in Figure 8. The orange circle symbolizes the root node, where the orange squares represent the destination nodes. Regarding to the links, the green lines symbolize the links allocated in the no redundancy case, the blue lines portray the additional links allocated in the half redundancy situation and the yellow lines represent the supplementary links for the full redundancy situation.

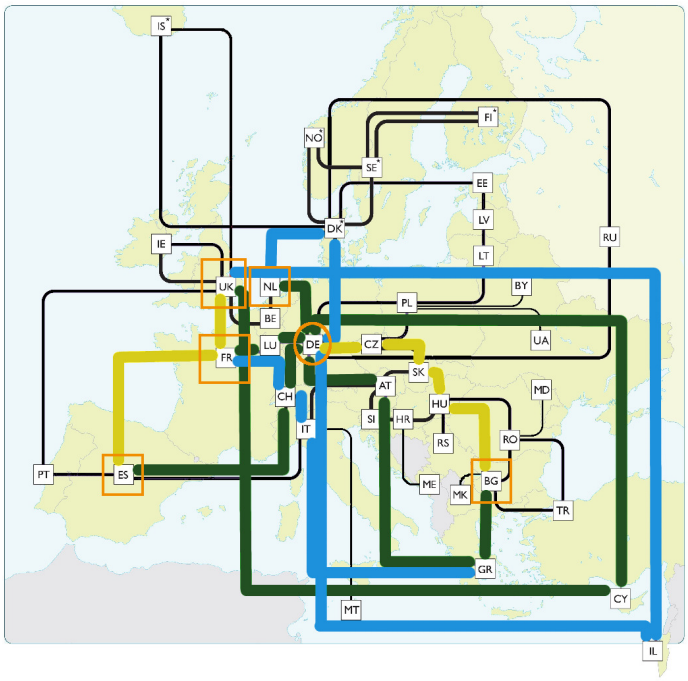

Fig. 8. Links Allocated in GEANT Network

C. Final Remarks

The showed experiments suggest that the proposed algorithm can meet the topology definition according to the reliability specifications set by the client, when feasible. The existing tradeoff between network reliability and the network size stays clear in the results, where the more paths exist between the root node and the destinations, the greater the network reliability.

At the same way, we realize another tradeoff that includes the virtual network size and the time to calculate the network reliability. This encompasses the most probable states definition and the probability of communication calculation. Figures 2 and 6 indicate an exponential behavior of runtime, due this tradeoff.

This fact explains why the runtime for GEANT scenario is lower that the Internet2 scenario. Despite the original topology of GEANT substrate network is bigger than Internet2 substrate network, the number of links and nodes allocated to be part of the virtual network in the GEANT scenario is smaller than in the Internet2 scenario, consequently the runtime too, attesting the tradeoff cited before.

V. Conclusion

This paper presented an algorithm for virtual network allocation in a network reliability negotiation context, where the reliability can be achieved through a partial disjoint path approach. The proposed algorithm defines a topology based on relative disjoint paths, which can reach different reliability levels. With that, the provider can adapt the network according to the client's requirement.

Experiments were run with real network topologies, showing the capacity of the proposal to generate the virtual topology according to the client's definition within acceptable running time. The tradeoffs between runtime, network reliability and the network size were identified and analyzed.

The use of different reliability calculation approaches to evaluate the defined topology is being explored. As future work, we intend to extend the algorithm to use a multicriteria topology definition considering an ISP policy context.

Acknowledgment

The authors would like to thank FAPESP (2012/04945-7), FAEPEX/Unicamp, CNPq, and RNP for the financial support.

References

[1] A. Urra, E. Calle, and J. Marzo, "Enhanced multi-layer protection in multi-service gmpls networks," in *Global Telecommunications Conference, IEEE GLOBECOM '05*, vol. 1, 2005.

[2] H.-W. Lee, E. Modiano, and K. Lee, "Diverse routing in networks with probabilistic failures," *IEEE/ACM Transactions on Networking*, vol. 18, no. 6, pp. 1895 –1907, 2010.

[3] M. Femminella, R. Francescangeli, G. Reali, J. W. Lee, and H. Schulzrinne, "An enabling platform for autonomic management of the future internet," *IEEE Network*, vol. 25, no. 6, pp. 24–32, 2011.

[4] R. L. Gomes, L. F. Bittencourt, and E. R. M. Madeira, "A generic sla negotiation protocol for virtualized environments," in *Proceedings of 18th IEEE International Conference On Networks (ICON 2012)*, 2012.

[5] H. Pouyllau and R. Douville, "End-to-end qos negotiation in network federations," *3rd IEEE/IFIP International Workshop on Bandwidth on Demand and Federation Economics*, 2010.

[6] F. Zaheer, J. Xiao, and R. Boutaba, "Multi-provider service negotiation and contracting in network virtualization," *12th IEEE/IFIP Network Operations and Management Symposium*, 2010.

[7] V. Li and J. Silvester, "Performance analysis of networks with unreliable components," *IEEE Transactions on Communications*, vol. 32, no. 10, pp. 1105 – 1110, oct 1984.

[8] J. W. Suurballe and R. E. Tarjan, "A quick method for finding shortest pairs of disjoint paths," *Networks*, vol. 14, no. 2, pp. 325–336, 1984.

[9] T. Gomes and J. Craveirinha, "Efficient calculation of the most reliable pair of link disjoint paths in telecommunication networks," *European Journal of Operational Research*, vol. 181, no. 3, 2007.

[10] J. Pu, E. Manning, and G. Shoja, "Routing reliability analysis of partially disjoint paths," in *IEEE Pacific Rim Conference on Communications, Computers and signal Processing.*, vol. 1, 2001.

[11] M. Rahman, I. Aib, and R. Boutaba, "Survivable virtual network embedding," in *NETWORKING*, 2010, vol. 6091, pp. 40 – 52.

[12] G. Sun, H. Di, H. Yu, L. Li, and V. Anand, "The framework and algorithms for the survivable mapping of virtual network onto a substrate network," *IETE Technical Review*, vol. 28, no. 5, 2011.