Synergy: An Overlay Internetworking Architecture

Minseok Kwon
Department of Computer Science
Rochester Institute of Technology
Rochester, NY 14623–5608
E-mail: jmk@cs.rit.edu

Sonia Fahmy
Department of Computer Science
Purdue University
West Lafayette, IN 47907–2066
E-mail: fahmy@cs.purdue.edu

***Abstract*— A multitude of overlay network designs for resilient routing, multicasting, quality of service, content distribution, storage, and object location have been recently proposed. Overlay networks offer several attractive features, including ease of deployment, flexibility, adaptivity, and an infrastructure for collaboration among hosts. In this paper, we explore cooperation among co-existing, possibly heterogeneous, overlay networks. We design *Synergy*, a utility-based overlay internetworking architecture that fosters overlay cooperation. Our architecture promotes fair peering relationships to achieve synergism. Results from Internet experiments with cooperative forwarding overlays indicate that our Synergy prototype improves delay, throughput, and loss performance, while maintaining the autonomy and heterogeneity of individual overlay networks.**

I. Introduction

Over the past few years, overlay networks have emerged as a flexible paradigm for sharing information and rapidly prototyping disruptive technology. Without violating the end-to-end principle [1], overlays can be used to deploy new functionality between the core IP network and applications. Overlay solutions have been designed for the multicasting [2], inter-domain routing [3], content distribution [4], storage [5], and peer-to-peer networking [6] problems.

As overlay networks become pervasive, a number of questions naturally arise. First, can autonomous and *heterogeneous* overlays cooperate? Current overlay networks have both heterogeneous performance goals and heterogeneous service goals, e.g., minimum latency versus maximum bandwidth overlays, or multicast distribution versus unicast forwarding versus distributed storage. The key incentive for overlay cooperation is to exploit the presence of hosts with unique resources in one overlay, when such resources are absent in other overlays that need them. Second, can co-existing overlays interfere? Overlay networks are currently being deployed independently. Consequently, several diverse overlay networks may simultaneously exist and compete for the same resources in the Internet [7], [8]. Third, can overlay cooperation facilitate the design and deployment of revolutionary Internet architectures? Several recent studies propose bold re-designs of the current Internet architecture [9]. Overlays can provide a framework to accelerate prototyping such approaches.

In this paper, we explore a spectrum of cooperative services among co-existing autonomous overlays, and discuss associated scalability, and heterogeneity problems (we give more examples in [10]). As a proof-of-concept, we design and implement a utility-based overlay internetworking architecture, *Synergy*, where overlay agents facilitate inter-overlay coordination. Overlay nodes may elect to join the Synergy network which incorporates nodes from several overlay networks. Long-lived flows can utilize Synergy forwarding for better performance. Synergy tackles a number of management challenges that arise when constructing a shared overlay network comprised of autonomous overlays. The routing protocol in Synergy considers load at hosts, time intervals between successive routing decisions, and peering relationships among overlays, to offer stable, scalable, and fair cooperative routing. We have implemented a prototype of Synergy cooperative forwarding, and conducted experiments on the PlanetLab wide-area experimentation platform [11]. Results indicate that Synergy improves latency, throughput, and loss, while preserving overlay heterogeneity.

The remainder of this paper is organized as follows. In Section II, we discuss routing and management in the Synergy overlay internetworking architecture. In Section III, we present our implementation details and preliminary results from our experiments. In Section IV, we give an overview of related work. Finally, we summarize our conclusions and future work in Section V.

II. Synergy Architecture

The objective of our prototype Synergy architecture is twofold: (i) to improve routing performance via cooperative forwarding, and (ii) to be easily extensible to new services. Given an overlay network o, we define an *overlay link* or *connection* as $e_o = (d_s, \eta_0, \ldots, \eta_l, d_r)$. This comprises a host d_s, followed by a sequence of routers η_i, followed by a host d_r. An *overlay flow* is a chain of overlay links in which the first and last hosts are the sender and destination, respectively. A host in an overlay network is connected to its neighbors in the overlay via overlay links. A *home* overlay is defined as the autonomous overlay network which a host originally joins. Each overlay network executes its own routing protocol, according to its own optimization metrics, e.g., delay, bandwidth, node identifier, etc. We now describe the components of the Synergy architecture. Synergy security issues are discussed in [10].

A. Overlay Agents

Each overlay network periodically designates one of its hosts as an *overlay agent* (for brevity, we will refer to it

as an "agent"). The agent facilitates joining Synergy via identifier assignment and utility estimation (details are given in Section II-C). Highly loaded or bandwidth-constrained nodes are eliminated from consideration for becoming agents. The selected agent can be the host with the estimated (i) highest number of co-located overlay nodes, or (ii) highest number of overlays represented in its current or past neighbors in the Synergy network, or (iii) minimum maximum delay to other hosts in its home overlay network. In this context, *co-located nodes* denote a single host connected to more than one overlay network, but the definition can be extended to hosts in the same subnet, or hosts in the same autonomous system (AS).

Hosts with co-located overlay nodes (heuristic (i)) are good candidates since they can serve as agents for multiple overlays, and reduce traffic among agents. Traffic can also be reduced if the agents of different overlays are in close proximity. In heuristic (ii), a host which is close to hosts from several other overlay networks is considered a good candidate. In heuristic (iii), all hosts in the home overlay can rapidly contact their agent since the agent is closest to the overlay centroid. Primitives such as those in [12] can supply network information required for agent selection. Observe that every agent must have a backup agent, which replaces the overlay agent when it fails or leaves its home overlay.

All agents form a delay-based overlay mesh, called the *agent network*. A bootstrap mechanism allows the discovery of other overlays and their agents. We assume that the address of at least one bootstrap node is globally known. Either one of the agents (which we refer to as the *primary agent*) or an independent server can serve as the Synergy bootstrap node. Each agent is restricted to connect to less than UBA (the upper bound in the agent network) neighbors on the agent mesh, where UBA is determined based upon a tradeoff between performance and scalability.

B. Utilizing the Synergy Network

The Synergy overlay internetwork (or "Synergy network" for short) constitutes hosts exported (i.e., temporarily contributed) by autonomous, but cooperating overlays. The Synergy network is a new overlay network that incorporates *selected* hosts from more than one overlay network. This form of cooperation is analogous to the notion of code sharing in airline systems. For example, a customer who has purchased a ticket from airline A actually uses airline B if the two air carriers have an agreement to share certain routes. In this case, airline A code shares certain routes with airline B, while still operating its own routes. Airlines employ code sharing for increased efficiency and better resource utilization. From the customer perspective, the notion of code sharing can reduce costs and improve customer experience.

Overlay flows are classified based on the sender and receiver identities in the packet header. Long-lived flows are better-suited for exploiting the Synergy *cooperative forwarding service*, since short-lived flows are unlikely to have time to find Synergy routes that are better than their home overlay routes. Each sender keeps track of the duration of its flows. A flow is declared *long-lived* once its time duration exceeds a predefined threshold. In this case, all the nodes on the overlay flow path (including sender and receiver) are exported, so that the sender can utilize the Synergy network to send to the receiver. Once these hosts are exported, their overlay flows use Synergy routing paths (instead of home overlay paths) to deliver packets. Synergy routing paths are computed *completely independently* of routing protocols of home overlays. Therefore, Synergy must support all routing metrics of participating overlays, as discussed in Section II-D. Observe that Synergy forwarding need not only be unicast – with replication, multicast and peer-to-peer overlays can be supported.

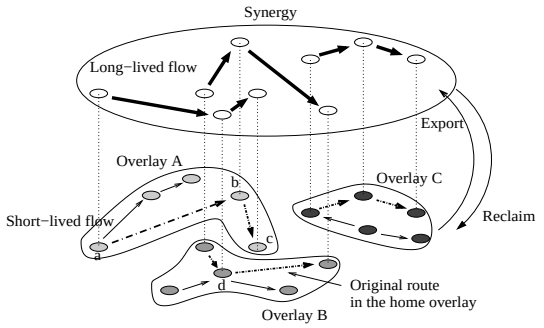


Fig. 1. Overlay internetworking architecture: An overlay internetwork is constructed among exported hosts. Long-lived fbws (dash-dot lines) in home overlays are replaced by Synergy routes (thick lines) if this will improve performance.

While a host is part of Synergy, the host executes both home overlay routing (for nodes on its home overlay) and Synergy routing. A host in Synergy thus maintains two forwarding tables: one for its home overlay, and another for Synergy. An overlay flow in Synergy routing can exploit hosts from both its home overlay and other overlays as intermediate hops on routing paths. This helps circumvent network hot spots by making a larger pool of hosts available as potential transit nodes. Fig. 1 depicts an example of Synergy forwarding. In the figure, the communication from sender host a to receiver host c in overlay network A benefits from cooperation between overlays A and B. The route from a to c employs host d in overlay B as a transit node. This cooperation can yield shorter (or wider) routes than the routes that only transit hosts in overlay network A.

C. Host Export

An exported host is assigned a unique identifier (e.g, a serial number) by the primary agent. Thus, hosts with the same identifier in different overlays (e.g., same nodeId in two Pastry overlays) can be easily distinguished in Synergy. An identifier from a leaving host can later be reused by new Synergy hosts. A mapping table (SynergyId, IP_address) is maintained by agents and distributed to exported hosts as well as hosts that utilize Synergy. Packets on Synergy can be thus routed using Synergy identifiers which map to IP addresses.

Operations for exporting and reclaiming hosts are facilitated by overlay agents. Hosts, however, communicate within

Synergy *without* the intervention of agents once they have successfully been exported. A host can leave Synergy by notifying its neighbors. The neighbors then disconnect the links to that host. When new hosts are exported to the Synergy network, an approximately equivalent number of hosts previously exported from the same home overlay (if any) may leave Synergy, i.e., be *reclaimed*, to maintain the Synergy network size manageable. Such hosts are clearly not allowed to leave Synergy if being used as intermediate nodes by any flow in the Synergy network.

The choice of which hosts to export (*in addition to* hosts in long-lived flows) significantly impacts the performance of Synergy. At least four heuristics may be used to make good export choices. First, an exported host should be likely to be useful to other overlays, either because it possesses unique resources/capabilities, or because of its strategic location. For example, an exported host in close proximity to several exported hosts may not significantly enhance delay performance, since it likely offers similar delays to these hosts in its close proximity. Second, if the exported host is already heavily loaded, it may not be as useful to other overlays as a lightly loaded host. Third, loads on different overlays may be considered, not just loads on hosts. For example, an overlay which is streaming audio can typically export a larger number of hosts than an overlay streaming high bandwidth video. Fourth, a trust-based priority mechanism must be used to determine which overlays are "cooperative," as we will discuss in Section II-D. The *export utility* of a host can therefore be computed based upon any subset of these factors.

In our implementation, the export utility approximates the number of hosts that are likely to transit this host when forwarding their packets on Synergy, multiplied by their performance gain when utilizing this host. To estimate this, we use the number of times this host serves as a next hop in its home overlay, which gives an indication of how strategically located this host is, divided by the number of possible connections. Unfortunately, comparing this number for all hosts in an overlay is $O(n^2)$ (where n is the number of overlay hosts), which is prohibitive for large overlays. Therefore, in our implementation, only p random hosts are used for computing the number of entries (where a host is the next hop) for q selected hosts. The number of entries examined in each routing table is limited to r. The agent estimates the export utility of each of the q hosts as the number of times they appear as next hop in the selected r entries of the routing tables of these selected p hosts. As long as $p, q, r \ll n$, the export utility computation is scalable. In the case of application level multicast, utility can be estimated as the size of the subtree rooted at that host. The utility value is periodically estimated, and hosts that exceed a specified threshold are exported, while those that fall below another threshold are reclaimed.

D. Routing and Priority Mechanisms

Our routing protocol employs three mechanisms to ensure that the Synergy network is fairly utilized. First, the inbound load at a host is limited to k overlay flows. This ensures that the host is not overloaded by many transit flows. Second, only one overlay flow is allowed to select its routes at a time. We serialize overlay flows for *route updates* in the order of flow identifiers. The primary overlay agent coordinates this process. Note that packets can still be forwarded concurrently while routing decisions are serialized. Third, an overlay flow is allowed to utilize host h as a transit node only if that host has a lower or equal priority than the source and destination of the flow, i.e., if $maximum(priority(d_i)) \geq priority(h)$, where d_i denotes source and destination hosts of a flow. We define $priority(d_i)$ as the number of overlay flows that d_i has assisted in the recent past by acting as a Synergy transit node. Our implementation computes $priority(d_i)$ using an exponentially weighted moving average (EWMA) of the number of overlay flows for a certain time duration. At time t, $priority(d_i)$ becomes $(1-\alpha) \times priority(d_i) + \alpha \times NumFlows(t)$. Observe that $maximum(priority(d_i))$ will increase if the source and destination have recently served as transit nodes through which other overlay flows have been routed. This approach ensures basic fairness in relationships among cooperating overlays. If an overlay network has refused to export potentially high-priority hosts in the recent past, its flows are allowed to use fewer hosts from other overlays due to the lower priority of these flows.

To compute routing paths, Synergy runs a link-state routing protocol on top of a network with overlay links for all possible node pairs. Each node maintains overlay link properties to all the other nodes for different metrics. Bandwidth is the most appropriate choice for overlays for high-throughput bulk data delivery (e.g., file download service in peer-to-peer systems), while real-time streaming overlays are latency-sensitive. An overlay flow on Synergy chooses the most representative primary routing metric(s), depending on its home overlay routing goals. Since each Synergy flow determines its routes independently of routes in home overlays, overlays with heterogeneous performance goals can cooperate. The overlay link available bandwidth is estimated through the TCP throughput formula as a function of round-trip latency and packet loss.

III. Performance Evaluation

We have analyzed the performance of our overlay internetworking implementation via both Internet experiments and simulations. Due to space limitations, we omit results from several Internet experiments, and simulation results with underlying routing pathologies; these results can be found in [10].

A. Implementation

We have implemented the techniques described in Sections II-A to II-D as a proof-of-concept for Synergy, with the exception of heuristic-based agent selection (Section II-A), which is included in our future work plans. Currently, we simply assign a random overlay host as the overlay agent. The primary goal of our implementation is to *transparently* provide a host with cooperative overlay services. Synergy is a separate component running on the same hosts where home overlays

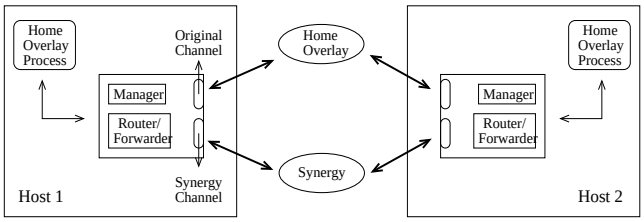


Fig. 2. Synergy implementation

are deployed. Synergy is implemented at the user-level, so as *not* to require superuser privileges or kernel modifications.

As shown in Fig. 2, Synergy comprises three modules: the channel, manager, and router/forwarder. Packets from the home overlay process are diverted to the Synergy component using divert mechanisms supported by several operating systems (e.g., ipchain on Linux). For our experiments, however, we emulate the divert mechanism via inter-process communication to avoid requiring superuser privileges. Therefore, our Synergy component sends periodic signals to a pre-specified port to which the home overlay process can listen.

The manager controls host export/reclaim, and incoming packet classification. If a host is exported, packets are sent to the Synergy channel; otherwise, the manager directs packets to the home overlay channel. Packets from the home overlay channel are transmitted following the paths established by the home overlay. The router/forwarder computes the Synergy routing paths. We reuse part of the RON implementation [3], specifically (i) classified routing and forwarding for different routing metrics (but no policy routing), (ii) ping protocols, and (iii) the performance database. The Synergy channel forwards its packets to the Synergy component at the next hop computed by this router/forwarder. When a packet arrives at the Synergy channel of the final destination, the channel passes the packet to the home overlay process at that destination.

B. Experimental Setup

We have implemented Synergy in C++ and conducted Internet experiments on PlanetLab [11]. We experiment with 8 RON overlays where each overlay contains 8 hosts. Therefore, we have a total of 64 nodes: 43 nodes belong to universities or companies in the United States, and the remaining 21 are located in Asia, Europe, Canada, and Australia. All the overlays except one have at least one host which is located outside the United States. Our experiments were conducted over several days in April−June 2004. We present in this paper a representative subset of the results.

In each overlay, a randomly chosen data source generates data streams to three other overlay nodes. Two streams are generated per source-destination pair, i.e., the total number of overlay flows is $2 \times 3 \times 8 = 48$. For the two streams per pair, one stream is transmitted over the home overlay only, while the other stream is concurrently transmitted over the Synergy network. The flows are bulk data transfers with application rate 1.7 Mbps. We use $r = 5$ for computing the export utility (p and q are set to all overlay hosts). The export algorithm uses utility threshold values 0.2 and 0.02. The time duration to detect long-lived flows is set to 5 seconds. The priority is updated every 14 seconds with $\alpha = 0.1$, and we use $k = 3$ for the inbound load. Latency and loss are measured by generating probe packets, and throughput is estimated from the TCP equation.[1]

C. Experimental Results

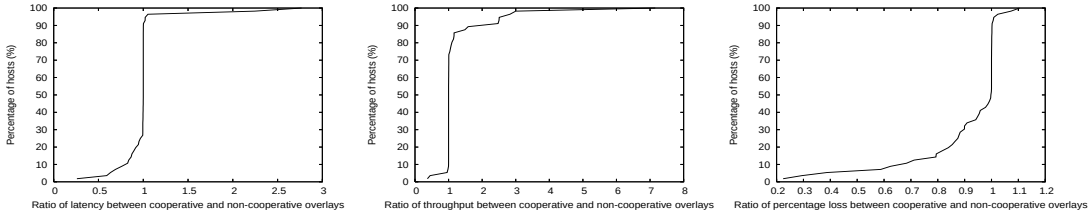
Fig. 3(a) illustrates the cumulative distribution of the ratio of latency observed with Synergy and that observed with independent home overlays, averaged for all hosts. The x-axis indicates the value of the ratio and the y-axis denotes the percentage of hosts that have latency ratios less than this value. The results are collected from a total of 120,000 latency samples from the sources to receivers. We find that Synergy reduces the latency of approximately 50% hosts compared to home overlays (30% hosts show substantial decrease while 20% exhibit marginal decrease). Synergy improves latency by a factor of five or more for many home overlay routing paths. About 45% of hosts exhibit a ratio close to one. The latency ratio for less than 5% of hosts is larger than one, due to overhead from the Synergy divert mechanism (inter-process communication overhead) and due to overloaded hosts.

Synergy also increases throughput for most participating hosts. To compare throughput with and without Synergy, we again measured and averaged 120,000 throughput samples at receivers. In Fig. 3(b), we plot the cumulative distribution of the ratio of the average Synergy throughput to the average throughput achieved by independent home overlays. The distribution reveals that about 52% of the hosts achieve higher throughput via Synergy than home overlays alone by a maximum factor of seven (9% hosts show substantial increase while 43% exhibit marginal increase). Among the remaining hosts, 44% use the same paths, while 4% traverse worse paths in Synergy than in their home overlays.

Fig. 3(c) illustrates the cumulative distribution of the ratio of loss percentage results obtained from Synergy and those observed with independent home overlays in the same experiment. About 50% of hosts in Synergy improve loss 1 to 5 times over independent home overlays. The remaining 50% exhibit a ratio close to or slightly larger than one. Again, Synergy packet processing overhead and overloaded hosts are possible reasons. In addition, the mechanism for RON loss measurements and averaging may give inaccurate long-term averages. Finally, RON uses bi-directional link information for uni-directional loss measurement assuming symmetric loss percentages. Network paths, however, are found to be asymmetric in losses. Our results also indicate that a route optimized for one of the three metrics promises high performance in terms of the other two metrics (see [10] for details).

To study how heterogeneous overlays maintain their individual performance goals, we examine the performance of

[1]To compute latency, the source timestamps each data packet sent. The latency is estimated when the packet arrives at the destination, and hence clock skew can introduce inaccuracies. Since our purpose is coarse granularity comparison between home overlays and Synergy, the results obtained are considered acceptable.



(a) Cumulative distribution of the ratio of average latency with Synergy to that achieved by independent home overlays.

(b) Cumulative distribution of the ratio of average throughput with Synergy to that achieved by independent home overlays.

(c) Cumulative distribution of the ratio of average loss percentage with Synergy to that achieved by independent home overlays.

Fig. 3. Latency, throughput, and loss comparison of cooperative and independent overlays.

Synergy versus independent home overlays for the particular metric of importance to each overlay (latency, loss, or throughput) in Table I. Synergy gives improvements in all the overlays for their respective metrics, except overlay 4. Overlay 5 suffered from performance variation as conditions changed significantly among experiments. Latency exhibits the most significant improvement – approximately 10% or more for overlays 1-3. Thus, Synergy allows heterogeneous overlays to maintain their routing goals while cooperating. More complex scenarios with multiple independent constraints, and overlays with heterogeneous service goals will be investigated in our future work.

TABLE I

EXPERIMENTAL RESULTS FOR THREE PERFORMANCE GOALS

Overlay	Metric	Mean	Max	Min
1	Latency	0.8986	1.0002	0.6347
2	Latency	0.8388	1.0003	0.2587
3	Latency	0.9363	1.0000	0.6898
4	Throughput	0.9997	1.0056	0.9817
5	Throughput	1.1341	2.9992	0.3708
6	Throughput	1.0011	1.0039	0.9999
7	% Loss	0.9663	1.0235	0.8392
8	% Loss	0.9573	1.0022	0.7945

The primary objective of cooperative forwarding is to overcome path outages that are commonly experienced in the Internet. As discussed in [3], an outage can be defined as the length of time during which no packets get through a path. We consider 30 and 50% loss as points at which performance is unacceptable. To measure outages in home overlays and Synergy, we compare the averaged packet loss in both cases, following a methodology similar to [3]. Fig. 4 shows that the majority of points in the scatter plot are located below the $y = x$ line, i.e., Synergy has lower packet loss than home overlays in most cases. There is a higher number of points for home overlays above 0.3 or 0.5 loss (corresponding to 30 or 50% loss) than Synergy. This means that Synergy decreases the number of path outages. The figure also illustrates that there are approximately 10 points with $x = 1$ and $y < 1$. For such network paths, Synergy helps overlay connections route around outages existing in home overlays. Many points

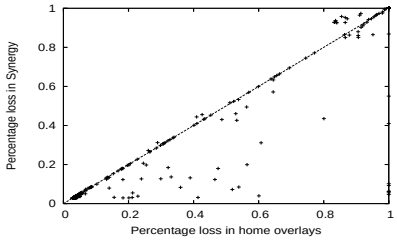


Fig. 4. Average packet loss for 660 network paths in home overlays and Synergy, where half of the paths are in home overlays and the other half is in Synergy. In the figure, 112 points exceed 0.3 for Synergy (horizontal line) while 128 points exceed 0.3 for home overlays (vertical line). In addition, 89 points exceed 0.5 for Synergy, but 105 points exceed 0.5 for home overlays.

are concentrated around the point of $x = 0$ and $y = 0$. Some points, however, can also be found around the point of $x = 1$ and $y = 1$. The packet loss for Synergy exceeds home overlays for a few of these points. We also observe that only 10% of transit nodes participate in relaying over 10,000 packets (see [10]). This result validates the conjecture that a few strategically located hosts can play a major role in Synergy routing, which confirms the effectiveness of our host export mechanism.

As previously discussed, Synergy imposes additional delay for diverting between the Synergy component and the ports of a home overlay. This is similar to the delay incurred by divert sockets to forward data in the home overlay itself, which is about 220 ms [3]. Since the current version of Synergy reuses the routing and probing mechanisms of RON, a similar overhead is added. Based upon [3], RON routing and probe traffic would be about 14 kbps for 32 hosts (this is the number of hosts which use Synergy in our experiment).

In addition to routing and probe traffic, Synergy injects extra traffic to broadcast the export utility and priority of each host. We analyze this overhead similar to the analysis in [3]. Our analysis shows that the bandwidth consumed by this traffic is $\frac{1280m(m-1)}{14} = 91.43m(m-1)$ bps where m is the number of hosts in the home overlay. Hosts also compute and send their priority value to the agent. The traffic consumed by an overlay in this case is $58n+60+10n+(48+10l)(n+1)$ bytes where $l = 8n$ (details in [10]). This overhead is reasonable, and can

be reduced by increasing the intervals, or the values of p and q.

IV. Related Work

The idea of overlay cooperation has been studied in different contexts. A grand challenge in networking research discussed in a recent NSF-sponsored workshop report [7] is "simultaneously co-existing overlays." Broadcast Federation [13] investigated cooperation among multicast/broadcast overlay and IP-level networks. Our focus, however, is not limited to overlay broadcasting – we consider more general characteristics of overlay interactions. Nakao *et al.* [12] have designed a shared routing underlay that provides large-scale and coarse-grained network layer information to overlay nodes. Tomography-based overlay network monitoring that requires $O(n)$ measurements for all the $O(n^2)$ overlay paths was proposed in [14]. The approach reduces measurement overhead, at the expense of computational overhead. MACEDON [15] is an infrastructure that enables users to generate code for overlay algorithms using a concise script language. Operating system support and network management for large-scale overlay networks were studied in [16]. Open DHT [17] is a publicly accessible distributed hash table (DHT) service that clients can use instead of running their own DHT nodes. The primary focus of Open DHT is on DHT service availability while ours is on general overlay interactions.

Several theoretical studies (e.g., [18], [19]) have recently shown that selfish routing may cause suboptimal performance. In contrast, Qiu *et al.* [8] have reported that selfish source and overlay routing indeed achieve close to optimal average latency in Internet-like environments, at the expense of significantly increased congestion on certain links. Overlay networks that detect performance degradation of current routing paths and re-route through other hosts include RON [3]. Rewaskar and Kaur [20] recently studied the tradeoff between performance gain in an overlay network and overhead incurred.

V. Conclusions and Future Work

We have designed an overlay internetworking architecture, Synergy, that supports transparent interactions among autonomous, possibly heterogeneous, overlay networks to improve performance and promote information sharing. To the best of our knowledge, this is the first work to propose a deployable and general overlay internetworking system. We have implemented a Synergy prototype and performed experiments on PlanetLab. We found that Synergy improves the performance of overlay connections with respect to latency, throughput, and loss.

We plan to conduct larger-scale experiments with heterogeneous overlays (e.g., complex constraint-based routing systems, overlay multicast, and peer-to-peer systems). These experiments will allow us to quantify the effectiveness and practicality of overlay cooperation in realistic environments. We will also extend Synergy to support other types of cooperative overlay services, and study the complexity, scalability, and security issues involved. Eventually, we hope that our Synergy system can serve as an enabler for the design and evaluation of revolutionary Internet routing and service architectures.

VI. Acknowledgments

This research has been sponsored in part by NSF grant ANI-0238294 (CAREER). The authors would like to thank David G. Andersen (CMU) for the RON code; Suman Banerjee (U. of Wisconsin) for the myns simulator; Akihiro Nakao (Princeton U.) for answering many of our questions; and Kihong Park, Dongyan Xu, and David Yau (Purdue U.) for several inspiring discussions.

References

[1] J. Saltzer, D. Reed, and D. Clark, "End-To-End Arguments In System Design," *ACM Transactions on Computer Systems*, pp. 277–288, November 1984.

[2] Y. Chu, S. Rao, S. Seshan, and H. Zhang, "Enabling Conferencing Applications on the Internet using an Overlay Multicast Architecture," in *Proc. of ACM SIGCOMM*, August 2001, pp. 55–67.

[3] D. Andersen, H. Balakrishnan, M. Kaashoek, and R. Morris, "Resilient Overlay Networks," in *Proc. of ACM SOSP*, October 2001, pp. 131–145.

[4] J. Byers, J. Considine, M. Mitzenmacher, and S. Rost, "Informed Content Delivery Across Adaptive Overlay Networks," in *Proc. of ACM SIGCOMM*, August 2002.

[5] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Wide-area Cooperative Storage with CFS," in *Proc. of ACM SOSP*, 2001, pp. 202–215.

[6] A. Rowstron and P. Druschel, "Pastry: Scalable, Decentralized Object Location and Routing for Large-scale Peer-to-Peer Systems," in *Proc. of ACM/IFIP Middleware*, 2001.

[7] M. Ammar and Co-authors, "Report of the National Science Foundation Workshop on Fundamental Research in Networking," April 2003, http://www.cs.virginia.edu/∼jorg/workshop1.

[8] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker, "On Selfish Routing in Internet-Like Environments," in *Proc. of ACM SIGCOMM*, August 2003.

[9] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe, and A. Warfield, "Plutarch: An Argument for Network Pluralism," in *Proc. of the ACM SIGCOMM Workshop on Future Directions in Network Architecture*, August 2003.

[10] M. Kwon and S. Fahmy, "Synergy: An Overlay Internetworking Architecture," Technical Report, July 2005, available at http://www.cs.purdue.edu/homes/fahmy/.

[11] L. Peterson, T. Anderson, D. Culler, and T. Roscoe, "A Blueprint for Introducing Disruptive Technology into the Internet," in *Proc. of the HotNets-I*, October 2002.

[12] A. Nakao, L. Peterson, and A. Bavier, "A Routing Underlay for Overlay Networks," in *Proc. of ACM SIGCOMM*, August 2003.

[13] Y. Chawathe and M. Seshadri, "Broadcast Federation: An Application-layer Broadcast Internetwork," in *Proc. of ACM NOSSDAV*, May 2002, pp. 117–126.

[14] Y. Chen, D. Bindel, and R. Katz, "Tomography-based Overlay Network Monitoring," in *Proc. of ACM SIGCOMM Internet Measurement Conference*, October 2003.

[15] A. Rodriguez, C. Killian, S. Bhat, D. Kostic, and A. Vahdat, "MACEDON: Methodology for Automatically Creating, Evaluating, and Designing Overlay Networks," in *Proc. of USENIX NSDI*, March 2004.

[16] A. Bavier et al., "Operating System Support for Planetary-Scale Network Services," in *Proc. of USENIX NSDI*, March 2004.

[17] B. Karp, S. Ratnasamy, S. Rhea, and S. Shenker, "Spurring Adoption of DHTs with OpenHash, a Public DHT Service," in *Proc. of IPTPS*, February 2004.

[18] E. Koutsoupias and C. Papadimitriou, "Worst-Case Equilibria," in *Proc. of the 16th Annual Symposium on Theoretical Aspects of Computer Science*, 1999, pp. 404–413.

[19] T. Roughgarden and E. Tardos, "How Bad is Selfish Routing?" *Journal of ACM*, vol. 49, no. 2, pp. 236–259, 2002.

[20] S. Rewaskar and J. Kaur, "Testing the Scalability of Overlay Routing Infrastructures," in *Proc. of the Passive and Active Measurements Workshop*, April 2004.