An Extension and Cooperation Mechanism for Heterogeneous Overlay Networks

Vincenzo Ciancaglini[12][⋆], Luigi Liquori[1],
Giang Ngo Hoang[12], and Petar Maksimović[13] [⋆⋆]

[1] Institut National de Recherche en Informatique et Automatique, France
[2] Hanoi University of Science and Technology, Vietnam
[3] Mathematical Institute of the Serbian Academy of Sciences and Arts, Serbia
Email: First.Last@inria.fr

Abstract. In real-world peer-to-peer applications, the scalability of data lookup is heavily affected by network artifacts. A common solution to improve scalability, robustness and security is to increase the local properties of nodes, by clustering them together. This paper presents a framework which allows for the development of distributed applications on top of interconnected overlay network. Here, message routing between overlays is accomplished by using co-located nodes, i.e. nodes belonging to more than one overlay network at the same time. These co-located nodes serve as distributed gateways, enabling the routing of requests across overlays, while keeping overlay maintenance operations local. The protocol has been evaluated via simulations and client deployment, showing that the ability, of reaching the totality of the overlays in a federated configuration can be preserved even with the simplest routing, proving the feasibility of federated overlay configurations.
Keywords. Peer-to-peer, overlay networks, interoperability.

1 Introduction and Related Work

Context. Overlay networks, structured and unstructured alike, have been broadly recognized as one viable solution for the implementation of various distributed applications: Distributed Hash Tables (DHTs), application-level multicast protocols, distributed object lookup services, file systems, etc. Although most of the protocols designed during the recent years have the theoretical properties of scalability, fault tolerance and handling of dynamic topologies, studies such as [10] have shown that, under real networking conditions, overlays are still vulnerable to severe performance degradation once deployed on a larger scale. The general problem on which we focus concerns the viability of deploying networks consisting of multiple smaller structured overlays, rather than one global overlay, which can interact with each other, thus exploiting locality and diverse topologies: being able to reduce the scope of overlay maintenance to a smaller diameter can benefit performance and fault tolerance, while having a mechanism allowing nodes to reach overlays other than their own facilitates the extensibility and flexibility of their design. The same mechanism could then also be exploited for the cooperation of existing overlay networks.

[⋆] Corresponding author.
[⋆⋆] Work partially supported by the Serbian Ministry of Education and Science (projects ON 174026 and III 044006).

Related work. Efficient cooperation between heterogeneous overlays has served as an inspiration to a number of research efforts, which include concepts such as *cooperation via gateway* (in [5, 6]), *cooperation via super-overlay* (in [11, 15]), *cooperation via hierarchy* (in [8, 9, 16]), *merging of overlays* (in [6, 7, 14]), and *hybrid overlay systems* (in [4, 13]). These efforts and concepts have been described and contrasted to our approach in more detail in [17], while here we only mention that we share their common idea of increasing locality in the network, and have the most features in common with the 'cooperation via hierarchy' approach.

Synapse. In [12], we have introduced a protocol, which we have named *Synapse*, capable of interconnecting heterogeneous overlays through utilization of co-located nodes as distributed gateways. This protocol was further explored and improved in [17], resulting in the *Synapse Framework*, which consists of the related Synapse protocol, software architecture, libraries in the OverSim simulator [3], and a peer client written in Java. This framework allows for a flexible implementation of different federated topologies, as well as the application of various routing strategies across these topologies, depending on the application scenario in question. In this workshop paper, we present a first evaluation of the Synapse protocol, accomplished via extensive simulations in OverSim, and deployment on the Grid'5000 platform of one of its possible routing strategies, namely 1-random-walk, on a topology comprising a cluster of heterogeneous structured Chord and Kademlia overlays. From the obtained results, we gain several insights concerning the design of such systems.

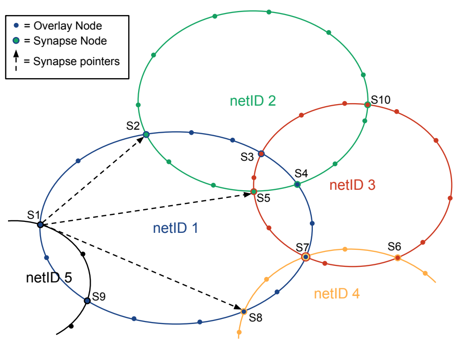
Summary. The remainder of the paper is structured as follows: in Section 2 we provide a brief description of the Synapse framework, as well as the routing strategy used for the simulations. Next, in Section 3 we present and discuss the results obtained from simulations performed in the OverSim simulator, and from the deployment of Synapse on the Grid'5000 platform. Finally, in Section 4 we give our conclusions and discuss further work. All of the details concerning the Synapse framework, as well as the full results of simulations and experiments performed are available at [2].

2 The Synapse Framework

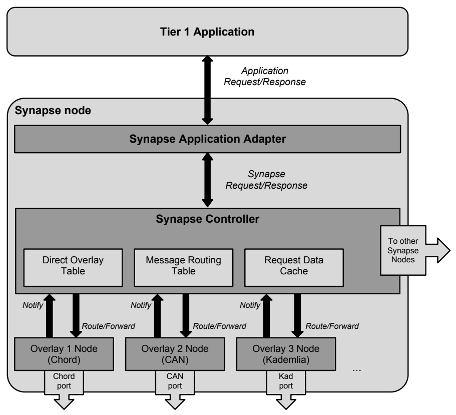
In this section, we briefly present the inter-overlay cooperation protocol which is part of the Synapse Framework, and the specific routing strategy implemented in our tests. Due to space restrictions, here we only provide an outline, while for a more articulate system description, we refer the reader to [17].

Synapse network topology. The Synapse Framework enables the creation of a network made of multiple overlays, each identified by a unique `netID`, capable of interacting with each other. Each overlay can exploit a different topology, different data indexing schemes, and different maintenance mechanisms.

Cooperation and message routing between overlays is achieved via co-located nodes which are connected to two or more networks at the same time, and can perform requests on behalf of other nodes. We will refer to these nodes as *gateway nodes*. Also, we will be referring to nodes which are aware of the Synapse protocol as *synapse nodes*. Finally, as the inter-routing of messages requires data other than the overlay-specific one to be exchanged between nodes, there are feasible scenarios in which, due to the interconnection of existing overlays (e.g. the Mainline DHT of BitTorrent, in [1]),



(a) Overlays with gateway nodes



(b) Structure of a synapse node

Fig. 1: Inter-overlay topologies and structure of synapse nodes

there can be nodes which are unable to understand the additional data embedded in the protocol messages. We refer to such nodes as *legacy nodes*.

In the Synapse Framework, synapse nodes and gateway nodes form an unstructured overlay, with each synapse node keeping a neighborhood of gateway nodes pointing to foreign overlays; this is used to exchange inter-routing requests and other information between the synapse nodes and gateway nodes. An example of a possible Synapse topology made up of 5 overlays, interconnected via a number of gateway nodes, is shown in Figure 1a.

Synapse node architecture. The architecture of a synapse node is shown in Figure 1b. A node usually maintains different instances of virtual overlay nodes, one for each overlay, and additional data structures to handle the inter-routing, namely a *Direct Overlay Table (DOT)* with pointers to gateway nodes arranged by overlay, a *Message Routing Table* to keep track of ongoing requests, and a *Request Data Cache*. Synapse and gateway nodes share the same architecture, with the only difference being in the number of overlays the node is connected to. This makes it possible to dynamically adjust the connectivity of each overlay, by increasing the number of overlays that nodes connect to. The effect is achieved via *social-based primitives*, i.e. invitation messages issued to gateway node candidates, asking them to increase their connectivity, and join a specific overlay.

Routing in Synapse. Whereas legacy nodes can only perform requests in the overlay they are connected to, synapse nodes are capable of reaching beyond their connected overlays by contacting gateway nodes. We have, therefore, two different routing mechanisms, one consisting simply of using the virtual node instances to issue a request in a connected overlay, and a second one consisting of sending a `SYNAPSE_REQUEST` message to known gateways. A `SYNAPSE_REQUEST` message carries various parameters within itself (e.g. *Request ID, TTL, Routing History, Target Overlay List*) which can be used to implement different routings between overlays, depending on the application and the desired QoS. The choice of gateway nodes at each routing step, and the choice of how should the request propagate and which overlays should it target is what constitutes a *Routing Strategy* in Synapse.

As mentioned above, routing strategies are application-dependent: for example, an application can decide to issue inter-routing requests following a Random Walk scheme, i.e. picking n gateway nodes connected to one randomly chosen overlay at each step. This strategy, henceforth referred to as *n-Random Walk*, helps limit the number of messages in the system, and is the one that we have tested in our performance evaluation of Synapse in Section 3. Other examples of routing strategies may include *n-Flooding* or *Direct Routing* to specific overlay networks, where a `SYNAPSE_REQUEST` message embeds the list of targeted `netID`s, and the message is routed along the gateway nodes until the desired overlays are reached.

Gateway node discovery. In order to reach beyond its own overlays, a synapse node which joins the network needs to discover gateway nodes connected to overlays other than its own. The most effective way of discovering gateway nodes would be *Message Embedding*: each node embeds its list of its connected overlays in every message sent in the overlays, In this way, every other node routing the message can update its DOT with the embedded gateway information.

If, however, the presence of legacy nodes makes it impossible to alter the message packets in the overlay, an alternative mechanism would be *Active Notification*: gateway nodes routing a message in the overlay contact the message originator by sending a `SYNAPSE_OFFER` message with the list of their connected overlays. There could also be a third scenario in which, due to the overlay routing being, for example, fully recursive, the message originator is not known. In this case, a *Peer Exchange* mechanism serves the purpose of populating the DOT of each node.

3 Simulation and Experimental Results

In this section, we present the results obtained by running simulations within the OverSim-based Synapse simulator, as well as those obtained from the deployment of Synapse on the Grid'5000 platform.

OverSim implementation of Synapse. In order to be able to handle multiple overlay networks of different types, we have modified the OverSim simulator to support dynamic run-time instantiation of overlay modules and the interaction of existing overlay modules in OverSim with the Synapse-controller module. A more detailed description of these changes can be found in [17].

Simulation settings. All of the simulations were run on 2000 nodes, clustered into an equal number of Chord and Kademlia sub-overlays. All of the nodes were treated as either synapse or gateway nodes, and no legacy nodes were present. The main purpose of our simulations was to test the reachability of each of the overlays, while varying the granularity of the network (i.e. the number of different overlays (o), given the same overall amount of nodes (n)), the number of gateway nodes present in each of the overlays (g_o), and the connection degree of gateway nodes (i.e. the number of different overlays a gateway node is connected to (d)). As the idea was to estimate a lower bound of system performance, we have adopted the simplest and least demanding routing strategy for synapse requests: a stateless 1-Random-Walk, as described in Section 2. Finally, the TTL has been set to 8, for all of the simulations.

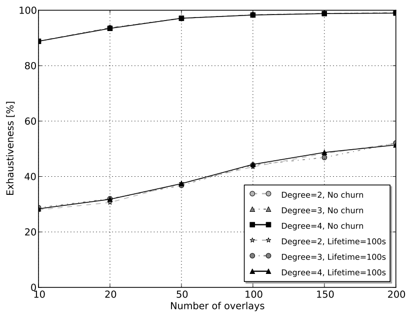
The tests consisted of inserting random keys throughout the entire system, and performing lookups for said keys, by a different node, not necessarily a member of the same sub-overlay in which the key is present. All replication within the sub-overlays has been disabled in order to create the most challenging conditions, and produce metrics as correlated as possible. We have tested different scenarios, without churn, to evaluate the topology built by the node discovery process, and with high churn, i.e. with a very short node lifetime, to test it in extreme conditions, such as those of a mobile application. In all of the simulations, the connection degree was equal for all gateway nodes. However, the percentage of gateway nodes and their interconnection degree have been correlated to guarantee the minimum number of gateway nodes-per-overlay to have a connected topology across all sub-overlays, without leaving any sub-overlay isolated due to, possibly, a lack of gateway nodes connected to it.

Topology construction. Topologies have been created statically, using n, o, d, and, depending on the simulation scenario, either the percentage of gateway nodes-per-overlay, or the overall percentage of gateway nodes in the system. Two algorithms were used to generate topologies:

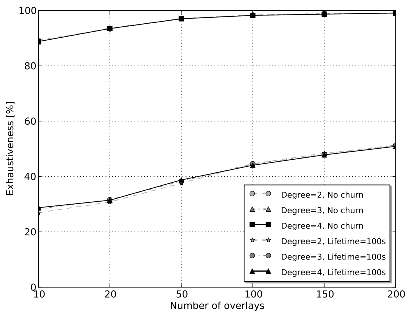
1. FIT – the topology is constructed to be fully-interconnected, in the sense that from any overlay there exists a path through gateway nodes of the system to any other overlay. This requires at least $\lceil \frac{o-1}{d-1} \rceil$ gateway nodes to be present in the system, and is accomplished using an algorithm described at [2];
2. RAT – The topology is constructed with fully random assignments of overlays to gateway nodes, using a uniform distribution over the o overlays.

Simulation 1: Effects of system granularity. Figure 2 shows the effect that system granularity (i.e. the number of sub-overlays) has on the general system exhaustiveness. We have simulated both a churn-less environment and one with high churn, to test the topology itself, as well as its resilience to extreme conditions. Figure 2a and Figure 2b compare a completely random topology vs. the one in which exhaustive connectivity has been forced. It is remarkable that the performances are substantially equivalent, suggesting that, in fact, a gateway topology can generally be built with just a partial knowledge of the system, by a simple random selection of overlays. Even with 200 overlays, the routing has proven to be exhaustive, reaching every sub-overlay, and suggesting that building a clustered overlay network is a feasible solution. Lower exhaustiveness with lower granularity is explained by the fact that, with having a higher number of edges for each overlay, there is a higher probability of loops being present, leading, with this simplest routing strategy, to requests bouncing back to their original overlay, an effect which can easily be avoided with a stateful routing strategy.

Simulation 2: Configuration of gateway nodes. Since maintaining a connection to multiple overlays is a costly operation, in this experiment we have tested the effectiveness of two opposite scenarios, one with very few gateway nodes maintaining a high degree of connectivity (much like a super-peer structure), and a second one, in which an increasing number of gateway nodes maintains a connectivity as low as possible (degree 2). It is worth noting that, despite the high connectivity degree, the gateway nodes in the first scenario were not exempted from churning. Figure 3 shows the system performance in the two scenarios. Interestingly enough, a decrease from degree 6 to degree 3 (Figure 3a) does not bring any visible decrease in performance, neither with nor without churn, partly due to the simple routing strategy adopted, and

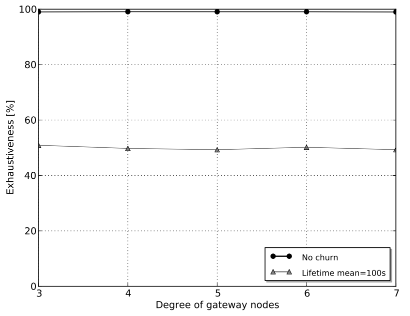


(a) FIT Topology

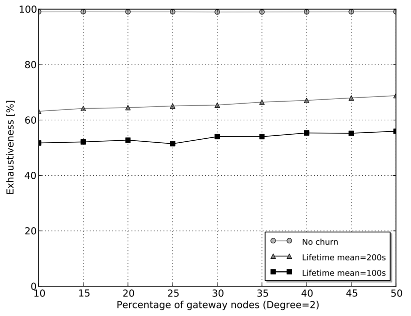


(b) RAT Topology

Fig. 2: Effects of system granularity, with and without churn

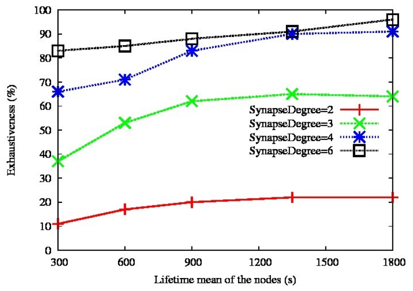


(a) Few high-degree gateways

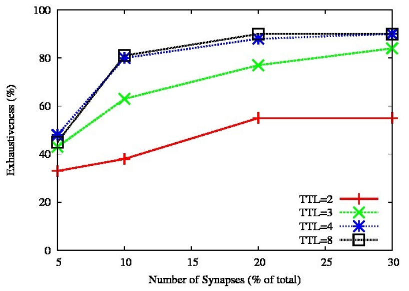


(b) Many low-degree gateways

Fig. 3: Performance comparison for different gateway topologies



(a)



(b)

Fig. 4: Results from experiments run on Grid'5000

it is an aspect that can be taken into account when designing a system by explicitly deploying synapse-gateways. In the second scenario (Figure 3b), on the other hand, the increase of gateway nodes brings a slight increase in the exhaustiveness under churn, which suggests a possible strategy to handle situations of sudden churn in a system, by having most of the nodes immediately increase their connectivity degree by 1.

Experiments on Grid'5000. In order to evaluate the behavior of Synapse within a real-world environment, we have developed a Java implementation of the Synapse protocol, which we have used to perform experiments on the Grid'5000 platform, which aims at providing an experimentation testbed to study large-scale parallel or distributed systems, and comprises thousands of interconnected computers across numerous sites in France. In all of the experiments performed, we have used 1000 nodes, distributed over 10 Chord and 10 Kademlia overlays, interconnected via the Synapse protocol.

In the first experiment, we have investigated the exhaustiveness of the interconnected systems under different mean lifetimes of the nodes and different degrees of connectivity of synapse nodes. We have placed an emphasis on high-churn-rate conditions (when the mean lifetime of the nodes is low), which should be observable in the near future, in overlay networks in which peers need not only be desktops and laptops, but also Internet TV and mobile devices, which are expected to join and leave the network at high frequency. In order to generate this high churn rate of nodes in the systems, we have used the Pareto distribution. The overall percentage of synapse nodes was fixed to 20% of the overall number of nodes, while the TTL value was fixed to 8, in all of the cases. The results obtained from this experiment are shown in Figure 4a, from which we can notice that, for a fixed degree of connectivity, the Synapse protocol is fairly resilient for values of the mean lifetime above 900s, and less resilient for lower values. However, in order to achieve a sufficient level of exhaustiveness, it is necessary to increase the degree of connectivity of synapse nodes to at least 4, for mean lifetime values above 900s, or to at least 6, for mean lifetime values below 600s.

In the second experiment, we have once again investigated the exhaustiveness of the interconnected systems, this time while varying the percentage of synapse nodes and the TTL. The degree of connectivity of synapse nodes has been fixed to 4, and the churn rate of the nodes to 1800s. The results obtained from this experiment are shown in Figure 4b, from which it can be noticed that the exhaustiveness significantly increases when the TTL is increased from 2 to 4, but remains the same as the TTL is increased from 4 to 8, giving rise to the conclusion that a TTL of 4 is efficient enough when interconnected networks of this scale are concerned (20 networks, 1000 nodes overall) Another inference which can be made from Figure 4b is that having 20% of overall nodes as synapses will result in sufficient exhaustiveness for this scale of interconnected overlays, as there is an obvious rise in exhaustiveness accompanying the increase of the number of synapse nodes from 5% to 10% and from 10% to 20%, while no significant rise occurs with a further increase of the number of synapses from 20% to 30%.

4 Conclusions and Further Work

In this paper, we have given a brief overview and presented the first evaluation of the Synapse Framework, the purpose of which is to enable the design of distributed applications based on multiple interconnected overlays, as well as to facilitate easier interconnection of already deployed overlays. The protocol has been developed in the OverSim overlay simulator, which has been modified to support multiple overlay types at run-time, and a Java client has been deployed and tested on the Grid'5000 platform. Simulations and experimental results suggest that the framework is suitable for providing efficient support for federated topologies with limited costs, in terms of messages and node connectivity. As we have just begun scratching the surface of all the possibilities offered by such an approach, our future work includes additional mathematical modeling of the protocol, an adaptation to unstructured overlays as well and further extensive testing of all the routing strategies under different system parameters, in order to be able to accurately quantify messaging overhead, resilience to churn and data consistency. Furthermore, we will aim to define a mechanism which would guarantee only a minimum level of interconnection between different overlays, i.e. that there is a constant presence of only a minimal number of gateway nodes within the overlays.

Acknowledgments. The authors warmly thank Erol Gelenbe for many precious discussions, and Riccardo Loti for careful reading.

References

1. Bittorrent website. http://www.bittorrent.com.
2. Synapse website. http://www-sop.inria.fr/teams/lognet/synapse.
3. I. Baumgart, B. Heep, and S. Krause. OverSim: A scalable and flexible overlay framework for simulation and real network applications, In *Proceedings of the IEEE P2P'09, p. 87-88.*
4. M. Castro, M. Costa, and A. Rowstron. Peer-to-peer overlays: Structured, unstructured, or both? Technical Report MSR-TR-2004-73, Microsoft Research, Cambridge, UK, 2004.
5. L. Cheng. Bridging distributed hash tables in wireless ad-hoc networks. In *Proc. of GLOBECOM '07*.
6. L. Cheng, R. Ocampo, K. Jean, A. Galis, C. Simon, R. Szabó, P. Kersch, and R. Giaffreda. Towards distributed hash tables (de)composition in ambient networks. In *Proc. of DSOM '06*.
7. A. Datta and K. Aberer. The challenges of merging two similar structured overlays: A tale of two networks. In *Proc. of EuroNGI '06*.
8. P. Ganesan, P. Krishna Gummadi, and H. Garcia-Molina. Canon in g major: Designing dhts with hierarchical structure. In *Proc. of ICDCS '04*.
9. L. Garcés-Erice, E. W. Biersack, P. Felber, K. W. Ross, and G. Urvoy-Keller. Hierarchical peer-to-peer systems. In *Proc. of Euro-Par '03*.
10. R. Jimenez, F. Osmani, and B. Knutsson. Connectivity properties of mainline bittorrent dht nodes. In *Proc. of IEEE P2P '09*.
11. M. Kwon and S. Fahmy. Synergy: an overlay internetworking architecture. In *Proc. of ICCCN '05*.
12. L. Liquori, C. Tedeschi, L. Vanni, F. Bongiovanni, V. Ciancaglini, and B. Marinkovic. Synapse: A scalable protocol for interconnecting heterogeneous overlay networks. In *Proc. of Networking '10*.
13. B. Thau Loo, R. Huebsch, I. Stoica, and J. M. Hellerstein. The case for a hybrid p2p search infrastructure. In *Proc. of IPTPS '04*.
14. T. M. Shafaat, A. Ghodsi, and S. Haridi. Dealing with network partitions in structured overlay networks. *Peer-to-Peer Networking and Applications*, 2(4), 2009.
15. G. Tan and S. A. Jarvis. Inter-overlay cooperation in high-bandwidth overlay multicast. In *Proc. of ICPP '06*.
16. Z. Xu, R. Min, and Y. Hu. Hieras: A dht based hierarchical p2p routing algorithm. In *Proc. of ICPP'03*.
17. V. Ciancaglini, L. Liquori and G. Ngo Hoang. Towards a Common Architecture to Interconnect Heterogeneous Overlay Networks. In *Proc of ICPADS '11*.