Differentiating Link State Advertizements to Optimize Control Overhead in Overlay Networks

Mathieu Bouet, Julien Boite, Jérémie Leguay and Vania Conan
Thales Communications & Security, Paris, France

***Abstract*—Routing in overlay networks typically involves engineering an overlay topology on top of the Internet to balance traffic along overlay paths so that quality and/or resilience of delivered services are improved. It can be used to reduce latency for delay-sensitive applications. It then consists in selecting, for any pair of nodes, an intermediate overlay node which reduces the latency on this one-hop overlay path against the latency on the direct overlay path between them. In this paper, we propose to optimize the overhead generated by the overlay route computation mechanism by introducing a differentiation between the nodes that are highly used as relay and those that are not. Our approach relies on disseminating at a high frequency the link states with the identified sub-set of nodes and at a lower frequency all the link states. We conduct large experimentations on PlanetLab to evaluate the trade-off between the performances in terms of RTT gain and the reduction of the control overhead compared to the state of the art.**

I. Introduction

Over the last decade, overlay networks have gained in interest from the research community. While their development began with application-specific Peer-to-Peer [1] services or Content Delivery Networks [2], their reach has been extended in the past recent years to enhance distributed Internet application back-ends or interconnect private networks with QoS and resilience support. In RON [3], authors show that overlay networks provide a better level of resilience for networks and services than the best-effort Internet and the routing protocol (BGP) it relies on. From this, other studies emerged, using overlay networks to improve the quality of services [4]–[6], and more generally for the purpose of providing value-added services with Service Overlay Networks (SON [7] / NG-SON [8]).

In this paper, we focus on the application of overlay routing to improve latency, e.g. for delivering delay-sensitive services such as VoIP or video broadcasting with high quality. It consists in engineering an overlay topology on top of the Internet to balance traffic along overlay paths (i.e. aggregates of physical nodes and links) with lower latency than the Internet path. There is a room for this kind of improvement due to the *Triangle Inequality Violations* (TIV) that occur in the Internet [9]. Fig. 1 illustrates this TIV phenomenon that takes place when the latency on the direct path between two overlay nodes A and C is higher than the sum of latencies on the two segments of the two-hop overlay path going through an intermediate node B.

Improving latency thus consists in finding overlay paths going through (at least) one intermediate node that provides a lower latency than the direct path between any pair of nodes. This

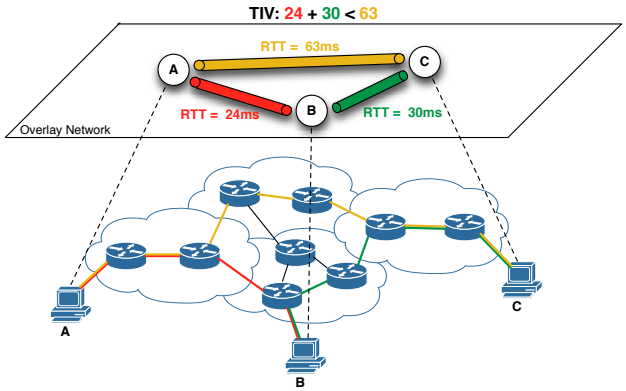

Fig. 1. A Triangle Inequality Violation (TIV) occurs when an overlay path through a relay node B offers a lower latency than on a direct path between two overlay nodes A and C.

implies: (i) monitoring latency between pairs of overlay nodes, (ii) disseminating link-state advertisements between nodes, including the monitored latency, and (iii) computing, for each node, the best two-hop path, which reduces latency, to reach each other node.

However, we show via experimentations on PlanetLab [10] that only a small sub-set of the overlay nodes are used as intermediate nodes in the majority of the two-hop paths that reduce the latency. From this observation, we propose to optimize the overhead generated by the overlay route computation mechanism by introducing a differentiation between the nodes that are highly used as relay and those that are not. Our approach relies on disseminating at a high frequency the link states with the identified sub-set of highly used nodes and at a lower frequency all the link states. This technique uses a notification mechanism to select from time to time the sub-set of nodes that are most likely to be used as relay. We conducted large and numerous experimentations on PlanetLab to evaluate the trade-off between the performances in terms of RTT gain and the reduction of the control overhead compared to the state of the art.

The remaining part of this paper is organized as follows. Section II motivates our idea of differentiating link state advertisement frequency. In Section III, we present our proposal and detail how we adapt it to the state of the art overlay routing mechanism. In Section IV, we evaluate our proposal on PlanetLab experimentations and compare it to the state of the art. Section V deals with related work. Finally, Section VI concludes this paper and mentions future work.

II. Motivations

With overlay routing, we use an intermediate relay node to reach a destination when this two-hop overlay path provides a better latency than the direct path. This situation takes place because of the Triangle Inequality Violations (TIV) occurring in the Internet [9].

The key factor motivating our proposal is relative to the observation of *how many times a node is used as relay node in the two-hop overlay paths*. We thus run experimentations on 49 PlanetLab nodes [10] and compute the load of nodes, i.e. how many times each node is used as intermediate node to improve the latency between two other nodes.

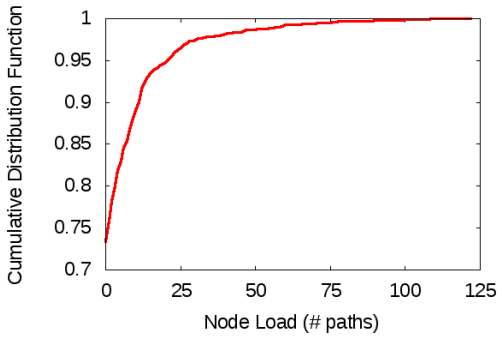

Fig. 2. Relayed communications per node in a network constituted of 49 PlanetLab nodes.

Fig. 2 represents the Cumulative Distribution Function (CDF) of the load experienced by nodes for an experiment with 49 nodes. The load of a node corresponds to the number of times it has been utilized as relay by two nodes that want to communicate. We observe that 5% of nodes are used as a next hop for more than 21 destinations, which is a very high value given the fact that 73.4% of nodes are never used as intermediate hop. We clearly see that some nodes attract traffic[1].

We base our proposal on this remarkable phenomenon. Indeed, the quality of the links between the sub-set of nodes used frequently as relays and all the nodes of the overlay are much more relevant to compute two-hop paths than the quality of the other links. We thus propose to introduce two different frequencies in the process of link state dissemination, one high frequency for the link states that are highly used to relay traffic and one low frequency for the whole link states. This approach enables to reduce the overhead generated by the overlay routing mechanism while assuring very good performance and reaction time.

III. Our Proposal

The goal of an overlay routing algorithm is to find the optimal route for all pairs of nodes in the network with as little per-node communication as possible. In the recent years, proposals have been made to reduce the overhead generated by the mechanisms necessary to compute paths in overlay network (see related work in Section V).

We choose the overlay routing mechanism proposed in [12] to evaluate our proposal of differentiating frequencies for the dissemination of link states in the overlay. Indeed, in [12] authors proposed a quorum-based approach that reduces the dissemination overhead to $\mathcal{O}(n\sqrt{n})$ while ensuring that best paths can still be discovered and maintaining a good level of resilience with regards to nodes or links failures. This proposal aims at finding, if it exists, a better two-hops route in the overlay than the direct path. In our case, we consider latency as the criteria to counter TIV issues.

Basically, overlay nodes are organized in a grid. Fig. 3(a) shows the grid quorums of nodes A and B. Each node i is assigned a set of Rendez-Vous (RDV) servers R_i composed of nodes on the same row and column (in grey in the figure). i is thus a RDV client for the RDV servers R_i. The grid organization ensures that these sets are constructed such that every pair of nodes share at least one RDV server (Fig. 3(a)). In this discussion, we assume that all links are bidirectional with identical cost[2]. This construction provides two important properties: first, every node pair share a RDV server, because every column and row intersect. This technique guarantees that at least one node (two if there is a perfect grid) is able to compute the best two-hops path for a given pair of nodes. Second, the traffic and computation load that the use of RDV servers induces is equally distributed among the nodes in the network. Hence, each node plays both roles of RDV client and server in the dissemination process.

This approach reduces the dissemination overhead. Indeed, instead of sending its link state information to all the overlay nodes, one node only reports to its RDV servers. Thanks to the grid organization, each node has maximum $2 * \sqrt{n}$ RDV servers. The dissemination overhead is thus equal to $\mathcal{O}(n\sqrt{n})$ instead of $\mathcal{O}(n^2)$ with flooding like RON [3], n being the number of overlay nodes.

The construction of the RDV server sets using the grid quorum results in every node knowing the best two-hops path to every other node in the overlay. We propose to introduce a differentiated update frequency in the dissemination of the link states in order to optimize the overhead.

The quality of the direct paths between overlay nodes is highly heterogeneous. Some paths offer a good delay while others are quite unusable for delay-sensitive applications. This second class of paths can be in general out-performed by two-hops paths. However, we show in Section II that a small sub-set of overlay nodes serves as relay, that is they are recommended by RDV servers, for a large portion of the two-hops paths. The link states of these highly recommended nodes with other nodes of the network are thus more important to compute recommendations than the others. They may be used at each computation iteration while some link states may rarely be used.

[1]This is due to inter-domain policy violations [11] caused by overlay routing. Some relay nodes used to forward overlay traffic bypass inter-domain policies defined by operators. This explains their attractiveness.

[2]Since we observed that the monitored RTT remains nearly equal in both directions, the one-way delay on an overlay link could be approximated to *RTT/2*.

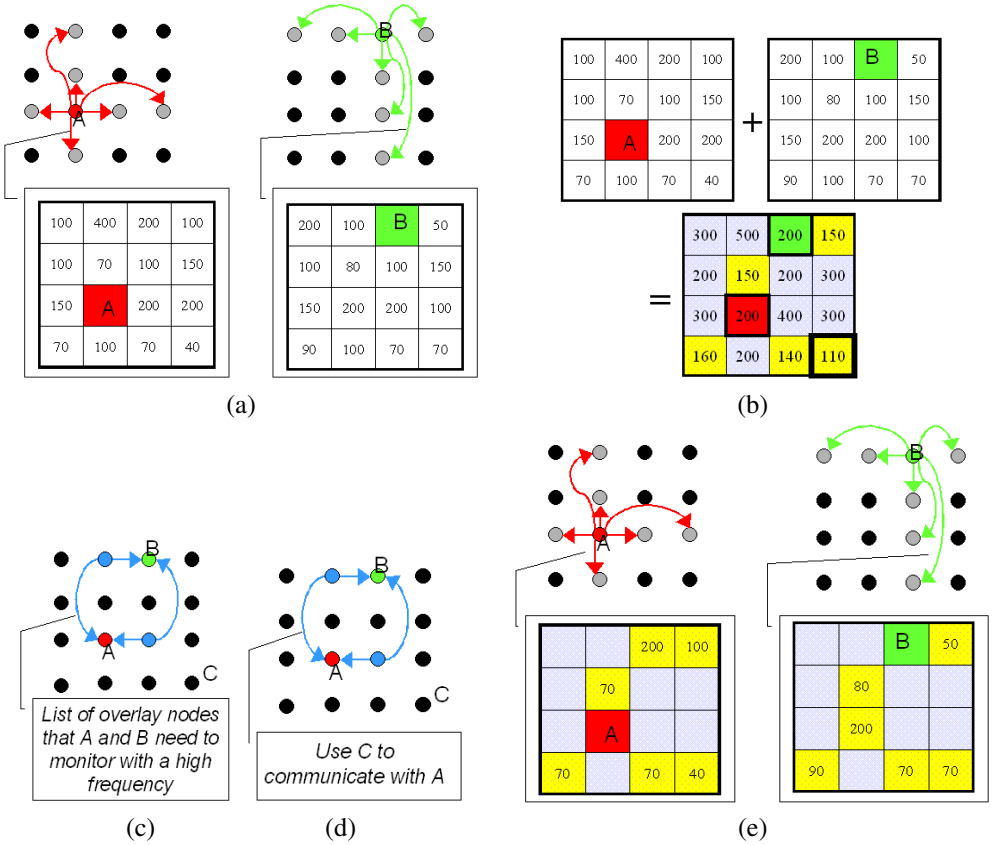

Fig. 3. Dissemination steps: (a) RDV clients send all their link-states (here containing RTTs) to their RDV servers, (b) RDV servers compute recommendations and select nodes to be monitored at a higher frequency, (c) RDV servers send notifications to their RDV clients, (d) RDV servers send recommendations to their RDV clients, (e) RDV clients send part of their link-states to their RDV servers.

To benefit from this, we propose to adapt the dissemination phase to reduce overhead while maintaining a fair delay reduction with two-hop paths. The basic dissemination is performed every T_i iterations, that is at a frequency $f_{flood.} = \frac{1}{T_i}$, in order for RDV servers to receive complete link states. We consider that the frequency of the dissemination phase is 1, that is $f_{flood.} + f_{opt.} = 1$. Therefore, the iterations in-between are done on a sub-set of the link states at a frequency $f_{opt.} = 1 - f_{flood.} = \frac{T_i - 1}{T_i}$.
As described in Fig. 3, every T_i iterations, the dissemination process is in four steps:

- *Step A-1*: RDV clients send complete link-state information to their RDV servers, as shown in Fig. 3(a).
- *Step A-2*: RDV servers receive link state information from their clients. They select the m more relevant nodes for each of their RDV clients, according to a selection threshold. Fig. 3(b) shows in yellow the m best relay nodes in term of delay.
- *Step A-3*: RDV servers send the list of the required link states to their RDV clients (Fig. 3(c)). In this example, the list for A and B is constituted by the yellow nodes. This type of messages is named *notifications* since a RDV server notifies its RDV clients of the link state it would like to receive more frequently.
- *Step A-4*: RDV servers send path recommendations they computed to their RDV clients (Fig. 3(d)).

The dissemination iterations in-between follow a three-steps process on a sub-set of the link states:

- *Step B-1*: RDV clients send the requested sub-sets of link-state information to their RDV servers (Fig. 3(e)). This sub-set has been notified to each RDV client during the *Step A-2*. It will remain the same until a new round is performed with the complete link states at frequency $f_{flood.} = \frac{1}{T_i}$.
- *Step B-2*: RDV servers compute path recommendations for each client and to each destination among these clients with the information on the link states they received.
- *Step B-3*: RDV servers send the path recommendations they computed to their RDV clients (Fig. 3(d)).

This differentiated dissemination mechanism enables to reduce the quantity of link states sent. At a frequency $f_{flood.} = \frac{1}{T_i}$ the RDV clients send their link state with all the nodes of the overlay network to their RDV servers (Step A-1), while at a frequency $f_{opt.} = \frac{T_i - 1}{T_i}$ they send only a sub-set of them. The selection threshold and the frequency have an impact on the gain since two-hop paths may not be found because of partial information, but also on the quantity of overhead generated by the notification mechanism. Their values have to be adjusted

to obtain a good trade-off between the relative performances and the reduction of the overhead.

IV. Evaluation

In this section, we present the results of the evaluation of our proposal compared to the state of the art. The goal of our experimentations is to evaluate the trade-off between the reduction of the dissemination overhead and the reduction of RTT gain offered by two-hop paths. Indeed, differentiating link states dissemination may lead to fewer recommendations, and hence smaller RTT gain on two-hops paths. The adjustment variables are the frequency of complete dissemination and the threshold of link state selection.

A. PlanetLab Experiments Parameters

We implemented in *Java* both the state of the art grid mechanism [12] , i.e. *without optimization*, and our proposal. Overlay nodes should know each others presence. As specified in RON [3], this can be done in a static way (e.g. each node has the list of all overlay nodes locally stored in a file), or dynamically with a bootstrap mechanism coupled to frequent announcements. Since we do not aim at evaluating nodes fail-over processes, we choose the static approach for our implementation.

Monitoring must be performed periodically to gather performance metrics and/or detect nodes or links failures. Our goal in this paper is to improve latency with two-hops overlay routing. We thus monitor Round-Trip-Time (RTT) on overlay links. If TCP-based methods as well as the UDP-based prober proposed in [3] could also be used, we simply perform *pings* between each pair of nodes to estimate the RTT on each overlay link. More precisely, we consider the average RTT over 3 consecutive *ping* results. Each node periodically probes other nodes simultaneously (separate threads) and results are stored/updated in a *Link-State Database* (LSDB).

We deployed our Java executable on 49 PlanetLab [10] nodes among Europe and ran several experiments for each combination of parameters.

Table I lists the parameters used for the experiments. We

TABLE I
PlanetLab experiments parameters.

# nodes	49
Monitoring period	8 sec.
Dissemination period (*Dp*)	10 sec.
# dissemination rounds (*r*)	61 ([0:60])
Duration (*r* × *Dp*)	61 × 10 = 610 sec.

use a dissemination period of 10 seconds because we aim at developing an overlay routing scheme able to react at this timescale. The monitoring period is lower (8 seconds) to ensure that new measurements are available between two dissemination rounds.

At the end of each dissemination round during an experiment, each node writes into files information regarding:

- Global results for the basic quorum-based approach such as the number of link states sent/received, of recommendations sent/received, and the difference of RTT between the direct path and this two-hops path (later called *gain*).
- The same information but with the optimized mechanism plus the number of notifications sent/received. The two mechanisms run in parallel in order to have consistent results in their comparison.

At the end of each experiment, we retrieve those files and generate results presented in the following sections.

B. Experiments Results

First, we evaluate the impact of both the selection threshold S and the frequency of complete dissemination $f_{flood.}$ on the overhead of the different steps of the process compared to the state of the art, that is without optimization. The frequency $f_{flood.} = \frac{1}{T_i}$ corresponds to the dissemination of complete link state information every T_i iterations, i.e. steps A-1 to A-4 that include notifications by RDV servers to their RDV clients. The others iterations are done with only a subset of the link state information (those that were required by the RDV servers), Steps B-1 to B-3, at a frequency $f_{opt.} = 1 - f_{flood.}$.

Fig. 4 presents the overhead generated by the exchange of link state information (or Link State Advertisement - LSA) between RDV clients and their RDV servers. The overhead is equal without optimization and with a frequency $ffood = 1$ since the complete LSAs are sent at each iteration. For a frequency $f_{flood.}$ and a selection threshold S, the LSA overhead is equal to $f_{flood.} * fullOverhead + (1 - f_{flood.}) * S * fullOverhead$. Therefore, it increases with the frequency of complete LSA exchange and with the threshold for the selection of nodes.

Fig. 5 shows the overhead generated by the notifications (Step A-3). The notifications are messages that enable the RDV servers to request specific link states from their RDV clients for the dissemination iterations that will follow this step. Indeed, the size of the sub-set of link states targeted by the notifications depends on the threshold of selection S. Note that notifications for a frequency $f_{flood.} = 1$ or a threshold $S = 100\%$ are useless since the complete LSAs are sent at each iterations. The overhead of the notifications increases linearly with selection threshold and depends on the frequency of their emission, that is on $f_{flood.}$.

Fig. 6 presents the overhead generated by the exchange of recommendations between the RDV servers and their RDV clients. Without optimization, the overhead is roughly equal to 11000 bytes. We can observe that the frequency of complete LSA exchange, when inferior to 1, has no impact on the overhead, that is on the quantity of recommendations sent. It means that the set of nodes used as relay on the two-hops paths stays consistent through time. Basically, the same nodes serve as relay. The selection threshold influences the recommendation overhead, hence the quantity of recommendations sent. However, it converges after 43% towards the overhead with no optimization. It also means that the recommendations are computed on a small set of link states. A sub-set of 43%

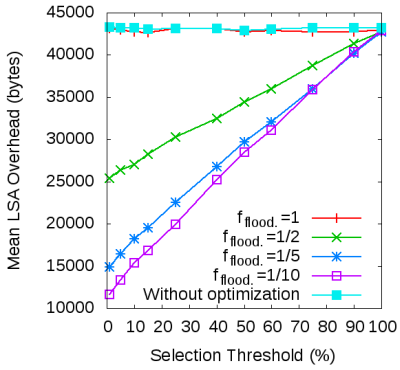

Fig. 4. Link state dissemination overhead per iteration for one node.

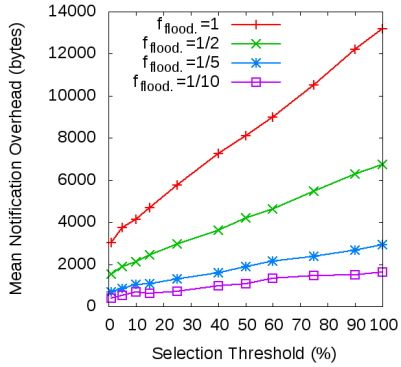

Fig. 5. Notifications dissemination overhead per iteration for one node.

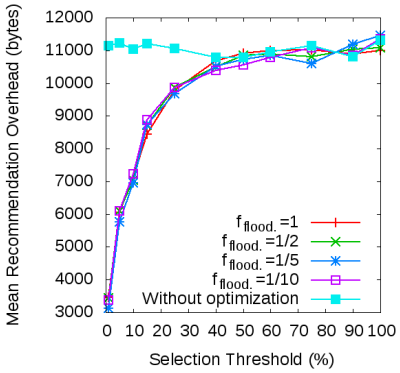

Fig. 6. Recommendations dissemination overhead per iteration for one node.

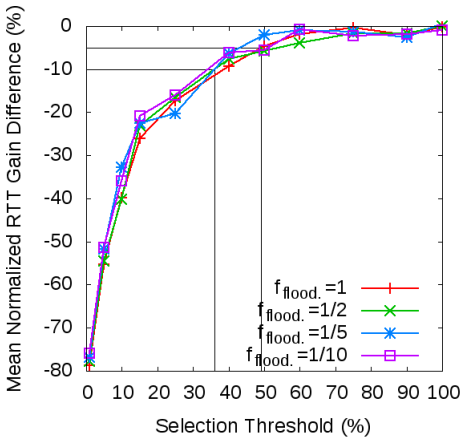

Fig. 7. Normalized difference between the RTT gain without optimization and with optimization.

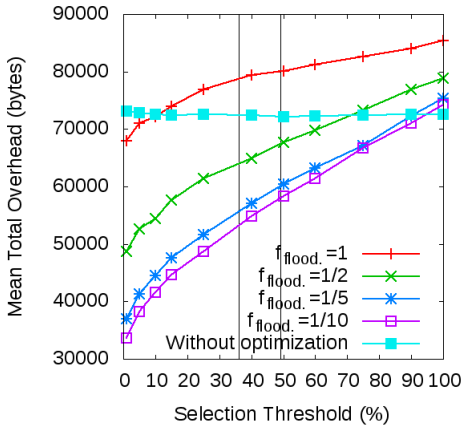

Fig. 8. Total monitoring and dissemination overhead per iteration for one node.

of the link states enables to find out almost all the two-hops paths that improve latency.

We now evaluate the trade-off between minimizing the whole dissemination overhead, by means of link states selection and frequency of complete LSA exchange, and the loss of latency gain compared to the state of the art approach without optimization. To quantify the gain, we compute, at the end of each dissemination round and for each destination where a relay node is used, the difference between the latency on the direct path and the latency through the selected two-hops path. We refer to this difference as the *gain* provided when using two-hops overlay routes. For destinations where the direct path is used, the value of gain is 0. For each node, we average the values of gain over all destinations at each dissemination round. We then compute the ratio between this average gain and the average of latencies measured onto the direct paths to obtain a global percentage value of gain. Finally, we compute in percentage the difference between the gain obtained without optimization and the gain obtained with our mechanism.

Fig. 7 presents this gain difference. When it equals 0 it means that the RTT gain of the two-hops paths recommended by the mechanism without optimization is the same as the one with the proposed mechanism. The same phenomenon observed in Fig. 6 for the quantity of recommendations sent can be seen here. Indeed, the performances of the optimized mechanism tends towards the performances of the state of the art mechanism when the selection threshold increases. In addition, for a given selection threshold, the RTT gain of the optimized mechanism is not function of the frequency at which complete LSAs are exchanged for the considered topology. The reason is that, as seen before, only a stable sub-set of the nodes serve as relay. We can observe that the gain difference is smaller than 10% for a selection threshold greater than 36% and smaller than 5% for 49%.

Finally, Fig. 8 enables to evaluate the total overhead of the proposed mechanism. This total overhead is composed of the overhead generated by the monitoring process and the optimized dissemination process that includes notifications. The state of the art mechanism generates per node and per iteration roughly 72500 bytes. The overhead with a frequency $f_{flood.} = 1$ is larger because notifications are sent at each iteration in addition to complete LSAs. As it has been observed, the overhead for link state dissemination (Fig. 4) and recommendation exchange (Fig. 6) is decreased compared to

the one of the mechanism without optimization. An additional overhead is introduced by notifications exchange, but it remains small (maximum 6300 bytes) compared to LSA exchange (maximum 43000 bytes) and recommendation exchange (maximum 11000 bytes). Selecting a sub-set of the link states and disseminating them with a differentiated frequency thus enable to reduce the total overhead. For example, for a selection threshold of 36%, it is reduced by 10.4%, 20% and 24.1% for a frequency of full dissemination of $1/2$, $1/5$, and $1/10$ respectively.

V. Related Work

Resilient Overlay Network (RON) is an end-user overlay network where overlay nodes aggressively probe their peers to detect links failures faster than the underlying Internet routing protocol (BGP) and choose another path that satisfies the QoS requirements [3]. However, it does not scale up well and supports only one routing metric. QoS-Aware Routing in Overlay Networks (QRON) proposes to add QoS service provisioning functionalities to RON utilizing an hierarchical organization, that is clustering in terms of network distance [4]. Nevertheless, paths are not always optimal and reliability is not completely assured. Bandwidth-Aware Routing in Overlay Networks (BARON) improves RON utilizing capacity between end hosts to identify viable overlay paths and pre-compute them [5]. However, while reducing monitoring overhead, this on-demand approach introduces latency. MCQoS [6] is a vector-based overlay routing mechanism that supports multiple QoS constraints. This on-demand approach provides a convergence time comparable to the one of clustered techniques such as QRON, but suffers from a very high re-stabilization time as the number of overlay nodes increases. Finally, [12] is also based on RON, but it considerably reduces its link state advertisement overhead and thus increases its scalability using a grid quorum technique while ensuring that the best paths can be found and maintaining a good level of resilience to nodes or links failures.

However, the approaches based on RON generate a large amount of overhead to monitor link states and disseminate this information to other nodes, which is actually $\mathcal{O}(n^2)$ with n being the number of overlay nodes. [13] address the scalability issue by pruning overlay links that share multiple underlay links. This approach requires to know the underlaying network structure. This strong assumption is out of our scope since the networks that connect the overlay nodes are considered as black boxes. The quorum-based approach [12] allows reducing the dissemination overhead to $\mathcal{O}(n\sqrt{n})$. The authors demonstrate that this is the minimal complexity to find out the best two-hop routes. We chose this technique to evaluate our proposal since the objective is to lower the overhead.

VI. Conclusion

Routing in overlay network offers benefits such as improving resilience and performances of Internet services and applications. In this paper, we have outlined the fact that a small sub-set of the overlay nodes are used as relay nodes by a majority of two-hop overlay paths. Based on this observation, we have proposed to optimize the overhead generated by an overlay route computation mechanism by introducing a differentiation between the nodes that are highly used as relay and those that are not. Our approach consists in disseminating at a high frequency the link states with the identified sub-set of nodes highly used as relays and at a lower frequency all the link states. The large number of experimentations on PlanetLab we conducted show the trade-off between the performances in terms of RTT gain and the reduction of the control overhead compared to the state of the art. The selection threshold and the frequency have an impact on the gain since two-hop paths may not be found because of partial information. They also have an impact on the quantity of overhead generated by the notification mechanism. Their values have to be adjusted to obtain a good trade-off between the relative performances and the reduction of the overhead. Our results show that the overhead can be substantially decreased while keeping a very good level of RTT gain in the two-hop paths. Future works thus include elaborating a mechanism that enables to dynamically adapt both the selection threshold and the frequency with respect to the dynamicity and the stability of the overlay network.

References

[1] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A survey and comparison of peer-to-peer overlay network schemes," *Communications Surveys & Tutorials, IEEE*, vol. 7, no. 2, pp. 72–93, 2005.

[2] H. Yin, X. Liu, G. Min, and C. Lin, "Content delivery networks: A bridge between emerging applications and future ip networks," *Network, IEEE*, vol. 24, no. 4, pp. 52–56, 2010.

[3] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP '01)*, 2001, pp. 131–145.

[4] Z. Li and P. Mohapatra, "QRON: Qos-aware routing in overlay networks," *Selected Areas in Communications, IEEE Journal on*, vol. 22, no. 1, pp. 29–40, 2004.

[5] S.-J. Lee, S. Banerjee, P. Sharma, P. Yalagandula, and S. Basu, "Bandwidth-aware routing in overlay networks," in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, 2008, pp. 1732–1740.

[6] A. Mondal, P. Sharma, S. Banerjee, and A. Kuzmanovic, "Supporting application network flows with multiple QoS constraints," in *Quality of Service, 2009. IWQoS. 17th International Workshop on*, 2009, pp. 1–9.

[7] Z. Duan, Z. L. Zhang, and Y. T. Hou, "Service overlay networks: SLAs, QoS, and bandwidth provisioning," *IEEE/ACM Transactions on Networking*, vol. 11, no. 6, pp. 870–883, 2003.

[8] IEEE-SA, "IEEE Standard for the Functional Architecture of Next Generation Service Overlay Networks," *IEEE Std 1903-2011*, 2011.

[9] C. Lumezanu, R. Baden, N. Spring, and B. Bhattacharjee, "Triangle inequality variations in the internet," in *Proceedings of the 9th ACM SIGCOMM conference on Internet Measurement Conference (IMC '09)*, 2009, pp. 177–183.

[10] PlanetLab. PlanetLab Europe website. [Online]. Available: www.planet-lab.eu

[11] S. Seetharaman and M. Ammar, "Characterizing and mitigating inter-domain policy violations in overlay routes," in *Network Protocols, 2006. ICNP'06. Proceedings of the 2006 14th IEEE International Conference on*, 2006, pp. 259–268.

[12] D. Sontag, Y. Zhang, A. Phanishayee, D. G. Andersen, and D. Karger, "Scaling all-pairs overlay routing," in *Proceedings of the 5th international conference on Emerging Networking Experiments and Technologies (CoNEXT '09)*, 2009, pp. 145–156.

[13] A. Nakao, L. Peterson, and A. Bavier, "Scalable routing overlay networks," *SIGOPS Oper. Syst. Rev.*, vol. 40, no. 1, pp. 49–61, Jan. 2006.