X-BOT: A Protocol for Resilient Optimization of Unstructured Overlay Networks (Supplementary Document)

João Leitão, *Member, IEEE,* João Pedro Marques, José Pereira, *Member, IEEE,* and Luís Rodrigues, *Senior Member, IEEE*

Abstract—This document provides complementary information to the paper with the same name submitted to IEEE Transactions on Parallel and Distributed Systems. This supplement discusses additional details regarding the configuration and operation of the X-BOT protocol and presents further experimental results that complement those presented in the main document.

Index Terms—Peer-to-Peer systems, Unstructured Overlay Networks, Topology Adaptation, Network Protocols

1 Introduction

This document provides complementary information to the paper with the same name submitted to IEEE Transactions on Parallel and Distributed Systems. The supplement is structured as follows:

Section 2 discusses some of the relevant properties of X-BOT and provides additional rationale for them. In particular the section discusses how X-BOT minimizes the occurrence of local minima configurations during the adaptation of the overlay topology. It also discusses what aspects of the X-BOT protocol contribute to maintain a low clustering coefficient and a low average shortest path in the overlay.

Section 3 provides additional insight on the design of oracles for X-BOT. The section discusses the class of oracles that can be used with X-BOT and their strengths and limitations. Several additional examples of oracle implementations are provided and some future work directions on the development of more complex oracles that can be plugged into X-BOT are addressed.

Section 4 provides a detailed description of the parameters that affect the operation of X-BOT and provides additional insight on how to configure these parameters adequately.

Section 5 provides additional experimental results that illustrate the impact of X-BOT on the average cost of links maintained by X-BOT, discussing its effect on both biased and unbiased links. Additionally, this section provides a breakout of the gossip-based dissemination latency in the PlanetLab test scenario, reported in the main document.

Section 6 briefly addresses some additional related work that covers additional techniques to adapt the topology of unstructured overlay networks.

Finally, Section 7 discusses the additional directions for research that we plan to pursue.

2 X-BOT Properties

This section provides additional insights on some of the X-BOT properties. In particular, the complexity of the protocol from the point of view of communication overhead is addressed. Also, the section presents informal arguments, claiming that the probability of X-BOT falling into local minima configurations is small. Finally, the section highlights the features of X-BOT that help in providing low clustering coefficient and low average shortest path to the resulting overlay.

2.1 Complexity

A complete X-BOT optimization round requires the sequential exchange of seven messages. Furthermore, in the most common run each node involved in the optimization only has to send and receive at most two messages (exceptions are runs where faults occur). Given that the optimization of the overlay can be executed as a background activity, the cost of the adaptive mechanisms can be easily tuned to become negligible when compared with the application traffic. This can be performed by adjusting the Period Between Optimizations parameter.

- *João Leitão is with the Distributed Systems Group of the INESC-ID laboratory and the Instituto Superior Técnico, Universidade Técnica de Lisboa. Email: jleitao@gsd.inesc-id.pt*
- *João Pedro Marques is with the Distributed Systems Group of the INESC-ID laboratory and the Instituto Superior Técnico, Universidade Técnica de Lisboa. Email: jmarques@gsd.inesc-id.pt*
- *José Pereira is with the Distributed Systems Group of the University of Minho. Email: jop@di.uminho.pt*
- *Luís Rodrigues is with the Distributed Systems Group of the INESC-ID laboratory and the Instituto Superior Técnico, Universidade Técnica de Lisboa. Email: ler@ist.utl.pt*

This work was partially supported by Fundação para a Ciência e Tecnologia (FCT) via the INESC-ID multi-annual funding through the PIDDAC Program fund grant and via the project "HPCI" (PTDC/EIA-EIA/102212/2008).

2.2 Avoiding Local Minima Configurations

This section provides informal arguments to backup the claim that the X-BOT avoids local minima with a probability greater than zero.

Assume that the overlay is in a given configuration, say $\mathcal{C}$, and there is some other possible configuration, say $\mathcal{C}'$, which has a lower overlay cost. Since X-BOT switches links in pairs to preserve the original degrees of the nodes, there must exist 4 nodes, a, b, c, d, such that links l_{ab} and l_{cd} in configuration $\mathcal{C}$ are replaced by links l_{ac} and l_{bd} in configuration $\mathcal{C}'$ and that:

$$\text{LinkCost}(l_{ac}) + \text{LinkCost}(l_{bd}) < \text{LinkCost}(l_{ab}) + \text{LinkCost}(l_{cd})$$

Assume that LinkCost(l_{ac}) $<$ LinkCost(l_{ab}). In order for node a to trigger an optimization round trying to replace his current link to b for a new link to c, c must be in a's passive view, and be sampled by its local oracle. As long as c is in a's passive view, there is always a probability greater than zero that c is selected to be sampled by the local oracle by the construction of the protocol. Therefore, in order to avoid the overlay from staying in configuration $\mathcal{C}$, without ever switching to configuration $\mathcal{C}'$, it is enough to ensure that eventually c will be part of a's passive view.

Passive views are updated periodically, through the exchange of samples of nodes active and passive views. These samples are exchanged through random walks, that are forwarded across neighbors in the active view of nodes. In the main paper, it has been shown that in X-BOT the in-degree of nodes is approximately the same for all nodes in the system (considering the overlay denoted by the closure of all nodes active views). Furthermore, when nodes initiate the process of updating their passive view (as described in previous work [1]) they include their identifier in the sample they exchange. Therefore, while c is active, there will always be some passive views in the system that contain c's identifier. As these views are shuffled, eventually c will be in a's passive view with some probability greater than zero.

This would allow node a to sample the link cost to c using its local oracle, and therefore trigger an optimization round that would replace his current link with b for a new link with c and the overlay to move from configuration $\mathcal{C}$ to $\mathcal{C}'$.

Note however that X-BOT is a localized algorithm. Therefore it cannot perform optimizations that require global knowledge.

2.3 Ensuring Low Clustering Coefficient

Experimental results have shown that X-BOT is able to maintain a small clustering coefficient. This section provides the rationale that justifies why the clustering coefficient only increases slightly despite the fact that the overlay topology is biased to promote some form of locality (taking into consideration an efficiency criteria encoded in the companion oracle). There are two main factors that contribute to maintain the clustering coefficient of the biased overlay network relatively low:

- The maintenance of unbiased neighbors which are implicitly biased to be distant neighbors selected at random (notice that the selection of the unbiased neighbors of a node is performed when nodes are filling their active views; during this time, as described in the paper, X-BOT do not execute its optimization protocol). As a result, the probability that two neighboring nodes share the same unbiased nodes is very low (this is supported by our experimental observations);
- Biased neighbors are not deterministically selected. On the contrary, X-BOT relies on a continually changing random sample of nodes (the passive view) to locate suitable candidates for its optimization protocol. Therefore there is a high probability that neighboring nodes sample different peers, and therefore select distinct neighbors during the execution of X-BOT.

2.4 Ensuring Low Average Shortest Path

Experimental results provided in the main paper show that the average shortest path between any pair of nodes in the overlay network that results from the operation of X-BOT is very similar to that of a random overlay.

This happens due to the existence of unbiased neighbors that, as discussed above, promote the existence of distant neighbors. Notice that if each node maintains a distant link, there are still a significative number of these links, that can be used to access remote areas of the overlay in a single hop. Also, because these links are selected at random (as explained before), there is a high probability that neighboring nodes have unbiased nodes which are positioned in different regions of the overlay. This contributes to the existence of short paths between any pair of nodes in the overlay.

3 Oracles on X-BOT

The main paper presents two possible oracle implementations, namely the *Latency Oracle* and the *Internet Service Provider Oracle*. This section further discusses the class of oracles that is supported by the X-BOT protocol as well as some alternative oracle designs. It also discusses some limitations on the supported class of oracles.

3.1 Class of Oracles

X-BOT employs a class of oracles that rely on *local knowledge*, i.e., a node makes optimization decisions based on local information regarding the distance to its own peers. Furthermore, in an optimization round, only 4 nodes are involved. Thus the optimization step is also localized. This makes X-BOT scalable.

Naturally, this prevents X-BOT from generating topologies that require global knowledge. For instance, to bias a network towards a complex topology such as an inter-connected clique of nodes, would require nodes to know the location of many other nodes in the overlay or to have *a priori* knowledge of their final coordinates in the overlay (such as in T-Man [2]). To bias the network towards such topologies is outside the scope of this paper. However, as discussed in the next paragraphs, there are several meaningful oracles that fit in the class of oracles considered by X-BOT.

3.2 Alternative Oracles Implementations

3.2.1 Stretch Oracle:

It is possible to devise an oracle that returns the number of hops, in the physical network, required to materialize a given overlay link. This can be implemented using tools such as the Unix *traceroute* command, similarly to what is proposed in the Araneola protocol [3]. Notice that the underlying topology of the Internet is relatively static, meaning that traceroute measurements do not need to be refreshed frequently, allowing the oracle to have a negligible overhead.

Such an oracle would allow to minimize the stretch of the overlay network, i.e., the ratio between the number of hops in the underlaying IP network and the number of hops in the overlay for a given path. Additionally, as reported in the work of Rostami & Habibi [4] there is an (approximately) linear correlation between the physical distance of two peers in the physical network, and the latency of communication between those peers, which implies that such an oracle could contribute to lower communication latency over peers in the overlay and the link stress due to the use of the overlay.

3.2.2 Landmark Oracle:

A Landmark Oracle is able to place nodes in a coordinate system. This allows to associate costs to links based on the Euclidean distance between the vertices, i.e., a given node may derive the link cost from its own coordinates and the coordinates of the peer.

Such oracle can be implemented measuring the round-trip-time (RTT) of each node to a globally agreed and well known set of servers, which are know as landmarks (*e.g.* DNS root servers however, this is valid for any set of well know, and geographically disperse servers), as suggested by Ratnasamy et al. [5]. Previous work [6] has shown that nodes with similar RTT measures to selected landmark servers had a higher probability of being geographically closer.

3.2.3 IP-based Clustering Oracle:

An IP-based Oracle assigns costs to links based on the IP addresses of the nodes (assigning low costs to links in the same network and higher costs to links that cross different autonomous systems). Such an oracle allows to improve the locality of neighboring relations in unstructured overlay networks with very reduced overhead.

Implementations of an IP-based Oracle have been proposed in the work of Piotr Karwaczyński and Piotr Karwaczyński et al. [7], [8]; these implementations are inexpensive as they not require the exchange of messages among peers. Such oracles operate by taking into consideration the IP of nodes to calculate the link cost between two peers. For instance, using the size of the match of common IP prefixes to calculate a measure of proximity between two peers. Additionally, other static information, such as the one employed in Skipnet [9] (which leverages in DNS names) can also be used to improve the operation of the oracle.

3.2.4 Content Similarity Oracle:

A Content Similarity Oracle assigns to links a cost that is inversely proportional to the similarity of the content stored by the nodes in its vertices. Such an oracle can be used to bias the network such that nodes with similar content are located in the same region in the overlay. This may help in implementing search algorithms, in combination with routing algorithms that guide queries to the regions where the desired content is more likely to exist.

Such an oracle can be implemented, for instance, by classifying resources in a set of (possibly finite) categories, and then locally calculate the percentage of stored resources that fall in each category. In order to compute the link cost, oracles could exchange a data structure containing the percentage of resources that fall in the c most significative categories, and then compute a similarity rate that can be directly used as the link cost [10].

3.3 On the Latency Oracle Implementation

The main paper discusses a simple implementation of the Latency Oracle: this implementation relies on the exchange of specific probe messages among peers, that are used to compute the link latency among them. For simplicity, in the paper it is assumed that latency values between nodes are static. Evidently, this is a simplification. In fact, a Latency Oracle can be implemented by maintaining an updated local record of the latency measured for each peer. Furthermore, for peers to which several measurements are performed (*i.e.* peers that remain in the active or passive views of nodes for long periods of time), the oracle can store a weighted average of the measured latency, similar to what TCP does, in order to avoid momentary spikes or errors in individual measurements.

3.4 Combining Oracles

The oracles that have been discussed in the previous paragraphs can be combined to create more complex oracles that take a combination of different criteria into consideration when optimizing the overlay. For instance the Content Similarity Oracle and the Latency Oracle can be combined to favor the selection of neighbors that have simultaneously similar content and are physically close.

To build such an oracle the output of the each contributing oracle would have to be normalized to a predetermined scale, taking into consideration the maximum (expected) cost value returned by each type of oracle. Subsequently, the output of the combined oracle can be defined as a weighted average of the output of each contributing oracle.

3.5 On the Use of Inconsistent Oracles

All the work reported in the paper assumes that the system is pre-configured such that all nodes use the same oracle. One could imagine scenarios where different nodes in the same overlay would have different (local) goals and that this would still allow the emergence of some global stable topology. Unfortunately, it is possible to show that if a set of independent agents in a system have conflicting goals, the convergence of the system is not guaranteed. Exploring adequate mechanisms to allow different peers to use different oracles in a single overlay network, aiming at optimizing distinct performance criteria will be further addressed in our future work.

4 CONFIGURING X-BOT

X-BOT has several configuration parameters that affect its operation. This section addresses the effect of each parameter in the behavior of the protocol and provide some insights on how to configure these parameters.

Active View Size:

This parameter controls the number of peers maintained in each node's active view. As discussed in the paper, X-BOT follows the architecture proposed in the HyParView protocol [1]. In HyParView, active views are symmetric and have a size optimized to support gossip-based dissemination protocols, i.e., their size is configured to be *fanout* $+ 1$, where *fanout* is the parameter used in the gossip broadcast protocol. It has been shown that, in order to ensure a high probability of atomic delivery of messages for gossip-based processes, the fanout used should be of $\ln(N)$, where N is the total number of nodes in the system [11]. However, these theoretical results do not assume an overlay that ensures global connectivity, and furthermore assume that some messages may be lost during the dissemination process. Furthermore, it is well known that in an undirected k-regular random graph 3 is the minimum degree to ensure global connectivity [12]. Taking into consideration these results, we suggest that the active view should have a size that falls between 3 and $\ln(N)$. We have opted to use a size of $\log(N) + 1$, which experimentally has shown to ensure global connectivity of the overlay with a probability close to 1.

Passive View Size Control Parameter (k):

The k parameter controls the size of the passive view employed by X-BOT. In fact, in our system, the passive view size is k times larger than the active view. As discussed in the paper, the passive view is used for two complementary purposes, namely: $i)$ fault-tolerance; $ii)$ as a source to extract unbiased samples of other nodes in the overlay, that X-BOT can use to bias the local active view.

From the point of view of fault-tolerance, as discussed earlier, the size of the passive view should be no smaller than $\ln(N)$, where N is the total number of nodes in the system. However, a larger value would offer more sample of nodes for X-BOT to operate and more opportunities to bias the active view of nodes. On the other hand, a much larger value may induce a non-negligible overhead, as more information need to be stored at each node.

In our experiments the value of k was set to 6, as this allows to have a view of size 30 which although being larger that $\ln(N)$, does not impose an noticeable overhead in the protocol. We have determined experimentally that this offers a good trade-off between the minimum size required for fault-tolerance, and also the necessity of X-BOT to have access to a varied sample of nodes.

Period Between Optimizations (PBO):

The period between optimizations (or simply PBO) determines the time between each attempt by the X-BOT protocol to locate one (or more) suitable neighbor to further bias its active view. It also determines the rate at which the active view of a node may be updated (*i.e.*, changed).

Large values of PBO slow the convergence of the overlay topology, leading to larger periods of operation with a sub-optimal overlay. On the other hand, large PBO values promote the stability of the overlay. Furthermore, low PBO values increase the X-BOT overhead, given that a more frequent operation induces additional message exchanges and more oracle invocations. In any case, the PBO value should be a multiple of the passive view update period; such that when a new optimization round start the passive views has been updated with new potential candidates for optimization.

Passive Scan Length (π):

The passive scan length (or simply π) determines the number of nodes extracted from the passive view to serve as candidates to bias the active view of nodes. The link cost to these nodes is then measured (through the companion oracle), and if their cost is below that of the most expensive active view neighbor (excluding the protected μ most expensive neighbors) kept by that node, the optimization process is triggered. This parameter also determines the maximum number of elements of an active view that can be updated in a single optimization round of X-BOT.

The value of π should be lower (or equal) to that of half the nodes in the active view that can be biased by X-BOT (which is equal to the size of the active view minus the number of unbiased neighbors). Setting π to a small value has two advantages: *i)* it promotes some stability in the overlay, avoiding to exchange the majority of nodes in the active view of a single node in the context of a single optimization execution and, *ii)* lowers the cost of the overall optimization process. Results presented in the main paper show that a conservative configuration of π allows to achieve fast convergence and a good level of optimization for the overlay.

Number of Unbiased Neighbors (μ):

This parameter controls the number of elements in an active view that are never targeted by the optimization process of X-BOT. Experimental results included in the main paper show the impact of this parameter on several properties of unstructured overlay networks; from those results it is possible to observe that a μ value of 1 provides good results in all scenarios (these results match those presented by the authors of GoCast [13]). Nevertheless, in the next section, we provide some additional experimental data that illustrate the effect of maintaining larger values of unbiased links.

5 Additional Experimental Results

This section provides additional experimental results, namely in terms of the effect of the X-BOT protocol on the average link cost of the overlay. These results aim at allowing to measure the effect of the protocol operation over the biased and unbiased overlay links. We then present detailed experimental results for the gossip-based dissemination latency in the PlanetLab scenario. A more concise description of these results has been presented in the main document; here we provide some additional detail as well as a breakout of the combined plot.

5.1 Effect on the Average Link Cost

We now present additional experimental data that illustrates the effect on the average link cost of the overlay (also discriminating between average biased and average unbiased link cost), when the protocol is configured to keep a different number of unbiased neighbors. For this purpose, we use the same scenarios as in the main paper but we vary μ from 1 to 4. Results were obtained by executing the protocol for $1,000$ simulation cycles after the join of nodes as described in the evaluation section of the main paper. At the end of the simulation the average cost of unbiased and biased links, kept by the protocol, is measured. For reference, the average link cost of a random overlay is also depicted (these results were obtained by allowing nodes to join the overlay in a similar fashion to X-BOT, but without executing the X-BOT optimization protocol). Table 1 summarizes the obtained results for 5 independent runs of each experience.

The results show that, for all test scenarios, with smaller values of μ we obtain lower values in the average link cost when compared with an overlay built at random. This is not a surprising effect, as it is the result of the X-BOT optimization on the (larger number of) biased links.

Also, in most scenarios, with smaller values of μ we obtain higher values in the average link cost of unbiased links. This is due to the fact that, as explained previously, X-BOT tends to avoid biasing the links that have a greater cost. There are however exceptions to this behavior. In particular, in the cartesian and inet-3.0 scenarios, the average cost of unbiased links is higher with $\mu = 2$ than with $\mu = 1$. This happens because in those topologies each node has many peers which are *close* in the distance metric applied in each scenario therefore, the random topology that serves as the base for X-BOT operation already included many links with a low cost, and X-BOT does not actively try to find distant peers to keep as unbiased neighbors.

Interestingly, in the PlanetLab scenario, as the number of unbiased neighbors kept by each node gets lower, the average link cost of unbiased neighbors slightly increases. This happens because in this scenario nodes have few *close* neighbors. Therefore, as nodes try to find more close neighbors they have to select peers which, despite the fact that are among the closest available peers, still have a slightly high link cost. Despite this, the average biased link cost is still dramatically lower than the average link cost of a random overlay in this scenario, clearly showing the benefits of our approach.

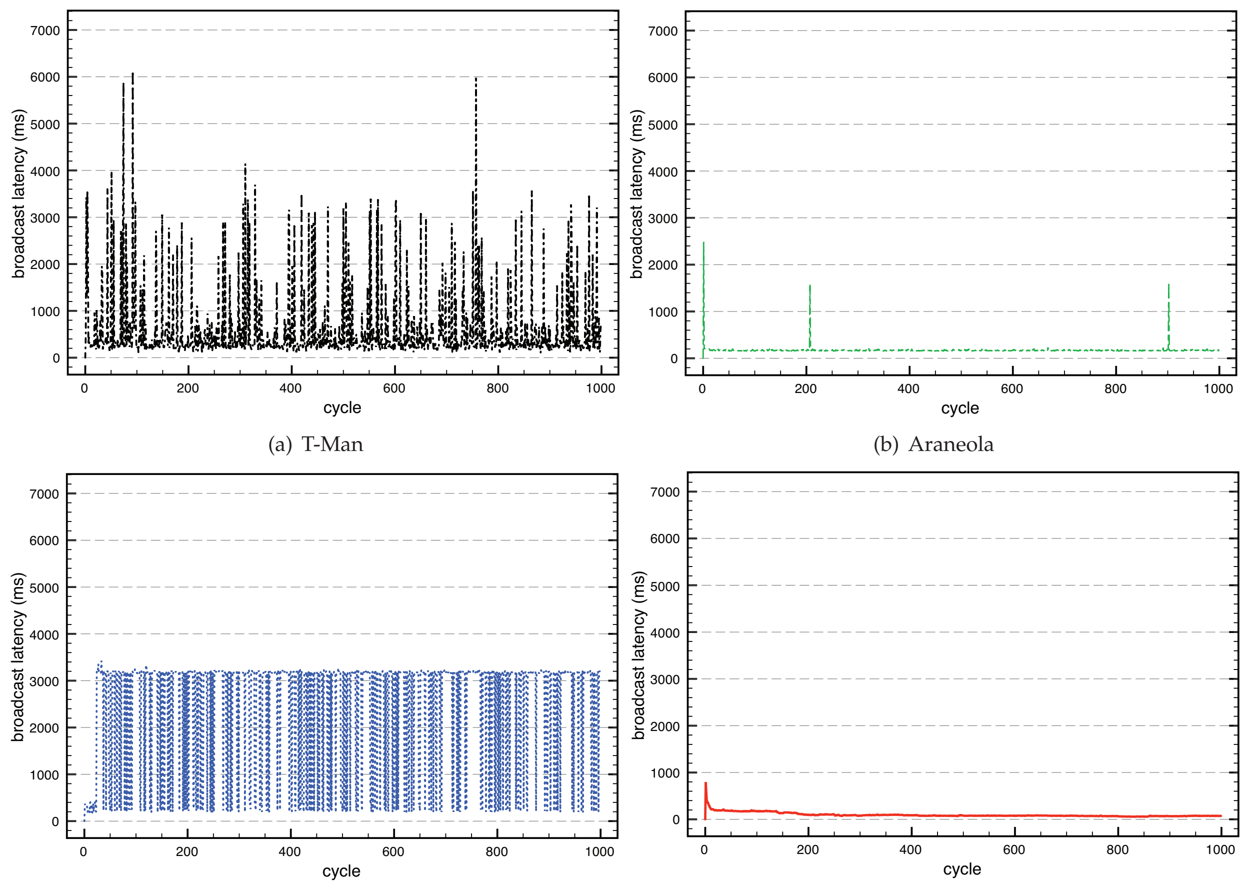
5.2 Message Dissemination Latency in the PlanetLab Scenario

In the main document we have presented experimental results for the latency of a gossip-based dissemination protocol operating of top of the overlays generated by each protocol (for each of the 3 experimental setups). In the PlanetLab scenario, both T-Man and GoCast present frequent fluctuations which result from the fact that the overlays generated by these protocols are not uniform.

In order to provide additional detail for the PlanetLab scenario (given that the combined figure presented in the main document obfuscates some information), Figure 1 presents individual plots for each individual protocol. As one can see both T-Man and GoCast present frequent fluctuations in the latency. This shows that the overlays that result from the operation of these protocols are not adequate to support a scalable and stable gossip-based dissemination protocol. Notice that during some periods of the simulation, T-Man was able to exhibit latency values below that of X-BOT. However, as we discuss in the main document, this happens because the aggressive strategy employed by the T-Man protocol breaks the overlay network, which results in broadcast reliability values below 100%.

	Cartesian Scenario		
	Average Link Cost	Average Biased Link Cost	Average Unbiased Link Cost
Random Overlay	520.05	–	–
X-BOT ($\mu = 4$)	461.36	24.73	570.62
X-BOT ($\mu = 3$)	372.85	22.87	606.47
X-BOT ($\mu = 2$)	258.44	18.17	619.05
X-BOT ($\mu = 1$)	125.93	12.35	581.79
	PlanetLab Scenario		
	Average Link Cost	Average Biased Link Cost	Average Unbiased Link Cost
Random Overlay	100, 865.86	–	–
X-BOT ($\mu = 4$)	88, 990.38	2, 225.70	110, 596.92
X-BOT ($\mu = 3$)	77, 511.78	3, 000.66	127, 111.88
X-BOT ($\mu = 2$)	57, 683.03	3, 795.05	138, 421.72
X-BOT ($\mu = 1$)	31, 456.59	4, 825.36	137, 902.78
	Inet-3.0 Scenario		
	Average Link Cost	Average Biased Link Cost	Average Unbiased Link Cost
Random Overlay	27404.44	–	–
X-BOT ($\mu = 4$)	27, 857.47	21, 793.35	29, 373.62
X-BOT ($\mu = 3$)	26, 406.80	21, 432.64	29, 722.99
X-BOT ($\mu = 2$)	24, 871.94	21, 404.65	30, 073.13
X-BOT ($\mu = 1$)	23, 186.46	21, 370.48	30, 450.88

TABLE 1
Comparison of average link cost, average biased link cost, and average unbiased link cost for several values of parameter μ in X-BOT and in a random overlay network with similar properties.



(a) T-Man (b) Araneola

(c) GoCast (d) X-BOT

Fig. 1. Message dissemination latency using overlays generated by each protocol in the PlanetLab scenario.

6 Additional Related Work

In this section we mention additional works that propose alternative mechanisms to bias the topology of unstructured overlay networks.

The work of Rostami & Habibi [4] the authors propose a mathematical model for measuring the degree of matching between an overlay network topology and the underlying physical topology. Similar to our approach, this metric is based on a link cost notion that can either be calculated by measuring the latency between two nodes or the number of physical hops that separate two overlay neighbors. The authors propose an heuristic and an algorithm to lower the mismatch between the overlay topology and the physical network topology. However, and contrary to our approach, their algorithm requires nodes to exchange their complete partial views. This creates a significant instability in the overlay, and may disrupt the operation of protocols that are executed on top of it.

Hsiao et al. [14] propose an approach to address the topology mismatch problem in unstructured overlay networks. Their solution however is only tailored for minimizing the latency between neighboring peers. Additionally, and contrary to X-BOT, their solution requires nodes to have an estimate of the total number of nodes in the system, and nodes are required to sample potentially hundred of nodes latencies before joining the overlay. The authors propose the use of a mechanism based on global coordinates to infer potential latency values between peers.

7 Future Work Directions

As future work we plan to further experiment X-BOT with other interesting oracles, such as oracles that reflect the similarity of the content stored by peers. These oracles could be used to build highly efficient resource location protocols on top of a (biased) unstructured overlay. Preliminary results of such a platform, which aims at assisting in the management and allocation of resources in cloud computing infrastructures, can be found in the work of Alveirinho et al. [10].

Another potential direction for the work presented in this paper, is to explore how to rely in a protocol such as X-BOT to efficiently build and maintain overlay networks that support distributed hash table functionality, by biasing the topology according to node identifiers. Comparing the churn resilience of these overlays with that of more classical structured overlays such as Chord [15] or Pastry [16] may lead us to propose a new class of algorithms to build and maintain very robust distributed hash tables.

Furthermore, we aim to study the properties of overlay networks that result from the operation of X-BOT when nodes can choose oracles that aim at biasing the overlay for different performance criteria. We will explore how to design a *universal oracle* that can exchange information with other oracles in order to facilitate their computations of link cost for the local node, while at the same time being able to be configured to bias the topology for the criteria that is more useful to the local node. Understanding how to ensure that the overlay remains connected and stable when different nodes are biasing its topology for a different specific criteria is still an open problem.

Acknowledgments

The authors wish to express their gratitude to the anonymous reviewers for the useful comments and suggestions, which were instrumental to improve the original manuscript.

References

[1] J. Leitão, J. Pereira, and L. Rodrigues, “HyParView: A membership protocol for reliable gossip-based broadcast,” in *Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Edinburgh, UK, 2007, pp. 419–429.

[2] M. Jelasity and O. Babaoglu, “T-Man: Gossip-based overlay topology management,” in *The Fourth International Workshop on Engineering Self-Organizing Applications*, Hakodate, Japan, May 2006.

[3] R. Melamed and I. Keidar, “Araneola: A scalable reliable multicast system for dynamic environments,” in *Proceedings of the 3rd IEEE International Symposium on Network Computing and Applications (NCA)*, Cambridge, MA, USA, 2004, pp. 5–14.

[4] H. Rostami and J. Habibi, “Topology awareness of overlay p2p networks,” *Concurrency and Computation: Practice & Experience - Autonomous Grid*, vol. 19, pp. 999–1021, May 2007.

[5] S. Ratnasamy, M. Handley, R. M. Karp, and S. Shenker, “Application-level multicast using content-addressable networks,” in *Proceedings of the Third International COST264 Workshop on Networked Group Communication*. London, UK: Springer-Verlag, 2001, pp. 14–29.

[6] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, “A scalable content-addressable network,” in *Proceedings of the SIGGCOMM 2001 Conference*, San Diego, CA, USA, 2001, pp. 161–172.

[7] P. Karwaczyński, “Fabric: Synergistic proximity neighbour selection method,” in *Proceedings of the 7th International Conference on Peer-to-Peer Computing (P2P)*, Galway, Ireland, 2007, pp. 229–230.

[8] P. Karwaczyński, D. Konieczny, J. Moçnik, and M. Novak, “Dual proximity neighbour selection method for peer-to-peer-based discovery service,” in *Proceedings of the 22nd ACM Symposium On Applied Computing (SAC)*, Seoul, Korea, 2007, pp. 590–591.

[9] N. Harvey, M. Jones, S. Saroiu, M. Theimer, and A. Wolman, “Skipnet: a scalable overlay network with practical locality properties,” in *Proceedings of the 4th conference on USENIX Symposium on Internet Technologies and Systems (USITS)*, Seattle, WA, USA, 2003, pp. 9–25.

[10] J. Alveirinho, J. Paiva, J. Leitão, and L. Rodrigues, “Flexible and efficient resource location in large-scale systems,” in *Proceedings of the 4th ACM SIGOPS/SIGACT International Workshop on Large-Scale Distributed Systems and Middleware (LADIS)*, Zurich, Switzerland, July 2010.

[11] P. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulie, “From Epidemics to Distributed Computing,” *IEEE Computer*, vol. 37, no. 5, pp. 60–67, 2004.

[12] B. Bollobás, “Random graphs,” *Combinatorics (ed. H. N. V. Temperley) London Mathematical Society Lecture Note Series*, no. 52, pp. 80–102, 1981.

[13] C. Tang and C. Ward, “GoCast: Gossip-enhanced overlay multicast for fast and dependable group communication,” in *Proceedings of the Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Yokohama, Japan, 2005, pp. 140–149.

[14] H.-C. Hsiao, H. Liao, and C.-C. Huang, “Resolving the topology mismatch problem in unstructured peer-to-peer networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 11, pp. 1668 –1681, Nov. 2009.

[15] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for internet applications,” in *Proceedings of the SIGCOMM 2001 Conference*, San Diego, CA, USA, 2001, pp. 149–160.

[16] A. Rowstron and P. Druschel, “Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems,” in *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, Heidelberg, Germany, 2001, pp. 329–350.