An Overlay Architecture for Throughput Optimal Multipath Routing

Nathaniel M. Jones, Georgios S. Paschos, Brooke Shrader, and Eytan Modiano, *Fellow, IEEE*

***Abstract*— Legacy networks are often designed to operate with simple single-path routing, like the shortest path, which is known to be throughput suboptimal. On the other hand, previously proposed throughput optimal policies (i.e., backpressure) require every device in the network to make dynamic routing decisions. In this paper, we study an overlay architecture for dynamic routing, such that only a subset of devices (overlay nodes) need to make the dynamic routing decisions. We determine the essential collection of nodes that must bifurcate traffic for achieving the maximum multi-commodity network throughput. We apply our optimal node placement algorithm to several graphs and the results show that a small fraction of overlay nodes is sufficient for achieving maximum throughput. Finally, we propose a threshold-based policy (BP-T) and a heuristic policy (OBP), which dynamically control traffic bifurcations at overlay nodes. Policy BP-T is proved to maximize throughput for the case when underlay paths do no overlap. In all studied simulation scenarios, OBP not only achieves full throughput but also reduces delay in comparison to the throughput optimal backpressure routing.**

***Index Terms*— Overlay networks, network control, backpressure routing.**

I. Introduction

WE STUDY optimal routing in networks where some legacy nodes are replaced with overlay nodes. While the legacy nodes perform only forwarding on pre-specified paths, the overlay nodes are able to dynamically route packets. *Dynamic backpressure* is known to be an optimal routing policy, but it typically requires a homogeneous network, where all nodes participate in control decisions. Instead, we assume that only a subset of the nodes are controllable; these nodes form a network overlay within the legacy network. The choice of the overlay nodes is shown to determine the *throughput region* of the network.

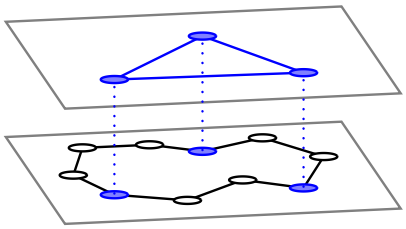


Fig. 1. Example of a network overlay. The bottom plane shows the full network graph, while the top plane shows a subset of network nodes and their conceptual overlay connectivity. In this work we study network throughput under the assumption that overlay nodes implement dynamic routing schemes and underlay nodes forward packets using pre-specified paths.

A first finding is that ring networks require exactly 3 controllable (overlay) nodes to enable the same throughput region as when all nodes are controllable, independent of the total number of nodes in the network. Motivated by this, we develop an algorithm for choosing the minimum number of controllable nodes required to enable the full throughput region. We evaluate our algorithm on several classes of regular and random graphs. In the case of random networks with a power-law degree distribution, which is a common model for the Internet, we find that fewer than 80 out of 1000 nodes are required to be controllable to enable the full throughput region.

Since standard backpressure routing cannot be directly applied to the overlay setting, we develop extensions to backpressure routing that determine how to route packets between overlay nodes. We confirm that maximum throughput can be attained with our policies in several scenarios, when only a fraction of legacy nodes are replaced by controllable nodes. Moreover, we observe reduced delay relative to the case where all nodes are controllable and operate under backpressure routing.

A. Motivation and Related Work

Backpressure (BP) routing, first proposed in [16], is a throughput optimal routing policy that has been studied for decades. Its strength lies in discovering multipath routes and utilizing them optimally without knowledge of the network parameters, such as arrival rates, link capacities, mobility, fading, etc. Nevertheless, the adoption of this routing policy has not been embraced for general use on the Internet. This is due, in part, to an inability of backpressure routing to coexist with legacy routing protocols. With few exceptions, backpressure routing has been studied in homogeneous networks, where all nodes are dynamically controllable and implement the backpressure policy across all nodes uniformly. As will be shown, backpressure routing — as proposed in [16] —

Manuscript received April 18, 2016; revised December 6, 2016 and April 24, 2017; accepted April 26, 2017; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor S. Chong. Date of publication May 31, 2017; date of current version October 13, 2017. The Lincoln Laboratory portion of this work was sponsored by the Department of the Air Force under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the authors and are not necessarily endorsed by the United States Government. This work was also supported by NSF grant CNS-1524317, and by DARPA I2O and Raytheon BBN Technologies under Contract No. HROO 1 l-l 5-C-0097. The work of G. S. Paschos was supported in part by the WiNC project of the Action 'Supporting Postdoctoral Researchers,' funded by national and Community funds (European Social Fund).

N. M. Jones was with the Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, MA 02421 USA. He is now with Akamai Technologies, Cambridge, MA 02142 USA (e-mail: njones@akamai.com).

G. S. Paschos was with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA. He is now with the Mathematical and Algorithmic Sciences Lab, France Research Center, Huawei Technologies, 92100 Boulogne-Billancourt, France (e-mail: georgios.paschos@huawei.com).

B. Shrader is with the Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, MA 02421 USA (e-mail: brooke.shrader@ll.mit.edu).

E. Modiano is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: modiano@mit.edu).

Digital Object Identifier 10.1109/TNET.2017.2703867

is suboptimal when applied only to a subset of nodes in the network.

Techniques to provide throughput-optimal multipath routing have been explored in various contexts. The work in [3] considers the problem of setting link weights provided to the Open Shortest Path First (OSPF) routing protocol such that, when coupled with bifurcating traffic equally among shortest paths, the network achieves throughput equal to the optimal multicommodity flow. The authors of [17] use an entropy maximization framework to develop a new throughput-optimal link state routing protocol where each router intelligently bifurcates traffic for each destination among its outgoing links. These techniques all require centralized control, universal adoption by all network nodes, or both; thus none of these techniques could provide incremental deployment of throughput optimal routing to wireless networks. Moreover, these techniques cannot be used in conjunction with throughput optimal dynamic control schemes, such as backpressure.

We would like to enable new network control policies to be deployed in existing networks, alongside legacy nodes that are unaware of the new control policies. There are many reasons to integrate controllable nodes into heterogeneous networks in a gradual manner, not the least of which is the financial cost of replacing all nodes at once. Other reasons include a need to maintain compatibility with current applications and special purpose hardware, a lack of ownership to decommission legacy equipment, and a lack of administrative privilege to modify existing software.

Conceptually, we model controllable nodes as operating in a network overlay on top of a legacy network. Network overlays are frequently used to deploy new communication architectures in legacy networks [13]. To accomplish this, messages from the new technology are encapsulated in the legacy format, allowing the two methods to coexist in the legacy network. Nodes making use of the new communication methods are then connected in a conceptual network overlay that operates on top of the legacy network, as shown in Fig. 1.

Several works have considered the use of network overlays to improve routing in the Internet. The work in [1] proposes resilient overlay networks (RON) to find paths around network outages on a faster timescale than BGP. Similarly, [5] proposed a method for choosing placement of overlay nodes to improve path diversity in overlay routes. While both of the preceding works show that their strategies choose high quality single-path routes, we go further and identify multipath routes that offer maximum throughput.

Delay reduction for BP routing has been studied in a variety of scenarios. While multipath routes are required to support the full throughput region, the exploratory phase of BP can lead to large queues when the offered load is low and single-path routes would suffice. In [9], a hybrid policy combining BP with shortest-path routing is proposed, where flows are biased towards shortest-path routes, yet still support the full throughput region. This hybrid policy is extended in [8] to also include digital fountain codes, and shown to achieve good end-to-end delay performance in the presence of random link failures. The work in [18] develops a policy that achieves a similar shortest-path result by minimizing the average hop count used by flows. In a scenario with multiple clusters that are intermittently connected, [15] combines BP with source routing in a network overlay model to separate the queue dynamics of intra-cluster traffic from longer inter-cluster delays. The work in [2] applies shadow queues to allow the use of per-neighbor FIFO queues instead of per-commodity queues, as is typical with differential backlog routing, and finds that this can improve network delay. A loop-free backpressure policy is developed in [14] that dynamically finds acyclic graphs for reducing delay while maintaining throughput optimality. These prior works assume a homogeneous scenario where all nodes use the same control policy and thus differ fundamentally from our approach. Our proposed algorithms for applying backpressure in overlay networks can help reduce delay by reducing the number of nodes between which differential backpressure is formed. While our original motivation for studying backpressure in overlay networks was not to reduce delay, we believe that our scheme can be used as part of a delay-reducing solution.

This paper is based on preliminary work that appeared in [7] and [11].

B. Problem Statement and Contributions

We consider two problem areas for control of heterogeneous networks. First, we develop algorithms for choosing the placement of controllable nodes, where our goal here is to allocate the minimum number of controllable nodes such that the full network stability region is available. Second, given any subset of nodes that are controllable, we also wish to develop an optimal routing policy that operates solely on these nodes.

In the first problem area, we are given a graph G with nodes $\mathcal{N}$ supporting shortest-path routes between each pair of nodes. We wish to identify a minimal set of controllable nodes $\mathcal{V} \subseteq \mathcal{N}$ such that if only these nodes are allowed to bifurcate traffic, maximum throughput can be achieved. Ideally, we would like to solve P1,

$$V_1^* = \min_{\mathcal{V} \subseteq \mathcal{N}} |\mathcal{V}| \quad \text{s.t. } \Lambda_G(\mathcal{V}) = \Lambda_G(\mathcal{N}), \tag{P1}$$

where $\Lambda_G(\mathcal{V})$ is the throughput region (i.e., the set of multi-commodity arrival rate vectors that can be stably supported by the network) for graph G when only nodes $\mathcal{V}$ are controllable, while $\Lambda_G(\mathcal{N})$ is the throughput region when all nodes are controllable. Note that comparing throughput regions directly can be difficult, so instead we identify a condition that is necessary and sufficient to guarantee the full throughput region, and then we search for the minimal $\mathcal{V}$ that satisfies this condition.

In the second problem area, we consider the design of dynamic network control policies that operate only at controllable nodes $\mathcal{V}$. These controllable nodes are connected by "tunnels" or paths through uncontrollable sections of the network, where the control policy can choose when to inject packets into a tunnel but the tunnel itself is uncontrollable. We develop an overlay control policy that stabilizes all arrival rate vectors in $\Lambda_G(\mathcal{V})$ for the case when tunnels do not overlap. We also develop a heuristic overlay control policy for use on

general topologies, and show through simulation that stability is achieved for all arrival rates considered.

Our solutions for the first and second problem areas are complementary, in the sense that they can be used together to solve the joint problem of providing maximum throughput when only a subset of nodes are controllable. However, our solutions can also be used in isolation; our node placement algorithm can be used with other control policies, and our BP extensions can yield maximal stability with any overlay node placement and legacy single-path routing.

Our contributions are summarized below.

- We formulate the problem of placing the minimum number of overlay (controllable) nodes in a legacy network in order to achieve the full multicommodity throughput region and provide an efficient placement algorithm.
- We apply our placement algorithm to several scenarios of interest including regular and random graphs, showing that in some cases, only a small fraction of overlay nodes is sufficient for maximum throughput.
- We propose a threshold-based control policy — BP-T — as a modification of BP for use at overlay nodes, and prove this policy to stabilize all arrival rates in $\Lambda_G(\mathcal{V})$ when tunnels do not overlap.
- We propose a heuristic overlay BP policy — OBP — for use at overlay nodes on general topologies. We show via simulation that OBP can outperform BP when limited to control at overlay nodes, and that OBP also has better delay performance compared to BP with control at all nodes.

II. Model

We model the network as a directed graph $G = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of nodes in the network and $\mathcal{E}$ is the set of edges. We assume that the underlay network provides a fixed realization for shortest-path routes between all pairs of nodes, and that uncontrollable nodes will forward traffic only along the given shortest-path routes. Further, we assume that only one path is provided between each pair of nodes. Let P_{ab}^{SP} be the shortest path from a to b, and let $\mathcal{P}^{\text{SP}} = (P_{ab}^{\text{SP}})$, for all pairs $a, b \in \mathcal{N}$, be the set of all shortest paths provided by the underlay network. If (i, j) is a link in G, then we assume that the single hop path is available, i.e. $P_{ij}^{\text{SP}} \in \mathcal{P}^{\text{SP}}$. Whenever a packet enters a forwarding node, the node inspects the corresponding routing table and sends the packet towards the pre-specified path. Therefore, the performance of the system depends on the available set of paths $\mathcal{P}^{\text{SP}}$. *Optimal substructure* is assumed for shortest-paths, such that if shortest-path P_{ac}^{SP} from node a to c includes node b, then path P_{ac}^{SP} includes shortest-paths P_{ab}^{SP}, from a to b, and P_{bc}^{SP}, from b to c. This optimal substructure is consistent with shortest-paths in OSPF, a widely used routing protocol based on Dijkstra's shortest-path algorithm [13], where OSPF allows for the use of lowest next-hop router ID as a method for choosing between multiple paths of equal length.

Next, we consider the subset of nodes $\mathcal{V} \subseteq \mathcal{N}$, called *overlay* or *controllable* nodes, which can bifurcate traffic by directing packets to the destination or other controllable nodes along the provided shortest-path routes. Intuitively, these nodes $\mathcal{V}$ can improve throughput performance by generating new paths and enabling multipath routing. The remaining uncontrollable nodes $u \in \mathcal{N} \setminus \mathcal{V}$ provide only shortest-path forwarding in the underlay network, with an exception that any uncontrollable node u can bifurcate all traffic that originates at u; this may occur, for example, in the source applications at uncontrollable nodes, or in a shim-layer between the network-layer and application-layer. Without such an exception, all sources may be required to be controllable nodes.

Controllable nodes can increase the achievable throughput region by admitting new paths to the network as concatenations of existing paths from shortest-path routing. A *2-concatenation* of shortest-paths P_{av}^{SP} and P_{vb}^{SP} is an acyclic path from a to b, P_{ab}, where $v \in \mathcal{V}$ is a controllable node and v is the only node shared between shortest-paths P_{av}^{SP} and P_{vb}^{SP}. Note that a 2-concatenation of acyclic paths will always be acyclic, as we only allow the concatenated paths to share the overlay node v at which concatenation is performed. An *n-concatenation* is then the concatenation of n shortest-paths at $n-1$ controllable nodes, performed as a succession of $(n-1)$ 2-concatenations, and is therefore acyclic. Consider the set of paths $\mathcal{P}(\mathcal{V})$, which contains all underlay paths $\mathcal{P}^{\text{SP}}$ as well as all possible n-concatenations of these paths at the controllable nodes $\mathcal{V}$. We will see that this set $\mathcal{P}(\mathcal{V})$ plays a role in the achievability of the throughput region.

III. Throughput Region

The *throughput region* $\Lambda_G(\mathcal{V})$ is the set of all arrival rates that can be achieved by any policy implemented at controllable nodes $\mathcal{V}$ on graph G. For the case where all nodes are controllable, i.e., $\mathcal{V} = \mathcal{N}$, the throughput region equals the stability region of graph G. This section characterizes this region for a given set of paths $\mathcal{P}(\mathcal{V})$.

Packets destined for node c are called *commodity c* packets. Let λ_a^c be the rate of exogenous arrivals at node a for commodity c, and let $\boldsymbol{\lambda} = (\lambda_a^c)$ be the multicommodity arrival rate vector for all sources a and commodities c. Let $f_{ij}^{ab,c}$ be the *edge-flow* for commodity c on edge (i, j) along the shortest-path from node a to b. Flow for a path is allowed only on the edges along that path, i.e. $f_{ij}^{ab,c} = 0$ unless $(i, j) \in P_{ab}^{\text{SP}}$. Let $\bar{f}_{ab}^c$ be *path-flow* for commodity c along shortest-path P_{ab}^{SP}, from node a to b. Decision variable $v_i = 1$ if node i is controllable, and $v_i = 0$ otherwise, for all nodes $i \in \mathcal{N}$. The capacity of edge (i, j) is R_{ij}. The controllable throughput region $\Lambda_G(\mathcal{V})$ is then the set of all arrival rate vectors (λ_a^c) such that Eqns. (1-6) can be satisfied.

Flow Conservation:

$$\lambda_v^c = \sum_{b \in \{c, \mathcal{V} \setminus v\}} \bar{f}_{vb}^c - \sum_{d \in \mathcal{V} \setminus v} \bar{f}_{dv}^c, \quad \forall v \in \mathcal{V},\ c \in \mathcal{N} \setminus v \quad (1)$$

$$\lambda_u^c = \sum_{b \in \{c, \mathcal{V}\}} \bar{f}_{ub}^c, \quad \forall u \in \mathcal{N} \setminus \mathcal{V},\ c \in \mathcal{N} \setminus u \quad (2)$$

Path Constraint:

$$\bar{f}_{ab}^c = f_{ij}^{ab,c}, \quad \forall (i, j) \in P_{ab}^{\text{SP}},\ \forall a, b, c \in \mathcal{N} \quad (3)$$

Overlay Neighbor Constraints:

$$f_{ij}^{ab,c} \leq (1 - v_i) R_{ij}, \quad \forall (i, j) \in P_{ab}^{\text{SP}},\ a \neq i,\ \forall c \in \mathcal{N} \quad (4)$$

$$f_{ij}^{ab,c} \leq (1-v_j)R_{ij}, \quad \forall (i,j) \in P_{ab}^{\text{SP}},\ b \neq j,\ \forall c \in \mathcal{N} \quad (5)$$

Edge Rate Constraint:

$$\sum_{a,b,c} f_{ij}^{ab,c} \leq R_{ij}, \quad \forall\, (i,j) \in \mathcal{E} \quad (6)$$

Eqn. (1) represents flow conservation of commodity c packets at controllable node v. Here, exogenous arrivals at node v equal network departures minus (endogenous) network arrivals at v. Similarly, Eqn. (2) represents flow conservation for exogenous arrivals at uncontrollable nodes. The exogenous arrivals for commodity c at uncontrollable node u are equal to network departures on the shortest-path to destination c plus network departures along shortest-paths to controllable nodes. This is the special case where uncontrollable node u is a source, in that u can dynamically route exogenous arrivals but not endogenous network arrivals. Eqn. (3) is a path constraint for each commodity c along the shortest-path from node a to node b, where the path-flow equals the edge-flow for each edge along path P_{ab}^{SP}. Overlay neighbor constraints in Eqns. (4-5) force edge-flow $f_{ij}^{ab,c} = 0$ (and therefore path-flow $\bar{f}_{ab}^{c} = f_{ij}^{ab,c} = 0$) if node i or j is a controllable node intermediate to path P_{ab}^{SP}, i.e., for $i \neq a$ and $j \neq b$; such paths would remove routing ability from intermediate controllable nodes i or j. Eqns. (4-5) are necessary to allow for dynamic choice of controllable nodes, and are redundant with Eqn. (6) when nodes i and j both are uncontrollable. Finally, Eqn. (6) is an edge rate constraint for every edge (i,j), such that total flow over an edge is upper bounded by the edge capacity.

If there are no controllable nodes, i.e. $\mathcal{V} = \emptyset$, then Eqn. (2) simplifies to

$$\lambda_a^c = \bar{f}_{ac}^{c}, \quad \forall a, c \in \mathcal{N},\ a \neq c, \quad (7)$$

where Eqns. (4-5) can be ignored as they are always redundant with Eqn. (6). The throughput region without controllable nodes, $\Lambda_G^{\text{SP}} \equiv \Lambda_G(\emptyset)$, is thus limited to the set of arrival rate vectors $\boldsymbol{\lambda}$ such that Eqns. (7), (3) and (6) are satisfied. Indeed, these equations specify the shortest-path formulation for the throughput region on graph G, defined as $\Lambda_G^{\text{SP}} \equiv \Lambda_G(\emptyset)$.

If all nodes are controllable, i.e. $\mathcal{V} = \mathcal{N}$, then there are no constraints from underlay paths and all dynamic routing decisions are allowed. Eqns. (1) and (6) simplify to

$$\lambda_a^c = \sum_{b:(a,b)\in\mathcal{E}} \bar{f}_{ab}^{c} - \sum_{d:(d,a)\in\mathcal{E}} \bar{f}_{da}^{c}, \quad \forall a, c \in \mathcal{N},\ a \neq c, \quad (8)$$

$$\sum_{c} \bar{f}_{ab}^{c} \leq R_{ab}, \quad \forall\, (a,b) \in \mathcal{E}. \quad (9)$$

There are no uncontrollable nodes here, so Eqn. (2) is unused, and Eqns. (3), (4), and (5) are redundant with Eqns. (8) and (9). The full region $\Lambda_G \equiv \Lambda_G(\mathcal{N})$ is then defined as the set of arrival rate vectors $\boldsymbol{\lambda}$ that satisfy Eqns. (8-9). This is the largest region supported by network G.[1]

Any work-conserving policy with shortest-path routing can support the region $\Lambda_G(\emptyset)$, while backpressure routing is known to support the full region $\Lambda_G(\mathcal{N})$. However, how to achieve the heterogeneous region $\Lambda_G(\mathcal{V})$ with a dynamic routing policy is not generally known. For heterogeneous networks, converting an uncontrollable node u into a controllable node v relaxes the constraints for node u from Eqn. (2) into Eqn. (1). Note that when node v becomes controllable, the overlay neighbor constraints from Eqns. (4-5) become active.

[1] Also known as the fractional Multicommodity Flow feasibility region.

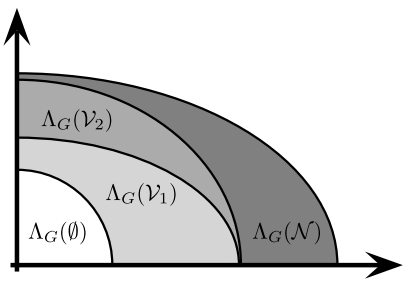


Fig. 2. Projection of throughput regions $\Lambda_G(\cdot)$ for sets of overlay nodes $\mathcal{V}_1, \mathcal{V}_2 : \mathcal{V}_1 \subseteq \mathcal{V}_2 \subseteq \mathcal{N}$, indicating subset relationship as described in Eqn. (10).

Recall that we assume optimal substructure for shortest-paths. We use this structure to find an additional property about the throughput region. Any path P_{ab}^{SP} that passes through a controllable node v can be split into two sub-paths P_{av}^{SP} and P_{vb}^{SP}, where optimal substructure guarantees that both sub-paths are in the set of underlay routes $\mathcal{P}^{\text{SP}}$. Node v can then concatenate these sub-paths to form the original path P_{ab}^{SP}. Therefore, if there exists a flow decomposition of $\boldsymbol{\lambda}$ that uses path P_{ab}^{SP}, then there is also a flow decomposition that uses sub-paths P_{av}^{SP} and P_{vb}^{SP}. Thus, with shortest-path routing, adding controllable nodes can allow the throughput region to grow, but never causes the region to shrink. This implies a subset relationship in the throughput region with shortest-path underlay routing, as represented in Fig. 2, such that for any overlay node sets $\mathcal{V}_1, \mathcal{V}_2 : \mathcal{V}_1 \subseteq \mathcal{V}_2 \subseteq \mathcal{N}$,

$$\Lambda_G^{\text{SP}} \equiv \Lambda_G(\emptyset) \subseteq \Lambda_G(\mathcal{V}_1) \subseteq \Lambda_G(\mathcal{V}_2) \subseteq \Lambda_G(\mathcal{N}) \equiv \Lambda_G. \quad (10)$$

IV. Optimal Placement of Overlay Nodes

We would like to place controllable nodes to solve P1, but the constraint $\Lambda_G(\mathcal{V}) = \Lambda_G(\mathcal{N})$ is difficult to evaluate directly. A simple implementation for P1 can use the fact that Λ_G is a convex polytope, choosing the minimum number of controllable nodes to satisfy all points in the throughput region, as

$$V_2^* = \min_{\mathcal{V}\subseteq\mathcal{N}} |\mathcal{V}| \quad \text{s.t. } \boldsymbol{\lambda}^{(i)} \in \Lambda_G(\mathcal{V}), \quad \forall \boldsymbol{\lambda}^{(i)} \in \Lambda_G, \quad \text{(P2)}$$

where the constraint enumerates all extreme points of Λ_G, and $\boldsymbol{\lambda}^{(i)}$ is the i^{th} such extreme point. Note that regions $\Lambda_G(\mathcal{V})$ and Λ_G are polytopes by Eqns. (1-6). Then $\Lambda_G(\mathcal{V}) \supseteq \Lambda_G$ by the constraint of P2. Combining this with $\Lambda_G(\mathcal{V}) \subseteq \Lambda_G(\mathcal{N}) = \Lambda_G$ from Eqn. (10), we have $\Lambda_G(\mathcal{V}) = \Lambda_G(\mathcal{N}) = \Lambda_G$, i.e. the constraints of P1 and P2 are equivalent. The objectives of P1 and P2 are identical, therefore it is clear that P2 and P1 are equivalent. Thus, P2 provides a straightforward implementation of P1, however enumerating all extreme points in P2 may be impractical.

Instead of evaluating P2, we propose a surrogate condition that is easier to evaluate while still leading to the same optimal

solution. Recall that the set of paths $\mathcal{P}(\mathcal{V})$ includes all underlay paths $\mathcal{P}^{\text{SP}}$ and all n-concatenations (for any n) of these paths at controllable nodes $\mathcal{V}$. Let $\mathcal{P}_G$ be the set of all acyclic paths between all pairs of nodes in G. A first observation is that $\mathcal{P}(\mathcal{N}) = \mathcal{P}_G$. This holds by the assumption that all 1-hop paths are included in the set $\mathcal{P}^{\text{SP}}$, and since all nodes are controllable we can produce any path in G as a concatenation of 1-hop paths. Next, we define an important condition.

Condition C.1 (All-Paths): A set of controllable nodes $\mathcal{V}$ is said to satisfy the all-paths condition *if $\mathcal{P}(\mathcal{V}) = \mathcal{P}_G$.*

The condition requires the formation of all acyclic paths in a network. Since some of the paths are already given (in our paper $\mathcal{P}^{\text{SP}}$), to satisfy the condition, a set of nodes $\mathcal{V}$ must enable all missing paths $\mathcal{P}_G \setminus \mathcal{P}^{\text{SP}}$ by path concatenations. The following result establishes that this condition is necessary and sufficient for $\Lambda_G(\mathcal{V}) = \Lambda_G$. In other words, to allow for maximum throughput achievability we must choose $\mathcal{V}$ to ensure that the path concatenations on these nodes form all missing paths in the network.

Theorem 1: Given a placement of controllable nodes $\mathcal{V}$, satisfying the all-paths condition is necessary and sufficient for maximizing the throughput region, i.e.,

$$\Lambda_G(\mathcal{V}) = \Lambda_G \text{ if and only if } \mathcal{P}(\mathcal{V}) = \mathcal{P}_G.$$

Sufficiency is proved by showing that any feasible network flow can be decomposed into single-path flows that are simultaneously feasible, where we are given that all acyclic paths are supported. Necessity is proved by showing that if any acyclic path is not supported, a feasible network flow can be constructed which requires the use of this unavailable path. The full proof of Theorem 1 is in the appendix.

Using the all-paths condition C.1, we define P3:

$$V_3^* = \min_{\mathcal{V} \subseteq \mathcal{N}} |\mathcal{V}| \quad \text{s.t. All-paths condition C.1} \tag{P3}$$

Corollary 1: P1 $\iff$ P3, therefore $V_1^* = V_3^*$.

A. Overlay Node Placement Algorithm

We design an algorithm to choose the placement of overlay nodes $\mathcal{V} \subseteq \mathcal{N}$ on a given graph $G = (\mathcal{N}, \mathcal{E})$ such that the choice of overlay nodes is sufficient to satisfy the full throughput region of the network, i.e. $\Lambda_G(\mathcal{V}) = \Lambda_G(\mathcal{N})$. At the end of this section we will show that the proposed algorithm optimally solves P3.

The algorithm consists of three phases: (1) removal of degree-1 nodes; (2) constraint pruning; and (3) overlay node placement. These phases are explained below, while each step is supported by a related claim which will help proving the optimality of the algorithm.

Phase 1 (Remove Degree-1 Nodes): An *attached tree* is a tree that is connected to the rest of graph G by only a single edge. An intuitive observation is that the throughput region does not increase by installing controllable nodes on attached trees, since shortest-paths are sufficient in trees. Thus, at this preparatory phase, we remove all attached trees by removing degree-1 nodes recursively, as follows. Start with original graph $G = (\mathcal{N}, \mathcal{E})$, and initialize $\mathcal{N}' := \mathcal{N}$ and $\mathcal{E}' := \mathcal{E}$.

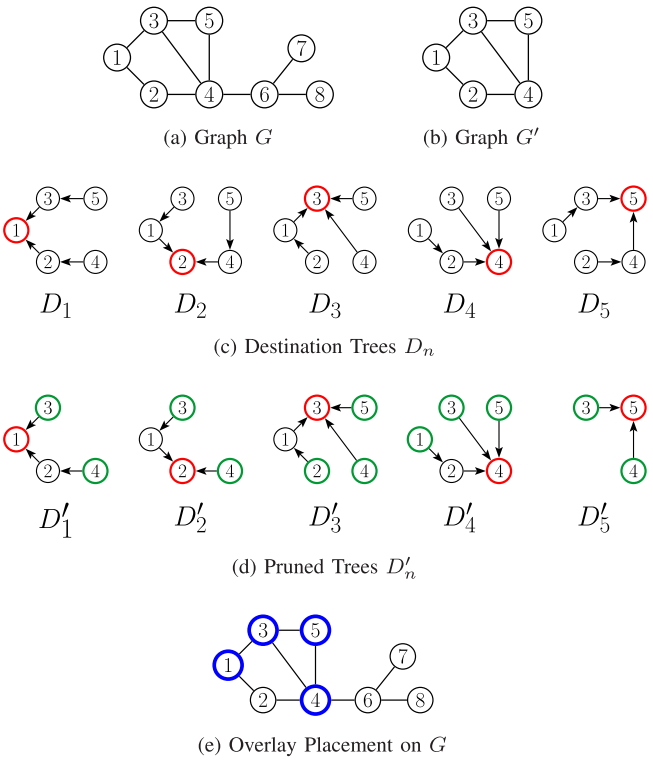


Fig. 3. Example illustrating the relationship between original graph G and various subgraphs used in the node placement algorithm. (a) Original graph G. (b) Modified graph G' at end of Phase 1, after all attached trees have been removed. (c) Destination trees D_n for each node $n \in \mathcal{N}'$. (d) Pruned trees D'_n for each node $n \in \mathcal{N}'$. (e) Optimal overlay node placement on original graph G. Destination nodes in red; leaf nodes in green; overlay nodes in blue.

While there exists any node $n \in \mathcal{N}'$ such that degree$(n) = 1$, set $\mathcal{N}' := \mathcal{N}' \setminus n$ and set $\mathcal{E}' := \mathcal{E}' \setminus e$, where e is the only edge that connects to node n. Repeat until no degree-1 nodes remain. All remaining nodes have a degree of at least 2, thus all attached trees have been removed. The graph that remains is $G' = (\mathcal{N}', \mathcal{E}')$. The relationship between graphs G and G' is shown for an example in Figs. 3a and 3b, where the subgraph connecting nodes 6, 7, and 8 formed an attached tree.

Lemma 1: Suppose that placement $\mathcal{V}$ satisfies the all-paths condition (C.1), and node $n \in \mathcal{V}$ lies on an attached tree. Then $\mathcal{V} \setminus n$ also satisfies the all-paths condition.

For any nodes $a, b \in \mathcal{N}$, the proof considers four arrangements for the locations of a and b: (i) a and b are on the same attached tree, (ii) a is on some attached tree while b is not, (iii) b is on some attached tree while a is not, and (iv) both a and b are on graph G'; this list is exhaustive. The proof shows that any acyclic path $P_{ab} \in \mathcal{P}(\mathcal{V})$ is also in $P_{ab} \in \mathcal{P}(\mathcal{V} \setminus n), n \notin G'$, for each of the four arrangements. The full proof is available in [6].

By induction, it suffices to allocate overlay nodes in G' to satisfy the all-paths condition.

Phase 2 (Constraint Pruning): In this phase, we define the destination trees which will be used to find the constraints for node placement. Exploiting a necessary condition from Lemma 2 regarding the placement of controllable nodes, we show that proper pruning of these destination trees will identify the sufficient constraints over which we minimize the allocation of controllable nodes.

By optimal substructure, the union of shortest-paths P_{xn}^{SP} to any destination n from all nodes $x \in \mathcal{N}' \setminus n$ forms destination tree D_n. Destination trees D_n are shown for the example graph in Fig. 3c. Define $\{P_{xn}^{\text{SP}}\} \setminus n$ to be the set of nodes on the shortest path from x to n, excluding node n. We have the following.

Lemma 2: If the degree of node x on tree D_n is less than the degree of x on graph G', and there is no overlay node along the shortest path from x to n (i.e. $\nexists v \in \mathcal{V} : v \in \{P_{xn}^{SP}\} \setminus n$), then the all-paths condition C.1 is not satisfied.

As an example, consider the network in Fig. 3 where node 3 has degree-2 on D_1 and has degree-3 on G'. If node 3 is not an overlay node, then the sequence of nodes 5-3-4-2-1 cannot be formed as a concatenation of shortest-paths, in which case the all-paths condition is not satisfied. The proof of Lemma 2 is in the appendix.

For Phase 2, we prune destination trees D_n *at* nodes x with degree less in D_n than in G' to obtain *pruned trees* D'_n. In other words, the incoming edges and associated children nodes are removed from x on D_n such that x becomes a leaf node on D'_n (unless x is also removed). Continuing with the example in Fig. 3, D_1 is pruned at node 3 to form pruned tree D'_1, where node 3 becomes a leaf node. By Lemma 2, for the all-paths condition to be satisfied it is necessary to have at least one overlay node on the shortest path to n from every leaf node of pruned tree D'_n. The pruned trees D'_n and this necessary condition from Lemma 2 will be used as constraints in Phase 3.

Phase 3 (Overlay Node Placement): Consider the following binary program to place the minimum number of overlay nodes to satisfy Lemma 2 for all nodes on all pruned trees D'_n:

$$\begin{aligned} V_4^* = \min & \sum_n v_n \\ \text{s.t.} & \sum_{a \in \{P_{bn}^{\text{SP}}\} \setminus n} v_a \geq 1, \quad \forall b \in \text{LeafNodes}(D'_n), \ \forall n \\ & v_n \in \{0,1\}, \quad \forall n \end{aligned} \tag{P4}$$

where $\text{LeafNodes}(D'_n)$ is the set of all leaf nodes on pruned tree D'_n, and where $\{P_{bn}^{\text{SP}}\} \setminus n$ is defined in Phase 2. Returning to the example in Fig. 3, the set $\text{LeafNodes}(D'_n)$ is highlighted in green for each pruned tree D'_n in Fig. 3d, and the overlay nodes identified by P4 are shown in blue in Fig. 3e.

Next, we show the placement from P4 to be sufficient.

Lemma 3: The overlay node placement of P4 satisfies the all-paths condition for graph G'.

This is proved by showing that every path is either (i) a shortest path or (ii) can be formed as a concatenation of shortest paths at overlay nodes which satisfy the leaf node constraint of P4. The proof of Lemma 3 is available in [6].

The following main result establishes the performance of the proposed placement algorithm.

Theorem 2: Let $\mathcal{V}^*$ be the solution produced by the overlay node placement algorithm. Then, $\mathcal{V}^*$ is an optimal solution to P3. It follows that

- $\Lambda_G(\mathcal{V}^*) = \Lambda_G$.
- *$\mathcal{V}^*$ is an optimal solution to P1.*

Proof: By Lemma 2, the constraint of P4 is necessary for the all-paths condition. By Lemmas 1 and 3 it is also sufficient.

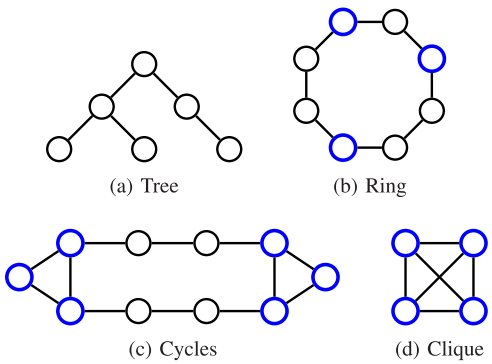


Fig. 4. Minimum node placement required to yield maximum throughput for several simple scenarios, where controllable nodes are bolded in blue. (a) No controllable nodes on trees. (b) Exactly 3 controllable nodes on a ring (for conditions specified in Lemma 5). (c) At least 3 controllable nodes on every cycle. (d) All nodes must be controllable on a clique.

Thus, we have $P4 \iff P3$. By Theorem 1, the remaining assertions follow. ■

Phases 1-2 of the algorithm have complexity $O(N^2)$. P4 solves a vertex cover problem, which is known to be NP-Hard in general. However, note that the constraints of our problem have optimal substructure, which might be exploitable. For our experiments on graphs with 1000 nodes, the solver found most solutions to P4 within 5 seconds, and we only rarely encountered scenarios that required more than a few minutes to solve. Thus, the algorithm is practical.

Program P4 is intended to be used offline to find an optimal node placement. If an online solution is desired with a polynomially-bounded runtime, then the following algorithm can be used in place of P4 for each disjoint graph in G'. Let $\mathcal{V} = \emptyset$, mark all nodes as unvisited, and create a to-visit list of nodes; choose node n with the highest number of edges in its pruned tree D'_n to add to the to-visit list and mark n as visited. While the to-visit list is not empty, remove node a from the front of the list, update $\mathcal{V} := \mathcal{V} \cup a$, and add all unmarked leafs-nodes b from D'_a to the to-visit list, marking such nodes b as visited. Repeat until the to-visit list is empty. Every node in G' will then either be included in $\mathcal{V}$ or will be on at least one pruned tree $D'_v : v \in \mathcal{V}$, Each of N nodes can be marked at most once, and the marked status of each node can be tested $O(N)$ times, yielding a complexity of $O(N^2)$.

V. Optimal Node Placement Examples

We provide results for various types of network graphs, including specific graph families and random graphs. By Theorem 1, the full throughput region is provided by the placement of our algorithm on all these cases.

A. Simple Scenarios

1) Trees and Forests: Consider trees with single-path underlay routes P_{ab}^{SP} for every pair of nodes a and b. A tree is loop free, and thus each path P_{ab}^{SP} is the unique acyclic path from node a to b, as shown in Fig. 4a. Thus, the all-paths condition is automatically satisfied, and $\Lambda_G(\emptyset) = \Lambda_G(\mathcal{N})$.

It follows that no controllable nodes are required for a forest, which is a disjoint union of trees.

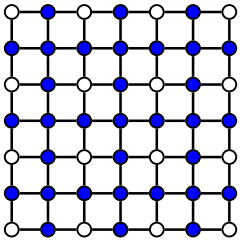


Fig. 5. Minimal placement of overlay nodes to support full throughput region on a 7×7 grid. Overlay nodes indicated in blue. Node placement from P5.

2) Cycles and Rings:

Lemma 4: Every cycle requires at least 3 controllable nodes to satisfy the all-paths condition.

For a ring, observe that shortest path P_{ab}^{SP} connects nodes a and b in only one direction, even when a and b are themselves controllable. At least one more controllable node is required to form path $P_{ab} \neq P_{ab}^{\text{SP}}$ in the counter direction. The proof of Lemma 4 can be found in the appendix, generalizing the above observation to consider all pairs of nodes on a cycle.

Further, the lower bound from Lemma 4 is tight for the case of a ring, where the entire graph is a single cycle. These scenarios are illustrated in Figs. 4b and 4c.

Lemma 5: Exactly 3 controllable nodes are required to satisfy the all-paths condition for a ring network with $N \geq 5$ nodes and hop-count as the metric for shortest-path routing.

This is proved by showing that shortest-paths exist from any node to at least $N/3$ other nodes in each direction around the ring. Thus, there exists a placement of 3 controllable nodes that can satisfy the all-paths condition. The proof is in the appendix.

3) Cliques: Consider cliques with single-path underlay routes P_{ab}^{SP} for every pair of nodes a and b. We require all edges (a, b) be included in the underlay routes, however there is an edge between every pair of nodes in a clique. Thus, all underlay routes are single edges, i.e. $P_{ab}^{\text{SP}} = (a, b)$ for all pairs $a, b \in \mathcal{N}$. A Hamiltonian path, traversing all nodes, will require all intermediate nodes to be controllable. Such paths can start and end at any node, therefore the all-paths condition requires all nodes to be controllable for a clique, i.e. $\mathcal{V} = \mathcal{N}$, as shown in Fig. 4d.

4) Regular Grids: Consider the regular grid topology (see Fig. 5). The network can be viewed as a tiling of nodes connected in squares of 2×2 nodes with total $N = L \times W$ nodes. Each 2×2 square tile is a cycle, so by Lemma 4 each cycle requires at least 3 controllable nodes. Let $\mathcal{T}_j$ be the set of four nodes on tile j. Then a simple program to place overlay nodes on a grid is given by P5.

$$\begin{aligned} \min \quad & \sum_n v_n \\ \text{s.t.} \quad & \sum_{n \in \mathcal{T}_j} v_n \geq 3, \quad \text{for each tile } j, \\ & v_n \in \{0, 1\}, \quad \forall n \end{aligned} \tag{P5}$$

In Fig. 5, we see that P5 chooses controllable nodes $\mathcal{V}$ in a crosshatch pattern. We can apply this pattern to grids of arbitrary size by choosing all nodes on even rows and even columns to be controllable. Note that no two uncontrollable nodes are adjacent in the crosshatching pattern.

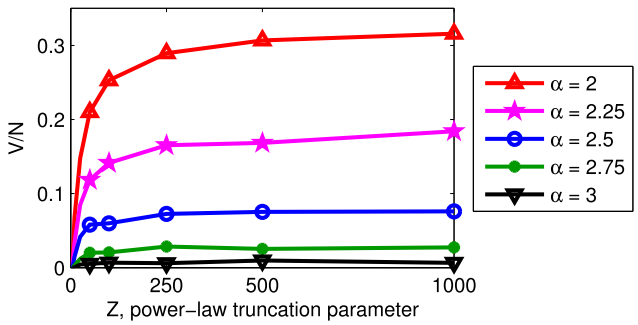


Fig. 6. Results of overlay node placement algorithm on random graphs where node degree follows a power-law distribution with exponent α. Graphs generated with configuration model and truncated Zipf distribution.

For the crosshatch overlay node allocation, the ratio of controllable nodes to total nodes, V/N, is shown in Eqn. (11).

$$\frac{V}{N} = \frac{L\lfloor W/2 \rfloor + \lfloor L/2 \rfloor \lceil W/2 \rceil}{L \times W}, \quad \text{for } L \geq 2 \text{ and } W \geq 2 \tag{11}$$

This ratio is exactly $3/4$ when both L and W are even and asymptotically approaches $3/4$ when either L, W, or both are odd.

B. Random Networks

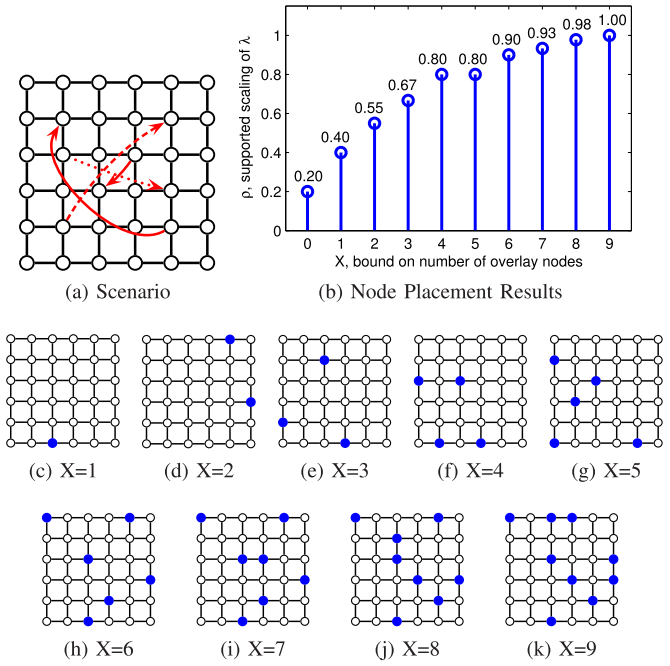
This section considers placement of overlay nodes to support the full throughput region on random graphs. We present here results about power-law graphs, where the degree of nodes is random and roughly follows a power-law distribution. This is recognized as a realistic model for the Internet [10]. We have experimented with several other models for random graphs, the results of which can be found in [6].

We construct random networks that have power-law degree distributions using the configuration model and a truncated Zipf distribution [10]. Zipf is a discrete distribution with parameters α and Z, where α is the power-law exponent and Z is a truncation parameter indicating the maximum degree of the distribution. The Zipf PMF is

$$\mathbb{P}(D = d) = \frac{d^{-\alpha}}{\sum_{k=1}^{Z} k^{-\alpha}}, \quad \text{for } d = 1, ..., Z.$$

For a given number of nodes N, the configuration model attaches a number of *stubs* to each node according to the Zipf distribution, where a stub is half of an edge. Pairs of unconnected stubs are then chosen randomly and connected to form edges. Thus, node degree follows a power-law distribution.

Fig. 6 shows results from the overlay node placement algorithm for random power-law graphs with $N = 1000$ nodes, averaged over 10 realizations per data point. Values of α between 2 and 3 are considered, with $\alpha = 2.5$ being a frequent estimate for the Internet [10]. For $\alpha = 2.5$, the overlay node placement algorithm finds that less than 8% of nodes need to be controllable for the full throughput region to be achievable.



(a) Scenario (b) Node Placement Results

(c) X=1 (d) X=2 (e) X=3 (f) X=4 (g) X=5

(h) X=6 (i) X=7 (j) X=8 (k) X=9

Fig. 7. Results of P6 for chosen rate vector on 6×6 grid. (a) Arrival rate vector $\boldsymbol{\lambda}$ includes traffic demands for all all pairs of nodes. Here we consider $\boldsymbol{\lambda}$ with four active traffic demands with symmetric rates, as indicated with arrows. (b) Fraction of $\boldsymbol{\lambda}$ supported when limited to $|\mathcal{V}| \leq X$ controllable nodes for a random shortest-path routing realization. (c)-(k) Node placements for (b); note that placements are not necessarily unique.

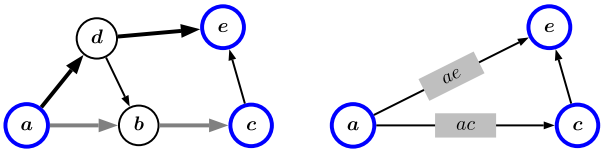
VI. Placing a Limited Number of Overlay Nodes

A formulation similar to P2 is useful in scenarios where only a small subset of arrival rate vectors require support, such that the constraints are limited to the specific vectors $\boldsymbol{\lambda}^{(i)}$ of interest. For example, this includes networks with nodes that neither generate nor consume information such as network routers. This approach can also use P2 to minimize the number of controllable nodes required to allow maximum flow between a specific source and destination.

A similar formulation can be used to maximize the achievable flow when the maximum number of controllable nodes is upper bounded by some number X, as shown in P6. This can be useful in scenarios where resource limitations don't allow enough controllable nodes to achieve maximum throughput. As in P2, multiple rate vectors $\boldsymbol{\lambda}^{(i)}$ can be supported with additional constraints $\rho\boldsymbol{\lambda}^{(i)} \in \Lambda_G(\mathcal{V})$.

$$\begin{aligned} &\max_{\mathcal{V}\subseteq\mathcal{N}} \ \rho \\ &\text{s.t. } \rho\boldsymbol{\lambda} \in \Lambda_G(\mathcal{V}) \\ &\quad |\mathcal{V}| \leq X \end{aligned} \tag{P6}$$

Fig. 7 shows results of P6 on a 6×6 grid for a specific rate vector $\boldsymbol{\lambda}$ with four equal traffic demands. Fig. 7b shows that a fraction 80% of throughput is supported in the direction $\boldsymbol{\lambda}$ with only $X = 4$ and diminishing returns from additional overlay nodes, with $X = 9$ required to provide maximum throughput for the four specified demands. Note that by Eqn. (11), $V = 27$ overlay nodes would be required to support maximum throughput for all possible traffic demands (i.e. Λ_G) on this grid network.



(a) Underlay Network (b) Overlay Network with Tunnels

Fig. 8. (a) An example network of controllable and uncontrollable nodes, where controllable (overlay) nodes $\mathcal{V} = \{a, c, e\}$ are shown in blue. We indicate with bold arrows the shortest paths available to node a by the underlay network. (b) The equivalent overlay network of controllable nodes and tunnels.

VII. Backpressure Overlay Policy

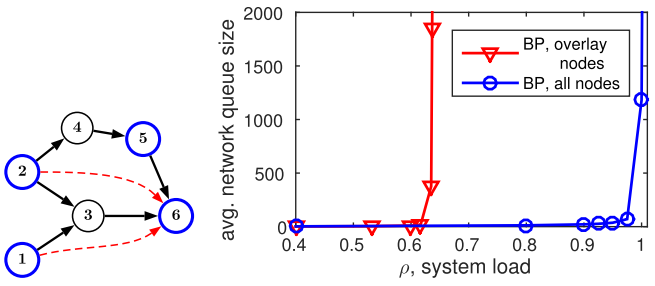
Subject to the placement of overlay nodes, we study the problem of throughput maximization using dynamic routing decisions at overlay nodes. We are interested in a dynamic routing policy that is stable for any arrival vector in the region $\Lambda_G(\mathcal{V})$, i.e. achieves maximum throughput.

For ease of exposition, we define the notion of "tunnels" which correspond to paths (in the underlay network) between controllable nodes. Let tunnel (i, j) correspond to a path in the underlay where end-points are overlay nodes i, j and intermediate nodes are underlay nodes. Thus, the overlay network $\mathcal{G}_R = (\mathcal{V}, \overline{\mathcal{E}})$ consisting of overlay nodes $\mathcal{V}$ and tunnels $\overline{\mathcal{E}}$. Fig. 8b depicts the overlay network for the physical network in Fig. 8a, assuming shortest path routing is used. Physically packets are stored at different underlay nodes along the tunnel. We assume that inside the tunnels packets are forwarded in a *work-conserving* fashion.[2]

Every overlay node $v \in \mathcal{V}$ maintains a queue for each commodity c and we denote its backlog with $Q_v^c(t)$ at slot t. For two overlay neighbors $v, w \in \mathcal{V}$, we define $F_{vw}^c(t)$ to be the number of commodity c packets that have departed overlay node v but have not yet reached overlay node w. We call these the *packets-in-flight* between overlay nodes v and w for commodity c. Moreover, let $F_{vw}(t)$ be the total number of packets-in-flight on the tunnel (v, w), across all commodities. We note that while it may not be possible to observe the individual queue sizes at uncontrollable nodes, the number of packets-in-flight can be estimated using a simple acknowledgment scheme. Note that in networks with reliable delivery, the number of packets-in-flight can be directly inferred from the available information. Even for cases where explicit control packets are required for the calculation of packets-in-flight, using [4, §4.7] delayed backlog information is sufficient for throughput optimality, and hence the number of control messages can be limited to a desired frequency (at the tradeoff of delay).

Under routing policy π, let $\mu_{vn}^c(t, \pi)$ be the service function on the link (v, n) for commodity c packets, where $v \in \mathcal{V}$ and $n \in \mathcal{N}$. Thus, $\mu_{vn}^c(t, \pi)$ is the number of packets allocated to commodity c on link (v, n), at time slot t. The edge rate constraint implies $\sum_c \mu_{vn}^c(t, \pi) \leq R_{vn}$ must be satisfied at every slot. Thus, at each overlay node, the policy chooses the number of packets to be sent to outgoing neighbors subject to the edge rate constraint. In what follows, we study the problem

[2]In our simulations we use proportionally-fair random service on a packet-by-packet basis.



(a) Scenario (b) Simulation Results

Fig. 9. Insufficiency of BP in overlay networks. (a) Scenario with contention at uncontrollable node 3. (b) Queue size of BP in overlay vs. BP in underlay.

of controlling this system by observing queue backlogs Q and packets-in-flight F, and choosing service function μ at overlay nodes only.

A. Insufficiency of Traditional Backpressure

For an interference-free wired network, the backpressure (BP) routing policy [16] is known to be throughput optimal. However, traditional backpressure requires knowledge of the queue sizes at all nodes in the network, and cannot be used at uncontrollable nodes. A natural alternative is to apply backpressure only at overlay nodes. However, as we show in the following example, applying backpressure at overlay nodes is not sufficient for network stability.

Traditional backpressure operates as follows. For each link (a,b), define the differential backlog $W_{ab}^c(t)$,

$$W_{ab}^c(t) = Q_a^c(t) - Q_b^c(t), \quad \forall\,(a,b) \in \mathcal{E},\ \forall\, c \in \mathcal{N},$$

and define commodity $c_{ab}^*(t)$ that maximizes this weight,

$$c_{ab}^*(t) \in \arg\max_{c\in\mathcal{N}} W_{ab}^c, \quad \forall\,(a,b) \in \mathcal{E}. \tag{12}$$

The BP policy chooses

$$\mu_{ab}^{c_{ab}^*}(t,\text{BP}) = \begin{cases} R_{ab} & \text{if } W_{ab}^{c_{ab}^*} > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{13}$$

where $\mu_{ab}^c(t,\text{BP}) = 0,\ \forall c \neq c_{ab}^*$.

In [16], this policy was shown to stabilize the network for any arrival rates in the region $\Lambda_G(\mathcal{N})$. The intuition behind the optimality of BP is that congestion information propagates through the network via queue backlogs. The policy balances neighboring backlogs, such that when node n becomes congested, any upstream neighbors of n also become congested. Ultimately each node can optimally route packets to avoid congestion based only on the observation of neighbor backlog. In our setting, the uncontrollable nodes do not use BP, and thus any congestion occurring on these nodes is not propagated.

Consider the example of Fig. 9a, where (controllable) overlay nodes $\mathcal{V} = \{1,2,5,6\}$ are indicated in blue, with directed unit-rate links. It can easily be verified that the all-paths condition C.1 is satisfied for this setting, thus $\Lambda_G(\mathcal{V}) = \Lambda_G(\mathcal{N})$. The dashed red arrows show two traffic demands with symmetric arrival rates λ. With unit-rate links, offered load $\rho = \lambda$, where $\rho < 1$ is required for this network to be stable. We examine two different cases. First, we run BP at all nodes; this achieves maximum throughout and it is stable for all $\rho < 1$. Second, we run BP only at overlay nodes, computing differential backlogs across the overlay edges, e.g. node 2 computes $W_{2,5}^6 = Q_2^6 - Q_5^6$ and $W_{2,6}^6 = Q_2^6 - Q_6^6$. Simulation results in Fig. 9b show that BP at the overlay nodes cannot stabilize $\rho > 2/3$, i.e. it is throughput suboptimal. The intuition is as follows. Note that $Q_6^6 = 0$, since node 6 is a destination. Then, any congestion at uncontrollable node 3 cannot be detected by source node 2, leading to positive traffic flow from source 2 through node 3.

Thus, using backpressure only in the overlay nodes hides any queue buildup in the underlay, and results in loss of throughput. When the overlay tunnels do not overlap,[3] it is possible to infer the congestion in the underlay by observing the packets-in-flight ($F_{vw}(t)$). This motivates our policy in the next section, which is throughput optimal when tunnels do not overlap. However in general tunnels do overlap, and $F_{vw}(t)$ no longer provides sufficient information about the backlog of flows along different tunnels. Thus, in Section VII-C we propose a simple heuristic scheme for general overlay networks with overlapping tunnels.

B. Threshold-Based Routing Scheme

Our threshold policy attempts to keep the number of packets inside the tunnel bounded, while at the same time keep traffic flowing through the tunnel whenever the backlog outside the tunnel is sufficiently large. Let M_{ij} be the number of underlay nodes associated with tunnel (i,j), $R_{ij}^{\max}$ be the maximum link capacity along the tunnel, $R_{ij}^{\min}$ be the minimum link capacity in the tunnel, and R_{ij}^{in} be the capacity of the input link to the tunnel. Now define the threshold, T, as follows:

$$T \triangleq \max_{(i,j)\in\overline{\mathcal{E}}} \left[M_{ij} R_{ij}^{\min} + \frac{M_{ij}(M_{ij}-1)}{2} R_{ij}^{\max} + R_{ij}^{\text{in}} \right] \tag{14}$$

The value of T is engineered for the following effect: if $F_{ij}(t) > T$ then we are sure that in slot $t+1$ the tunnel will output $R_{ij}^{\min}$ packets.

Due to work-conservation, if tunnels do not overlap, a tunnel with "sufficiently many" packets has instantaneous output equal to its bottleneck capacity. Thus, we propose the following Threshold-based Backpressure (BP-T) Policy which is designed to keep the tunnel backlogs close to the threshold T.

At each time slot t and tunnel (i,j), let

$$c_{ij}^* \in \arg\max_{c\in\mathcal{C}} \left[Q_i^c(t) - Q_j^c(t) \right],$$

be a session that maximizes the differential backlog between overlay nodes i,j, ties resolved arbitrarily. Then route into that tunnel

$$\mu_{ij}^{c_{ij}^*}(t,\text{BP-T}) = \begin{cases} R_{ij}^{\text{in}} & \text{if } Q_i^{c_{ij}^*}(t) > Q_j^{c_{ij}^*}(t) \\ & \text{AND } F_{ij}(t) < T \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

and $\mu_{ij}^c(t,\text{BP-T}) = 0,\ \forall c \neq c_{ij}^*$.

BP-T is similar to applying backpressure in the overlay, with the striking difference that *no packet is transmitted to*

[3]Two tunnels are said to overlap if they share an underlay link.

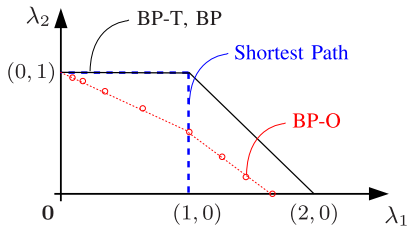


Fig. 10. Throughput comparison of routing schemes: BP-T, BP-O, Shortest Path, and BP.

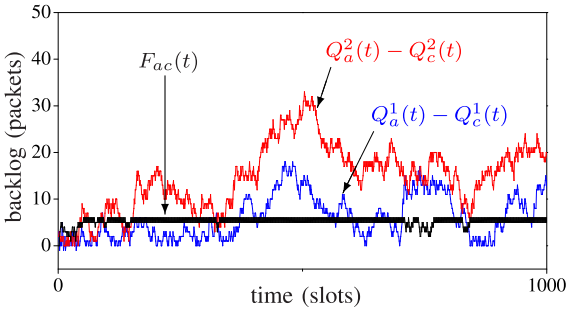


Fig. 11. Sample path evolution of the system under BP-T, $\lambda_1 = \lambda_2 = .97$.

a tunnel if $F_{ij}(t) \geq T$. Therefore, when tunnels do not overlap, the total tunnel backlog is limited to at most $T+R_{\max}$, where $R_{\max}$ is the maximum number of packets that may enter the tunnel in one slot. Thus, the threshold-based policy keeps the tunnel occupancy bounded, and ensures that when the tunnel is full, a minimum rate of flow out of the tunnel is guaranteed. It is the combination of these two conditions that guarantees the throughput optimality of BP-T when tunnels do not overlap.

Theorem 3: If underlay nodes use a work-conserving scheduler and the tunnels are non-overlapping, BP-T is stable for all arrival rates within the network stability region $\Lambda_G(\mathcal{V})$.

The proof is based on a pipelining argument combined with a K-slot Lyapunov drift analysis. The proof details can be found in [11].

BP-T is a distributed policy since it utilizes only local queue information and the capacity of the incident links, while it is agnostic to arrivals, or capacities of remote links, e.g. note that the decision does not depend on the capacity of the bottleneck link $R_{ij}^{\min}$. Moreover, tunnel backlogs $F_{ij}(t)$ can be estimated at each overlay node using acknowledgments. In practice, these estimates may be delayed; however, it is easy to show that bounded delays do not affect stability [9].

We simulate the BP-T scheme on the simple network topology of Fig. 8, where we define two sessions sourced at a; session 1 destined to e and session 2 to c. We assume that $R_{ab} = 2$ and all other links have unit capacity as shown in the figure. We choose R_{ab} in this way to make the routing decisions of session 1 non-trivial. We use a threshold value of $T = 6$, which satisfies (14) in this example setting. Since the example satisfies the non-overlapping tunnel condition, by Theorem 3 our policy achieves the full throughput region ($\Lambda_G(\mathcal{V})$), as can be seen in Fig. 10. Also shown in the figure is the performance of backpressure at overlay nodes only (BP-O), and of backpressure at all nodes (BP). As can be seen, BP-T achieves the same throughput region as BP and greater throughput region than BP-O and shortest path routing.

To better understand the operation of BP-T, we examine a sample path evolution of this system under BP-T for the case where $\lambda_1 = \lambda_2 = 0.97$. In this setting, in order to achieve stability, session 1 must use its dedicated path (a, d, e), and send almost no traffic through tunnel ac. Focusing on the tunnel ac, Fig. 11 shows the differential backlogs per session $Q_a^c(t) - Q_c^c(t)$ and the corresponding tunnel backlog $F_{ac}(t)$ for a sample path of the system evolution.

In most time slots a is congested, which is indicated by high differential backlogs. In such slots, the tunnel has more than 1 packet, which guarantees that it outputs packets at highest possible rate, hence the tunnel is fully utilized. Recall that when the tunnel is full ($F_{ac}(t) > T$=6) no new packets are inserted to the tunnel preventing it from exceeding $F_{\max}$. Observe that the differential backlog of session 2 always dominates the session 1 counterpart, and hence whenever a tunnel is again ready for a new packet insertion, session 2 will be prioritized for transmission according to (15). Therefore, the proportion of session 2 packets in this tunnel is close to 100%, which is the correct allocation of the tunnel resources to sessions for this case.

C. Overlay Backpressure Heuristic Algorithm

Although we are able to show that the BP-T is throughput optimal when tunnels do not overlap, its performance in the general case of overlapping tunnels is not guaranteed. Nonetheless, simulation results on simple overlapping tunnel topologies indicate good throughput performance even when tunnels overlap [12]. In this section, we propose a heuristic scheme that is inspired by BP-T, yet is much simpler to implement. In particular, our heuristic takes tunnel congestion into account, but does not require the threshold computation and associated knowledge of the underlay topology.

1) Overlay Backpressure (OBP): Redefine the differential backlog as,

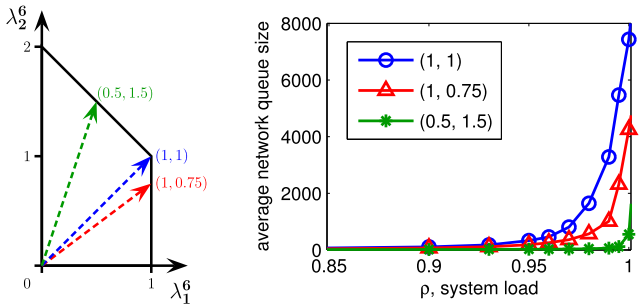
$$W_{vw}^c(t) = Q_v^c(t) - Q_w^c(t) - F_{vw}^c(t), \quad \forall\,(v,w) \in \overline{\mathcal{E}},\ \forall\, c \in \mathcal{N},$$

then determine c_{vw}^* and $\mu_{vw}^c(t, \text{OBP})$ as in Eqns. (12)-(13).

Intuitively, this policy takes into account both the packet accumulation at the neighbor overlay node v, as well as any packets-in-flight on the path P_{vw}, in the form of *negative pressure*.

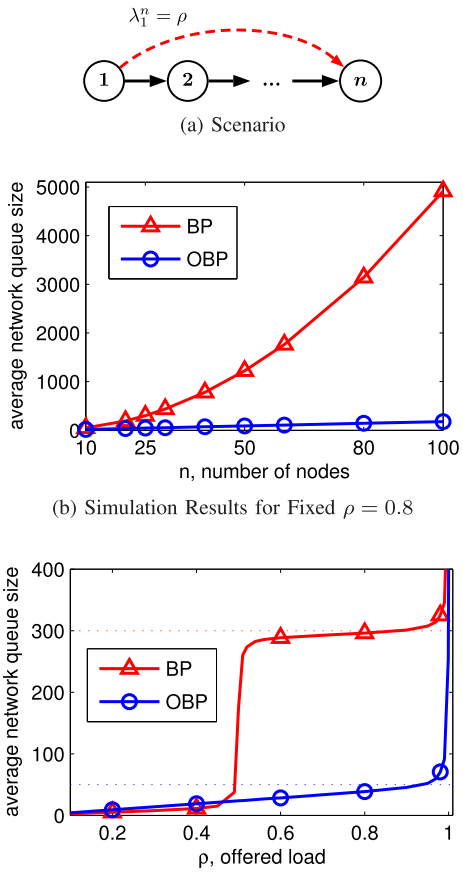
Although we are not able to demonstrate the throughput optimality of OBP, we studied its performance through extensive simulations. We observe the following properties of the algorithm. (i) OBP maximizes throughput in all examined scenarios, including the one of Fig. 9a, (ii) OBP outperforms BP applied only at overlay nodes, and (iii) OBP has good delay properties, outperforming BP even when the latter is applied at all nodes.

In Fig. 12, we study different arrival vectors for the network of Fig. 9a. The simulation results in Fig. 12b show that all studied vectors are supported by the OBP policy.



(a) Throughput Region (b) OBP Simulation Results

Fig. 12. Evaluation of OBP policy on scenario from Fig. 9a. (a) Throughput region of Fig. 9a, with select rate vectors indicated. (b) Average queue backlog of OBP, after 1e6 time steps, for rate vectors indicated in (a).



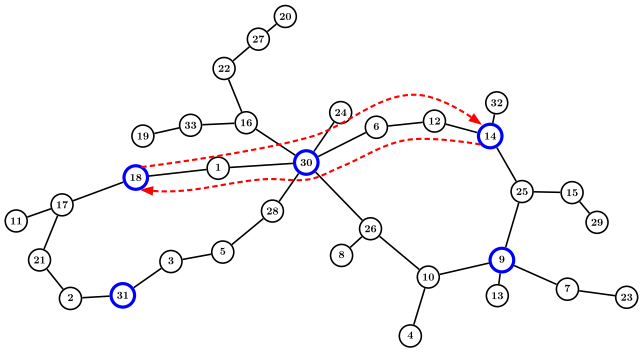
(a) Scenario

(b) Simulation Results for Fixed $\rho = 0.8$

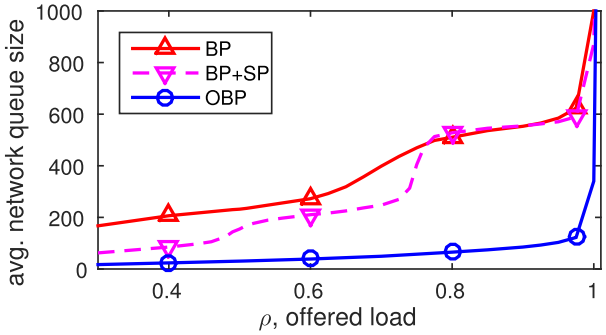
(c) Simulation Results for Fixed $n = 25$

Fig. 13. Directed tandem with n nodes. (b) BP versus OBP for offered load $\rho = 0.8$. Quadratic growth for BP. Linear growth for OBP. (c) BP versus OBP for $n = 25$. Quadratic backlog in BP results for $\rho > 0.5$. Dotted horizontal line at $n(n-1)/2$ for BP, and at $2n$ for OBP.

In Fig. 13, we study a directed tandem network for the purpose of illustrating the delay properties of OBP. From [2] it is known that for BP on a tandem network, per-node queues grow linearly with distance from the destination, and thus network queue size grows quadratically with the total number of nodes. However, for the OBP policy we observe this linear growth of per-node queues only at controllable nodes, implying smaller total network queues size and improved



(a) Scenario



(b) Simulation Results

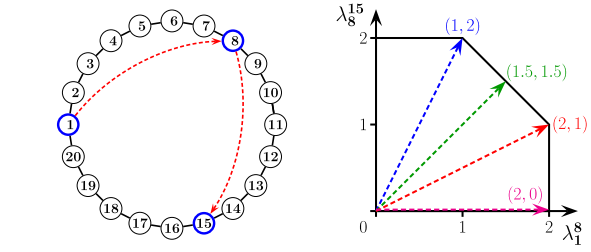
Fig. 14. Comparing OBP with BP on a random graph. (a) Scenario with two symmetric traffic demands. (b) Average queue size for BP, BP+SP, and OBP.

delay performance when there are few controllable nodes. In this particular example, only the source is controllable, with $n-1$ legacy nodes, a setting that corresponds to the maximum benefit. Delay is compared between BP and OBP for a fixed offered load in Fig. 13b and for a fixed number of nodes in Fig. 13c. Although BP is applied at all nodes it is still outperformed by OBP applied only at the source.

Finally, in Fig. 14, we show simulation results from three policies: OBP, BP at all nodes, and BP with shortest-path bias (BP+SP) from [9].

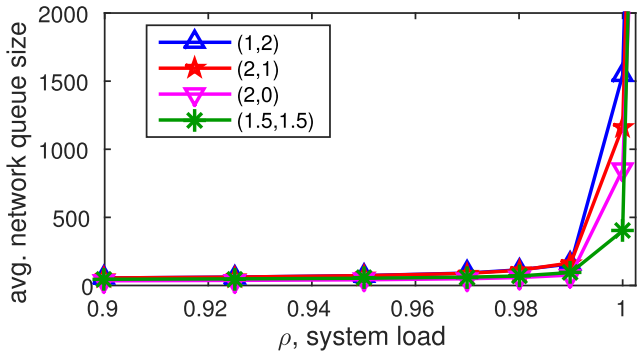
Although the latter two are both throughput optimal policies, they yield worse delay than OBP. The reason is threefold: (i) the quadratic network queue size of BP is proportional to the number of controllable nodes used (in this scenario, OBP uses only 5 overlay nodes), (ii) no packets are sent to attached trees in case of OBP, and (iii) under light traffic, packets under BP perform random walks.

Finally, we consider the performance of OBP on a ring network with $N = 20$ nodes and $V = 3$ overlay nodes, where $V = 3$ was proved sufficient to achieve $\Lambda_G(\mathcal{V}) = \Lambda_G$ by Lemma 5. The scenario is shown in Fig. 15a, with two competing traffic demands indicated with red arrows. Fig. 15b shows the throughput region for these two traffic demands, with 4 rate vectors identified, and results for the OBP policy on these rate vectors is shown in Fig. 15c. For each rate vector, we see the queues remain small for all points internal to the throughput region, indicating that OBP can stabilize the system for these vectors.



(a) Scenario

(b) Stability Region



(c) Simulation Results

Fig. 15. Evaluation of OBP on a ring with $N = 20$ nodes. (a) Overlay nodes indicated in blue; two traffic demands shown with red arrows. (b) Throughput region for competing traffic demands λ_1^8 and λ_8^{15}. Various rate vectors identified for simulation. (c) Queue size of OBP policy after 1 million time steps for rate vectors indicated in (b).

While our OBP policy seems to perform well in simulations, we do not believe that it is optimal in general settings. A promising future direction of research is to identify a maximally stable dynamic routing policy for our overlay architecture.

VIII. Overlay Nodes in Wireless Networks

The goal of this section is to motivate the need for additional study into the placement of overlay nodes for networks with wireless interference.

The all-paths condition C.1 is sufficient to achieve $\Lambda_G(\mathcal{V}) = \Lambda_G$ in all networks, but this condition is not always a necessary condition in wireless networks. In other words, satisfying the all-paths condition may over allocate controllable nodes under certain wireless interference models. To see this, consider a clique where all edges have unit-capacity and all transmissions mutually interfere. Due to interference, the maximum network sum throughput in this scenario is one, and this maximum throughput can only be achieved when each source a sends to destination b directly over edge (a, b). Thus no multi-hop paths are required, and the all-paths condition is sufficient but not necessary for this scenario.

To illustrate an overlay network in a wireless scenario, we study the performance of the overlay node placement algorithm on random geometric graphs, which is a simple model for wireless networks with omnidirectional antennas. The geometric model has parameters N and r, where N is the number of nodes and r is the edge range. Random graphs are then generated by randomly placing N nodes in a unit square,

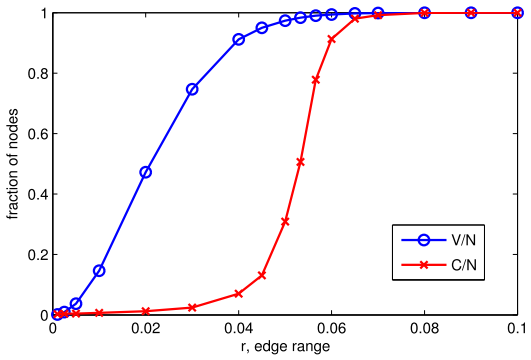


Fig. 16. Results of overlay node placement algorithm on random geometric graphs with $N = 500$ nodes. The blue curve shows the ratio V/N of overlay nodes to total nodes. The red curve shows the ratio C/N for size of largest connected component to total nodes.

and creating all edges (a, b) for which the Euclidean distance between nodes a and b is within range r. Fig. 16 shows results of the overlay node placement algorithm on random graphs with $N = 500$, averaged over 10 realizations per data point. Here, we see for the geometric model that the number of overlay nodes, V, placed by our algorithm grows much faster than the size of the largest connected component, C. The reason is twofold: (i) triangles appear in minor components,[4] and (ii) multiple large components grow simultaneously.

The results for random geometric graphs show that the overlay node placement algorithm chooses most nodes to be controllable. However, as noted above, the placement of controllable nodes by this algorithm is sufficient but may not be necessary for wireless networks. Thus, the minimum number of controllable nodes required to provide full throughput in wireless networks is unclear. A topic for future work is a study of the necessary conditions for $\Lambda_G(\mathcal{V}) = \Lambda_G$ under various interference models.

IX. Conclusions

We study optimal routing in legacy networks where only a subset of nodes can make dynamic routing decisions, while the legacy nodes can forward packets only on pre-specified shortest-paths. This model captures evolving heterogeneous networks where intelligence is introduced at a fraction of nodes. We propose a necessary and sufficient condition for the overlay node placement to enable the full multicommodity throughput region. Based on this condition, we devise an algorithm for optimal controllable node placement. We run the algorithm on large random graphs to show that very often a small number of intelligent nodes suffices for full throughput. Finally, we propose dynamic routing policies to be implemented in a network overlay. We provide a threshold-based policy that is optimal for overlays with non-overlapping tunnels, and provide and alternate policy for general networks that demonstrates superior performance in terms of both throughput and delay.

[4]If edges (a, b) and (a, c) exist at range r, then the distance between b and c is at most $2r$. Thus, every degree-2 node at range r is on a triangle at range $2r$.

APPENDIX

Proof of Sufficiency for Theorem 1: We will show that the all-paths condition is sufficient for supporting any multicommodity vector $\boldsymbol{\lambda} \in \Lambda_G$ while bifurcating traffic only at nodes $\mathcal{V}$. Feasibility of $\boldsymbol{\lambda}$ implies existence of a feasible flow decomposition of $\boldsymbol{\lambda}$. Without loss of generality, choose any one component of $\boldsymbol{\lambda}$ that sends flow from node a to node b with corresponding arrival rate λ_a^b. This arrival rate λ_a^b is supported by flow $f_{ab}^{\boldsymbol{\lambda}}$, where $f_{ab}^{\boldsymbol{\lambda}}$ can be decomposed into subflows $f_{ab}^{\boldsymbol{\lambda}}(p)$ for paths $p \in \mathcal{P}_{ab}$. Since $\mathcal{V}$ satisfies all path condition it follows that all-paths $\mathcal{P}_{ab}$ can be formed as concatenations of available shortest paths on nodes $\mathcal{V}$, and thus the feasible flow decomposition can be constructed with a stationary policy using underlay routes and the given set of controllable overlay nodes. ■

Proof of Necessity for Theorem 1: We will show that given a $\mathcal{V}$ such that there is a path that is not available either as a shortest path or as a concatenation, i.e. the all-paths condition is not satisfied, the full throughput region cannot be achieved. Support of the full throughput region requires support for all arrival rate vectors interior to the rate region allowed by the network. Assume $\Lambda_G(\mathcal{V}) = \Lambda_G$ and some path P_{ab}^{X} is unavailable, both as a shortest-path and as an n-concatenation of shortest-paths at controllable nodes $\mathcal{V}$. Without loss of generality, assume that this unavailable path does not traverse any controllable nodes. Otherwise, split the unavailable path at controllable nodes and choose an unavailable segment induced from the split as path P_{ab}^{X}; such an unavailable segment must exist, otherwise the original path could be formed as an n-concatenation of the induced segments. We will show that there exists a feasible arrival rate vector that requires use of the unavailable path P_{ab}^{X}.

Construct an arrival rate vector $\boldsymbol{\lambda}$ that includes component λ_a^b equal to the maximum flow allowed for path P_{ab}^{X}, plus edge rate R_{ab} if edge (a,b) exists. In vector $\boldsymbol{\lambda}$, also include one-hop traffic demands for all edges $(i,j) \in \mathcal{E} \setminus (a,b)$ by choosing λ_i^j to equal any remaining capacity on edge (i,j). This rate vector $\boldsymbol{\lambda}$ is then feasible by construction.

Let $\mathcal{N}_{ab}^{\text{X}}$ be the set of nodes on path P_{ab}^{X}. For every node j not on path P_{ab}^{X}, i.e., $j \in \mathcal{N} \backslash \mathcal{N}_{ab}^{\text{X}}$, the arrival rate vector $\boldsymbol{\lambda}$ was constructed such that $\sum_i \lambda_i^j = \sum_i R_{ij}$. Applying the edge rate constraints from Eqn. (6) at node j and taking the sum over all neighbors i, we have $\sum_i \sum_{x,y,c} f_{ij}^{xy,c} \leq \sum_i R_{ij} = \sum_i \lambda_i^j$ for all $j \in \mathcal{N} \setminus \mathcal{N}_{ab}^{\text{X}}$, where the final equality comes from the previous equation. Then flow conservation requires that $f_{ij}^{xy,c} = 0$ for all commodities $c \neq j$. Thus, no feasible flow decomposition of $\boldsymbol{\lambda}$ can route flow for λ_a^b through any nodes in $\mathcal{N} \backslash \mathcal{N}_{ab}^{\text{X}}$. Therefore, it remains to consider only nodes in $\mathcal{N}_{ab}^{\text{X}}$ to support λ_a^b.

If P_{ab}^{X} is the only path from node a to b using nodes from the set $\mathcal{N}_{ab}^{\text{X}}$, then P_{ab}^{X} is clearly necessary to support flow λ_a^b. Otherwise, recall that by assumption there are no controllable nodes intermediate to path P_{ab}^{X}. Then it remains only to consider the case where the shortest-path from node a to b uses a strict subset of nodes in $\mathcal{N}_{ab}^{\text{X}}$, as no controllable nodes are available for path concatenation. Consider edge (i,j) such that nodes i and j are on path P_{ab}^{X}, where edge (i,j) is on P_{ab}^{SP} but not on P_{ab}^{X}. Here, $P_{ij}^{\text{SP}} = (i,j)$ is the only available path from i to j with unused capacity, because no controllable nodes are available. Then, $f_{ij}^{ij,j} = \lambda_i^j = R_{ij}$, and Eqn. (6) requires $f_{ij}^{ab,b} = 0$. Therefore, there is no unused capacity on path P_{ab}^{SP}, so λ_a^b and λ_i^j cannot be supported simultaneously. There are no other paths to consider from node a to b for a feasible flow decomposition of $\boldsymbol{\lambda}$.

Therefore, $\Lambda_G(\mathcal{V}) \subset \Lambda_G$ if any path is not available. Thus, we have proved the necessity of the all-paths condition for wired networks with shortest-path routing. ■

Proof for Lemma 2: Let (b,x) be an edge in G' but not in D_n, where such an edge exists by the premise of Lemma 2. Consider path p formed from the concatenation of (b,x) and shortest-path P_{xn}^{SP}. We will show that this path cannot be formed if there are no controllable nodes in the shortest path from x to n, and thus the all-paths condition C.1 is not satisfied.

First, observe that since edge (b,x) is not on tree D_n, shortest-path P_{bn}^{SP} does not include this edge. Thus, the path p requires a concatenation of two or more shortest-paths. Such a concatenation must occur at a controllable node on path P_{xn}^{SP}. However, this is impossible since there are no controllable nodes on path P_{xn}^{SP}. Thus, C.1 is not satisfied. ■

Proof for Lemma 4: Consider controllable nodes $v, w \in \mathcal{V}$ on a cycle, and without loss of generality assume shortest-path P_{vw}^{SP} is on the cycle. Then path P_{vw}^{SP} allows one direction of flow on the cycle, and at least one additional controllable node is required to allow flow in the counter direction on the cycle. Note that the same problem occurs in scenarios with 0 or 1 controllable node on the cycle, and when path P_{vw}^{SP} is not on the cycle. Thus, at least 3 controllable nodes are required on each cycle in the network. ■

Proof for Lemma 5: Lemma 4 establishes the necessity of at least 3 controllable nodes, so it only remains to show that 3 controllable nodes are sufficient to satisfy the all-paths condition.

Starting from any node x, consider nodes y and z that are neighbors, i.e., $(y,z) \in \mathcal{E}$, where shortest-paths P_{xy}^{SP} and P_{xz}^{SP} are disjoint. Without loss of generality assume $|P_{xy}^{\text{SP}}| \leq |P_{xz}^{\text{SP}}|$ where $|p|$ is the length of path p. With hop-count as the shortest-path metric, the length of these disjoint shortest-paths can differ at most by 1. Otherwise, there would exist a contradiction, as the path formed as a concatenation of P_{xy}^{SP} with edge (y,z) would be shorter than shortest-path P_{xz}^{SP}. Then the following inequality holds for any number of nodes $N \geq 5$.

$$|P_{xy}^{\text{SP}}| \geq \left\lfloor \frac{N-1}{2} \right\rfloor \geq \frac{N}{3} \tag{16}$$

Therefore, any node can reach a minimum of $N/3$ nodes in either direction around the ring using shortest-path routing. Conversely, any node can be reached by a minimum of $N/3$ nodes in either direction. Then we can place 3 controllable nodes, v_1, v_2, and v_3, such that shortest-paths $P_{v_i v_j}^{\text{SP}}$ and $P_{v_i v_k}^{\text{SP}}$ are edge-disjoint for all permutations $i,j,k \in \{1,2,3\}$. The overlay edges between these controllable nodes then form a bidirectionally connected ring as shown in Fig. 1, making use of all-paths between the controllable nodes. Every uncontrollable u is on the shortest-path between two controllable nodes

v_i and v_j; thus, by optimal substructure, paths $P^{\text{SP}}_{uv_i}$ and $P^{\text{SP}}_{uv_j}$ are edge-disjoint paths from u to v_i and v_j, and paths $P^{\text{SP}}_{v_iu}$ and $P^{\text{SP}}_{v_ju}$ are edge-disjoint paths from v_i and v_j to node u. Then every path in the network is either a shortest-path or can be formed as an n-concatenation of shortest paths, and the all-paths condition is satisfied with exactly 3 controllable nodes. ■

REFERENCES

[1] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. ACM SOSP*, Oct. 2001, pp. 131–145. [Online]. Available: http://doi.acm.org/10.1145/502034.502048

[2] L. Bui, R. Srikant, and A. Stolyar, "Novel architectures and algorithms for delay reduction in back-pressure scheduling and routing," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 2936–2940.

[3] B. Fortz and M. Thorup, "Internet traffic engineering by optimizing OSPF weights," in *Proc. IEEE INFOCOM*, Mar. 2000, pp. 519–528.

[4] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Found. Trends Netw.*, vol. 1, no. 1, pp. 1–144, 2006. [Online]. Available: http://dx.doi.org/10.1561/1300000001

[5] J. Han, D. Watson, and F. Jahanian, "Topology aware overlay networks," in *Proc. IEEE INFOCOM*, Mar. 2005, pp. 2554–2565. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1498540

[6] N. M. Jones, "Practical algorithms for distributed network control," Ph.D. dissertation, Dept. Aeronautics Astronautics Massachusetts Inst. Technol., Cambridge, MA, USA, 2013.

[7] N. M. Jones, G. S. Paschos, B. Shrader, and E. Modiano, "An overlay architecture for throughput optimal multipath routing," in *Proc. ACM MobiHoc*, Aug. 2014, pp. 73–82.

[8] W. Khan, L. B. Le, and E. Modiano, "Autonomous routing algorithms for networks with wide-spread failures," in *Proc. IEEE MILCOM*, Oct. 2009, pp. 1–6. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5379792

[9] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic power allocation and routing for time-varying wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 1, pp. 89–103, Jan. 2005. [Online]. Available: http://ieeexplore.ieee.org/search/srchabstract.jsp?tp=&arnumber=1208724%&queryText%3Djsac+neely+modiano+rhors%26openedRefinements%3D*%26searchField%3D%Search+All

[10] M. E. J. Newman, *Networks: An Introduction*. New York, NY, USA: Oxford Univ. Press, 2010.

[11] G. S. Paschos and E. Modiano, "Throughput optimal routing in overlay networks," in *Proc. IEEE Allerton*, Oct. 2014, pp. 401–408.

[12] G. S. Paschos and E. Modiano. (Sep. 2014). "Throughput optimal routing in overlay networks." [Online]. Available: http://arxiv.org/abs/1409.1739

[13] L. L. Peterson and B. S. Davie, *Computer Networks: A Systems Approach*, 4th ed. San Francisco, CA, USA: Morgan Kaufmann, 2007.

[14] A. Rai, C.-P. Li, G. Paschos, and E. Modiano, "Loop-free backpressure routing using link-reversal algorithms," in *Proc. ACM MobiHoc*, Jun. 2015, pp. 87–96 .

[15] J. Ryu, L. Ying, and S. Shakkottai, "Back-pressure routing for intermittently connected networks," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–5. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5462224

[16] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Autom. Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.

[17] D. Xu, M. Chiang, and J. Rexford, "Link-state routing with hop-by-hop forwarding can achieve optimal traffic engineering," *IEEE/ACM Trans. Netw.*, vol. 19, no. 6, pp. 1717–1730, Dec. 2011.

[18] L. Ying, S. Shakkottai, and A. Reddy, "On combining shortest-path and back-pressure routing over multihop wireless networks," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 1674–1682. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5062086

Nathaniel M. Jones received the degree of B.S. in Electrical Engineering and the degree of M.S. in Electrical and Computer Engineering from the Georgia Institute of Technology in 2000 and 2004, respectively, and the Ph.D. degree in the field of communications and networks from the Massachusetts Institute of Technology (MIT) in 2013. He was with the Georgia Tech Research Institute from 2000 to 2005, and MIT Lincoln Laboratory from 2005 to 2016. He is currently with Akamai Technologies.

Georgios S. Paschos received the diploma degree in electrical and computer engineering from the Aristotle University of Thessaloniki, Greece, in 2002, and Ph.D. degree in wireless networks from the Electronics Communication Engineering Department, University of Patras (supervisor Prof. Stavros Kotsopoulos), Greece, in 2006. From 2007 to 2008, he was an ERCIM Post-Doc Fellow in VTT, Finland, with the team of Prof. Norros. He was an Adjunct Lecturer with the Department of Electrical and Computer Engineering, University of Thessaly, from 2009 to 2011. He was with the Massachusetts Institute of Technology for two years with the team of Prof. Eytan Modiano. From 2008 to 2014, he was affiliated with The Center of Research and Technology Hellas–Informatics and Telematics Institute CERTH-ITI, Greece, joined with Prof. Leandros Tassiulas. He is currently a Principal Researcher with Huawei Technologies, Paris, France, since 2014, leading the Network Control and Resource Allocation team. Two of his papers won the Best Paper Award, in GLOBECOM 07 and IFIP Wireless Days 09, respectively. He serves as an Associate Editor for the IEEE/ACM TRANSACTIONS ON NETWORKING, and as a TPC member of INFOCOM, WiOPT, and Netsoft.

Brooke Shrader received the B.S. degree from Rice University, the M.S. degree from the Swedish Royal Institute of Technology, and the Ph.D. degree from the University of Maryland, College Park, MD, USA, all in electrical engineering. She is a Senior Member of Technical Staff with the Massachusetts Institute of Technology Lincoln Laboratory, where she has been since 2008. Her research interests lie in communication systems, wireless networks, and related disciplines, including information theory, control, and queueing models.

Eytan Modiano (S'90–M'93–SM'00–F'12) received the B.S. degree in electrical engineering and computer science from the University of Connecticut, Storrs, CT, USA, in 1986, and the M.S. and Ph.D. degrees in electrical engineering from the University of Maryland, College Park, MD, USA, in 1989 and 1992, respectively. He was a Naval Research Laboratory Fellow from 1987 and 1992 and a National Research Council Post-Doctoral Fellow from 1992 to 1993. From 1993 to 1999, he was with the Massachusetts Institute of Technology (MIT) Lincoln Laboratory. Since 1999, he has been with the Faculty at MIT, where he is currently a Professor with the Department of Aeronautics and Astronautics and with the Laboratory for Information and Decision Systems. His research is on communication networks and protocols with emphasis on satellite, wireless, and optical networks. He is an Associate Fellow of the AIAA, and served on the IEEE Fellows Committee. He was the co-recipient of the MobiHoc 2016 Best Paper Award, the WiOpt 2013 Best Paper Award, and the SIGMETRICS 2006 Best Paper Award. He is an Editor-at-Large for the IEEE/ACM TRANSACTIONS ON NETWORKING, and served as Associate Editor for the IEEE TRANSACTIONS ON INFORMATION THEORY and the IEEE/ACM TRANSACTIONS ON NETWORKING. He was the Technical Program Co-Chair for the IEEE WiOpt 2006, the IEEE INFOCOM 2007, the ACM MobiHoc 2007, and DRCN 2015.