Randomized Smoothing Networks ⋆

Maurice Herlihy

Computer Science Dept., Brown University, Providence, RI, USA

Srikanta Tirthapura *

Dept. of Electrical and Computer Engg., Iowa State University, Ames, IA, USA

⋆ A preliminary version of this article has appeared in the Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS) 2004

* Corresponding Author

Email addresses: mph@cs.brown.edu (Maurice Herlihy), snt@iastate.edu (Srikanta Tirthapura).

1 Introduction

A *k-smoothing network* is a distributed data structure that accepts tokens on input wires and routes them to output wires. It ensures that no matter how imbalanced the traffic on the input wires, the numbers of tokens emitted on the output wires are approximately balanced, lying within k of one another, where k is a constant, independent of the number of active tokens. Smoothing networks are well suited for load balancing applications where tokens represent requests for service. Clients send tokens to arbitrary input wires; these tokens are routed to servers in such a way that all servers receive approximately the same number of tokens.

In a real distributed system, network switches may be rebooted or replaced dynamically, and it may not be feasible to determine the correct initial state for each switch. Hence, an attractive approach to fault-recovery is simply to initialize the new switch to a random state, thus eliminating the need for a global coordination. In this paper, we analyze the smoothness properties of *randomized balancing networks*, whose constituent balancers are initialized to random states. Randomized networks are easy to maintain and can recover from faults without a global reconfiguration.

We show that randomized networks produce outputs that are remarkably smooth. We consider the *block* smoothing network initialized to a random state; we assume that an off-line adversary, one that does not know the initial state, chooses an input sequence. We show that the output of the network is $2.36\sqrt{\log w}$-smooth with high probability. As a direct consequence, we show that the output smoothness of a randomly initialized *bitonic* or a *periodic* network is also $O\left(\sqrt{\log w}\right)$ with high probability.

In prior work [7] we showed that certain well-known 1-smoothing networks, when started in an arbitrary initial state (perhaps chosen by an adversary), produce outputs that are at worst $(\log w)$-smooth, where w is the number of input and output wires. This bound is tight for each of the networks that we considered there. Thus, our results show that the outputs of randomized networks are significantly smoother than the worst-case outputs of the same networks initialized deterministically.

Our results lead to extremely practical smoothing networks. If an application requires approximate smoothing, then a block network with randomized balancers will work very well. The block network of width w has depth exactly $\log w$ (there are no hidden constants), and the smoothness bound grows very slowly with w. For example, if the width $w = 2^{24}$, then the output smoothness is less than 12 with overwhelming probability.

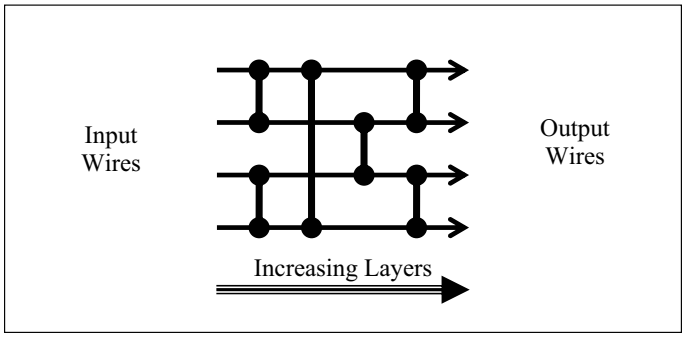

Fig. 1. A balancing network of width 4 and depth 3. Horizontal segments are wires and the vertical segments balancers.

We conclude this article with a brief discussion of informal but suggestive experimental results. By feeding random sequences to randomized smoothing networks, we observed a degree of smoothness worse than constant, and consistent with our bound. On random input sequences, we discovered that randomized networks were dramatically smoother than the same networks initialized to the default initial state.

1.1 Model

1.1.1 Balancers and Smoothing Networks

A *balancer* is an asynchronous switch with two input wires and two output wires, labeled "top" and "bottom". A balancer accepts a stream of tokens on its input wires. A balancer has two states: it is oriented *up* or *down*. If the balancer is oriented *up*, then the next input token leaves on the top output wire and the balancer becomes oriented *down*. If the balancer is oriented *down*, then the next input token leaves on the bottom output wire and the balancer becomes oriented *up*. The above definition applies to balancers of width two. All the balancers that we consider in this paper have a width of two.

A *balancing network* is an acyclic network of balancers where output wires of some balancers are linked to input wires of others. An example is shown in Figure 1. The network's *input wires* are not linked to the output of any balancer, and similarly the network's *output wires* are not linked to the input of any balancer. In this paper, we consider balancing networks with the same number of input and output wires, called the network's *width*. Tokens enter the network on the input wires, typically several per wire, propagate asynchronously through the balancers, and leave on the output wires, typically several per wire. A balancing network is *quiescent* if every token that has entered the network has also left. Because balancing networks are acyclic (as directed graphs), each balancer can be assigned a unique *layer*, which is the length of the longest path from an input wire to that balancer.

Definition 1 *A sequence of natural numbers* $X = x_0, \ldots, x_{n-1}$ *is* k-smooth *if* $|x_i - x_j| \leq k$, *for any* $0 \leq i, j < n$.

Definition 2 *A balancing network is a* k-smoothing network *if, starting from its initial state, the overall distribution of output tokens across the output wires in any quiescent state is* k*-smooth.*

Note that if a network is k-smooth, it is also ℓ-smooth for any $\ell > k$. In this paper, we will be concerned with the block network [6] (which is isomorphic to the popular butterfly network), the periodic network [6,3] and the bitonic network [4,3].

We now prove a simple but useful lemma.

Lemma 3 *If a sequence* $X = x_0, \ldots, x_{n-1}$ *is* k*-smooth, then the result of passing* X *through a balancing network is* k*-smooth.*

PROOF. We show that the result of passing any two elements of X through a balancer leads to a sequence that is k-smooth, and the lemma follows by induction. Let Z be the result of passing two elements of X through a balancer. The smallest element in Z is greater than or equal to the smallest element in X, and the largest element in Z is lesser than or equal to the largest element in X. Since X is k-smooth, Z is also k-smooth. □

Lemma 3 leads to the following corollary.

Corollary 4 *If a balancing network* $\mathbf{N}$ *contains another network* $\mathbf{M}$ *embedded inside it, and* $\mathbf{M}$ *is a* k*-smoothing network, then* $\mathbf{N}$ *is also a* k*-smoothing network.*

1.1.2 Randomized Balancers and Networks

Definition 5 *A* randomized balancer *is a balancer which has been initialized to a random initial state, i.e.* up *or* down *with equal probability.*

Definition 6 *A* randomized balancing network *is a balancing network whose component balancers are independent randomized balancers.*

Let x denote the total number of input tokens to a randomized balancer b, and y_1, y_2 denote the number of tokens routed to the top and bottom output wires respectively. Let $x_b = D(x)r_b$, where:

- r_b is a random variable specific to balancer b, which can take values $+1/2$ or $-1/2$ with equal probability.

- D is the odd-characteristic function: $D(x) = 1$ if x is an odd number, and 0 otherwise.

By the definition of the randomized balancer:

$$\begin{aligned} y_1 &= x/2 + x_b \\ y_2 &= x/2 - x_b \end{aligned} \tag{1}$$

Note that if x is even, then the input tokens are divided equally among the output wires. The randomness comes into play only when x is odd. In such a case, the excess token is routed to a randomly chosen output wire.

1.2 Our Results

We ask the following question. *What are the smoothness properties of randomized balancing networks?* We show the following results in response.

- The output of a randomized block network of width w is $(2.36\sqrt{\log w})$-smooth with probability at least $1 - 4/w$.

- We show that the bitonic network contains a block network embedded inside it. The periodic network consists of many pipelined block networks, hence trivially contains a block network embedded inside. Hence, the outputs of periodic and bitonic networks of width w are also $(2.36\sqrt{\log w})$-smooth with probability at least $1 - 4/w$.

1.3 Related Work

Aiello, Venkatesan and Yung [1] build smoothing networks using a different kind of randomized balancer: odd-numbered input tokens to the balancer leave on a randomly chosen wire, while even-numbered input tokens leave on the opposite wire from their immediate predecessors. It can be easily verified that our result for the smoothness of the block network still holds if we replace our random balancers with theirs. Using a combination of deterministic and randomized balancers, Aiello *et al.* [1] construct smoothing networks whose outputs are 2-smooth with high probability. In our model, however, an adversary could set the orientations of the deterministic balancers, and their analysis does not hold under these conditions.

Klugerman and Plaxton [10] show the existence of counting networks (which are 1-smoothing networks with other additional properties) of depth $O(\log w)$

Analysis of width w network	Depth	Balancers	Global Initialization	Smoothness
Klugerman and Plaxton [10,9]	$O(\log w)$	Det.	Required	1
Aiello et al. [1]	$O(\log w)$	Det. and Rand.	Required	2 (w.h.p.)
Herlihy et al. [7]	$\log w$	Det.	Not required	$\log w$
This paper	$\log w$	Rand.	Not required	$2.36\sqrt{\log w}$ (w.h.p.)

Fig. 2. Comparison of various analyses of smoothing networks. Note: w.h.p. = "with high probability", det. = "deterministic" and rand. = "randomized".

which use deterministic balancers. They also give an explicit construction of a counting network of depth $O(c^{\log^* w} \log w)$ (where c is a constant) using deterministic balancers. Klugerman [9] extends this to give a polynomial time construction of an $O(\log w)$ depth counting network of width w. The above networks use the AKS network [2] as a subnetwork, and are hence not practical since the constants in $O(\log w)$ are huge. For a survey of work on low-depth counting networks and related topics, we refer the reader to Busch and Herlihy [5].

In earlier work [7], we showed that the worst case output smoothness of the block, periodic and bitonic networks of width w is *exactly* $\log w$ when the switches are initialized adversarially. Our results in this paper show that randomization helps significantly to reduce the output smoothness of the block network. A comparison of various analyses of smoothing networks appears in Figure 2.

Roadmap: The rest of the paper is organized as follows. Section 2 contains the analysis of the block network. Section 3 contains the analysis of the bitonic network. Section 4 contains the experimental results, and we conclude by listing some open problems in Section 5.

2 The Periodic and Block Networks

The periodic smoothing network [3] is isomorphic to the periodic sorting network of Dowd *et al.* [6]. At its heart is a component Block$[w]$ network, defined inductively as follows.

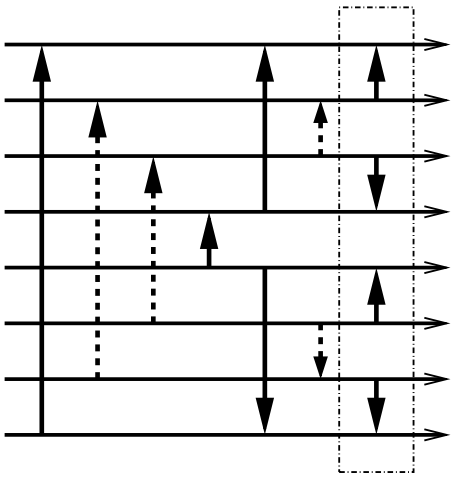

Fig. 3. A BLOCK[8] Network consists of two BLOCK[4] networks: one with balancers denoted by solid lines, and the other with balancers denoted by dashed lines. The outputs of the BLOCK[4] networks are fed into an EVENODD[8] network, shown inside the box.

The Block Network: The BLOCK[2] network is a single balancer. The BLOCK[$2w$] network is constructed from two BLOCK[w] networks as follows. Given an input sequence X of length $2w$, represent each index (subscript) as a binary string. The *A-cochain* of X, denoted X^A, is the subsequence whose indexes have low-order bits 00 or 11. For example, the *A-cochain* of the sequence $x_0, \ldots, x_7$ is x_0, x_3, x_4, x_7. The *B-cochain* x^B is the subsequence whose low-order bits are 01 and 10.

The input sequence X is fed into two parallel BLOCK[w] networks, which we will call the *A-block* and the *B-block*. Sequence X^A goes to the A-block, and X^B to the B-block. Suppose the output sequence of the A-block is $Y^A = y_0^A, y_1^A, \ldots, y_{w-1}^A$ and the output sequence of the B-block is $Y^B = y_0^B, y_1^B, \ldots, y_{w-1}^B$. These are fed into an EVENODD[$2w$] network, which simply balances each element of Y^A with the corresponding element of Y^B. More formally, EVENODD[$2w$] consists of w balancers numbered 0 to $w-1$, and the ith balancer balances y_i^A with y_i^B. Figure 3 shows a BLOCK[8] network.

The BLOCK[w] network consists of $\log w$ layers of balancers, numbered from 1 (input layer) to $\log w$ (output layer). The output wires of a balancer in layer ℓ belong to layer ℓ. The input wires to the network belong to layer 0. Each layer consists of $w/2$ balancers, so that the network has a total of $(w \log w)/2$ balancers.

The Periodic Network: The PERIODIC[w] network is the cascade of $(\log w)$ BLOCK[w] networks. Since the PERIODIC[w] network trivially contains a BLOCK[w] embedded inside it, because of Corollary 4, our upper bounds for the block network directly carry over to the periodic network.

2.1 *Smoothness of the Randomized* BLOCK[w] *Network*

Consider an execution of the randomized smoothing network, taking it from a random initial state to a quiescent state, where a total of M tokens have entered and left the network.

We consider BLOCK[w] as a tree. Each node in this tree is a set of one or more balancers, as described below.

- The root of the tree is the set of all the $w/2$ balancers at the input layer of the network (layer 1). This node is labeled $v_{1,1}$, since it is the first node in layer 1.

- For each $i = 2 \ldots \log w$, the ith layer is divided into 2^{i-1} nodes, numbered from $v_{i,1}$ till $v_{i,2^{i-1}}$, with each node consisting of $w/2^i$ balancers. Given the nodes in the $(i-1)$th layer, the nodes in the ith layer are defined as follows.

  The node $v_{i,2k-1}$ consists of all the balancers that the top output wires of the balancers in node $v_{i-1,k}$ point to. Similarly, the node $v_{i,2k}$ consists of all the balancers that the bottom output wires of the balancers in node $v_{i-1,k}$ point to. In the tree, there is an edge from node $v_{i-1,k}$ to nodes $v_{i,2k-1}$ and $v_{i,2k}$.

- The leaves of the tree are the balancers at the output layer.

Let $m_{i,j}$ denote the total number of tokens entering node $v_{i,j}$. Since the tokens on the top output wires of the balancers in $v_{i,j}$ enter $v_{i+1,2j-1}$, the number of tokens entering node $v_{i+1,2j-1}$ is (using Equation 1):

$$m_{i+1,2j-1} = \frac{m_{i,j}}{2} + \sum_{b \in v_{i,j}} x_b \tag{2}$$

We will now express the number of tokens that are output at the top output wire of $v_{\log w,1}$ as a function of the number of input tokens and the random variables corresponding to the different balancers. The tokens that exit from the topmost output wire must follow the path $v_{1,1} \rightarrow v_{2,1} \rightarrow v_{3,1} \rightarrow \ldots \rightarrow v_{\log w,1}$ and further exit on the top output wire of balancer $v_{\log w,1}$.

The number of tokens entering $v_{1,1}$ is M. Let X_1 denote the number of tokens exiting the top output wire of $v_{\log w,1}$, which is a single output balancer. Applying Equation 2 repeatedly, and finally Equation 1, we get:

$$
\begin{aligned}
m_{2,1} &= \tfrac{M}{2} + \textstyle\sum_{b \in v_{1,1}} x_b \\
m_{3,1} &= \tfrac{m_{2,1}}{2} + \textstyle\sum_{b \in v_{2,1}} x_b \\
&\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
X_1 &= \tfrac{m_{\log w,1}}{2} + \textstyle\sum_{b \in v_{\log w,1}} x_b
\end{aligned}
$$

Combining the above, we get:

$$X_1 = \frac{M}{w} + \frac{\sum_{b \in v_{1,1}} x_b}{w/2} + \frac{\sum_{b \in v_{2,1}} x_b}{w/4} + \ldots + \frac{\sum_{b \in v_{\log w - 1,1}} x_b}{2} + \sum_{b \in v_{\log w,1}} x_b$$

Rewriting the above: $X_1 = M/w + V$ where

$$V = \sum_{i=1}^{\log w} \sum_{b \in v_{i,1}} \frac{2^i x_b}{w} \tag{3}$$

We will now analyze V. The main problem is that V is not the sum of independent random variables. Clearly the various random variables x_b (for different balancers b) are heavily dependent on each other, since the event that the number of tokens entering balancer b is even (resp., odd) depends upon the corresponding events for balancers at earlier layers.

Recall that $x_b = D(n_b)r_b$. We consider an alternate random variable W, defined as follows.

$$W = \sum_{i=1}^{\log w} \sum_{b \in v_{i,1}} \frac{2^i r_b}{w} \tag{4}$$

Since the random variables r_b for different balancers b are independent, W is easier to handle than V.

We rewrite Equations 3 and 4 as follows:

$$V = \sum_{\{(b \in S) \wedge (D(n_b)=1)\}} c_b r_b \tag{5}$$

$$W = \sum_{\{b \in S\}} c_b r_b \tag{6}$$

S is a set of balancers, c_b denotes a constant specific to balancer b, and r_b is the random variable corresponding to balancer b. The set S and the c_bs can

be derived from Equations 3 and 4, but their exact nature is not important here.

We now prove a key lemma, showing that in order to bound the probability that V deviates significantly from its expected value, it is enough to bound the probability that W deviates from its expected value.

Lemma 7 *For any $\delta > 0$, $\mathbf{Pr}\{|V| > \delta\} \leq 2\mathbf{Pr}\{|W| > \delta\}$.*

PROOF.

Consider the conditional probability

$$\alpha = \mathbf{Pr}\{W > \delta | V > \delta\}$$

From Equations 5 and 6,

$$\mathbf{Pr}\{W \leq \delta | V > \delta\} \leq \mathbf{Pr}\left\{\sum_{\{(b \in S) \wedge (D(n_b) \neq 1)\}} c_b r_b < 0\right\} \leq 1/2$$

The right hand side inequality ($\leq 1/2$) is true because: for each balancer b, the random variable r_b is distributed symmetrically around zero, and is chosen independent of n_b. Thus, we have $\alpha \geq 1/2$.

$$\begin{aligned}\mathbf{Pr}\{W > \delta\} &\geq \mathbf{Pr}\{(W > \delta) \wedge (V > \delta)\} \\ &\geq \mathbf{Pr}\{W > \delta | V > \delta\} \cdot \mathbf{Pr}\{V > \delta\} \\ &\geq \mathbf{Pr}\{V > \delta\}/2\end{aligned}$$

Similarly, we can show the other direction, that $\mathbf{Pr}\{V < -\delta\} \leq 2\mathbf{Pr}\{W < -\delta\}$, and the claim follows. □

We will use the following lemma in further proofs.

Lemma 8 *[Hoeffding's bound [8]]*
Suppose $S_n = Y_0 + Y_1 + \ldots Y_{n-1}$ where the Y_j's are independent random variables, and for each $j = 0 \ldots n-1$, $Y_j \in [a_j, b_j]$. Then, for any $\epsilon > 0$,

$$\mathbf{Pr}\{S_n - E[S_n] > n\epsilon\} \leq \exp\left\{\frac{-2n^2\epsilon^2}{\sum_{j=0}^{n-1}(b_j - a_j)^2}\right\}$$

Lemma 9

$$\mathbf{Pr}\left\{|V| > 1.177\sqrt{\log w}\right\} < \frac{4}{w^2}$$

PROOF. We will first bound the probability that $|W|$ is large, and then use Lemma 7 to bound the probability that $|V|$ is large.

Since W is the sum of independent random variables, we use Hoeffding's inequality (stated in Lemma 8) to bound the probability that W is large. For $i = 1 \ldots \log w$, the number of balancers in node $v_{i,1}$ is $w/2^i$. Thus, W is the sum of $(w/2 + w/4 + \ldots + 1) = w - 1$ independent random variables.

The expectation of W: $E[W] = 0$, since $E[r_b] = 0$ for every balancer b.

Next, the ranges of various random variables. For $i = 1 \ldots \log w$,

$$\frac{2^i r_b}{w} \in \left[\frac{-2^{i-1}}{w}, \frac{+2^{i-1}}{w}\right]$$

In Lemma 8, the term $\sum (b_j - a_j)^2$ is:

$$\frac{w}{2}\left(\frac{2}{w}\right)^2 + \frac{w}{4}\left(\frac{4}{w}\right)^2 + \ldots + 1 = 2 - \frac{2}{w}$$

Setting $\epsilon = \frac{\sqrt{2 \ln w}}{w-1}$ in Lemma 8, we get

$$\mathbf{Pr}\left\{W > \sqrt{2 \ln w}\right\} \leq \exp\left\{-2 \ln w\right\} = \frac{1}{w^2} \tag{7}$$

Since W is the weighted sum of r_bs, each of which is symmetrically distributed around zero, W is also symmetrically distributed around zero. Hence

$$\mathbf{Pr}\left\{W < -\sqrt{2 \ln w}\right\} = \mathbf{Pr}\left\{W > \sqrt{2 \ln w}\right\} \leq \frac{1}{w^2} \tag{8}$$

Equations 7 and 8 combined with Lemma 7 yield the claim. Note that we have stated our lemma using logarithms to base 2. □

Theorem 10 *The output sequence of* Block$[w]$ *with randomized balancers is* $2.36\sqrt{\log w}$*-smooth with probability at least* $1 - 4/w$.

PROOF. Let $X_1, X_2, \ldots, X_w$ denote the output sequence of Block$[w]$ with randomized balancers, where X_1 corresponds to the topmost output wire.

From Lemma 9, we have

$$\mathbf{Pr}\left\{|X_1 - M/w| > 1.177\sqrt{\log w}\right\} \leq 4/w^2$$

The random variables $X_1 \ldots X_w$ all have the same distribution, due to symmetry, and a similar equation holds for all of them. Using the union bound on probabilities, the probability that for some i the value $|X_i - M/w|$ exceeds $1.177\sqrt{\log w}$ is not greater than $4/w$. Thus, with probability at least $1 - 4/w$, all outputs X_i are within $1.177\sqrt{\log w}$ of M/w, and hence within $2.36\sqrt{\log w}$ of each other, as needed. □

3 The Bitonic Network

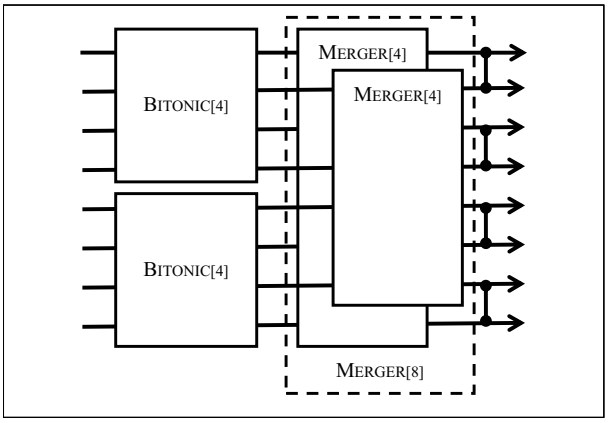

Fig. 4. Recursive Structure of a BITONIC[8] Counting Network.

The BITONIC[w] smoothing network [3] is isomorphic to the *bitonic* sorting network of Batcher [4]. This network has a simple inductive structure, shown in Figure 4. The BITONIC[2] network is a single balancer. The BITONIC[$2w$] network is constructed by feeding the $2w$ input wires into two parallel BITONIC[w] networks, and feeding their outputs into a MERGER[$2w$] network.

The MERGER[$2w$] network takes two input w-sequences X and Y and produces an output $2w$-sequence Z. The MERGER[w] network is also defined inductively. The MERGER[2] network is a single balancer. We construct the MERGER[$2w$] network from two MERGER[w] networks and a EVENODD[$2w$] network which was described in Section 2. Let X^E denote the even subsequence $x_0, x_2, \ldots, x_{w-2}$ of X, and X^O denote the odd subsequence $x_1, x_3, \ldots, x_{w-1}$. Similarly define Y^E and Y^O.

The input to the first MERGER[w] network is the w-sequence formed by the concatenation of X^E and Y^O, denoted by $X^E \cdot Y^O$. Call that network's output

sequence U. Symmetrically, the input to the second MERGER$[w]$ network is the w-sequence $X^O \cdot Y^E$. Call that output sequence V. The final layer of the network, EVENODD$[w]$, simply joins the each output wire of the first MERGER$[w]$ network with the corresponding output wire of the second MERGER$[w]$ network.

Two networks **N** and **M** are *isomorphic* if the underlying directed graphs are isomorphic.

Lemma 11 *The* MERGER$[w]$ *network is isomorphic to the* BLOCK$[w]$ *network.*

PROOF. We argue by induction on w. When w is 2, both networks consist of a single balancer. Assume that MERGER$[w]$ is isomorphic to BLOCK$[w]$, and consider the MERGER$[2w]$ and BLOCK$[2w]$ networks.

In the BLOCK$[2w]$ network, the final layer connects the i-th wire of one component BLOCK$[w]$ network with the i-th wire of the other. Likewise, in the MERGER$[2w]$ network, the final layer connects the i-th wire of one component MERGER$[w]$ network with the i-th wire of the other, preserving the isomorphism. □

Lemma 11, Theorem 10 and Corollary 4 together lead to the following corollary.

Corollary 12 *The* MERGER$[w]$ *network, and hence, the* BITONIC$[w]$ *network are* $2.36\sqrt{\log w}$*-smooth with probability at least* $1 - 4/w$.

4 Experiments

To develop an intuition about whether our bound can be improved, we conducted a number of simple experiments testing the behavior of randomized block networks of different sizes. These results do not prove anything, but they are suggestive. The results are shown in Figure 5.

In our experiments, the number of tokens input into each wire is randomly chosen between 1 and 100000, and the output smoothness is averaged over 10 runs. We simulated networks of width up to 2^{24}. The results show that when $\log w$ changes from 1 to 24, the (average) output smoothness increases very gradually, from 0.7 to 3.2. We do not expect to be able to simulate much larger networks.

To compare with the randomized network, we simulated a block network initialized in the standard state; all balancers were initialized to *up*. Using the same setup as for the random network, (the number of tokens into each wire is a random number between 1 and 100000, average taken over 10 runs), the results obtained are shown below. It can be seen that the output smoothness is very nearly equal to $\frac{\log w}{2}$.

While we know that the worst case smoothness of a block network initialized by an adversary is $\log w$, these experiments suggest that the smoothness of the network over random inputs when initialized very regularly (not by an adversary) is no better than $\log w/2$. In other words, these experiments suggest that the average smoothness of the block network initialized to the standard state is $O(\log w)$, while we have shown that the smoothness of a block network initialized randomly is $O(\sqrt{\log w})$.

We conclude that for applications needing approximate load balancing, a randomized block network works much better than a deterministic one.

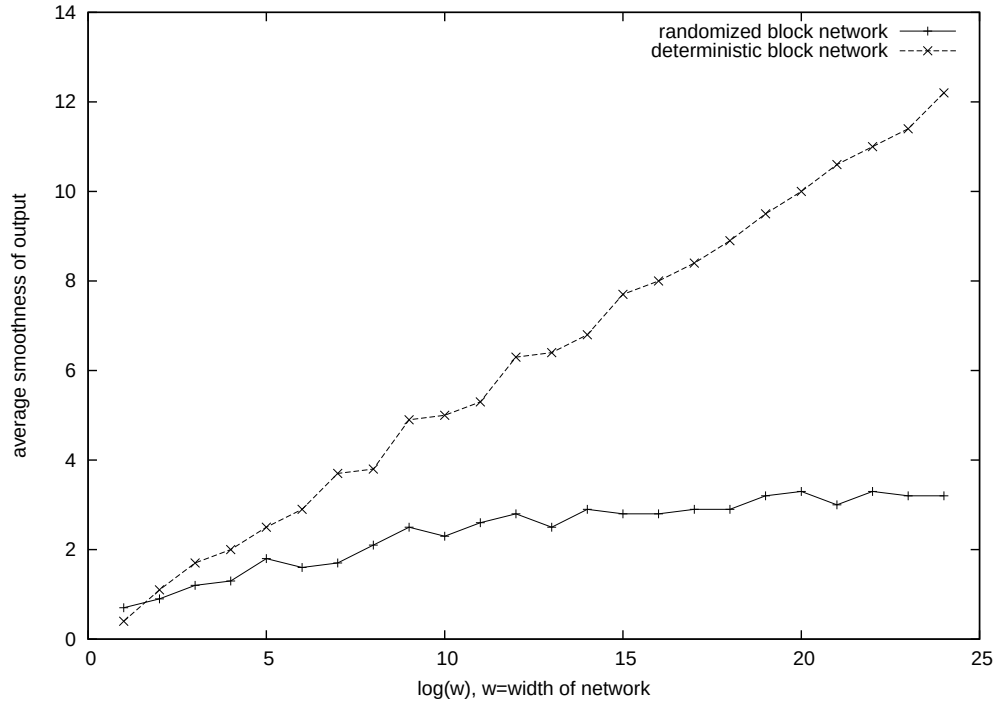

Fig. 5. The observed smoothness of a Block$[w]$ network with randomized and deterministic balancers

5 Open Problems

We conclude by listing some interesting open problems.

(1) Our bounds for the smoothness of the block network does not make use of structure that may be present in the input sequence. Can we obtain better bounds if the input is already fairly smooth?

(2) Can we get better bounds on the output smoothness of the randomized periodic or bitonic networks?

(3) How tight is the $O(\sqrt{\log w})$ upper bound for the block network? Can we get a matching lower bound?

References

[1] W. Aiello, R. Venkatesan, and M. Yung. Coins, weights and contention in balancing networks. In *Proceedings of the annual ACM symposium on Principles of Distributed Computing*, pages 193–205, August 1994.

[2] M. Ajtai, J. Komlós, and E. Szemerédi. Sorting in $c \log n$ parallel steps. *Combinatorica*, 3:1–19, 1983.

[3] J. Aspnes, M. Herlihy, and N. Shavit. Counting networks. *Journal of the ACM*, 41(5):1020–1048, 1994.

[4] K.E. Batcher. Sorting networks and their applications. In *Proceedings of the AFIPS Spring Joint Computer Conference*, volume 32, pages 307–314, 1968.

[5] C. Busch and M. Herlihy. A survey on counting networks. In *Proceedings of Workshop on Distributed Data and Structures (WDAS)*, March 1998.

[6] M. Dowd, Y. Perl, L. Rudolph, and M. Saks. The periodic balanced sorting network. *Journal of the ACM*, 36(4):738–757, October 1989.

[7] M. Herlihy and S. Tirthapura. Self stabilizing smoothing and counting. In *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS)*, pages 4–11, 2003.

[8] W. Hoeffding. Probability inequalities for sums of bounded random variables. *American Statistical Association Journal*, 58:13–30, 1963.

[9] M. Klugerman. *Small-depth Counting Networks and Related Topics*. PhD thesis, Massachusetts Institute of Technology, Department of Mathematics, Sept 1994.

[10] M. Klugerman and C.G. Plaxton. Small-depth counting networks. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*, pages 417–428, 1992.