# Investigation of Backpressure Based Policies for Routing in Developing Regions

Manikandan Somasundaram
mani@cs.umass.edu

Arun Venkataramani
arun@cs.umass.edu

Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003

## ABSTRACT

Wireless edge networks can play a huge role in bridging the digital divide between the developing and the developed world. But routing across and within these wireless edge networks remains a challenge. Backpressure-based scheduling can theoretically achieve the capacity region in wireless networks, but realizing this property in practice has been an interesting and hard challenge. We implement classical backpressure on Disruption Tolerant Networks (DTNs) and evaluate its utility in practice. We also augment the classical backpressure algorithm with replication and a novel *label exchange* mechanism. Our results indicate that classical backpressure (both with and without replication) underperforms when compared to the state of the art heuristic based DTN routing algorithms. We also consider the various ways in which backpressure can be used in wireless networks of developing regions.

## 1. INTRODUCTION

Communication technologies and information have a direct and vital impact on social and economic well being of people in developing regions. Emerging wireless edge networks (like meshes, MANETs and DTNs) provide an inexpensive and rapid way for communication and information access in rural regions. Due to the monetary overhead associated with establishing traditional infrastructures, these networks form a feasible solution for rural communication and "last mile" information delivery. But routing within and across these wireless edge networks remains a challenge.

Backpressure based scheduling has been extensively studied in theory for routing and scheduling in wireless networks [21, 20, 13, 16]. Due to its throughput-optimal[1] properties, it is intuitive to consider backpressure as a panacea for routing within and across the different wireless edge networks. Nevertheless, in practice classical backpressure[2] suffers from two drawbacks. First, it requires the solution of a complex optimization problem and is centralized. Second, under light load, the packets in the network essentially perform random walks, adversely affecting delay and transient throughput.

Our work is motivated by the observation that we can circumvent the first drawback of classical backpressure on Disruption Tolerant Networks (DTNs), while retaining the theoretical throughput-optimal properties. Disruption Tolerant Networks (DTNs) is a collective term given to communication in emerging wireless networking scenarios, which are characterized by highly unstable wireless links and increased absence of contemporaneous end-to-end paths between the nodes. Since classical backpressure can achieve the capacity region and it can be easily implemented on DTNs, it seems to be a great candidate for routing and scheduling on DTNs.

We evaluate classical backpressure on real traces against other heuristic based DTN routing algorithms. To the best of our knowledge, this is the first practical investigation of backpressure on DTNs. In our experiments classical backpressure underperforms in terms of both transient throughput and delay. The low transient throughput suggests that the limited contacts and frequently changing topologies in DTNs adversely affect the ability of backpressure to learn the network characteristics. The underperformance with respect to delays is not surprising as classical backpressure is not designed to optimize delay. But it is definitely a cause of concern, since delay is an important metric in practice.

We augment classical backpressure with replication and a novel *label exchange* mechanism to address the above issues. Replication also helps in addressing the first drawback of classical backpressure. With these enhancements we obtain a significant improvement in throughput and delay, but still not overwhelmingly so. Moreover, the improvement in performance mainly stems from replication rather than from backpressure itself. Even after addressing the two drawbacks of classical backpressure, we are far from realizing benefits of classical backpressure for routing in practice on DTNs

Our work points out that classical backpressure – even augmented backpressure – is not a panacea for routing in wireless networks, despite the rather intuitive appeal of using classical backpressure for routing, driven by its theoretical properties. However, implementations using backpressure to augment existing routing schemes have been shown to be successful [10, 23, 15]. Perhaps it is the case that

[1]A routing/scheduling policy is throughput-optimal if it can stabilize a network traffic which can be stabilized by any other routing/scheduling policy.

[2]Throughout this paper, when we use the term "classical backpressure", we will be referring to the algorithm in [21].

.

using backpressure in conjunction with existing schemes is more likely to be beneficial, rather than starting out with backpressure itself for routing.

The remainder of the paper is organized as follows. §2 discusses the feasibility of classical backpressure on DTNs. We evaluate classical backpressure on DTNs in §3. §4 outlines the augmented backpressure algorithm for DTNs. The augmented algorithm is evaluated in §5. Related work is discussed in §6. We discuss the results in §7 and conclude the paper in §8.

## 2. CLASSICAL BACKPRESSURE ON DTNS

In this section, we describe the classical backpressure based scheduling for routing and flow control in wireless networks in necessary detail. [21] presents the classical backpressure algorithm in complete detail. Later in this section, we discuss classical backpressure routing for DTNs.

### *Classical Backpressure*

Consider a wireless network modeled as a graph $G(V, E)$, where $V$ is the set of nodes in the network and $E$ is the set of links in the network. Each node maintains a queue for each destination. Upon reception of a packet at a node, the node examines the destination field of the packet. If the node is the destination of the packet, it delivers the packet to the appropriate application layer, else it enqueues it in the appropriate queue in a FIFO manner.

Time is slotted into equal sized epochs. In each epoch, some links are *active* and the rest are *inactive*. Transmissions can take place only over the active links. Let $i, j \in V$ and $l$ be a link from $i$ to $j$. Let $Q_{id}(t), i, d \in V$ denote the FIFO queue at node $i$ for destination node $d$ at time slot $t$. Then the differential backlog over link $l$ for destination $d$ is given by

$$D_{ld}(t) = |Q_{id}(t-1)| - |Q_{jd}(t-1)|$$

The weight of link $l$ at time $t$ is given by

$$D_l(t) = \max_{d \in V} \{D_{ld}(t)\}$$

$D(t) = (D_l(t) : l = 1, 2, \ldots, |E|)$ is the weight vector at time $t$.

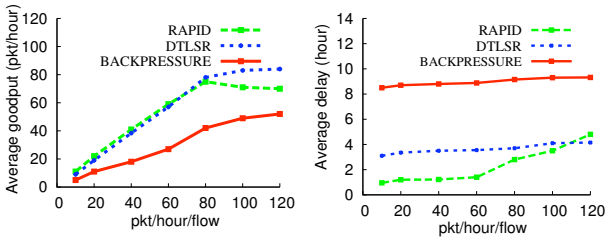
An *activation set* is the set of links which can be active simultaneously (i.e., the links don't interfere with each other). The activation set is represented by its activation vector (a binary vector of size $|E|$). A constraint set $S$ contains all activation vectors of the system. The backpressure scheduling algorithm has to pick a $c \in S$ for slot $t$, such that $D^T(t)c$ is maximized. In other words we have to pick the maximum weighted activation vector $\hat{c}$, where

$$\hat{c} = \arg\max_{c \in S} \{D^T(t)c\}$$

If link $l$ is set to active from the previous step, then node $i$ transmits a packet to node $j$ from the destination queue which has the largest differential backlog. The above steps are then repeated for each time slot.

### *Why Classical Backpressure is Easy on DTNs*

Classical backpressure is known to achieve the maximum throughput region. But, implementing the backpressure policy requires solving the NP-complete optimization problem – $\max_{c \in S} \{D^T(t)c\}$ – at each time slot $t$. Solving this NP-complete problem for each time slot in a distributed manner is challenging. Fortunately, we can circumvent this problem in DTNs using a simple and realistic assumption. We can assume that the size of the largest connected component in the graph $G$ is only 2. This assumption is realistic in DTNs, where pairs of nodes meet infrequently. Using this assumption, any link which is discovered in the DTN environment can be considered to be an active link. This makes the implementation of classical backpressure algorithm on DTNs trivial. It should be noted that the above assumption does not hold true for other kinds of networks (meshes, MANETs), where connected components are larger in size. Hence implementing classical backpressure on these kinds of networks remains challenging.




(a) Throughput (b) Delay

**Figure 1: Average throughput and delay on DieselNet for single day of traces**

## 3. EVALUATION OF CLASSICAL BACKPRESSURE ON DTNS

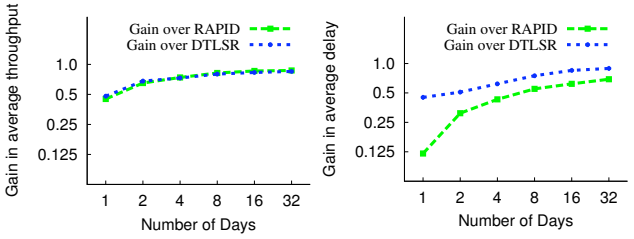
Given that classical backpressure can theoretically achieve the capacity region and is easy to implement on DTNs, it is interesting to see how it actually performs on DTNs. We use delay and throughput metrics to evaluate classical backpressure against other heuristic based DTN routing algorithms.

We compare classical backpressure with two other DTN routing protocols: DTLSR [7] and RAPID [3]. RAPID is a replication routing protocol for DTNs, which was shown to outperform many other replication routing protocols [3]. We choose DTLSR as a representative protocol of forwarding based DTN routing protocols.

We use DieselNet and Haggle traces for our evaluation. DieselNet is a repository of traces collected from a vehicular DTN testbed consisting of 40 buses in Amherst, Massachusetts. Haggle comprises of traces of a number of mobile devices carried by people in Cambridge, UK. A more detailed description of the traces can be found at [6, 17]

We use Qualnet [2] simulator for the evaluation. Qualnet is a commercial network simulator originating from the DARPA Global Mobile communications Networking project (GlomoSim) [1]. Qualnet provides a platform for comparison of alternative protocols at each layer. It has a modular, layered stack design and uses a discrete event simulator. It has been used in many recent works [1, 4].

We generate packets of size 1KB for the buses on road. While generating packets for each day, we make sure that the source bus and the destination bus are available on road on that day. We stop generating packets for/from a bus after its last meeting. This avoids creation of undeliverable




(a) Throughput (b) Delay

**Figure 2: Average gain in throughput and delay on DieselNet over multiple days of traces**

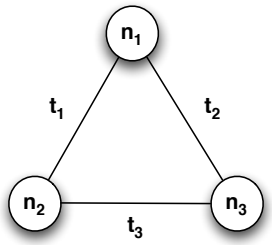



**Figure 3: Drawbacks of forwarding approach of classical backpressure: $n_1$ meets $n_2$ at time $t_1$ and $n_3$ at time $t_2$; $n_3$ meets $n_2$ at time $t_3$**

packets. For buses that are on road, each bus generates packets for every other bus at a constant rate. Each data point is an average of five different runs on five different days of the traces. Each run comprises of 30 concurrent flows between randomly chosen source destination pairs.

We perform two kinds of experiments. In the first, we evaluate the throughput and delay of the protocols on single day traces. In the second, we use traces of multiple days stitched together. We quantify the load in the network in terms of the rate at which the packets are injected into the network.

## 3.1 Single Day Traces

Figure 1 show the average throughput and average delay of the packets on DieselNet traces with different packet injection rates in the network. We increase the load (packet injection rate) in the network, till the network is not able to sustain it (above 80 pkt/hour/flow). The delay of classical backpressure is more than twice of the other protocols and the throughput of classical backpressure is less than half of the other protocols.

The consistent underperformance of classical backpressure on single day traces across all kinds of loads indicates that most of the packets end up essentially doing random walks. In other words, the contact opportunities on single day traces are not enough for backpressure to learn about the network characteristics.

## 3.2 Stitched Traces

For experiments in this subsection, we stitch traces of multiple consecutive days together. In other words, each experiment will simulate multiple days of traces, instead of a single day of traces as in the previous subsection. The continuity of the traces over the multiple days is ensured. That is, if a bus is not able to deliver some packets on a day, it can deliver them on any of the subsequent days. Further, for all the experiments in this subsection, the packet injection rate into the network is constant (60 pkt/hour/flow).

The gain in average delay of classical backpressure over RAPID is the ratio of the average delay of RAPID over the average delay of classical backpressure. The gain in average throughput of classical backpressure over RAPID is the ratio of the average throughput of classical backpressure over the average throughput of RAPID. Gains over DTLSR are also defined in similar manner. For both average delay and average throughput, a higher gain indicates a better performance of classical backpressure.

Figure 2(a) and Figure 2(b) show gain in average throughput and delay respectively of the classical backpressure algorithm against RAPID and DTLSR, with traces of different number of consequent days stitched together. The gain in average delay as well as average throughput improve significantly as the number of days stitched together increases. For example, the gain in average throughput over RAPID is only 0.45 over a single day of trace, but when 32 days of traces are stitched together, the gain in average throughput over RAPID almost doubles (0.87). Similarly the gain in delay over RAPID increases by 5× as more number of days are stitched together.

We conclude from the above experiments that the classical backpressure algorithm requires a large number of packets and contacts in the network for high gains. In the next subsection we look at the drawbacks of the forwarding approach of classical backpressure on DTNs.

## 3.3 Drawbacks of Forwarding in Classical Backpressure on DTNs

In classical backpressure, by default the node will delete the copy of the packet once it knows that the packet has been delivered to a next hop. We observe two main cases against this *forwarding* component of classical backpressure algorithm.

First, it prevents the classical backpressure algorithm from utilizing the transmission opportunities that might arise in the future. As an example, consider the scenario with three nodes in Figure 3. Assume that the meeting durations are large enough that a node can successfully transmit a packet to the other node. Let $n_1$ have a packet destined to $n_3$ and let no other packet be in the network. Assume $t_2 > t_1$ and $t_3 > t_1$. If $t_2 \ll t_3$, then according to classical backpressure $n_1$ will transmit the packet to $n_2$ and delete the packet from its storage. Later, when $n_1$ has a direct contact with $n_3$ at time $t_2$, even though it has an opportunity to transmit the packet, it will not have a copy of the packet with it.

Second, it results in many unnecessary transmissions. In the same scenario, if $n_1$ and $n_2$ meet each other after regular intervals and if $n_3$ is disconnected from both of them for a long time ($t_2, t_3 \sim \infty$), then the packet will end up shuttling between the first two nodes resulting in unnecessary transmissions.

Both the effects are highly pronounced in DTNs. We observe that in bandwidth constrained DTNs, it is beneficial to keep the copy of the packet in the storage till the node is sure that the packet has been delivered to the destination. This approach is known as *forwarding with caching*. To avoid looping of packets (as in the second case), the nodes can

first exchange information about the packets in the storage at the start of every contact. This exchange ensures that the same node does not receive the same packet more than once, resulting in efficient use of the limited bandwidth in DTNs.

Based on the above insights, we develop an augmented backpressure based algorithm for DTNs called AUGBP. It uses *forwarding with caching* and a novel *label exchange mechanism*. We describe AUGBP in detail in the next section.

# 4. AUGMENTED BACKPRESSURE FOR DTNS

We augment classical backpressure and create AUGBP, a DTN routing protocol. We describe AUGBP in this section.

Initially, when a packet originates, the network has only one copy of the packet in the network. We call it the *primary* copy of the packet. As the packet is transmitted among the nodes in the network, multiple nodes can have a copy of the packet with a *forwarding with caching* approach. At any time, only one of the copies of the packet in the network will be designated as the primary copy of the packet. The remaining copies of the packet in the network are designated as *secondary* copies.

| | |
|---|---|
| $\mathrm{P}(X)$ | Set of primary packets at node $X$ |
| $\mathrm{P}(X, D)$ | Set of primary packets at node $X$ destined to $D$ |
| $\mathrm{S}(X)$ | Set of secondary packets at node $X$ |
| $\mathrm{S}(X, D)$ | Set of secondary packets at node $X$ destined to $D$ |
| $\mathrm{Promoted}(X)$ | Set of packets promoted from being a secondary copy to a primary copy in node $X$ |
| $\mathrm{Demoted}(X)$ | Set of packets demoted from being a primary copy to a secondary copy in node $X$ |

**Table 1: Common variables used**

**Algorithm 1** $\mathrm{AUGBP}(X, Y)$

***Direct Delivery:***
Receive packets in $Y$ destined to $X$.

***Metadata Exchange:***
Receive $\mathrm{P}(Y)$ and $\mathrm{S}(Y)$.
Receive cumulative ack for delivered packets and delete stale packets.

***Label Exchange:***
**for all** destination $D$ **do**
  Select $\min(|\mathrm{P}(X, D) \cap \mathrm{S}(Y, D)|, \frac{|\mathrm{P}(X,D)|-|\mathrm{P}(Y,D)|}{2})$ number of packets from $\mathrm{P}(X, D) \cap \mathrm{S}(Y, D)$ and add them to $\mathrm{Demoted}(X)$
**end for**
Receive $\mathrm{Demoted}(Y)$
$\mathrm{Promoted}(X) \leftarrow \mathrm{Demoted}(Y)$

***Data Transfer:***
Apply classical backpressure described in §2 with caching on $\mathrm{P}(X)$ and $\mathrm{P}(Y)$.
Transmit packets in $\mathrm{S}(X) - (\mathrm{P}(Y) \cup \mathrm{S}(Y))$ based on packet creation time.

When node $X$ meets node $Y$, node $X$ executes the stages in $\mathrm{AUGBP}(X, Y)$. We use the variables presented in Table 1 at every node. The neighbor discovery can be done by an underlying layer or by periodic broadcast of beacons.

### *Direct Delivery:*

When the two nodes discover each other, they first deliver all the packets destined to the neighbor.

### *Metadata Exchange:*

The nodes exchange information about the primary and secondary packets with them. This is followed by exchange of information about delivered packets in the network. This helps in removing stale packets from the network. It should be noted that no actual data packets are exchanged at this stage.

### *Label Exchange:*

Consider a primary copy of a packet $p$ which is tranmitted from $X$ to $Y$ by classical backpressure. Since $p$ is performing a random walk, it is possible that $Y$ might have come across $p$ earlier. With a *forwarding with caching* approach, $Y$ will have a secondary copy of $p$. In this case, it is unnecessary to transmit $p$. It is sufficient if $X$ demotes $p$ from being a primary copy to a secondary copy and $Y$ promotes $p$ from being a secondary copy to a primary copy. In other words, if $p$ has been marked as primary in $X$ and secondary in $Y$ and if by applying classical backpressure, it is possible that eventually the packet $p$ will be transfered to $Y$ (assuming the two nodes stay in contact for sufficient time), then $X$ will demote packet $p$ and $Y$ will promote packet $p$.

This stage simulates actual exchange of packets by classical backpressure, but incurs negligible amount of overhead compared to the classical backpressure. It is effectively reducing a number of transmissions which would have been required by the classical backpressure to a single transmission. Replication helps in reducing the delay and mitigating the impact of the first drawback of classical backpressure.

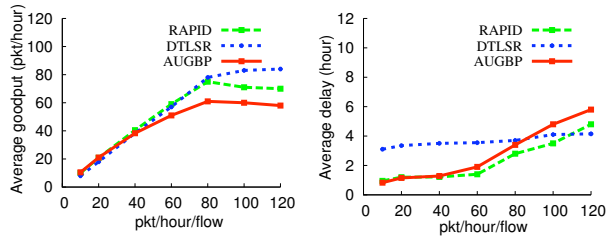
### *Data Transfer:*

After the label exchange process, classical backpressure is applied on the remaining packets (as described in §2) in the primary queues of both the nodes. This process stops when the corresponding queue size differences between the two nodes is $\leq 1$.

In the remaining time, the nodes can exchange secondary copies of packets which they have, but are not present with the neighbor in any form. Our algorithm uses packet creation time as a metric to order the packets in this stage of the algorithm. This process explicitly replicates packets.

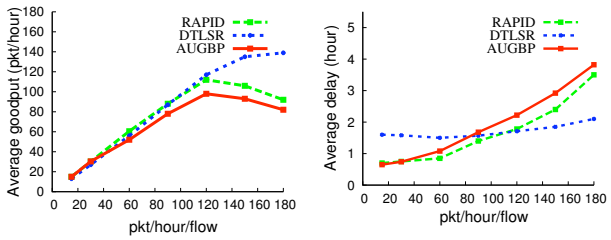
The contact opportunity might not be large enough for all these stages to take place, in which case only the feasible stages of transmissions occur. The way the stages are ordered ensures that the important transmissions are given higher priority. The initial stages of the algorithm resemble a pure forwarding algorithm. When extra transfer opportunities are available, the later stages of the algorithm are executed which resemble replication.

# 5. EVALUATION OF AUGMENTED BACKPRESSURE ALGORITHM ON DTNS




(a) Throughput (b) Delay

**Figure 4: Average throughput and delay on DieselNet for single day of traces**




(a) Throughput (b) Delay

**Figure 5: Average throughput and delay on Haggle for single day of traces**

In this section, we evaluate the performance of the augmented backpressure based algorithm for DTNs. We restrict our attention to only single day traces. The experimental setup is similar to the one described in §3.

Figures 4 and 5 give the average throughput and delay of the three protocols for varying load. AUGBP performs well under low to moderate load, but under high load, its performance degrades. The experiments suggest that the incorporation of replication element into the backpressure algorithm has boosted the performance in low to moderate loads. But as is common with any replication scheme, AUGBP faces the challenge of avoiding over-replication under high load. AUGBP relies on backpressure to constrain replication under high load. Unfortunately, backpressure proves to be inefficient, when compared to the heuristics used by the other protocols. AUGBP ends up over-replicating and degrading its performance under high load.

# 6. RELATED WORK

The works related to this paper can be classified into two categories. In the first, there has been focus on adapting classical backpressure for wireless mesh networks. There has also been efforts to enhance existing schemes with the concept of backpressure. In the second, there is a whole bunch of work on routing for DTNs. We are not aware of any work which has tried to adapt backpressure for DTNs and analyze the benefits of backpressure on DTNs.

## 6.1 Work on backpressure

Backpressure was first investigated to handle data bursts [12, 14] Tassiulas and Ephremides were the first to characterize the stability region of a network under stochastic arrival rate and showed that a backpressure based scheduling algorithm in fact achieves it [21]. But the algorithm is centralized and requires the solution of a complex optimization problem at every epoch. Tassiulas in a later work showed that the computation at each epoch can be reduced to linear complexity using a randomized scheduling approach [20] under certain conditions. But the algorithm is centralized.

Modiano et al [13] provided a distributed scheduling framework focusing on node-exclusive spectrum sharing models (i.e., having primary interference constraints). In this case a maximum matching is a feasible solution at every epoch. Sanghavi et al [16] provided a set of algorithms which guarantees a fixed fraction of the capacity region while using small and 'constant' overhead. These works are theoretical contributions, while we focus on actual implementations of backpressure.

Recently, architectures like Horizon [15], DiffQ [23] and Hop [10] have been proposed, which adapt the classical backpressure idea for practical implementations. They use backpressure to assist existing routing schemes. In contrast, our work focuses on using backpressure itself for routing.

## 6.2 Routing in DTNs

A number of routing protocols have been proposed for routing in DTNs. We can classify them using the following criteria.

### *Global vs Decentralized Protocols*

Global protocols gather network wide information like node meeting times or contact duration distributions to predict the next best hop. Global protocols for DTNs rely on the fact that social networks exhibit regularities over a long time with some deviations [8, 24, 25, 11, 5]. Buses and transit vehicles generally stick to a schedule. Movements of students in campus or people in a settlement is most likely to exhibit regularity with periodicity equal to a day [8, 24]. Decentralized protocols, on the other hand, try to make decisions locally, with minimal global information.

### *Single-copy vs Multi-copy Protocols*

Most of the DTN routing protocols use multiple copies of the same packet to increase the probability of the packet being delivered to the destination [3, 5, 22, 11, 18, 19, 9]. They are also known as replication-based protocols. Single-copy protocols on the other hand, have only one copy of the packet in the network [7, 25]. A single copy approach might not be as effective as a multi-copy approach due to the obvious gains of replication. However, replication based protocols have to also ensure that they do not replicate too much and overwhelm the al ready resource constrained DTNs. Hence, they use various heuristics to constrain replication [3].

# 7. DISCUSSION

Our analysis thus far has been restricted to backpressure on DTNs. In this section, we look at the inferences which can be drawn from our work on the general utility of backpressure on wireless networks.

Even after circumventing the implementation issues of classical backpressure on DTNs, we are not able to realize any practical benefits on DTNs. This result leads us to believe that backpressure by itself is unlikely to be beneficial for routing in any kind of wireless edge network. The reasoning behind this claim stems from the fact that the implementation of classical backpressure on other forms of

wireless edge networks (Mesh, MANET) is far more challenging than that on DTNs (refer §2).

However, this classical idea, in conjunction with existing schemes can be beneficial. Practical implementations using backpressure with existing schemes have been able to get better flow control, load balancing across links/paths, congestion control and fairness [23, 15, 10]. Perhaps it is the case that using backpressure in conjunction with existing schemes is more promising than using backpressure itself for routing. We can schedule the packet transmissions within a link based on the differential backlogs across the queues in the link. The destination queue which has the highest differential backlog can be given higher priority for routing. By using this backpressure-based scheme for scheduling (and not for routing) with existing routing protocol, the performance of the existing routing protocol can only become better. In addition, we will also be able to do better flow control, congestion control and load distribution.

Though classical backpressure underperforms under low load, the fact that classical backpressure can achieve the capacity region can be used to our advantage. Any non backpressure based routing policy can achieve only a subset of the capacity region attainable by backpressure. While using an existing routing policy, if the policy is not able to stabilize the input traffic, then we can dynamically switch to backpressure for routing. The routers can start performing backpressure dynamically when the existing routing policy is not able to stabilize the packet queues. This can be a theoretically motivated as well as practically beneficial approach towards routing.

# 8. CONCLUSION

In this paper, we have studied the utility of backpressure based policies for routing in wireless edge networks. It is appealing to use classical backpressure for routing in wireless networks due to its throughput optimal properties. But our experiments indicate that there are limited benefits to be had by using classical backpressure for routing. In a broader sense, we observe that classical backpressure can give benefits when used with existing routing schemes or under high load, but starting out with classical backpressure itself for routing is unlikely to be beneficial.

# 9. REFERENCES


[1] Delay tolerant networking - bundle protocol simulation. NASA, smc-it.jpl.nasa.gov/docs/Abstracts/M39.pdf.

[2] Qualnet. www.scalable-networks.com/.

[3] A. Balasubramanian, B. Levine, and A. Venkataramani. Dtn routing as a resource allocation problem. *SIGCOMM Comput. Commun. Rev.*, 37(4):373–384, 2007.

[4] A. Balasubramanian, R. Mahajan, A. Venkataramani, B. N. Levine, and J. Zahorjan. Interactive wifi connectivity for moving vehicles. In *SIGCOMM '08*, pages 427–438, New York, NY, USA, 2008. ACM.

[5] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. In *INFOCOM 2006*, pages 1–11, 2006.

[6] J. Burgess, B. N. Levine, R. Mahajan, J. Zahorjan, A. Balasubramanian, A. Venkataramani, Y. Zhou, B. Croft, N. Banerjee, M. Corner, and D. Towsley. CRAWDAD data set umass/diesel (v. 2008-09-14). Downloaded from http://crawdad.cs.dartmouth.edu/umass/diesel, Sept. 2008.

[7] M. Demmer and K. Fall. Dtlsr: delay tolerant routing for developing regions. In *NSDR '07*, pages 1–6, New York, NY, USA, 2007. ACM.

[8] J. Ghosh, C. Qiao, S. J. Philip, H. Ngo, and S. Yoon. Sociological orbit aware location approximation and routing (solar) in dtn. In *Technical Report, University at Buffalo, The State University of New York*, 2005.

[9] S. Jain, M. Demmer, R. Patra, and K. Fall. Using redundancy to cope with failures in a delay tolerant network. *SIGCOMM Comput. Commun. Rev.*, 35(4):109–120, 2005.

[10] M. Li, D. Agrawal, D. Ganesan, and A. Venkataramani. Block-switched networks: a new paradigm for wireless transport. In *NSDI'09*, pages 423–436, Berkeley, CA, USA, 2009. USENIX Association.

[11] C. Liu and J. Wu. An optimal probabilistic forwarding protocol in delay tolerant networks. In *MobiHoc '09*, pages 105–114, New York, NY, USA, 2009. ACM.

[12] P. P. Mishra and H. Kanakia. A hop by hop rate-based congestion control scheme. *SIGCOMM Comput. Commun. Rev.*, 22(4):112–123, 1992.

[13] E. Modiano, D. Shah, and G. Zussman. Maximizing throughput in wireless networks via gossiping. *SIGMETRICS Perform. Eval. Rev.*, 34(1):27–38, 2006.

[14] C. Özveren, R. Simcoe, and G. Varghese. Reliable and efficient hop-by-hop flow control. In *SIGCOMM '94*, pages 89–100, New York, NY, USA, 1994. ACM.

[15] B. Radunović, C. Gkantsidis, D. Gunawardena, and P. Key. Horizon: balancing tcp over multiple paths in wireless mesh network. In *MobiCom '08*, pages 247–258, New York, NY, USA, 2008. ACM.

[16] S. Sanghavi, L. Bui, and R. Srikant. Distributed link scheduling with constant overhead. *SIGMETRICS Perform. Eval. Rev.*, 35(1):313–324, 2007.

[17] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau. CRAWDAD data set cambridge/haggle (v. 2009-05-29). Downloaded from http://crawdad.cs.dartmouth.edu/cambridge/haggle, May 2009.

[18] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *WDTN '05*, pages 252–259, New York, NY, USA, 2005. ACM.

[19] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility. In *PERCOMW '07*, pages 79–85, Washington, DC, USA, 2007. IEEE Computer Society.

[20] L. Tassiulas. Linear complexity algorithms for maximum throughput in radio networks and input queued switches. In *INFOCOM '98. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 533–539 vol.2, Mar-2 Apr 1998.

[21] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. In *Decision and Control, 1990., Proceedings of the 29th IEEE Conference on*, pages 2130–2132 vol.4, Dec 1990.

[22] A. Vahdate and D. Becker. Epidemic routing for partially connected ad hoc networks. In *Technical Report, Duke University*, 2002.

[23] A. Warrier, S. Ha, P. Wason, I. Rhee, and J. Kim. Diffq: Differential backlog congestion control for wireless multi-hop networks. In *SECON '08*, pages 585–587, June 2008.

[24] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. In *Nature 393*, pages 440–442, 1998.

[25] Q. Yuan, I. Cardei, and J. Wu. Predict and relay: an efficient routing in disruption-tolerant networks. In *MobiHoc '09*, pages 95–104, New York, NY, USA, 2009. ACM.