O(congestion+dilation) Steps," *Combinatorica*, Vol. 14, No. 2, pp. 167–180, 1994.

[LMRi] T. Leighton, B. Maggs and A. Richa, "Fast Algorithms for Finding O(Congestion+Dilation) Packet Routing Schedules," *Combinatorica*, to appear.

[M] M. Mihail, "Conductance and Convergence of Markov Chains—A Combinatorial Treatment of Expanders," *Proc. of 30th FOCS*, pp. 526–531, 1989.

[MP] Y. Mansour, and B. Patt-Shamir, "Greedy Packet Scheduling on Shortest Paths", *Journal of Algorithms*, Vol. 14, No. 3, pages 99–129, 1993.

[MV] F. Meyer auf der Heide and B. Vöcking. "A packet routing protocol for arbitrary networks". *Proc. of STACS '95.*

[OR] R. Ostrovsky and Y. Rabani, "Local Control Packet Switching Algorithm," *Proc. of 29th STOC*, pp. 644–653, 1997.

[PG] A. Parekh, and R. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case," *IEEE/ACM Transactions on Networking*, 1 (3) pp. 344–357, 1993.

[PG2] A. Parekh, and R. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple-Node Case," *IEEE/ACM Transactions on Networking*, 2 (2) pp. 137–150, 1994.

[RT] Y. Rabani and É. Tardos. "Distributed packet switching in arbitrary networks". *Proc. of STOC '96.*

[ST] A. Srinivasan and C.-P. Teo. "A constant-factor approximation algorithm for packet routing, and balancing local vs. global criteria". *Proc. of STOC 97.*

[STs] G. Stamoulis and J. Tsitsiklis, "The Efficiency of Greedy Routing in Hypercubes and Butterflies," *IEEE Transactions on Communications*, 42 (11), pp. 3051–208, 1994.

[SV] C. Scheideler and B. Vöcking, "Universal Continuous Routing Strategies," *Proc. of 8th SPAA*, 1996.

[AL] B. Awerbuch and T. Leighton, "A Simple Local-Control Approximation Algorithm for Multicommodity Flow", *Proc. of 34th FOCS*, pp. 459–468, 1993.

[AL2] B. Awerbuch and T. Leighton, "Improved Approximation Algorithms for the Multicommodity Flow Problem and Local Competitive Routing in Dynamic Networks", *Proc. of 26th STOC*, pp. 487–496, 1994.

[AMS] B. Awerbuch, Y. Mansour, N. Shavit, "End-to-End Communication with Polynomial Overhead," *Proc. of 30th FOCS*, pp. 358–363, 1989.

[BFU] A.Z. Broder, A.M. Frieze, and E. Upfal, "A General Approach to Dynamic Packet Routing with Bounded Buffers", *Proc. of STOC '96.*

[BKR+] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, and D. Wiliamson, "Adversarial Queuing Theory", *Proc. of 28th STOC*, pp. 376–385, 1996.

[BU] A.Z. Broder, and E. Upfal, "Dynamic Deflection Routing on Arrays," *Proc. of 28th STOC*, 1996.

[C] R. Cruz, "A Calculus for Network Delay, Part I: Network Elements in Isolation," *IEEE Transactions on Information Theory*, pp. 114–131, 1991.

[C2] R. Cruz, "A Calculus for Network Delay, Part II: Network Analysis," *IEEE Transactions on Information Theory*, pp. 132–141, 1991.

[CMSV] R. Cypher, F. Meyer auf der Heide, C. Scheideler, and B. Vöcking. "Universal algorithms for store-and-forward and wormhole routing", *Proc. of STOC '96.*

[GL+] B. Ghosh, T. Leighton, B. Maggs, S. Muthukrishnan, G. Plaxton, R. Rajaraman, A. Richa, R. Tarjan, and D. Zuckerman, "Tight Analyses of Two Local Load Balancing Algorithms," *Proc. of 27th STOC*, pp. 548–558, 1995.

[GM] B. Ghosh and S. Muthukrishnan, "Dynamic Load Balancing on Parallel and Distributed Networks by Random Matchings," *Proc. of 6th SPAA*, pp. 226–235, 1994.

[HB] M. Harchol-Balter and P. Black, "Queuing Analysis of Oblivious Packet-Routing Algorithms," *Proc. of 5th SODA*, 1994.

[HW] M. Harchol-Balter and D. Wolfe "Bounding Delays in Packet Routing Networks," *Proc. of 27th STOC*, 1995.

[KPP] A. Kamath, O. Palmon, and S. Plotkin, "Routing and Admission Control in General Topology Networks with Poisson Arrivals," *Proc. of 7th SODA*, 1996.

[KT] J. Kleinberg and E. Tardos. "Disjoint Paths in Densely Embedded Graphs." *Proc. of FOCS'95.*

[L] F.T. Leighton. "Methods for message routing in parallel machines". Invited paper in *Proc. of STOC '92.*

[LMR] T. Leighton, B. Maggs, S. Rao, "Packet Routing and Job-Shop Scheduling in

modifications. If we assume that when an edge comes up after being down, the nodes are notified as to the status of their neighbors queues then the distributed protocol works with only minor modifications.

Directed networks: Our results still hold even if the network is directed. If the network is not strongly connected then we have to add a slight modification to our protocol; namely, the protocol never sends a packet with destination d, over an edge leading to node v, if there is no directed path from v to d. With this modification, our proofs still hold in the directed case. We still need to allow control bits to flow across edges in both directions (or to assume that nodes know the size of the buffers across their adjacent edges).

Other potential functions: In this paper we analyze our protocol using a "linear potential function" Φ (it is "linear" in the sense that the contribution of every packet is linear in its height). When "diffusion-type" algorithms have been used in other contexts, "exponential potential function" variants of the algorithms have yielded improved bounds (See [AL2, GL+]). It is worthwhile checking if exponential potential function variants of our protocols would yield improved bounds.

Acknowledgments We thank Matthew Andrews, Allan Borodin, Tom Leighton and Yuval Rabani for useful discussions.

References

[AAF+] M. Andrews, B. Awerbuch, A. Fernández, J. Kleinberg, T. Leighton, and Z. Liu, "Universal Stability Results for Greedy Contention-Resolution Protocols", *Proc. of 37th FOCS*, pp. 380–389, 1996.

[AAG+] Y. Afek, B. Awerbuch, E. Gafni, Y. Mansour, N. Shavit, A. Rosén. Slide - The Key to Polynomial End-to-End Communication. *Journal of Algorithms*, Vol. 22, No. 1, pages 158–186, January 1997.

[AAMR] W. Aiello, B. Awerbuch, B. Maggs, and S. Rao, "Approximate Load Balancing on Dynamic and Asynchronous Networks," *Proc. of 25th STOC*, pp. 632–641, 1992.

[AAP] B. Awerbuch, Y. Azar, and S. Plotkin, "Throughput Competitive On-Line Routing," *Proc. of 34th FOCS*, pp. 32–40, 1993.

[AAPW] B. Awerbuch, Y. Azar, and S. Plotkin, and O. Waarts, "Competitive Routing of Virtual Circuits with Unknown Duration," *Proc. of 5th SODA*, pp. 321–330, 1994.

[AFH+] M. Andrews, A. Fernández, M. Harchol-Balter, and T. Leighton, L. Zhang, "General Dynamic Routing with Per-Packet Delay Guarantees of O(distance + 1/session rate)," *Proc. of 38th FOCS*, pp. 294–302, 1997.

[AGR] Y. Afek, E. Gafni, A. Rosén, "Slide—A Technique for Communication in Unreliable Networks," *Proc. of 11th PODC*, pp. 35–46, 1992.

tion is at least $\Omega(m^3n^3W^4)$ when the window of time starts, then the potential function cannot grow over that window of time. We now use the upper bound we established above on the size of any buffer at the time that the estimate reaches 2^j. Using this upper bound, we have that at the time that the estimate reaches 2^j the value of the potential function is at most $g(W) = O(2mn(f(2^{j-1}) + 11W)^2) = O(m^4n^4W^4)$. We get that the value of the potential function will never grow over $O(m^4n^4W^4)$, which means that the height of any buffer will not exceed $O(m^2n^2W^2)$. This is an upper bound on the height of the buffers at times which are the boundaries of window of size 2^j. Using Claim 1, between these boundaries any buffer can grow by at most another $O(2^j) = O(W)$, which gives us an upper bound of $O(m^2n^2W^2)$ on the size of any buffer at any time. The above still holds even if the estimate grows above 2^j.

We now give an upper bound on the delivery time of any packet. By the above, we know that the protocol never holds more than $O(2mn \cdot m^2n^2W^2) = O(m^3n^3W^2)$ packets altogether. We also observe that we can write the bound on any single buffer as $f(c \cdot m^{1/4}n^{1/4}W)$, for some constant c. Therefore the estimate that the protocol uses will never exceed $c \cdot m^{1/4}n^{1/4}W$. This means that, every at most $2c \cdot m^{1/4}n^{1/4}W$ time steps, the drainage protocol is given control of the network. Thus every packet will be delivered within $(O(m^3n^3W^2)+n) \cdot 2cm^{1/4}n^{1/4}W^2 = O(m^{13/4}n^{13/4}W^3)$ time steps. [5]

We can conclude with the following theorem.

Theorem 9: If the sequence of packets is given by an $A(W, \varepsilon)$ adversary, then the total number of packets stored by BOUNDEDDT at any given time is at most $\mathcal{M}(W, \varepsilon) = O(m^3n^3W^2)$. Every packet is delivered to its destination in at most $\mathcal{T}(W, \varepsilon) = O(m^{13/4}n^{13/4}W^3)$ time steps.

[5] The bound can be easily made to be a function of ε.

5 Extensions and Discussion

In this section we briefly mention some extensions of our results.

Larger capacity: We can allow edges to have integer capacities. That is, in each time step each c-capacity edge can deliver c packets in each direction. The algorithm and analysis view any c-capacity edge as c 1-capacity edges. The proof works in a straightforward manner for this case except that the complexity now is polynomial in $\sum_e \text{capacity}(e)$ instead of m.

Robustness Diffusion-type protocols are often robust to edge failure. To model edge failure in our context, Definition 1 should be modified as follows. An $A(w, \varepsilon)$ adversary is allowed to inject packets, as before, and to have edges operational, i.e., "up", during certain time steps or not operational, i.e., "down," subject to the following constraint. For any time $t \in \mathbf{N}$, let $\mathcal{I}^t$ be the set of packets injected during the w time steps from t to $t + w - 1$, inclusive. Then, the adversary can associate with each packet $p = (s, d) \in \mathcal{I}^t$, a simple path from s to d, such that for each edge $e \in E$ and each direction of e, the number of times steps that the edge is up minus the number of paths in $\mathcal{I}^t$ which use the edge in that direction is at least $\lceil \varepsilon w \rceil$. With this definition, protocol BASIC should only be modified as to not send a packet over a "down" edge (the buffers associated with such edge are still maintained as before). The proof that the protocol has bounded queues is the same as before with only minor

to "realize" that the buffers are empty). Altogether we get that a packet can spend in the network at most $2((M'(w,\varepsilon)+n)\cdot 2w)+2w = O(m^{5/2}n^{5/2}w^2/\varepsilon)$ time steps, as needed. □

4.2 When w is Unknown

We now consider the general case of a protocol with a bound on the delivery time, and without the protocol having any information on the adversary. The high-level structure of this protocol, BOUNDEDDT, is to have an estimate w, and run BOUNDEDDT(w) until one of the processors realizes that the estimate is too low with respect to the real value W according to which the adversary injects the packets. Then, the estimate is doubled and the new version of the protocol is run. Each node maintains a value MAX which holds the maximum size of a buffer it has ever locally seen. This value is used to decide if a processor is "happy" with the estimate w or not. At the first time step of every window of $2w$ time steps, each processor sets its local `happy` flag according to the current estimate and its value MAX. Then a procedure similar to the one used for the `busy` flag is used, in a way that after exactly $2w$ time steps *all* the processors are "not happy" with the current estimate iff there was at least one processor that was not happy. Then, each processor doubles its estimate if it is not happy. We show below that the buffers are bounded, and that there exists an upper bound on the delivery time, in spite of the fact that the estimate w may at times be incorrect.

Let the sequence of packets be injected by an adversary $A(W,\varepsilon)$. Let j be such that $2^{j-1} \le W < 2^j$. Observe that the main difference between BOUNDEDDT(w) and BOUNDEDDT(w') is in how often does the drainage protocol get control of the network. Let $f(w)$ be a function such that as long as the size of the buffers in node v is bounded by $f(w)$ then node v is "happy" with the estimate w for the real value W. Let $w_i = 2^i$. We start with $w = w_{\lceil \log n \rceil}$ and do doubling of w whenever is necessary by at least one processor being unhappy, as described above.

We first prove an upper bound on the size of any single buffer. Let $f(w) = \Theta(m^{3/2}n^{3/2}w^2)$. If no buffer ever exceeds $f(2^{j-1}) \le f(W)$, we can take this as a bound. Now let t be the first time that in some processor v some buffer exceeds $f(2^{j-1})$. Assume that, at time t, the current estimate on W used by the processors is $w_i = 2^i$. Thus, by time step $t + 2w_i$ processor v will set its `happy` flag to false; it will continue to be false as long as the estimate is not increased to 2^j (possibly more if the size of the buffers grows even higher). After $j-i$ doublings happen the estimate reaches $w_j = 2^j$. This will take at most $(5/2)w_j \le 5W$ time steps ($2w_i$ steps until the `happy` flag is set to false, $2w_i$ until the first doubling, $2w_{i+1}$ until the second doubling and so on where finally we need $2w_{j-1}$ for the last doubling). During these time steps any buffer can grow by at most $11W$ by Claim 1. We get that if any buffer gets bigger than $f(2^{j-1})$ then after a bounded amount of time the estimate reaches the value 2^j, and at that time all buffers are bounded by $f(2^{j-1})+11W$. We therefore argue about the time after the estimate reaches 2^j. We will use an argument similar to the proof of Theorem 1.

We look inductively at windows of size $2\cdot 2^j$. Going over the proof of Theorem 8, one sees that if we give the drainage protocol one time step only every 2^j time steps *or less frequently* (i.e., when the estimate is even bigger) then the argument of "free" time steps in each window still holds. Thus, the potential function cannot increase over $2\cdot 2^j$ time steps by more than $O(mnW^2) = O(mn2^{2j})$. If the potential func-

1. If $t = iw + 1$ for some i perform the following: If `busy` $= 0$ then empty *all* buffers of BASIC and move the packets as input to the drainage protocol in node v; if `busy` $= 1$ and the buffers of the drainage protocol (in v) are empty set `busy` to 0; otherwise (if `busy` $= 1$ but the buffers of the drainage protocol are not empty) set `busy` to 1.

2. If $t = iw + 1$ for i even, allow the drainage protocol to operate for one time step (that is, send packets according to its decisions). Otherwise run protocol BASIC (that is, send packets according to the decisions of BASIC).

3. For every t, send the value of `busy` on all outgoing edges (this bit is to be piggybacked on packets if such are sent according to Step 2).
   OR the flag `busy` with all the values for `busy` received from the adjacent nodes.

We now prove that this protocol has bounded buffers and, at the same time, guarantees delivery in a bounded amount of time.

Theorem 8: If the packets are injected by an adversary $A(W, \varepsilon)$, for any $\varepsilon > 0$, and any integer $W \leq w$, then the total number of packets stored by BOUNDEDDT(w) at any time is at most $O(m^{5/2}n^{5/2}w/\varepsilon)$. In addition, each packet is delivered at most $O(m^{5/2}n^{5/2}w^2/\varepsilon)$ times steps after its injection.

Proof sketch: We first consider the buffers of BASIC maintained by BOUNDEDDT(w). Observe that these buffers are sometimes emptied (when the packets are moved to the drainage protocol), and that by Step 2 of BOUNDEDDT(w) every $2w$ time steps protocol BASIC is not allowed to use the edges. Apart from that, the behavior of BASIC is exactly the same as in the case when it operates independently. Also observe that our analysis of BASIC only depends on the definition of the adversary and that we had "free" time steps in each window of time of size W. Therefore, repeating the arguments of Lemma 3 with respect to pairs of windows of size w time steps (i.e., in one of them we use one "free" time step for the drainage algorithm and in the other we use the "free" steps for decreasing the potential), we get that the buffers of BASIC will hold at most $O(M(W, \varepsilon)) = O(M(w, \varepsilon)) = O(m^{5/2}n^{5/2}w/\varepsilon)$ packets, at any time. As a result we also get that the buffers of the drainage protocol never hold more than $O(m^{5/2}n^{5/2}w/\varepsilon)$ at any time. We get that the total number of packets stored by BOUNDEDDT(w) at any given time is at most $M'(w, \varepsilon) = O(M(w, \varepsilon)) = O(m^{5/2}n^{5/2}w/\varepsilon)$.

We now turn to prove that each packet is delivered in a bounded amount of time steps. First, note that once a packet is moved to the drainage protocol, the number of time steps until it is delivered is at most $(M'(w, \varepsilon) + n) \cdot 2w$. This is because at most $M'(w, \varepsilon)$ packets are moved to the empty buffers of the drainage protocol, which thus needs at most $M'(w, \varepsilon) + n$ time steps *in which it has control* to deliver all packets [MP]. However, the drainage protocol has control of the network once every $2w$ time steps. By the same argument, the buffers of the drainage protocol get emptied at most $(M'(w, \varepsilon) + n) \cdot 2w$ time steps after packets have been moved to them. Thus a packet, if not delivered earlier by BASIC, will be transfered to the drainage protocol at most $(M'(w, \varepsilon) + n) \cdot 2w + 2w$ time steps after it is injected (the additional $2w$ time steps come from the fact that it takes the network time

(BASIC) another "underground" drainage protocol. This protocol gets at certain time steps *all* the packets stored in the buffers of the main protocol, and will be responsible for their delivery. The buffers of the main protocol become empty at such an event. The drainage protocol receives more packets from the main protocol only after the buffers of the drainage protocol become empty (that is, all its packets are delivered). The advantage of the drainage protocol is that it receives only a bounded amount of packets (since the buffers of the main protocol are bounded), and all of them at the same time, and thus can deliver the packets in some bounded amount of time. To enable the drainage protocol to operate we have to assign it from time to time some time steps in which it can move its packets across edges (we shall prove that this can be done without significantly affecting the properties of the main protocol; this is in-spite of the fact that the main protocol will not be assigned to run in some of the time steps). Thus, we base our construction on a protocol, to be used as the "underground" protocol, as follows.

Drainage Protocol The drainage protocol is a protocol that is injected *once* with packets at the nodes. If the total number of packets injected is M, then all packets are delivered within $T(M)$ time steps (where we count only time steps in which the drainage protocol operates). Clearly there are such protocols, the simplest of them, maybe, being the one that routes any packet with destination d along the shortest path to d, with some arbitrary greedy congestion resolution rule. By the results of Mansour and Patt-Shamir [MP], for such protocols $T(M) \leq M + n$. Obviously, the buffers used by such protocol are bounded by M.

We also devise a mechanism that will allow all processors to know if the drainage protocol has delivered all its packets.[4] To do that, we send at every time step an additional bit across any edge (piggybacked on a packet if such is sent). We want that at the beginning of each time window of size w all processors will be synchronized as to this information. The mechanism works as follows. At the first time step of every window of size w, each processor checks if its buffers of the drainage protocol are empty or not. It sets a flag `busy` to 0 or 1 accordingly. Then each processor sends the value of this flag along all of its edges. When such a bit is received by a processor the flag is ORed with the bit received. The new value of the flag is then sent in the next time steps. Since we assume that $w \geq n$, at the beginning of the first time step of the next window the value of *every* flag is 1 iff there is at least one processor that had packets in the "drainage" buffers exactly w times steps before.

We now define the protocol. The protocol will maintain in each node the flag `busy`, all the buffers of the protocol BASIC, and in addition all the buffers and variables that the drainage protocol maintains. We define the protocol below using the drainage protocol as a black box: At each step either protocol BASIC or the drainage protocol will assume control in *all* processors, and will send packets according to their own separate decisions.

Protocol BOUNDEDDT(w):

Initially, in all nodes the flag `busy` is set to 0. Then, the following is performed by every node $v \in V$, at each time step $t \geq 1$.

[4] Alternatively, the processors can simply wait "enough time" (i.e., $T(M)$ steps, for the largest possible M according to the bound on the buffer size). However, we will anyway need the notification mechanism later for other purposes and therefore we already use it here.

Lemma 7: Let $q^t_{\max}$ be the size of the maximal buffer in the whole network at time t. Then $\Phi^{(t+w-1)''} - \Phi^t \leq 190mnw^2 - q^t_{\max}\lceil \varepsilon w \rceil$, for $n \geq 3$, $w \geq 4$, $m \geq 4$.

Proof: We follow and modify the proof of Lemma 3. The difference between the proof of the present lemma and the proof of Lemma 3, is that in the present case we cannot ignore any packet transfer, because packet transfers may *increase* the potential function. However we have by Lemma 5 an upper bound on this increase.

Let p_k, α_k, π^k and T^k be as in the proof of Lemma 3. By Lemma 6, the decrease in the potential function due to packet transfers along π^k at times T^k is at least $q^t_{v_k,u_k,d_k} - \ell(47w+4) \geq q^t_{v_k,u_k,d_k} - n(47w+4)$.

Using the $\lceil \varepsilon w \rceil$ "free" time steps we have another decrease of at least

$$\lceil \varepsilon w \rceil (q^t_{\max} - n(47w+4)) \geq \lceil \varepsilon w \rceil q^t_{\max} - nw(47w+4).$$

The increase in the potential function occurs for insertion of packets, and may also occur for packet transfers. The increase due to packet insertions is $\sum_{p_k \in \mathcal{I}^t} \alpha_k$, while the increase due to packet transfers is upper bounded by $2mw \cdot 22w$. Summing up we have

$$\begin{aligned}
&\Phi^{(t+w-1)''} - \Phi^t \\
&\leq \sum_{p_k \in \mathcal{I}^t} \alpha_k + 2mw \cdot 22w \\
&\quad - \sum_{p_k \in \mathcal{I}^t} [q^t_{v_k,u_k,d_k} - n(47w+4)] \\
&\quad - (\lceil \varepsilon w \rceil q^t_{\max} - nw(47w+4)) \\
&\leq \sum_{p_k \in \mathcal{I}^t} [q^t_{v_k,u_k,d_k} + 2w + 1] + 44mw^2 \\
&\quad - \sum_{p_k \in \mathcal{I}^t} [q^t_{v_k,u_k,d_k} - 48nw] - \lceil \varepsilon w \rceil q^t_{\max} + 48nw^2 \\
&\leq \sum_{p_k \in \mathcal{I}^t} [2(w+1) + 48nw] \\
&\quad + 44mw^2 + 48nw^2 - \lceil \varepsilon w \rceil q^t_{\max} \\
&\leq 2mw(nw + 48nw) + 44mw^2 + 48nw^2 - \lceil \varepsilon w \rceil q^t_{\max} \\
&\leq 190mnw^2 - \lceil \varepsilon w \rceil q^t_{\max} \, .
\end{aligned}$$

□

Finally, to give an upper bound on the size of any buffer, we use the above lemma in the same argument as those in the proof of Theorem 1. We get a bound of $O(m^{3/2}n^{3/2}w/\varepsilon)$ on the size of any buffer.

4 A Protocol with a Bound on Delivery Time

Protocol BASIC of the previous section guarantees bounded buffers as long as the packets are injected by some adversary $A(w,\varepsilon)$. However, this protocol does not guarantee a bound on the delivery time of the packets. Indeed, some packets may get stuck in the network forever. In this section we extend the protocol so as to guarantee that each packet is delivered within a bounded amount of time steps $T(w,\varepsilon)$. Whenever we mention protocol BASIC we refer to its totally distributed version.

4.1 The case where w is known

First we define a protocol that provides such guarantees on the delivery time, but has to know an upper bound w on the value W according to which the adversary injects the packets; we also assume that this bound $w \geq n$. We later eliminate this requirement.

The main idea of the extension is as follows. We will run in parallel to the basic protocol

that

$$q^t_{u,v,d}+1-q^t_{v,u,d} \leq 22w+a^t_{u,v,d}+1-q^t_{v,u,d} \leq 22w \ .$$

□

We now give a version of Lemma 2 that applies to the current protocol.

Lemma 6 : Let $e_1, e_2, \ldots, e_\ell$, for $e_i = (v_{i-1}, v_i) \in E$, be a simple path in G. Denote v_ℓ by D. Let $t_1, t_2, \ldots, t_\ell$ and t be any set of times satisfying $t \leq t_i \leq t + w - 1$ for all $1 \leq i \leq \ell$. Let Δ_i, for $1 \leq i \leq \ell$, be the *decrease* in the potential function due to a packet transfer in time step t_i on e_i from v_{i-1} to v_i. Then $\sum_{i=1}^{\ell} \Delta_i \geq q^t_{v_0,u,D} - \ell(47w+4)$, for any $(v_0, u) \in E$.

Proof: First, we consider a single Δ_i. Let d be such that a packet with destination d is sent along edge e_i at time t_i (we will deal with the case that no packet is sent later). The potential, corresponding to the buffers of destination d, along the edge e_i at time t_i (just before any packet was sent at this time step) is $\Phi^{t_i}_{v_{i-1},v_i,d} + \Phi^{t_i}_{v_i,v_{i-1},d}$. Therefore, the decrease in the potential caused by such packet transfer would be $q^{t_i}_{v_{i-1},v_i,d} - (q^{t_i}_{v_i,v_{i-1},d} + 1)$. Using Lemma 4, we have

$$\begin{aligned} \Delta_i &= q^{t_i}_{v_{i-1},v_i,d} - (q^{t_i}_{v_i,v_{i-1},d} + 1) \\ &\geq q^{t_i}_{v_{i-1},v_i,d} - (a^{t_i}_{v_i,v_{i-1},d} + 22w + 1) \ . \end{aligned}$$

Since the protocol sent a packet with destination d, by the resolution rule it follows that

$$q^{t_i}_{v_{i-1},v_i,d} - a^{t_i}_{v_i,v_{i-1},d} \geq q^{t_i}_{v_{i-1},v_i,D} - a^{t_i}_{v_i,v_{i-1},D} \ ,$$

and using Lemma 4 again we get

$$\begin{aligned} \Delta_i &\geq q^{t_i}_{v_{i-1},v_i,d} - (a^{t_i}_{v_i,v_{i-1},d} + 22w + 1) \\ &\geq q^{t_i}_{v_{i-1},v_i,D} - (a^{t_i}_{v_i,v_{i-1},D} + 22w + 1) \\ &\geq q^{t_i}_{v_{i-1},v_i,D} - (q^{t_i}_{v_i,v_{i-1},D} + 1) - 44w \ . \end{aligned}$$

In the special case where no packet is sent, $\Delta_i = 0$. This special case occurs only if for all d, $q^{t_i}_{v_{i-1},v_i,d} - a^{t_i}_{v_i,v_{i-1},d} \leq 0$. It follows that that for all d (including D)

$$\begin{aligned} & q^{t_i}_{v_{i-1},v_i,d} - (q^{t_i}_{v_i,v_{i-1},d} + 1) - 44w \\ &\leq q^{t_i}_{v_{i-1},v_i,d} - (a^{t_i}_{v_i,v_{i-1},d} - 22w + 1) - 44w \\ &= q^{t_i}_{v_{i-1},v_i,d} - a^{t_i}_{v_i,v_{i-1},d} - 1 - 22w \leq 0 \ . \end{aligned}$$

Thus the lower bound on Δ_i holds also in this case.

Note that we refer in the above to the size of buffers at time t_i. However, using Claim 1, we can relate these sizes to the sizes of buffers at time t. That is,

$$\begin{aligned} \Delta_i &\geq (q^t_{v_{i-1},v_i,D} - w - 1)) \\ &\quad - (q^t_{v_i,v_{i-1},D} + (2w+1) + 1) - 44w \\ &= q^t_{v_{i-1},v_i,D} - q^t_{v_i,v_{i-1},D} - (47w+3) \ . \end{aligned}$$

Therefore,

$$\begin{aligned} \sum_{i=1}^{\ell} \Delta_i &\geq \sum_{i=1}^{\ell} [q^t_{v_{i-1},v_i,D} - q^t_{v_i,v_{i-1},D} - (47w+3)] \\ &= -\ell(47w+3) + q^t_{v_0,v_1,D} - q^t_{v_\ell,v_{\ell-1},D} \\ &\quad + \sum_{i=1}^{\ell-1} (-q^t_{v_i,v_{i-1},D} + q^t_{v_i,v_{i+1},D}) \end{aligned}$$

Now since $v_\ell = D$ then $q^t_{v_\ell,v_{\ell-1},D} = q^t_{D,v_{\ell-1},D} = 0$. In addition, each of the terms $(q^t_{v_i,v_{i+1},D} - q^t_{v_i,v_{i-1},D})$ is at least -1, by invariant (1) (with respect to node v_{i+1}), and similarly the difference between $q^t_{v_0,v_1,D}$ and $q^t_{v_0,u,D}$ (for any u adjacent to v_0) is at most 1. Altogether, we get that $\sum_{i=1}^{\ell} \Delta_i \geq q^t_{v_0,u,D} - \ell(47w+4)$. □

We now give an analogue of Lemma 3.

2. (send updates:) For each $e = (v,u) \in E$, let $b^t_{v,u,d} = q^t_{v,u,d} - A^t_{v,u,d}$. If $b^t_{v,u,d} \leq 0$ let $r^t_{v,u,d} = \max(b^t_{v,u,d}, -4n)$. If $b^t_{v,u,d} > 0$ let $r^t_{v,u,d} = \min(b^t_{v,u,d}, 4n)$. Send $r^t_{v,u,d}$ to u (actually, the value $r^t_{v,u,d}$ is piggybacked on the packet sent from v to u on e at time t). Let $A^{t''}_{v,u,d} \leftarrow A^t_{v,u,d} + r^t_{v,u,d}$.

3. (receive updates:) For each $e = (v,u) \in E$, receive the value $r^t_{u,v,d}$ on edge e. Let $a^{t''}_{u,v,d} \leftarrow a^t_{u,v,d} + r^t_{u,v,d}$.

We now give a lemma that shows that the estimates on the buffer sizes that the nodes hold are always close to the real sizes of the buffers. Note that we assume that $w \geq n$; this implies that every estimate is updated at least once during each window of w time steps. On the other hand, it is possible that during the n time steps between two updates of $a_{u,v,d}$ the change in the buffer size will be much larger than $4n$ which is the maximal quantity by which $a_{u,v,d}$ is changed; yet, $q_{u,v,d}$ is changed by at most $O(w)$ during a single window of w time steps.

Lemma 4: For any t, any $e = (v,u) \in E$, and any $d \in V$, $|q^t_{u,v,d} - a^t_{u,v,d}| \leq 22w$.

Proof: We make the following observations about the changes in the values $q_{u,v,d}$ and $a_{u,v,d}$ during w time steps. For any t_1, t_2 such that $|t_1 - t_2| \leq w$, $|q^{t_1}_{u,v,d} - q^{t_2}_{u,v,d}| \leq 2w + 1 \leq 3w$, by Claim 1. For the same times t_1, t_2, $|a^{t_1}_{u,v,d} - a^{t_2}_{u,v,d}| \leq 8w$ (if w is divisible by n it is at most $4w$; otherwise it is $4w + 4n \leq 8w$). With these observations, the proof of the lemma is by induction on t. When the protocol starts we have $q^1_{u,v,d} = a^1_{u,v,d} = 0$ and in the next w steps $q_{u,v,d}$ and $a_{u,v,d}$ cannot grow to more than $3w$, and $8w$, respectively, by the above observations, and so the lemma holds for $t \leq w$. For the induction step, consider $t > w$ and let $t' = t - w$. The induction hypothesis implies that $|q^{t'}_{u,v,d} - a^{t'}_{u,v,d}| \leq 22w$. Consider two cases: (a) if $|q^{t'}_{u,v,d} - a^{t'}_{u,v,d}| \leq 11w$ then by the observations on the changes in $q_{u,v,d}$ and $a_{u,v,d}$ the difference in time t cannot grow to more than $22w$. (b) if $q^{t'}_{u,v,d}$ is larger (smaller) than $a^{t'}_{u,v,d}$ by more than $11w$ then it follows, again by the above observations, that $q_{u,v,d}$ will remain larger (smaller) than $a_{u,v,d}$ during each of the next w steps. Therefore, in *all* updates of $a_{u,v,d}$ during this time it will be increased (decreased) and so it will be increased (decreased) by total of at least $4n \cdot \lfloor w/n \rfloor$. In the worst case, $q_{u,v,d}$ is increased at the same time by $3w$, and so the difference at t is at most the difference at t' (for $w > 4n$). □

We now turn to prove an upper bound on the size of the buffers of the modified protocol. The proof follows the proof for protocol BASIC. The main difference stems from the fact that now the decision on sending a packet is based on the estimates of the sizes of buffers in adjacent nodes, rather than on the real sizes (each node still uses for its own buffers the real size). This may cause a different decision, and may lead to an *increase* in the potential function due to a packet transfer. We first give an upper bound on the possible increase in the potential function due to a packet transfer.

Lemma 5: For any time t and any $e = (v,u) \in E$, the transfer of a packet from v to u can increase the potential function by at most $22w$.

Proof: If a packet is sent from $Q_{v,u,d}$ to $Q_{u,v,d}$ at t then the *increase* in the potential function is exactly $q^t_{u,v,d} + 1 - q^t_{v,u,d}$. For the packet to be sent it must hold that $q^t_{v,u,d} - a^t_{u,v,d} > 0$. But $q^t_{u,v,d} - a^t_{u,v,d} \leq 22w$, by Lemma 4. It follows

v^*, u^*, and d^*, be such that $q^t_{\max} = q^t_{v^*,u^*,d^*}$. Using any simple path leading from v^* to d^* and the "free" time slots available for each of the edges along this path, we have by Lemma 2 that the *decrease* in the potential along this path, and at these time steps, is at least

$$\lceil \varepsilon w \rceil (q^t_{\max} - n(3w+4)) \geq \lceil \varepsilon w \rceil q^t_{\max} - nw(3w+4).$$

We can now sum all the increases and decreases identified above.

$$\begin{aligned}
&\Phi^{(t+w-1)''} - \Phi^t \\
&\leq \sum_{p_k \in \mathcal{I}^t} \alpha_k - \sum_{p_k \in \mathcal{I}^t} [q^t_{v_k,u_k,d_k} - n(3w+4)] \\
&\quad - (\lceil \varepsilon w \rceil q^t_{\max} - nw(3w+4)) \\
&\leq \sum_{p_k \in \mathcal{I}^t} [q^t_{v_k,u_k,d_k} + 2w + 1] - \\
&\quad \sum_{p_k \in \mathcal{I}^t} [q^t_{v_k,u_k,d_k} - 4nw] - \lceil \varepsilon w \rceil q^t_{\max} + 4nw^2 \\
&\leq \sum_{p_k \in \mathcal{I}^t} [2(w+1) + 4nw] + 4nw^2 - \lceil \varepsilon w \rceil q^t_{\max} \\
&\leq 2mw \cdot (nw + 4nw) + 4nw^2 - \lceil \varepsilon w \rceil q^t_{\max} \\
&\leq 11mnw^2 - \lceil \varepsilon w \rceil q^t_{max} \ ,
\end{aligned}$$

for $n \geq 3$, $w \geq 4$, $m \geq 4$. □

To complete the proof of Theorem 1, we consider times $\tau_i = i \cdot w + 1$, for $i \geq 0$. Now consider the value of the potential function at Φ^{τ_i} and at Φ^{τ_i+1}. Let $B = (A+1)/\sqrt{2}$ where $A = 11mnw/\varepsilon$.

We consider two cases. First, suppose that $\Phi^{\tau_i} \geq 2mn \cdot B^2$. This implies that there is at least one buffer with potential at least B^2. Recall that the height q and potential Φ of a buffer satisfy $q^2/2 < \Phi < (q+1)^2/2$. Thus, the height of this buffer is at least A. By Lemma 3, $\Phi^{\tau_i+1} \leq \Phi^{\tau_i}$. Now suppose that $\Phi^{\tau_i} < 2mn \cdot B^2$. By Lemma 3 (since the buffer size is always non-negative), the increase in the potential function is bounded by $11mnw^2$, and we have that $\Phi^{\tau_i+1} \leq \Phi^{\tau_i} + 11mnw^2$.

Since $\Phi^{\tau_0} = 0$ we can conclude that for $\tau_i = iw + 1$, $\Phi^{\tau_i} \leq 2mnB^2 + 11mnw^2$. Using the fact that $n \geq 3, w \geq 4, m \geq 4$ we can bound $B \leq 8mnw/\varepsilon$. Since the height of each buffer is at most the square root of twice the potential of the buffer, it follows that the height of each buffer at times τ_i is at most $17m^{3/2}n^{3/2}w/\varepsilon$. Since by Claim 1 each buffer can increase by at most $2w+1$ in any w consecutive time steps, we have an upper bound on the height of any buffer, at *any time*, of $O(m^{3/2}n^{3/2}w/\varepsilon)$. □

3.3 The Totally Distributed Protocol

The protocol BASIC defined above is not totally distributed. In particular, each node v uses for its decisions the size of the buffers at the other end of its edges. In order to make the protocol totally distributed, we replace in step 1 of BASIC the values $q^t_{u,v,d}$ stored in node v (that is, the real sizes of the buffers in adjacent nodes u) with values $a^t_{u,v,d}$, which will be estimates on $q^t_{u,v,d}$. Furthermore, node u will also hold values $A^t_{u,v,d}$, which the protocol will ensure is kept equal to the values $a^t_{u,v,d}$ at adjacent nodes v. The estimates will be updated using the following procedure, executed at each time step $t \in \mathbf{N}$ and node $v \in V$. This procedure allocates each edge $e = (v,u)$, in a round-robin way, to one of the n destinations. When e is allocated for destination d, node v sends to u the difference between the actual size of the buffer $q^t_{v,u,d}$ and the value of the estimate $A^t_{v,u,d}$, but only up to values in the range $[-4n, +4n]$. Thus, the number of bits that v uses for each such message is only $\lceil \log n \rceil + 4$. For convenience, we number the destinations $d \in V$ by numbers from 0 to $n-1$.

1. Let $d = (t-1) \bmod n$.

The following is our main lemma which implies that once some buffer gets sufficiently large the potential will not increase over w consecutive time steps.

Lemma 3: Let $q^t_{\max}$ be the size of the maximal buffer in the whole network at time t. Then $\Phi^{(t+w-1)''} - \Phi^t \leq 11mnw^2 - q^t_{\max}\lceil \varepsilon w \rceil$, for $n \geq 3$, $w \geq 4$, $m \geq 4$.

Proof: Consider the set of all packets $\mathcal{I}^t$ injected in time frame $[t,(t+w-1)'')$ for some time t. For each packet $p_k = (v_k, d_k) \in \mathcal{I}^t$ which was injected into the network during this time frame we will count, as promised, all the potential increase due to the injection and identify some packet transfers that will cause a decrease in potential.

We will first identify the increases in potential. Each packet $p_k = (v_k, d_k) \in \mathcal{I}^t$ was added at some time τ_k to a buffer Q_{v_k,u_k,d_k} in node v_k. Let α_k be the potential increase due to the injection of this packet. By our definitions, $\alpha_k \leq q^{\tau_k''}_{v_k,u_k,d_k}$. By Claim 1, $\alpha_k \leq q^t_{v_k,u_k,d_k} + 2w + 1$ for each $p_k \in \mathcal{I}^t$.

Now let us identify some of the decrease in potential. For each packet $p_k = (v_k, d_k) \in \mathcal{I}^t$, consider the path from its injection point, v_k, to its destination d_k, guaranteed by the definition of the $A(w,\varepsilon)$ adversary. Let this path be $\pi^k = e^k_1, e^k_2, \ldots, e^k_{\ell_k}$. The set of paths associated with all the packets of $\mathcal{I}^t$ have, by the definition of the adversary, the property that no edge is used more than $\lfloor (1-\varepsilon)w \rfloor$ times in either direction. This leads to the following claim

Claim 2: For each path π^k we can associate a sequence of times $T^k = \{t^k_1, \ldots, t^k_{\ell_k}\}$, where $t \leq t^k_j \leq t + w - 1$ for $1 \leq j \leq \ell_k$ (i.e., a time step for each edge in the path), in a way that satisfies the following two conditions:

1. for each edge, at most $\lfloor (1-\varepsilon)w \rfloor$ time steps (in each direction) are assigned to that edge; and,

2. for each edge, no two time steps assigned to the edge are the same.[3]

Thus, each edge, in each direction, will still have at least $\lceil \varepsilon w \rceil$ "free" time steps among the time steps from t to $t + w - 1$, inclusive.

Proof: An assignment can be achieved in a greedy manner by making the assignment first of π^1 then the assignment for π^2, etc. For each edge $e \in E$ and each direction $* \in \{+,-\}$ define $S^0_{e,*}$ to be the null set. If π^1 through π^{i-1} have been assigned denote $S^{i-1}_{e,*} \subset \{t, \ldots, t+w-1\}$ to be the set of times assigned to edge e in direction $*$. Now assign π^i as follows. For each edge e, and direction of the edge $*$, in π^i pick an arbitrary element of $\{t, \ldots, t+w-1\} \setminus S^{i-1}_{e,*}$ and add that element to $S^{i-1}_{e,*}$ to obtain $S^i_{e,*}$. Such element must exist by the definition of the adversary. If $S_{e,*}$ is the set after all paths have been assigned, by the restrictions on the adversary, $|S_{e,*}| \leq \lfloor (1-\varepsilon)w \rfloor$ for each $e, *$. □

By Lemma 2, for every packet $p_k \in \mathcal{I}^t$, the decrease in the potential function, due to packets transfers along π^k at times T^k is at least $q^t_{v_k,u_k,d_k} - \ell_k(3w+4) \geq q^t_{v_k,u_k,d_k} - n(3w+4)$.

As observed above these paths leave $\lceil \varepsilon w \rceil$ "free" time steps (during the time window) for each edge. We use this "free" time to get an additional reduction in the potential function. Let

[3] We emphasize that the paths are not given to the protocol and are used only for its analysis. In particular the protocol is likely to use different paths for the packets. Also, we emphasize that we make no hidden assumptions on the times in T^k other than that no edge is assigned more than $\lfloor (1-\varepsilon)w \rfloor$ time steps (e.g., we do not assume that $t^k_1 \leq t^k_2 \leq \ldots \leq t^k_{\ell_k}$).

transfer of a packet across an edge (can either decrease the potential function or not change it); (3) the redistribution of packets in buffers in a node (either decreases the potential function, or does not change it); and (4) the removal of packets at their destination (only decreases the potential function).

We will analyze the behavior of the potential function over windows of w time steps. Therefore, if we want to give an upper bound on the increase of the potential function from time t_1 to time t_2'', for $t_2 = t_1 + w - 1$, we can consider the increase due to *all* packets injected in this time frame, and the decrease due to *some* packet transfers during the same time, some of them transfered to their destination, and being removed. Below we state the central technical lemma of our analysis which quantifies the potential decrease due to some packet movements associated with packets in node v_0 destined for node D and with a simple path from v_0 to D.

Lemma 2 : Let $e_1, e_2, \ldots, e_\ell$, for $e_i = (v_{i-1}, v_i) \in E$, be a simple path in G. Denote v_ℓ by D. Let $t_1, t_2, \ldots, t_\ell$ and t be any set of times satisfying $t \leq t_i \leq t + w - 1$ for all $1 \leq i \leq \ell$. Let Δ_i, for $1 \leq i \leq \ell$, be the *decrease* in the potential function due to a packet transfer in time step t_i on e_i from v_{i-1} to v_i. Then $\sum_{i=1}^{\ell} \Delta_i \geq q^t_{v_0,u,D} - \ell(3w+4)$, for any $(v_0, u) \in E$.

The lemma implies that if $q^t_{v_0,u,D}$ is "very large" then $\sum \Delta_i$ is "very large" as well.

Proof: First, we consider a single Δ_i. The potential, corresponding to the buffers of destination D, along the edge e_i at time t_i (just before any packet was sent at this time step) is $\Phi^{t_i}_{v_{i-1},v_i,D} + \Phi^{t_i}_{v_i,v_{i-1},D}$. If a packet whose destination is D is indeed sent from v_{i-1} to v_i then the size of $Q_{v_{i-1},v_i,D}$ is decreased by 1 and the size of $Q_{v_i,v_{i-1},D}$ is increased by 1. Therefore, the decrease in the potential caused by such packet transfer would be $q^{t_i}_{v_{i-1},v_i,D} - (q^{t_i}_{v_i,v_{i-1},D} + 1)$. By the resolution rule that we use (Step 1 of protocol BASIC), the actual packet that is sent along edge e_i at time t_i is one that corresponds to some destination d that makes this difference maximal (and non-negative). Therefore, the decrease in potential can only be bigger; i.e.,

$$\Delta_i \geq q^{t_i}_{v_{i-1},v_i,D} - (q^{t_i}_{v_i,v_{i-1},D} + 1).$$

(In the special case where there is no destination d for which $q^{t_i}_{v_{i-1},v_i,d} - q^{t_i}_{v_i,v_{i-1},d}$ is positive no packet is sent on this edge in this direction at this time step; in such a case $\Delta_i = 0$ and the inequality still holds.) Note that we refer here to the size of buffers at time t_i. However, using Claim 1, we can relate these sizes to the sizes of buffers at time t. That is,

$$\begin{aligned}
\Delta_i &\geq (q^t_{v_{i-1},v_i,D} - w - 1)) \\
&\quad - (q^t_{v_i,v_{i-1},D} + (2w+1) + 1) \\
&= q^t_{v_{i-1},v_i,D} - q^t_{v_i,v_{i-1},D} - (3w+3) .
\end{aligned}$$

Therefore,

$$\begin{aligned}
\sum_{i=1}^{\ell} \Delta_i &\geq \sum_{i=1}^{\ell} [q^t_{v_{i-1},v_i,D} - q^t_{v_i,v_{i-1},D} - (3w+3)] \\
&= -\ell(3w+3) + q^t_{v_0,v_1,D} - q^t_{v_\ell,v_{\ell-1},D} \\
&\quad + \sum_{i=1}^{\ell-1} (-q^t_{v_i,v_{i-1},D} + q^t_{v_i,v_{i+1},D}) .
\end{aligned}$$

Now since $v_\ell = D$ then $q^t_{v_\ell,v_{\ell-1},D} = q^t_{D,v_{\ell-1},D} = 0$. In addition, each of the terms $(q^t_{v_i,v_{i+1},D} - q^t_{v_i,v_{i-1},D})$ is at least -1, by invariant (1) (with respect to node v_i), and similarly the difference between $q^t_{v_0,v_1,D}$ and $q^t_{v_0,u,D}$ (for any u adjacent to v_0) is at most 1. Altogether, we get that $\sum_{i=1}^{\ell} \Delta_i \geq q^t_{v_0,u,D} - \ell(3w+4)$. □

An immediate corollary to the above theorem is that the total number of packets stored by BASIC at any given time is at most $M(w,\varepsilon) = O(m^{5/2}n^{5/2}w/\varepsilon)$.

In the following we assume that the sequence of packets is injected by an $A(w,\varepsilon)$ adversary. Note however that the protocol does *not* know the values of w and ε (i.e., these values are not used by the protocol).

Proof: We adopt the following convention on the notation of times. For times which are the beginning of a time step we use the simple notation t (without any primes). To indicate the time just after sub-step 2 of the protocol is performed in time step t we use the notation t', and to indicate the time just after the last sub-step of the protocol has been executed we use the notation t'' (i.e., $t < t' < t'' < t+1$). We start with a simple claim about the change in the height of a given buffer.

Claim 1: For any $t_1 \leq t_2 \leq t_1 + w - 1$, and for all $u, v, d \in E$,

$$q^{t_1}_{v,u,d} - w - 1 \quad \leq q^{t''_2}_{v,u,d} \leq \quad q^{t_1}_{v,u,d} + 2w + 1.$$

(We remark that one can incorporate ε into the statement of this claim. This, however, yields only a minor improvement.)

Proof: By invariant (1), $\lfloor q^{t_1}_{v,*,d}/\delta_v \rfloor \leq q^{t_1}_{v,u,d} \leq \lfloor q^{t_1}_{v,*,d}/\delta_v \rfloor + 1$ for all $(v,u) \in E$. Consider the number of packets with destination d stored in v at time t''_2, which is denoted $q^{t''_2}_{v,*,d}$. In time frame $[t_1, t''_2]$ at most $\delta_v w$ such packets can arrive into node v across the edges. In addition, at most $\delta_v w$ such packets can be injected by the adversary into v (otherwise, the restriction on the adversary is violated). Thus, at most $2w\delta_v$ such packets can be added to the node. On the other hand, at most $\delta_v w$ such packets can be sent out by v in time frame $[t_1, t''_2]$. We get that $q^{t_1}_{v,*,d} - \delta_v w \leq q^{t''_2}_{v,*,d} \leq q^{t_1}_{v,*,d} + 2w\delta_v$. It follows that $q^{t''_2}_{v,u,d} \leq \lfloor q^{t''_2}_{v,*,d}/\delta_v \rfloor + 1 \leq \lfloor q^{t_1}_{v,*,d}/\delta_v \rfloor + 2w + 1 \leq q^{t_1}_{v,u,d} + 2w + 1$ since $\lfloor q^{t_1}_{v,*,d}/\delta_v \rfloor \leq q^{t_1}_{v,u,d}$. Similarly, $q^{t_1}_{v,u,d} - w - 1 \leq \lfloor q^{t_1}_{v,*,d}/\delta_v \rfloor - w \leq \lfloor q^{t''_2}_{v,*,d}/\delta_v \rfloor \leq q^{t''_2}_{v,u,d}$ since $q^{t_1}_{v,u,d} \leq \lfloor q^{t_1}_{v,*,d}/\delta_v \rfloor + 1$. □

We now define a potential function Φ on which our proof is based. For each buffer of size q we assume that each packet p in that buffer is assigned a unique *height* $h(p)$ from 1 to q as if the packets are stored one on top of the other. Let the potential of a buffer be the sum of the heights of packets in the buffer. That is,

$$\Phi^t_{v,u,d} = \sum_{p \in Q^t_{v,u,d}} h(p) = \binom{q^t_{v,u,d}+1}{2}.$$

Let $\mathcal{P}^t$ be the set of all packets stored in all nodes at time t. The value of Φ at time t is defined by

$$\Phi^t = \sum_{p \in \mathcal{P}^t} h(p) = \sum_{d \in V} \sum_{e=(v,u) \in E} \left[\Phi^t_{v,u,d} + \Phi^t_{u,v,d} \right].$$

For the purpose of analysis, we assume that the packets that are moved in sub-step 1 of the protocol are the packets with maximum height in the buffer. That is, in sub-step 1 of time step t, if a packet is to move from $Q_{v,u,d}$ to $Q_{u,v,d}$ then the packet which is taken from $Q_{v,u,d}$ is of height $q^t_{v,u,d}$. Moreover, when it is inserted into $Q_{u,v,d}$ it is assigned height $q^t_{u,v,d} + 1$. Thus, the potential decreases by $q^t_{v,u,d} - (q^t_{u,v,d} + 1)$ due to this packet movement.

We observe the following. The potential function can only change upon the following events: (1) the addition of a new injected packet into a buffer (increases the potential function); (2) the

bits per packet; note that $\lceil \log n \rceil$ bits are inherently required to transmit the destination of the packet. For clarity of exposition, we first give our protocol with the assumption that each node knows the sizes of buffers on the other ends of its adjacent edges (and does not use any control bits except for specifying the destination). In Section 3.3 we show how to modify the protocol and proof so as to eliminate this assumption. Also, we note that the protocol need *not* have any topological information about the underlying graph (such information will be needed for the protocol presented in Section 4).

3.1 The Protocol

The protocol maintains several buffers in each node $v \in V$: for each edge $e = (v, u) \in E$ adjacent to v, and for every destination $d \in V$, the protocol maintains a buffer for packets bound for d. Thus, there is a buffer of packets for each triplet (v, u, d), that we denote $Q_{v,u,d}$. Denote the set of packets in $Q_{v,u,d}$ at time t by $Q^t_{v,u,d}$ and by $q^t_{v,u,d}$ the size (i.e., number of packets) of the buffer $Q_{v,u,d}$ at the time t (i.e., $q^t_{v,u,d} = |Q^t_{v,u,d}|$). At each destination d, we consider the value of $q^t_{d,u,d}$, for any $u \in V$ and any time t which is the beginning of a time step, to be always 0 (as packets that arrive to their destinations are immediately removed).

Let $q^t_{v,*,d}$ be the number of packets at node v destined for node d at time t, i.e., the sum of all $q^t_{v,u,d}$ over $(v, u) \in E$. At the end of every time step, each node v will distribute as evenly as possible all the packets destined for d among its edges, for every d. That is, at the beginning of time step $t \geq 2$, $\forall v, d \in V \quad \forall (v, u) \in E$,

$$\lfloor q^t_{v,*,d}/\delta_v \rfloor \leq q^t_{v,u,d} \leq \lceil q^t_{v,*,d}/\delta_v \rceil, \qquad (1)$$

where δ_v is the degree of node v. Since at the beginning of time step 1, when the network is empty, (1) is satisfied, this invariant is maintained by the protocol at the beginning of every time step.

The following protocol is performed by each node $v \in V$, at each time step $t \in \mathbf{N}$.
Protocol BASIC:

1. For each $e = (v, u) \in E$, let $d \in V$ be such that $q^t_{v,u,d} - q^t_{u,v,d}$ is maximal over all $d \in V$ (break ties arbitrarily). If $q^t_{v,u,d} - q^t_{u,v,d}$ is positive then send one packet over the edge e from $Q_{v,u,d}$ to $Q_{u,v,d}$.

2. Accept all packets injected by the adversary to the node v.

3. Remove any packets that arrive at their destination.

4. Redistribute all packets which have arrived in steps 1 and 2 among the buffers so as to maintain invariant (1), breaking ties arbitrarily.

3.2 Analysis

In this section we prove the following theorem.

Theorem 1 : If the sequence of packets is given by an $A(w, \varepsilon)$ adversary, then the number of packets stored at any given time in any of the buffers of BASIC is at most $O(m^{3/2} n^{3/2} w/\varepsilon)$ where $\varepsilon \geq 1/w$ without loss of generality.[2]

[2] Note that since $\varepsilon > 0$ and since an $A(w, \varepsilon)$ adversary is limited to injecting $\lfloor (1 - \varepsilon) w \rfloor$ packets at each window (where the floor is used since the number of packets is obviously an integer), then $\varepsilon \geq 1/w$ does *not* pose any additional constraint.

that guarantees that every packet has polynomial worst-case latency. Our protocol operates in the face of bursty and adversarial injections. Moreover, our protocol is totally distributed and does not need any information on the injection rates.

Organization: In Section 2, we briefly give some definitions including a formal definition of the adversary. In Section 3, we give our basic protocol. This protocol solves the problem in that it requires only polynomial size buffers but it does not guarantee bounded delivery time for the packets. In Section 4 we modify the protocol so as to get bounded delivery time as well. Section 5 briefly discusses several extensions.

2 The Model

We model a communication network by a graph $G = (V, E)$, where $|V| = n$, $|E| = m$. Each node $v \in V$ models a processor, and each edge $e \in E$ models a link between two processors. The processors store and forward packets. Packets at each node are stored in *buffers*. The network is synchronous; we number the time steps, known to all processors, by $t \in \mathbf{N} = \{1, 2, 3, \ldots\}$. We model each edge as bidirectional, i.e., in each time step each edge can deliver one packet in each direction. (See Section 5 for various extensions.)

Each time step is partitioned into three substeps: first, packets are sent between neighbors across edges, at most one packet per edge per time step in each direction; next, each node accepts all new packets injected from the outside; finally each node removes all packets that have reached their destination.

New packets are being injected into the network at each time step. Each packet is injected into an arbitrary source processor s, and has some arbitrary destination processor d that it has to reach. Thus, a packet $p = (s, d)$ is specified by the node into which it is injected and the destination of the packet. No route is given. The sequence of injected packets is controlled by an *adversary*.

Definition 1: We say that the adversary injecting packets is an $A(w, \varepsilon)$ adversary, for some $\varepsilon > 0$ and some integer $w > 1$, if the following holds: For any time $t \in \mathbf{N}$, let $\mathcal{I}^t$ be the set of packets injected during the w time steps from t to $t + w - 1$, inclusive. Then, the adversary can associate with each packet $p = (s, d) \in \mathcal{I}^t$, a simple path from s to d, such that each direction of every edge $e \in E$ is used by these paths at most $\lfloor (1 - \varepsilon)w \rfloor$ times.

3 The Main Protocol

In this section we present our main protocol. This protocol guarantees that the buffers at each node remain bounded for any input sequence given by any $A(w, \varepsilon)$ adversary. That is, the size of the buffers is at most some (polynomial) function of the size of the network, and the parameters w and $1/\varepsilon$ (although the values of w and ε are *not* known to the protocol). In Section 4, we extend this protocol to guarantee that each packet is delivered within a given bounded amount of time. Our protocols are local in nature; that is, decisions are taken in each processor separately, based on the information it has in its node only. This information includes the sizes of its own buffers as well as information gathered from control bits piggybacked on the packets. Our protocol uses $O(\log n)$ control

eral bounds on the buffer sizes of several protocol are derived in [BKR+] using this model.

In [AAF+] the model of [BKR+] is generalized to allow bounded bursts. Now the adversary is characterized by both a rate parameter $\varepsilon > 0$ and a window parameter w such that in every window of time of size w, the number of paths corresponding to packets injected in that window which pass through any edge must be at most $(1-\varepsilon)w$. The maximum queue size of a protocol becomes a function of both the network size and the parameters w and $1/\varepsilon$. [1] Andrews, et al. show that several well-known and simple deterministic greedy queuing protocols yield queues (and latencies) which are bounded albeit exponentially large. They also describe a randomized greedy protocol with polynomial queue size and latency.

OUR RESULTS: We consider the setting of dynamic and adaptive packet routing without probabilistic assumptions on the injections of packets. One of the main open problems left in [AAF+] is to deal with adaptive path-selection, adversarial routing; i.e., the case where an adversary generates the sequence of packets, and specifies for each packet a destination but not a path. In particular, the goals were to use some version of the adversarial injection model appropriate for adaptive path-selection routing, to find a universal protocol that keeps buffers bounded in this model, and to determine whether there is a universal, distributed, and deterministic protocol which requires only polynomial size buffers.

In this paper we resolve these problems. Our model is a natural extension of the "adversarial queuing theory" model [BKR+, AAF+] to the setting of adaptive path selection. In fact, we consider the same (w, ε) adversaries as in the non-adaptive case; that is, the limitation on the sequence of injections is that there exist paths for the packets injected in every consecutive w time steps that do not use any edge more than $(1-\varepsilon)w$ times. The difference is that in the adaptive setting these paths are *not* provided to the protocol with the packets; only the destination is specified. For this model we present a protocol to route the packets, that works for any network and for any sequence of insertions generated by an (w, ε) adversary with any $w > 1$, and any $\varepsilon > 0$. Moreover, our algorithm is distributed and deterministic and it requires buffers of only polynomial size (in the size of the network and the parameters w and $\min(1/\varepsilon, w)$).

OUR TECHNIQUES AND RELATED WORK: Our algorithm is a "diffusion-type" algorithm and as such it is not a greedy algorithm. Packets in an edge buffer are not sent if doing so would take the packet "up hill." Our algorithm uses and extends several previous techniques. In particular it uses "diffusion-type" techniques similar to those used, for example, for communication networks in [AGR, AMS, AAG+], and continuous or discrete load balancing (see [AAMR, GM, GL+, M]). Our protocol is also related to the [AL, AL2] algorithms for constant rate traffic. It is worth pointing out the differences though. The results in [AL, AL2] do not guarantee a bound on the latency of the packets, and indeed packets may get stuck in the network forever. Moreover, the algorithms there require that each node know the insertion rate for each source-target pair. Thus neither the algorithm nor the analysis allow for bursty and adversarial injection. In contrast, we present a protocol

[1] The burst models of [C, C2] and [AAF+] are the same for $\varepsilon > 0$. However, due to the fact that the [C, C2] adversary is limited to virtual circuits, "traffic shaping" can be applied at the source so that the traffic across edges is limited to a maximum rate as in [BKR+].

whether the insertion of packets is static, (i.e., a finite number of packets are inserted once and then routed to completion), or dynamic (i.e., packets arrive continuously and are routed continuously).

NON-ADAPTIVE VS. ADAPTIVE PATH SELECTION: In non-adaptive path selection routing, the entire path of each packet is determined before it crosses its first edge. Thus, any contention resolution protocol at the nodes cannot influence the packets path. This allows the routing algorithm to be divided into two separate steps: selection of paths for packets and contention resolution at edges congested by the selected paths [LMR]. Separating the path selection from the contention-resolution nicely models virtual circuit routing or (randomized) oblivious routing, among others. Moreover, it is extremely helpful from the standpoint of algorithmic design and analysis. Indeed, using this approach there is a large body of work on path selection and contention resolution for minimizing queue size and latency. See, for example, [ST, AAP, KT, KPP] for path selection and [AAF+, BFU, LMR, LMRi, PG, PG2, RT, OR] for packet-scheduling.

The non-adaptive approach, however, does not allow packets to dynamically adapt to congestion and faults along their routes. In adaptive path routing, the contention resolution protocol of an intermediate node may "reroute" a packet in an attempt to achieve smaller packet latency or higher network throughput. For example, adaptive routing algorithms for general networks appear in work on end-to-end communication [AMS, AGR, AAG+], and in work on multicommodity flow [AL, AL2]. There, the packets choose where to go depending on the local traffic conditions.

STATIC VS. DYNAMIC ROUTING: Until recently, routing algorithms which afforded worst-case analysis on latency and queue size were limited to the static routing problem. There is a long history of work in this area. See, for example, [L, LMR, LMRi, RT, OR]. In the case of dynamic routing, one can still employ algorithms for the static case by using synchronization barriers between injections, but this is not an efficient use of network resources and better dynamic routing algorithms have been considered.

Obviously, some assumptions regarding the traffic are necessary since if the traffic is constantly larger than the capacity of the edges then clearly one needs unbounded buffers. One way to view continuous injection of packets is by using a probabilistic model such as assuming Poisson arrivals at each node with destinations chosen uniformly and independently. In such cases the performance is measured in terms of the *expected* latency and queue size. See, for example, [BU, BFU, HB, HW, SV, STs].

The first attempt to develop a model of dynamic routing for analyzing the queue size and latency in the worst-case was developed by Cruz [C, C2]. In his model, one assumes arbitrary virtual circuits are established each with a source of fixed rate (with bounded bursts allowed) subject to the constraint that the total rate of all sessions using a given edge is strictly less than one. Borodin et al. [BKR+] introduce "adversarial queuing theory" which models general non-adaptive path routing. In their model, packets are not restricted to virtual circuits—each packet is given an arbitrary path at injection by an adversary. The adversary is characterized by a rate constraint, $\varepsilon > 0$: in every window of time of length t, the number of paths from packets injected in that window which pass through any edge must be strictly less than $\lceil (1-\varepsilon)t \rceil$. Sev-

Adaptive Packet Routing for Bursty Adversarial Traffic

William Aiello* Eyal Kushilevitz† Rafail Ostrovsky‡ Adi Rosén§

Abstract

One of the central tasks of networking is *packet-routing* when edge bandwidth is limited. Tremendous progress has been achieved by separating the issue of routing into two conceptual sub-problems: *path selection* and *congestion resolution* along the selected paths. However, this conceptual separation has a serious drawback: each packet's path is *fixed* at the source and cannot be modified adaptively en-route when traffic conditions change. The problem is especially severe when packet insertions are modeled by an *adversary*, whose goal is to cause "traffic-jams".

In this paper, we consider this adversarial setting, motivated by the "adversarial queuing theory model" of Borodin et al. [BKR+]. More precisely, we consider an adversary who injects packets, with only their destinations specified, into network nodes in a continuous manner subject to the limitation that–roughly speaking–the injection sequence does not inherently overload the network. The question whether it is possible to deal with such an adversary and to design protocols that would "discover" routes which avoid "traffic jams" so that nodes only needed to buffer a bounded number of packets, was left as an open problem in the paper of Andrews et al. [AAF+] (which deals with the case where the adversary does provide routes for the packets). In the present paper, we resolve this open problem. In particular, we present a simple, deterministic, local-control protocol that applies to any network topology. Our protocol guarantees that, for any injection sequence generated by the adversary, the buffers at the nodes are polynomially-bounded and that each packet has a polynomially-bounded delivery time.

1 Introduction

Packet routing is one of the central issues in the areas of parallel computing and networking (see the survey of Leighton [L]). In this paper, we consider packet routing in the setting of arbitrary synchronous networks. The study of packet routing in this setting can be categorized along two different axes: (1) whether path selection is non-adaptive (i.e., each packet's path through the network is fixed at its source) or adaptive (i.e., each packet's path through the network can be modified en-route); and (2)

*Bellcore. e-mail:aiello@bellcore.com

†Dept. of Computer Science, Technion, Haifa, Israel. e-mail: eyalk@cs.technion.ac.il; Part of this research was done while visiting DIMACS. URL:http//www.cs.technion.ac.il/~eyalk

‡Bellcore. e-mail:rafail@bellcore.com

§Dept. of Computer Science, University of Toronto, Toronto, Canada. Part of this research was done while visiting DIMACS. e-mail: adiro@cs.toronto.edu