An Efficient Counting Network*

Costas Busch
Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180
buschc@cs.rpi.edu

Marios Mavronicolas†
Department of Computer Science
University of Cyprus
Nicosia, CY-1678 Cyprus
mavronic@ucy.ac.cy

Abstract

We present a novel *counting network* construction, where the number of input wires w is smaller than or equal to the number of output wires t. The *depth* of our network is $\Theta(\lg^2 w)$, which depends only on w. In contrast, the *amortized contention* of the network depends on the number of concurrent processes n and the parameters w and t. This offers more flexibility than all previously known networks, with the same number w of input and output wires, whose contention depends only on two parameters, w and n. As a result, when choosing $n, t \geq w \lg w$, the contention of our network is $O(n \lg w/w)$, which improves by a logarithmic factor of w over all previously known networks with w wires.

1 Introduction

1.1 Background

A fundamental problem in distributed computing is the efficient implementation of a *shared counter*. In the shared counter problem, the distributed processes access the counter through Fetch&Increment operations in order to obtain successive integer values from a given range. Distributed problems such as load balancing and barrier synchronization can be expressed and solved as counting problems. In a seminal work, Aspnes, Herlihy and Shavit [5] introduced counting networks as a class of distributed data structures used to construct concurrent, low-contention implementations of distributed counters that support the Fetch&Increment operation.[1]

*A preliminary version of this work appears in the *Proceedings of the 1st Merged International Parallel Processing Symposium and Symposium on Parallel and Distributed Processing (IPPS/SPDP'98),* pp. 380–384, Orlando, Florida, March/April 1998.

†Supported by funds for the promotion of research at University of Cyprus.

[1]See [2] for an extention to counting networks which support also Fetch&Decrement operations.

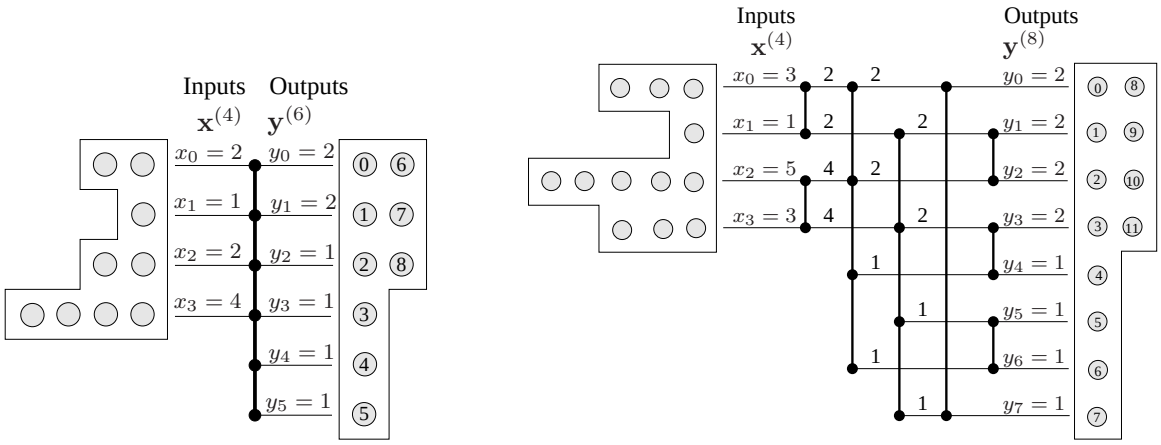


Figure 1: Left: a $(4, 6)$-balancer; Right: a balancing network of input width 4 and output width 8

Counting networks are constructed from p-input q-output asynchronous *switches* called (p, q)-*balancers* [1, 5, 10, 11]; p is the balancer's *input width,* while q is its *output width.* As illustrated in Figure 1, a balancer accepts a stream of tokens on its p input wires. The tokens arrive asynchronously to the balancer and the balancer processes a token at a time in an atomic operation so that the i-th token to be processed by the balancer leaves on output wire $i \bmod q$ (where $i = 0, 1, \ldots$). In the same figure, we show the number of tokens that enter on each input wire and leave from each output wire. A balancer for which $p = q$ will be called *regular*, while a balancer for which $p \neq q$ will be called *irregular.*

A *balancing network* [5], denoted $\mathcal{B}$, is an acyclic network of balancers, where the output wires of the balancers are linked to input wires of others; see Figure 1 for an illustration. The network's *input wires* are those input wires not linked to any balancer's output, and similarly for the network's *output wires.* The number of input wires is called the network's *input width,* denoted w; the number of output wires is called the network's *output width,* denoted t.

A balancing network in which each balancer is regular will be called a *regular* network. In a regular network it holds that the input width is equal to the output width, that is, $w = t$. If a network uses irregular balancers, it will called *irregular.* Note that in irregular networks it may be that $w \neq t$. For example the network in Figure 1, is irregular. Examples of regular balancing networks are shown in Figure 2.

Tokens enter the balancing network on the input wires, typically several per wire, propagate asynchronously through the balancers, and leave on the output wires, typically several per wire. The *depth* of the balancing network is the maximum number of balancers that any token has to traverse from an input wire to an output wire. The depth of a balancing network determines its *latency* which is a delay due to the physical characteristics of a balancing network. A significant source of delay are token collisions (contention) which we discuss below.

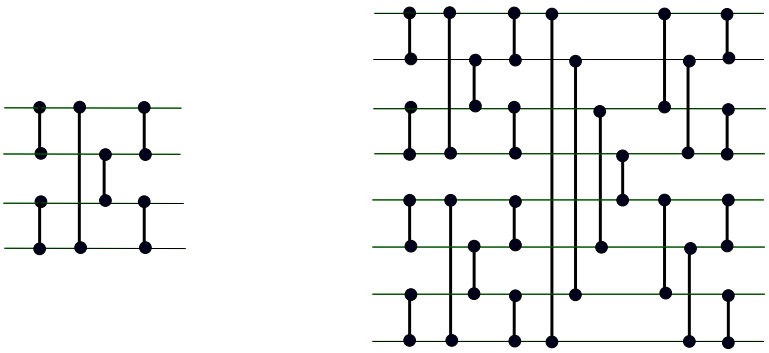


Figure 2: Regular balancing networks of width 4 and 8 built from $(2, 2)$-balancers

A balancing network is a *counting network* [5] if the overall distribution of output tokens across the output wires satisfies the *step property*: exiting tokens are divided uniformly among the output wires, while any excess tokens emerge on the upper wires. The balancing networks illustrated in Figures 1, 2, and 3 are all counting networks. In Figure 1, we show a particular distribution of tokens on the input and output wires; note that the step property holds on the distribution of tokens across the output wires.

The primary purpose of a counting network is to support distributed Fetch&Increment operations. Each token corresponds to a request by a process to increment a distributed counter. Suppose that the output width of a counting network is t. At each output wire i on the counting network there is variable v_i that assigns counter values to the tokens. The initial value of v_i is i. If a token τ exits on wire i, in an atomic operation, τ is assigned the value v_i, and the value of v_i is increased by t. If in total m tokens traverse the network, each token is assigned a counter value between 0 and $m - 1$. In figure 1, we show the respective counter value that it assigned for each token that exits from the counting network and the (p, q)-balancer.

1.2 Contention

On an MIMD shared memory multiprocessor machine, the balancing network $\mathcal{B}$ is implemented as a shared data structure. Each balancer is a memory location shared by all the processes; wires are pointers from one memory location to another. The memory location of the balancer contains the necessary information that determines the wire the next token will exit from; this information may be accessed by any process's *token*. Each of the machine's n asynchronous *processes* runs a program that repeatedly traverses the data structure from some input pointer to some output pointer, each time shepherding a new token through the network.

Call n the *concurrency*. Tokens generated by process p_l, $l \in \{0, \ldots, n-1\}$, enter the network on input wire $l \mod w$, where w is the input width of the network. Since each process can have at most one token traversing the network at any time, the total number of

tokens simultaneously traversing the network is no more than n. Note that in an execution, the total number of tokens that will traverse the network may exceed n, since a process may issue a token several times.

Contention in balancing network $\mathcal{B}$ occurs when two or more tokens are trying to access the same balancer simultaneously. In such a case, the tokens content for which one will atomically access the memory location of the balancer; all unsuccessful tokens must wait and try again. Each time a token passes through a balancer, it incurs a *stall step* or *stall* for short [9] to all other tokens pending at the balancer; equivalently, every time a token is bypassed by another token, a stall is incurred to it.

The number of stalls has been proposed by Dwork *et al.* [9] as a complexity-theoretic measure of contention in shared memory algorithms. Roughly speaking, the *contention* incurred by the traversal of m tokens through the network $\mathcal{B}$ at concurrency n, denoted $\mathrm{cont}(\mathcal{B}, n, m)$, is the maximum number of stalls, over all possible executions, induced by an adversary scheduler. Furthermore, the *amortized contention* of the network $\mathcal{B}$ at concurrency n, denoted $\mathrm{cont}(\mathcal{B}, n)$, is the limit supremum of $\mathrm{cont}(\mathcal{B}, n, m)$ divided by m, as m goes to infinity.

Naturally, the higher the (amortized) contention, the smaller the network's *throughput,* since a token is delayed less time in the network when tokens abound, and vice versa. Clearly, amortized contention is an appropriate measure of the average delay experienced by any token traversing a network. The amortized contention measure is both simple and practical in the sense that the only parameters that turned out to be needed for its analysis are the concurrency n and the width of the network. Even more so, it does not require any timing information on the arrival and departure time of tokens, as more complicated queueing theory models do.

1.3 Contribution

Almost all known counting network constructions are regular and they are built from regular balancers (see, e.g., [1, 5, 7, 8, 10]). The prime example of such networks is the *bitonic* counting network [5, Section 3]; built from $(2, 2)$-balancers, it achieves input and output width $w = 2^k$, for any integer $k > 0$. The depth of the bitonic network is $\Theta(\lg^2 w)$, while its amortized contention is $\Theta(n \lg^2 w/w)$ [9, Section 3.2]. Figure 2 depicts two instances of the bitonic counting network, one of width 4 and the other of width 8. Another notable construction with similar performance characteristics is the *periodic* counting network [5, Section 4], which achieves amortized contention $O(n \lg^3 w/w)$ [9, Section 3.4].

In this work, we depart from the regular approach to counting networks, and we built irregular counting networks. The principal motivation for our study is to improve the efficiency of counting networks by relaxing regularity. More specifically, we are interested in understanding whether, and by how much, irregular networks may improve on efficiency regarding amortized contention, at the same level of latency (network depth) and concurrency, over their regular counterparts. We are able to provide an affirmative, quantitative answer to this question.

1.3.1 Bounds

We present a novel construction of an irregular counting network $\mathcal{C}(w,t)$, where the input width w is smaller than or equal to the output width t. Specifically, $w = 2^k$ and $t = p2^l$, for some integers $k, l, p > 0$, so that $w \leq t$. Our network is constructed from $(2,2)$-balancers and $(2, 2t/w)$-balancers. For example, the network $\mathcal{C}(4,8)$ is illustrated in Figures 1 and 3, while the network $\mathcal{C}(8,48)$ appears in Figure 3.

The depth of network $\mathcal{C}(w,t)$, depends solely on the input width w; it is $\Theta(\lg^2 w)$. We would like to note that the depth of $\mathcal{C}(w,t)$ is exactly the same with the depth of a bitonic network of width w.

We discover that the amortized contention of the network $\mathcal{C}(w,t)$ depends on all three parameters w, t and n:

$$\text{cont}\left(\mathcal{C}(w,t), n\right) \;=\; O\left(\frac{n \lg w}{w} + \frac{n \lg^2 w}{t} + \frac{w \lg^3 w}{t} + \lg^2 w\right).$$

Since the amortized contention is now determined by three parameters, we expect this to offer some more flexibility and trade-offs when one must choose the right network for the specific needs of any particular counting problem. Apparently, our network provides more options than previous (regular) networks, like the bitonic and the periodic, whose contention depends only on two parameters (w and n).

To demonstrate this additional flexibility, but also to compare our network against previous constructions, we adjust the output width t for achieving efficiency.

- When $t = w$, we obtain a new regular counting network $\mathcal{C}(w,w)$, with depth $\Theta(\lg^2 w)$ and amortized contention $O(n \lg^2 w/w + \lg^3 w)$. For $n \geq w \lg w$, the amortized contention becomes $O(n \lg^2 w/w)$ which is the same as in the bitonic network of width w.

- By increasing the output width t the contention of network $\mathcal{C}(w,t)$ decreases, while its depth remains the same (since it only depends on w).

  - Specifically, by taking $t = w \lg w$ the amortized contention becomes $O(n \lg w/w + \lg^2 w)$. For $n \geq w \lg w$ the amortized contention becomes $O(n \lg w/w)$ which is better by a logarithmic factor of w over the amortized contention of the bitonic counting network with the same depth and width w. Therefore, out network can handle better higher concurrency.

Since our network achieves a decreased amortized contention as a function of the concurrency n, we naturally expect that it offers the option of a higher throughput for the same latency. No such options were available for any of the previously known networks.

1.3.2 Structural Interpretation

We attribute the improved performance characteristics enjoyed by our network construction to some features of its unique structure, as we argue below.

When we look inside the structure of our network, we identify three series blocks, $\mathcal{N}_a$, $\mathcal{N}_b$, and $\mathcal{N}_c$, as illustrated in Figure 3:

- Block $\mathcal{N}_a$ has input and output width w and depth $\lg w - 1$; it is built from $(2,2)$-balancers. This block is regular.
- Block $\mathcal{N}_b$ has input width w, output width t, and depth 1; it is built from $(2, 2t/w)$-balancers. This block serves as a transition block from block A to block C. This block is irregular.
- Block $\mathcal{N}_c$ has input and output width t, and depth $\Theta(\lg^2 w)$; it is built from $(2,2)$-balancers. This block is regular.

The dominant block with respect to depth is block $\mathcal{N}_c$. Thus, intuitively, tokens spend most of their time in this block of the network. It is therefore expected that contention will be heavily influenced from parameters of this block. By increasing the output width t, block $\mathcal{N}_c$ becomes wider and fatter with respect to the number of balancers, so that tokens will then have there less chance to collide at the same balancer. Consequently, as t increases, the contention in block $\mathcal{N}_c$ decreases, so that the contention of the entire network decreases. Even more so, by unboundedly increasing t, the contention in block $\mathcal{N}_c$ approaches $\lg^2 w$ (independent of the concurrency n). However, for a fixed w, as t becomes large, block $\mathcal{N}_a$ remains the same; thus, block $\mathcal{N}_a$ will be the one to determine the network's contention when $w \ll t$. Nevertheless, since the depth of block $\mathcal{N}_a$ is only $\Theta(\lg w)$, block $\mathcal{N}_a$ cannot affect the performance of the entire network very much, so that the gain in contention due to increasing t is preserved.

1.3.3 Remarks and Comparison

We remark that increasing t causes a corresponding increase to the number of balancers in block $\mathcal{N}_c$. For really large values of t, this may seem to cause a resource burden when implementing the network in a real shared memory multiprocessor system. Thus, there is an implementation tradeoff between the two cases $w = t$ and $w \ll t$. This tradeoff will have to depend on the particular intricacies and requirements of the counting problem in hand. A compromise where $t = w \lg w$ seems to provide a reasonable solution.

There are only two other known irregular counting networks. The first one, called a *diffracting tree,* is given by Shavit and Zemach [13]; built from $(1,2)$-balancers, it has the form of a binary tree with 1 input wire, w output wires, and depth $\lg w$. This construction employs randomization to implement a *diffraction scheme,* which allows a pair of colliding tokens to combine and eliminate themselves. Experiments have revealed some nice performance results for this construction; similar results have also been in [14] for the steady state, and under certain probabilistic assumptions on the frequency of traversals. Nevertheless, the amortized

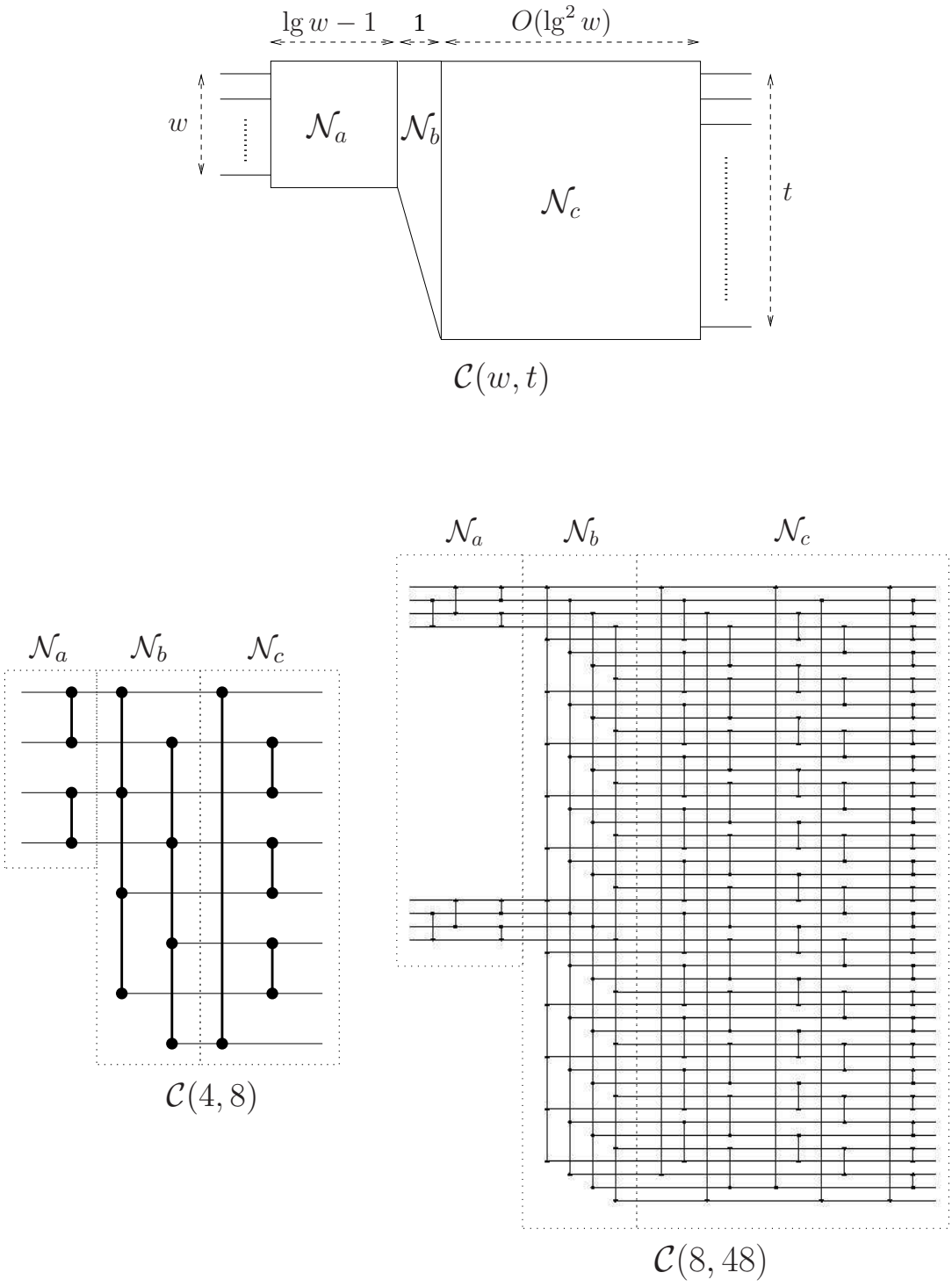


Figure 3: Partition of network $\mathcal{C}(w,t)$ into blocks $\mathcal{N}_a$, $\mathcal{N}_b$ and $\mathcal{N}_c$

contention of the diffracting tree is $\Theta(n)$, since it is possible for an adversary scheduler to accumulate all tokens at the root of the tree.

The second irregular construction is given by Aiello *et al.* [3]; it has input width w, output width $w \lg w$, and depth $O(\lg w)$; it is built from $(2,2)$-balancers and $(1,2)$-balancers. This construction uses as a building block the AKS sorting network [4], whose depth is $O(\log n)$, but the asymptotic depth notation hides huge constant factors which makes this counting network construction to be of no practical interest. On the other hand, our construction has small constants in the asymptotic notation of its depth and can be easily implemented in practice.

1.4 Road Map

In Section 2, we offer some necessary preliminaries for integer sequences, and give some definitions and basic results for balancing networks. We give in Section 3 the construction of a *merging* network, which is a major building block for the construction of our counting network in Section 4. In Section 5, we give the *butterfly* network, which is a special network that will be useful for the analysis of the amortized contention of our counting network in Section 6. Finally, in Section 7 we discuss our results and give some open problems.

2 Preliminaries

2.1 Step Sequences

The number of tokens that enter of exit a balancing network will be represented with sequences. We denote an integer sequence of length w with a boldface letter such as $\mathbf{x}^{(w)}$. The elements of the sequence are denoted with small letters; so, $\mathbf{x}^{(w)} = x_0, x_1, \ldots, x_{w-1}$. The sum of the elements of the sequence is denoted as $\sum(\mathbf{x}^{(w)}) = x_0 + x_1 + \cdots + x_{w-1}$. The *maximum value* is $\max_i(x_i)$, while the *minimum value* is $\min_i(x_i)$.

A subsequence of $\mathbf{x}^{(w)}$ is any sequence of elements $x_{i_0}, x_{i_2}, \ldots, x_{i_{k-1}}$, such that $i_j < i_{i_{j+1}}$, for all $0 \leq j \leq k-1$. The *even* subsequence of $\mathbf{x}^{(w)}$ is $\mathbf{x}_{\mathrm{e}}^{(w/2)} = x_0, x_2, \ldots$, while the *odd* subsequence is $\mathbf{x}_{\mathrm{o}}^{(w/2)} = x_1, x_3, \ldots$. If w is even, the *first half* of $\mathbf{x}^{(w)}$ is subsequence $x_0, x_1, \ldots, x_{w/2-1}$, while the *second half* of $\mathbf{y}^{(w)}$ is subsequence $y_{w/2}, y_{w/2+1}, \ldots, y_{w-1}$.

A sequence $\mathbf{x}^{(w)}$ has the *step property* [5] if $0 \leq x_i - x_j \leq 1$, for any pair of indices i and j such that $0 \leq i < j < w$; alternatively, we say that the sequence $\mathbf{x}^{(w)}$ is *step*. For a step sequence $\mathbf{x}^{(w)}$, its *step point* is either the unique index i such that $x_i < x_{i-1}$, or w if all x_i are equal; that is, $1 \leq i \leq w$, For any element x_i of a step sequence $\mathbf{x}^{(w)}$, it holds that [5]:

$$x_i = \left\lceil \frac{\sum(\mathbf{x}^{(w)}) - i}{w} \right\rceil . \tag{1}$$

We continue with two basic results for step sequences.

Lemma 2.1 *Any subsequence of a step sequence is step.*

Proof: Suppose that $\mathbf{x}^{(w)}$ is a step sequence. Let $\widehat{\mathbf{x}}^{(m)} = x_{i_0}, x_{i_2}, \ldots, x_{i_{m-1}}$ be a subsequence of $\mathbf{x}^{(w)}$. We have that $0 \leq x_{i_j} - x_{i_k} \leq 1$, for all $0 \leq j < k < m$, since $\mathbf{x}^{(w)}$ is step. Therefore, $\widehat{\mathbf{x}}^{(m)}$ is step too. ■

Lemma 2.2 *Consider a pair of step sequences $\mathbf{x}^{(w)}$ and $\mathbf{y}^{(w)}$, where $w \geq 2$, with maximum values a and b, respectively. If there is an integer δ such that*

$$0 \leq \sum \left(\mathbf{x}^{(w)}\right) - \sum \left(\mathbf{y}^{(w)}\right) \leq \delta,$$

then,

$$0 \leq a - b \leq \left\lfloor \frac{\delta}{w} \right\rfloor + 1.$$

Proof: Since a and b represent the maximum values of the step sequences $\mathbf{x}^{(w)}$ and $\mathbf{y}^{(w)}$, respectively, it holds that

$$w(a-1) < \sum \left(\mathbf{x}^{(w)}\right) \leq wa,$$

and

$$w(b-1) < \sum \left(\mathbf{y}^{(w)}\right) \leq wb.$$

By subtracting these two inequalities, we get

$$w(a-b-1) < \sum \left(\mathbf{x}^{(w)}\right) - \sum \left(\mathbf{y}^{(w)}\right) < w(a-b+1).$$

By inequality

$$0 \leq \sum \left(\mathbf{x}^{(w)}\right) - \sum \left(\mathbf{y}^{(w)}\right) \leq \delta$$

we get that

$$w(a-b-1) < \delta,$$

and,

$$w(a-b+1) > 0.$$

Since $w \geq 2$, it follows that

$$a - b < \frac{\delta}{w} + 1,$$

and

$$a - b > -1.$$

Since a and b are integers, this implies that

$$0 \leq a - b \leq \left\lfloor \frac{\delta}{w} \right\rfloor + 1,$$

as needed. ■

Next, we give properties for the even and odd subsequences of step sequences.

Lemma 2.3 *If $\mathbf{x}^{(w)}$ is a step sequence, and w is even with $w \geq 2$, then*

$$0 \leq \sum \left(\mathbf{x}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) - \sum \left(\mathbf{x}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right) \ \leq \ 1 \, .$$

Proof: Let a be the maximum value of $\mathbf{x}^{(w)}$. Let k be the step point of $\mathbf{x}^{(w)}$. All the elements x_i with $i < k$ have value a, while the remaining elements have value $a-1$. Thus, all elements $x_0, \ldots, x_{k-1}$ have value a, while all elements in $x_k, \ldots, x_{w-1}$ have value $a-1$.

If k is even, we have $x_{2i} = x_{2i+1} = a$, for $i < k/2$, while $x_{2i} = x_{2i+1} = a-1$, for $i \geq k/2$. Thus, $\mathbf{x}_{\mathrm{e}}^{(w)} = \mathbf{x}_{\mathrm{o}}^{(w)}$, which implies

$$\sum \left(\mathbf{x}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) - \sum \left(\mathbf{x}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right) = 0.$$

If k is odd, we have $x_{2i} = x_{2i+1} = a$, for $i < (k-1)/2$, while $x_{2i} = x_{2i+1} = a-1$, for $i > (k-1)/2$. Further, $a - 1 = x_k < x_{k-1} = a$. Thus, $\mathbf{x}_{\mathrm{e}}^{(w)}$ and $\mathbf{x}_{\mathrm{o}}^{(w)}$ differ only on their kth element, which implies

$$\sum \left(\mathbf{x}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) - \sum \left(\mathbf{x}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right) = 1.$$

■

Lemma 2.4 *Consider two step sequences $\mathbf{x}^{(w)}$ and $\mathbf{y}^{(w)}$, where $w \geq 2$. If there is an even integer δ such that*

$$0 \leq \sum \left(\mathbf{x}^{(w)} \right) - \sum \left(\mathbf{y}^{(w)} \right) \leq \delta,$$

then

$$0 \leq \sum \left(\mathbf{x}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) - \sum \left(\mathbf{y}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) \leq \frac{\delta}{2},$$

and

$$0 \leq \sum \left(\mathbf{x}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right) - \sum \left(\mathbf{y}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right) \leq \frac{\delta}{2}.$$

Proof: Denote

$$\begin{aligned} A &= \sum \left(\mathbf{x}_{\mathrm{e}}^{(w/2)} \right) - \sum \left(\mathbf{y}_{\mathrm{e}}^{(w/2)} \right), \\ B &= \sum \left(\mathbf{x}_{\mathrm{o}}^{(w/2)} \right) - \sum \left(\mathbf{y}_{\mathrm{o}}^{(w/2)} \right). \end{aligned}$$

We will show that $0 \leq A \leq \delta/2$ and $0 \leq B \leq \delta/2$. We have:

$$\begin{aligned} \sum \left(\mathbf{x}^{(w)} \right) &= \sum \left(\mathbf{x}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) + \sum \left(\mathbf{x}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right), \\ \sum \left(\mathbf{y}^{(w)} \right) &= \sum \left(\mathbf{y}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) + \sum \left(\mathbf{y}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right). \end{aligned}$$

Since by assumption, $0 \leq \sum(\mathbf{x}^{(w)}) - \sum(\mathbf{y}^{(w)}) \leq \delta$, it follows that

$$0 \leq \left(\sum \left(\mathbf{x}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) + \sum \left(\mathbf{x}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right) \right) - \left(\sum \left(\mathbf{y}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) + \sum \left(\mathbf{y}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right) \right) \leq \delta$$

or

$$0 \leq A + B \leq \delta. \tag{2}$$

From Lemma 2.3,

$$0 \leq \sum \left(\mathbf{x}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) - \sum \left(\mathbf{x}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right) \leq 1, \tag{3}$$

and

$$0 \leq \sum \left(\mathbf{y}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) - \sum \left(\mathbf{y}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right) \leq 1. \tag{4}$$

By subtracting Inequalities 3 and 4 we get

$$-1 \leq \left(\sum \left(\mathbf{x}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) - \sum \left(\mathbf{y}_{\mathrm{e}}^{\left(\frac{w}{2}\right)} \right) \right) - \left(\sum \left(\mathbf{x}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right) - \sum \left(\mathbf{y}_{\mathrm{o}}^{\left(\frac{w}{2}\right)} \right) \right) \leq 1$$

or

$$-1 \leq A - B \leq 1. \tag{5}$$

By adding Inequalities 2 and 5, we get

$$-\frac{1}{2} \leq A \leq \frac{\delta}{2} + \frac{1}{2},$$

and by subtracting Inequalities 2 and 5, we get

$$-\frac{1}{2} \leq B \leq \frac{\delta}{2} + \frac{1}{2}.$$

Since A and B are integers we get

$$0 \leq A \leq \frac{\delta}{2},$$

and

$$0 \leq B \leq \frac{\delta}{2},$$

as needed. ■

2.2 Smooth Sequences and Permutations

A sequence $\mathbf{x}^{(w)}$ has the *k-smooth property* [1, 5] if $|x_i - x_j| \leq k$, for any pair of indices i and j such that $0 \leq i, j < w$; we say also that the sequence $\mathbf{x}^{(w)}$ is *k-smooth*. Notice that the elements of any k-smooth sequence take values in a range $a, a+1, \ldots, a+k$, for some integer a. Clearly, any step sequence is also 1-smooth.

Consider the set $H = \{0, \ldots, w-1\}$. A permutation on H is a correspondence (one to one and onto function) $\pi : H \rightarrow H$, that maps each element of H to another element of H. We define the permutation of a sequence $\mathbf{x}^{(w)}$ to be $\pi(\mathbf{x}^{(w)}) = \mathbf{y}^{(w)}$ if $x_i = y_{h(i)}$, for each $i \in H$. Since permutation π is a correspondence, it has a *reverse* permutation, denoted π^R, such that $\pi^R(h(i)) = i$. Note that if $\pi(\mathbf{x}^{(w)}) = \mathbf{y}^{(w)}$, then $\mathbf{x}^{(w)} = \pi^R(\mathbf{y}^{(w)})$.

Lemma 2.5 *If a sequence $\mathbf{x}^{(w)}$ is k-smooth and π is a permutation, then $\pi(\mathbf{x}^{(w)})$ is k-smooth.*

Proof: Let $\pi(\mathbf{x}^{(w)}) = \mathbf{y}^{(w)}$. We have that for any pair of elements y_i and y_j of $\mathbf{y}^{(w)}$, $x_{\pi^R(i)} = y_i$ and $x_{\pi^R(j)} = y_j$. Since $|x_{\pi^R(i)} - x_{\pi^R(j)}| \leq k$, we get $|y_i - y_j| \leq k$. ■

2.3 Balancing networks

Consider a balancing network $\mathcal{B}$ of input width w and output width t. Each balancer b in $\mathcal{B}$ has a *depth* which is the length of the longest path, in terms of number of balancers, that a token has to traverse from an input wire of $\mathcal{B}$ until the token reaches an output wire of balancer b. The depth of the network, denoted depth($\mathcal{B}$), is the maximum depth of any balancer in $\mathcal{B}$. Suppose that $d = \text{depth}(\mathcal{B})$. Network $\mathcal{B}$ can be decomposed into d *layers* of balancers, $\ell_1, \ldots, \ell_d$, such that layer ℓ_i contains all the balancers with depth i. Note that a layer is itself a balancing network with input and output width. Note also that in a regular balancing network all the layers have input and output width equal to the width of the regular network.

Consider now a (p, q)-balancer b. At any moment during an execution where tokens traverse b, the balancer b has a state which is the index of the output wire on which b will forward the next token that it processes. Thus, the state is a number in $\{0, \ldots, q-1\}$. A *transition* $\alpha(\tau, b)$ is the action of taking a token τ from the input and forwarding it to an output wire of b. The transition increases the state of the balancer by one (in a modulo-q operation).

Consider now a balancing network $\mathcal{B}$ with balancers $b_1, \ldots, b_k$. The state of the network is the collection of the states of its balancers. Each transition in the network brings the network from one state to another. An execution E in $\mathcal{B}$ that involves tokens $\tau_1, \ldots, \tau_m$ in $\mathcal{B}$ is a sequence of transitions, namely, $E = \alpha(\tau_{i_1}, b_{i_1}), \alpha(\tau_{i_2}, b_{i_2}), \ldots, \alpha(\tau_{i_k}, b_{i_k})$, where k is the length of the execution. At the end of the last transition in the execution (transition $\alpha(\tau_{i_k}, b_{i_k})$), there are no tokens traversing the network; in this case we say that network has reached a *quiescent* state.

Consider a (p, q)-balancer b in a quiescent state. Let x_i denote the number of tokens that have entered the balancer on input wire i. The sequence $\mathbf{x}^{(p)} = x_0, \ldots, x_{p-1}$ is the *input sequence* to balancer b (see Figure 1). Let y_i denote the number of tokens that have left from output wire i of balancer b. The sequence $\mathbf{y}^{(q)} = y_0, \ldots, y_{q-1}$ is the *output sequence* of balancer b. The output sequence $\mathbf{y}^{(q)}$ satisfies the step property. The input and output sequences satisfy the *sum preservation property*, which expresses the fact that in a quiescent state all tokens that have entered the balancer has also left it: $\sum(\mathbf{x}^{(p)}) = \sum(\mathbf{y}^{(q)})$.

Since $\mathbf{y}^{(w)}$ is step, from Equation 1 it holds for any output wire i that $y_i = \left\lceil \frac{\sum(\mathbf{x}^{(w)}) - i}{w} \right\rceil$. Therefore, $\mathbf{y}^{(w)}$ is a function on the number of tokens that have gone through the balancer, $\sum(\mathbf{y}^{(q)})$. From the sum preservation property, we have $y_i = \left\lceil \frac{\sum(\mathbf{x}^{(w)}) - i}{w} \right\rceil$. Therefore, the output sequence $\mathbf{y}^{(q)}$ is function on the number of tokens that have entered b. Thus, any two executions that involve balancer b such that the same number of tokens traverse the balancer, will leave the the balancer in the same state with the same values on the output sequence $\mathbf{y}^{(w)}$.

Consider now a balancing network of input width w and output width t. For any quiescent state, we define the input sequence $\mathbf{x}^{(w)}$ and output sequence $\mathbf{y}^{(w)}$ of $\mathcal{B}$, similarly as for the balancer (see Figure 1). It can be easily shown by induction on the layers of $\mathcal{B}$ that $\mathcal{B}$ satisfies the *sum preservation property:* $\sum(\mathbf{x}^{(w)}) = \sum(\mathbf{y}^{(t)})$. Similarly, it can be shown by induction

on the number of layers of $\mathcal{B}$ that the output sequence *vyt* only depends on the particular values on each entry of $\mathbf{x}^{(w)}$. Thus, for any two executions the corresponding values in $\mathbf{y}^{(t)}$ are the same, as long the number of tokens on each input wire are the same.

We will consider the following balancing network families, which are described according to their behavior in a quiescent state:

- *Counting network*: For any values in the input sequence, the output sequence satisfies the step property.
- *k-Smoothing network*: For any values in the input sequence, the output sequence satisfies the k-smooth property.
- *Difference merging network*: Suppose that the width of the network is w. Let $\mathbf{u}^{(w)}$ be the input sequence. The input sequence $\mathbf{u}^{(w)}$ is decomposed into two sequences $\mathbf{x}^{(w/2)}$ (*first input sequence*) and $\mathbf{y}^{(w/2)}$ (*second input sequence*), consisting of the first and second half, respectively, of $\mathbf{u}^{(w)}$. There is a *merging parameter* $\delta \geq 1$ that specifies the behavior of the network. If both $\mathbf{x}^{(w/2)}$ and $\mathbf{y}^{(w/2)}$ satisfy the step property, and $0 \leq \sum(\mathbf{x}^{(w/2)}) - \sum(\mathbf{y}^{(w/2)}) \leq \delta$, then the output sequence satisfies the step property too. That is, a difference merging network merges two step input sequences into a unique step output sequence, if the sums of the input sequences differ by at most δ.

The following result states that in a regular balancing network, if the input to a layer is k-smooth, then the output of the layer (thus, of every subsequent layer) is k-smooth too.

Lemma 2.6 *Consider a regular balancing network $\mathcal{B}$ of width w. Let ℓ denote a layer of $\mathcal{B}$. If the input sequence to ℓ is k-smoothing, then the the output sequence of ℓ is k-smoothing too.*

Proof: Suppose that the input sequence to ℓ is k-smoothing. We will show that its output sequence is k-smooth too. The output sequence of any particular balancer in ℓ is step, which is is trivially k-smooth. Thus, we only need to examine the difference on the output sequences of two different balancers b_1 and b_2 in ℓ. Suppose that a is the minimum value on the input sequence of ℓ. Then, the maximum value on the output sequence of ℓ is at most $a + k$.

The maximum difference between the output sequences of b_1 and b_2 is when one balancer, say b_1, receives a tokens on each input wire, while the other balancer, say b_2, receives $a + k$ tokens on each input wire. For such inputs, on each output wire of b_1 will exit a tokens, and on each output wire of b_2 will exit $a + k$ tokens. Thus, the difference on the number of tokens between any two wires of the two balancers is bounded by k. Therefore, the output sequence of ℓ is k-smooth. ■

2.4 Isomorphic Balancing Networks

Consider two balancing networks $\mathcal{B}_1$ and $\mathcal{B}_2$ that have the same input width w, and the same output width t. We say that the networks $\mathcal{B}_1$ and $\mathcal{B}_2$ are *isomorphic* if three conditions hold:

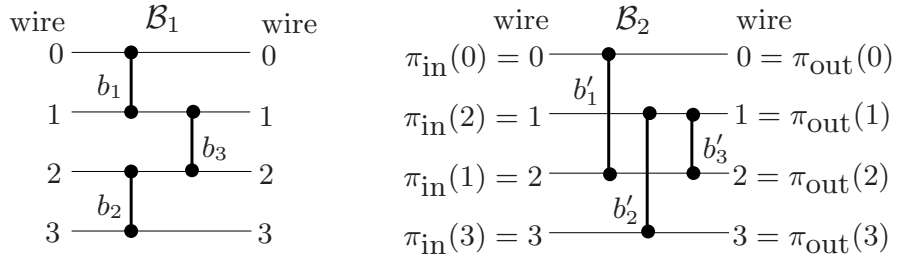


Figure 4: Isomorphic

i. There is a correspondence between the balancers of $\mathcal{B}_1$ and $\mathcal{B}_2$ so that any (p, q)-balancer b in $\mathcal{B}_1$ has a corresponding (p, q)-balancer b' in $\mathcal{B}_2$.

ii. For any balancer b_i in $\mathcal{B}_1$ whose k-th output wire is connected to an input wire of a balancer b_j, it holds that in $\mathcal{B}_2$ the k-th output wire of balancer b'_i is connected to some input wire of balancer b'_j (the input wire in b_j may not be necessarily the same as the input wire in b'_j).

iii. There is a correspondence between the input wires of $\mathcal{B}_1$ and $\mathcal{B}_2$. Similarly there is a correspondence between the output wires of $\mathcal{B}_1$ and $\mathcal{B}_2$.

Let $\mathbf{x}^{(w)}$ and $\mathbf{y}^{(t)}$ be the respective input and output sequences of $\mathcal{B}_1$, and let $\mathbf{u}^{(w)}$ and $\mathbf{z}^{(t)}$ be the respective input and output sequences of $\mathcal{B}_2$. Let π_{in} be the correspondence (permutation) that maps input wires of $\mathcal{B}_1$ to input wires of $\mathcal{B}_2$. Similarly, let π_{out} be the correspondence (permutation) that maps output wires of $\mathcal{B}_1$ to output wires of $\mathcal{B}_2$. From condition (iii) of isomorphism, if token τ enters on input wire j in B_1, then t enters on wire $\pi_{\text{in}}(j)$ in B_2. Similarly, if token τ exits on output wire j in B_1, then t exits on wire $\pi_{\text{out}}(j)$ in B_2. An example of two isomorphic networks is shown in Figure 4.

Any execution $E = \alpha(\tau_{i_1}, b_{i_1}), \alpha(\tau_{i_2}, b_{i_2}), \ldots, \alpha(\tau_{i_k}, b_{i_k})$ in B_1 with tokens $\{\tau_1, \ldots, \tau_m\}$, has a corresponding execution $E' = \alpha(\tau'_{i_1}, b'_{i_1}), \alpha(\tau'_{i_2}, b'_{i_2}), \ldots, \alpha(\tau'_{i_k}, b'_{i_k})$ in B_2 with tokens $\{\tau'_1, \ldots, \tau'_m\}$, where token τ_i corresponds to token τ'_i. At the end of executions E and E', the number of tokens that have left the ith wire of a balancer b in B_2 is the same with the the number of tokens that have left the ith wire of balancer b' in B_2.

Lemma 2.7 *If $\mathcal{B}_1$ and $\mathcal{B}_2$ are isomorphic and in are in quiescent states that holds $\mathbf{u}^{(w)} = \pi_{in}(\mathbf{x}^{(w)})$, then $\mathbf{z}^{(w)} = \pi_{out}(\mathbf{y}^{(w)})$.*

Proof: Let $\mathbf{c}_{\text{in}}^{(w)}$ be an arbitrary sequence of values. Consider two executions E_1 and E_2 of $\mathcal{B}_1$ in which the input sequence to $\mathcal{B}_1$ is $\mathbf{x}^{(w)} = \mathbf{c}_{\text{in}}^{(w)}$. In both executions, the resulting output sequence of $\mathcal{B}_1$ has to be the same, namely $\mathbf{y}^{(w)} = \mathbf{c}_{\text{out}}^{(t)}$, for some assignment of values to a sequence $\mathbf{c}_{\text{out}}^{(t)}$. For any execution E in $\mathcal{B}_1$ there is a corresponding execution E' in $\mathcal{B}_1$, such that on each input and output wire of $\mathcal{B}_2$ there will be the same number of tokens as on the corresponding wires on $\mathcal{B}_1$ on execution E.

Suppose that in execution E the input sequence is $\mathbf{x}^{(w)} = \mathbf{c}_{\text{in}}^{(w)}$; thus, the output sequence is $\mathbf{y}^{(w)} = \mathbf{c}_{\text{out}}^{(w)}$. In execution E' the corresponding input sequence is $\mathbf{u}^{(w)} = \pi_{\text{in}}(\mathbf{c}_{\text{in}}^{(w)})$. The output sequence on E' is $\mathbf{z}^{(w)} = \pi_{\text{out}}(\mathbf{c}_{\text{out}}^{(w)})$. Actually, any other execution on $\mathcal{B}_2$ with input sequence $\mathbf{u}^{(w)} = \pi_{\text{in}}(\mathbf{c}_{\text{in}})$ results to the same output sequence $\mathbf{z}^{(w)} = \pi_{\text{out}}(\mathbf{c}_{\text{out}}^{(w)})$. Since $\mathbf{y}^{(w)} = \mathbf{c}_{\text{out}}^{(t)}$, we obtain $\mathbf{z}^{(w)} = \pi_{\text{out}}(\mathbf{y}^{(w)})$. ■

Lemma 2.8 *If $\mathcal{B}_1$ and $\mathcal{B}_2$ are isomorphic and $\mathcal{B}_1$ is k-smoothing, then $\mathcal{B}_2$ is k-smoothing.*

Proof: Suppose that sequence $\mathbf{u}^{(w)}$ takes arbitrary values. Then set the values on $\mathbf{x}^{(w)}$ so that $\mathbf{x}^{(w)} = \pi_{\text{in}}^{R}(\mathbf{u}^{(w)})$. Since $\mathcal{B}_1$ is k-smoothing, $\mathbf{y}^{(w)}$ is k-smooth. From Lemma 2.5, $\pi_{\text{out}}(\mathbf{y}^{(w)})$ is k-smooth too. From Lemma 2.7, $\pi_{\text{out}}(\mathbf{y}^{(w)}) = \mathbf{z}^{(w)}$. Therefore, $\mathbf{z}^{(w)}$ is k-smooth. ■

3 Difference Merging Network

We present the construction of a difference merging network $\mathcal{M}(t, \delta)$ where t is the network width (recall that difference merging networks are regular) and δ is the merging parameter, such that $t = p2^l$, $\delta = 2^k$, $p \geq 1$, and $1 \leq k < l$ (such assignments to parameters t and δ will be called *valid*).

3.1 Construction of $\mathcal{M}(w, t)$

We will give the construction of network $\mathcal{M}(w, \delta)$. To describe the network, we will use input and output sequences to refer to their corresponding wires in the network. In the construction, we will say that any two sequences $\mathbf{c}^{(w)}$ and $\mathbf{e}^{(y)}$ are *directly-connected* if c_i is connected with a wire to e_i.

Let $\mathbf{u}^{(t)}$ and $\mathbf{z}^{(t)}$ denote the input and output sequences, respectively, of $\mathcal{M}(t, \delta)$. Let $\mathbf{x}^{(t/2)}$ (*first input sequence*) and $\mathbf{y}^{(t/2)}$ (*second input sequence*), denote the first and second half of $\mathbf{z}^{(t)}$, respectively.

The construction of $\mathcal{M}(t, \delta)$ is recursive on the parameter δ; parameter t take any valid value (see Figure 5).

- *Recursive basis, $\delta = 2$.* The network $\mathcal{M}(t, 2)$ consists of a single layer of $t/2$ copies of the $(2, 2)$-balancer, denoted $b_0, \ldots, b_{t/2-1}$ (see Figure 5). For $1 \leq i < t/2$, the first and second input wires of balancer b_i are connected to y_{i-1} and x_i, respectively, and the first and second output wires are connected to z_{2i-1} and z_{2i}, respectively. For balancer b_0, the first and second input wires are connected to x_0 and $y_{t/2-1}$, respectively, the first and second output wires are connected to z_0 and z_{t-1}, respectively.

- *Recursive step, $\delta > 2$.* Suppose that we have constructed the network $\mathcal{M}(t', \delta/2)$, for any valid t'. The network $\mathcal{M}(t, \delta)$ is constructed in two sub-steps, as follows (see Figure 5).

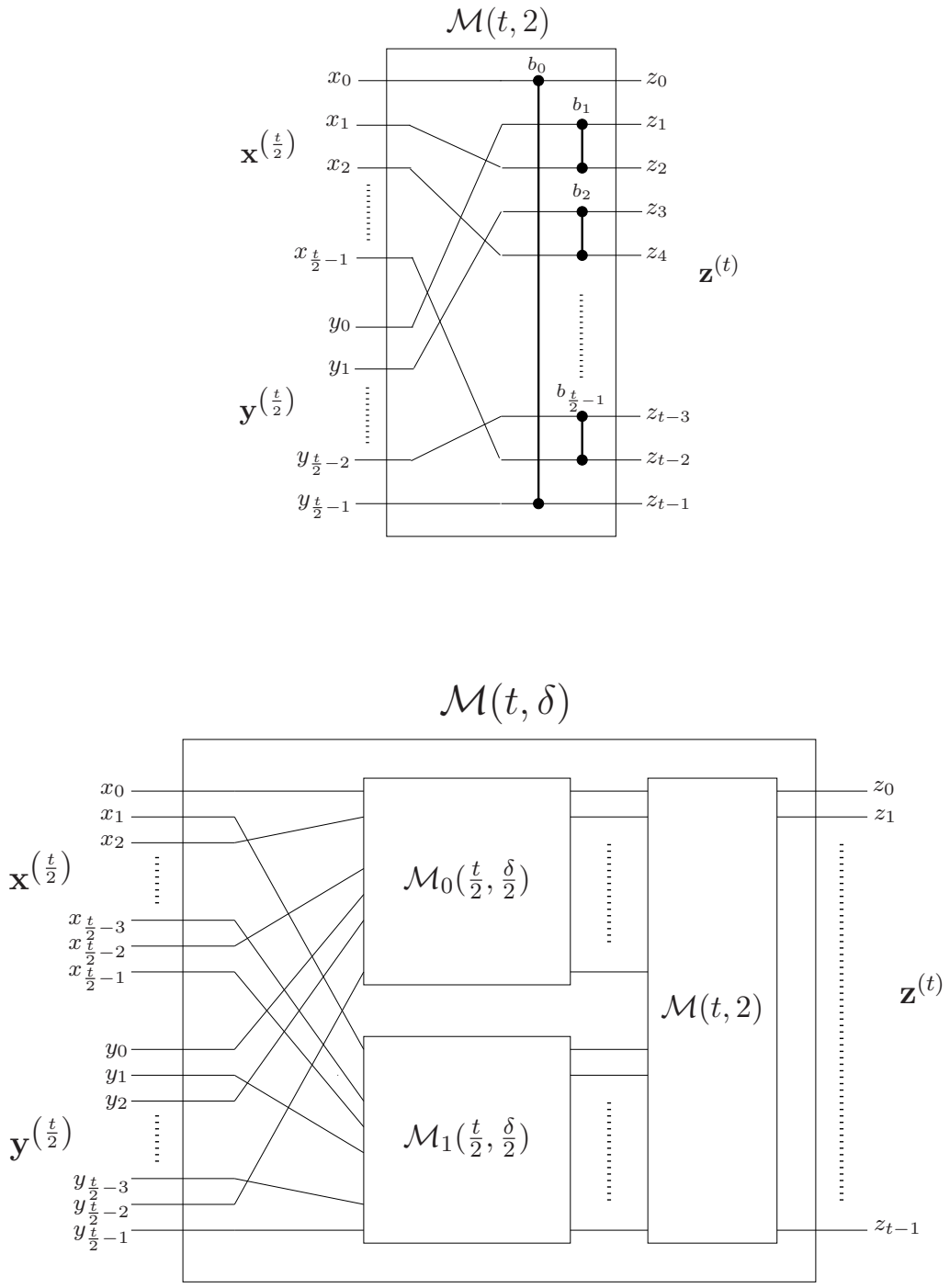


Figure 5: Top: the network $\mathcal{M}(t,2)$; Bottom: the recursive construction of $\mathcal{M}(t,\delta)$

– *Sub-step 1.* Take two copies of the network $\mathcal{M}(t/2, \delta/2)$, denoted $\mathcal{M}_0(t/2, \delta/2)$ and $\mathcal{M}_1(t/2, \delta/2)$. The first input sequence of $\mathcal{M}_0(t/2, \delta/2)$, is directly-connected to the even subsequence of $\mathbf{x}^{(t/2)}$, namely, $\mathbf{x}_{\mathrm{e}}^{(t/4)}$, while the second input sequence is directly-connected to the even subsequence of $\mathbf{y}^{(t/2)}$, namely, $\mathbf{y}_{\mathrm{e}}^{(t/4)}$. The first input sequence of $\mathcal{M}_1$ is directly-connected to the odd subsequence of $\mathbf{x}^{(t/2)}$, namely, $\mathbf{x}_{\mathrm{o}}^{(t/4)}$, while the second input sequence is directly-connected to the odd subsequence of $\mathbf{y}^{(t/2)}$, namely, $\mathbf{y}_{\mathrm{o}}^{(t/4)}$. Let $\mathbf{g}^{(t/2)}$ and $\mathbf{h}^{(t/2)}$ denote the output sequences of networks $\mathcal{M}_0(t/2, \delta/2)$ and $\mathcal{M}_1(t/2, \delta/2)$, respectively.

– *Sub-step 2.* Take a copy of the network $\mathcal{M}(t, 2)$ which is given by the recursion basis. The first input sequence of $\mathcal{M}(t, 2)$ is directly-connected to the output sequence $\mathbf{g}^{(t/2)}$ of $\mathcal{M}_0(t/2, \delta/2)$ and the second input sequence is directly-connected to the output sequence $\mathbf{h}^{(t/2)}$ of $\mathcal{M}_1(t/2, \delta/2)$. Finally, the output sequence of $\mathcal{M}(t, 2)$ is directly-connected to the output sequence $\mathbf{z}^{(t)}$ of $\mathcal{M}(t, \delta)$.

Next, we calculate the depth of network $\mathcal{M}(t, \delta)$.

Lemma 3.1 *depth*$(\mathcal{M}(t, \delta)) = \lg \delta$.

Proof: By the recursive construction of $\mathcal{M}(t, \delta)$, $\mathrm{depth}(\mathcal{M}(t, 2)) = 1$ and

$$\begin{aligned}
\mathrm{depth}\left(\mathcal{M}(t,\delta)\right) &= \mathrm{depth}\left(\mathcal{M}\left(\frac{t}{2},\frac{\delta}{2}\right)\right) + \mathrm{depth}\left(\mathcal{M}(w,2)\right) \\
&= \mathrm{depth}\left(\mathcal{M}\left(\frac{t}{2},\frac{\delta}{2}\right)\right) + 1 \\
&\quad (\text{since } \mathrm{depth}\left(\mathcal{M}(t,2)\right) = 1) \\
&= \mathrm{depth}\left(\mathcal{M}\left(\frac{t}{4},\frac{\delta}{4}\right)\right) + 1 + 1 \\
&= \mathrm{depth}\left(\mathcal{M}\left(\frac{t}{2^2},\frac{\delta}{2^2}\right)\right) + 2 \\
&= \ldots \\
&= \mathrm{depth}\left(\mathcal{M}\left(\frac{t}{2^k},\frac{\delta}{2^k}\right)\right) + k \\
&= \ldots \\
&= \mathrm{depth}\left(\mathcal{M}\left(\frac{t}{2^{\lg\delta-1}},\frac{\delta}{2^{\lg\delta-1}}\right)\right) + \lg\delta - 1 \\
&= \mathrm{depth}\left(\mathcal{M}\left(\frac{2t}{\delta},2\right)\right) + \lg\delta - 1 \\
&= 1 + \lg\delta - 1 \\
&\quad (\text{since } \mathrm{depth}\left(\mathcal{M}\left(\tfrac{2t}{\delta},2\right)\right) = 1) \\
&= \lg\delta,
\end{aligned}$$

as needed. ∎

3.2 Correctness of $\mathcal{M}(w,t)$

We prove the correctness of $\mathcal{M}(t,\delta)$. We start with the correctness of $\mathcal{M}(t,2)$.

Lemma 3.2 *$\mathcal{M}(t,2)$ is a difference merging network.*

Proof: Suppose that $\mathcal{M}(t,2)$ is quiescent. Suppose that each of the input sequences $\mathbf{x}^{(t/2)}$ and $\mathbf{y}^{(t/2)}$ satisfies the step property and $0 \leq \sum(\mathbf{x}^{(t/2)}) - \sum(\mathbf{y}^{(t/2)}) \leq 2$. We will show that the the output sequence $\mathbf{z}^{(t)}$ satisfies the step property.

Denote by a and b the maximum values in sequences $\mathbf{x}^{(t/2)}$ and $\mathbf{y}^{(t/2)}$, respectively. Denote by k and l the step points, respectively. Since $t/2 \geq 2$, Lemma 2.2 implies that we have $0 \leq a - b \leq 1$.

The various possible values on the input and output sequences of $\mathcal{M}(t,2)$ are illustrated in Figures 6, 7, and 8. Each sequence is represented with a linear array were an entry is darker for a higher value, and lighter for a lower value. In the figures, $\widehat{\mathbf{u}}^{(t)}$ denotes a permutation of the input sequence $\mathbf{u}^{(t)}$ where $\mathbf{x}^{(t/2)}$ (the first half of $\mathbf{u}^{(t)}$) corresponds to the even subsequence of $\widehat{\mathbf{u}}^{(t)}$, and $\mathbf{y}^{(t/2)}$ (the second half of $\mathbf{u}^{(t)}$) corresponds to the odd subsequence of $\widehat{\mathbf{u}}^{(t)}$.

We consider the cases $a = b$ and $a = b + 1$, separately.

- $a = b$. In this case, $\widehat{\mathbf{u}}^{(t)}$ is 1-smooth. The step points k and l differ by at most 2, which gives the following three possibilities.
  - $k = l$. It holds $1 \leq k \leq t/2$. The case $k < t/2$ is depicted in Figure 6.a, where only balancers b_k and b_0 receive two different values on their inputs. The case $k = t/2$ is depicted in Figure 7.a, where all balancers receive the same input values. The output sequence $\mathbf{z}^{(t)}$ is step in both cases.
  - $k = l+1$. It holds $2 \leq k \leq t/2$, and $1 \leq l \leq t/2-1$. The case $k < t/2$ is depicted in Figure 6.b, while the case $k = t/2$ is depicted in Figure 7.b. In both cases, only balancer b_0 receives two different values on its inputs. The output sequence $\mathbf{z}^{(t)}$ is step in both cases.
  - $k = l+2$. It holds $3 \leq k \leq t/2$ and $1 \leq l \leq t/2-2$. The case $k < t/2$ is depicted in Figure 6.c, while the case $k = t/2$ is depicted in Figure 7.b. In both cases, only balancers b_{k-1} and b_0 receive two different values on their inputs. The output sequence $\mathbf{z}^{(t)}$ is step in both cases.
- $a = b+1$. In this case, it is possible that $\widehat{\mathbf{u}}^{(t)}$ is 2-smooth. There are three possibilities with respect to the step points k and l that we examine separately.
  - $k = 1$ *and* $l = t/2 - 1$. This case is depicted in Figure 8.a. Note that $\widehat{\mathbf{u}}^{(t)}$ is 2-smooth, and balancer b_0 receives values a and $a-2$ on its inputs. The resulting sequence $\mathbf{z}^{(t)}$ contains $a-1$ in all of its entries. Thus, $\mathbf{z}^{(t)}$ is step.
  - $k = 1$ *and* $l = t/2$. This case is depicted in Figure 8.b. Here, $\widehat{\mathbf{u}}^{(t)}$ is 1-smooth. Only balancer b_0 receives different values on its inputs. The output sequence $\mathbf{z}^{(t)}$ is step.

– $k = 2$ *and* $l = t/2$. This case is depicted in Figure 8.b. Note that $\widehat{\mathbf{u}}^{(t)}$ is again 1-smooth. Only balancers b_0 and b_1 receive different values on their inputs. The output sequence $\mathbf{z}^{(t)}$ is step.

So, in all cases the output sequence $\mathbf{z}^{(t)}$ satisfies the step property, as needed. ■

Next, we show the correctness of network $\mathcal{M}(t, \delta)$ for any δ.

Lemma 3.3 *$\mathcal{M}(t, \delta)$ is a difference merging network.*

Proof: Suppose that $\mathcal{M}(t, 2)$ is quiescent, and that each of the input sequences $\mathbf{x}^{(t/2)}$ and $\mathbf{y}^{(t/2)}$ satisfy the step property with $0 \leq \sum(\mathbf{x}^{(t/2)}) - \sum(\mathbf{y}^{(t/2)}) \leq \delta$. We will show that the the output sequence $\mathbf{z}^{(t)}$ satisfies the step property. We will prove the claim by induction on δ.

For the basis case $\delta = 2$, Lemma 3.2 proves that $\mathcal{M}(t, 2)$ is a difference merging network.

Consider now the case $\delta > 2$, and suppose that the network $\mathcal{M}(t', \delta/2)$ is a difference merging network (induction hypothesis), for any valid value of t'. We will show that the network $\mathcal{M}(t, \delta)$ is a difference merging network.

Since $\mathbf{z}^{(t)}$ is the output sequence of the difference merging networks $\mathcal{M}(t, 2)$, $\mathbf{z}^{(t)}$ will be step if each of $\mathbf{g}^{(t/2)}$ and $\mathbf{h}^{(t/2)}$ satisfy the step property and $0 \leq \sum(\mathbf{g}^{(t/2)}) - \sum(\mathbf{h}^{(t/2)}) \leq 2$.

First we show that the sequence $\mathbf{g}^{(t/2)}$ is step. The input sequences of network $\mathcal{M}_0(t/2, \delta/2)$ are the even subsequences of $\mathbf{x}^{(t/2)}$ and $\mathbf{y}^{(t/2)}$, namely $\mathbf{x}_{\mathrm{e}}^{(t/4)}$ and $\mathbf{y}_{\mathrm{e}}^{(t/4)}$. By the induction hypothesis, network $\mathcal{M}_0(t/2, \delta/2)$ is a difference merging network. Thus $\mathbf{g}^{(t/2)}$ is step if each of $\mathbf{x}_{\mathrm{e}}^{(t/4)}$ and $\mathbf{y}_{\mathrm{e}}^{(t/4)}$ are step and $0 \leq \sum(\mathbf{x}_{\mathrm{e}}^{(t/4)}) - \sum(\mathbf{y}_{\mathrm{e}}^{(t/4)}) \leq \delta/2$. Since each of $\mathbf{x}^{(t/2)}$ and $\mathbf{y}^{(t/2)}$ is step, Lemma 2.1 implies that that each of the subsequences $\mathbf{x}_{\mathrm{e}}^{(t/4)}$ and $\mathbf{y}_{\mathrm{e}}^{(t/4)}$ is step too. Furthermore, since $0 \leq \sum(\mathbf{x}^{(t/2)}) - \sum(\mathbf{y}^{(t/2)}) \leq \delta$, Lemma 2.4 implies that that $0 \leq \sum(\mathbf{x}_{\mathrm{e}}^{(t/4)}) - \sum(\mathbf{y}_{\mathrm{e}}^{(t/4)}) \leq \delta/2$. Hence, the sequence $\mathbf{g}^{(t/2)}$ is step.

In an exactly similar way, we can prove that the sequence $\mathbf{h}^{(t/2)}$ is step.

Now, we show that $0 \leq \sum(\mathbf{g}^{(t/2)}) - \sum(\mathbf{h}^{(t/2)}) \leq 2$. By the sum preservation property of networks $\mathcal{M}_0(t/2, \delta/2)$ and $\mathcal{M}_1(t/2, \delta/2)$ we have

$$\sum\left(\mathbf{g}^{\left(\frac{t}{2}\right)}\right) = \sum\left(\mathbf{x}_{\mathrm{e}}^{\left(\frac{t}{4}\right)}\right) + \sum\left(\mathbf{y}_{\mathrm{e}}^{\left(\frac{t}{4}\right)}\right),$$

and

$$\sum\left(\mathbf{h}^{\left(\frac{t}{2}\right)}\right) = \sum\left(\mathbf{x}_{\mathrm{o}}^{\left(\frac{t}{4}\right)}\right) + \sum\left(\mathbf{y}_{\mathrm{o}}^{\left(\frac{t}{4}\right)}\right).$$

From Lemma 2.3,

$$0 \leq \sum\left(\mathbf{x}_{\mathrm{e}}^{\left(\frac{t}{4}\right)}\right) - \sum\left(\mathbf{x}_{\mathrm{o}}^{\left(\frac{t}{4}\right)}\right) \leq 1,$$

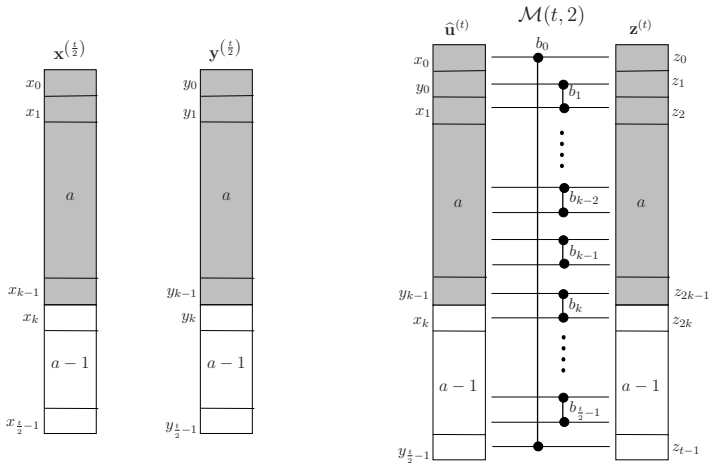
and

$$0 \leq \sum\left(\mathbf{y}_{\mathrm{e}}^{\left(\frac{t}{4}\right)}\right) - \sum\left(\mathbf{y}_{\mathrm{o}}^{\left(\frac{t}{4}\right)}\right) \leq 1.$$

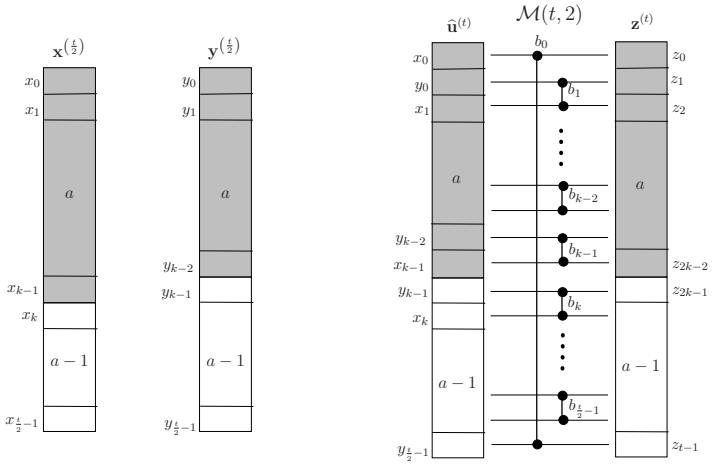
Adding these two inequalities we get

$$0 \leq \left(\sum\left(\mathbf{x}_{\mathrm{e}}^{\left(\frac{t}{4}\right)}\right) + \sum\left(\mathbf{y}_{\mathrm{e}}^{\left(\frac{t}{4}\right)}\right)\right) - \left(\sum\left(\mathbf{x}_{\mathrm{o}}^{\left(\frac{t}{4}\right)}\right) + \sum\left(\mathbf{y}_{\mathrm{o}}^{\left(\frac{t}{4}\right)}\right)\right) \leq 2,$$

(a): $k = l$



(b): $k = l + 1$



(c): $k = l + 2$

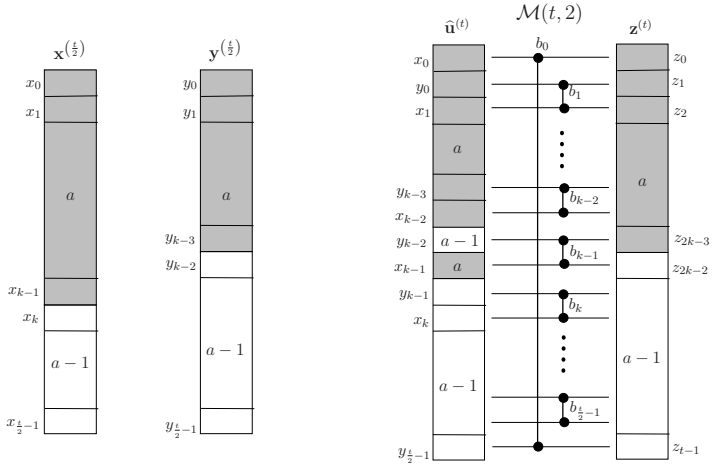
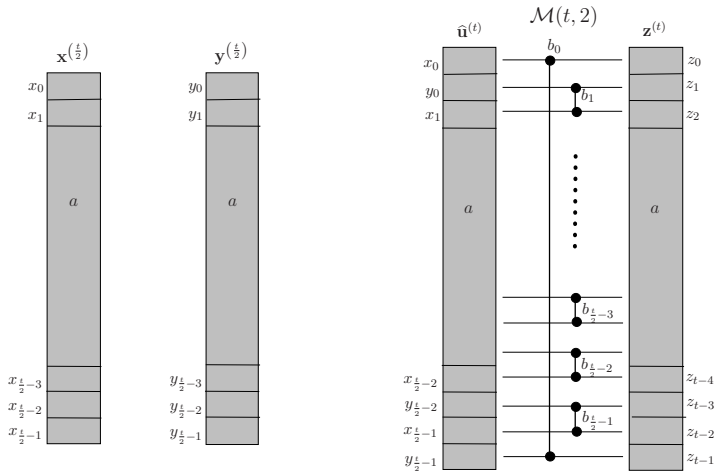
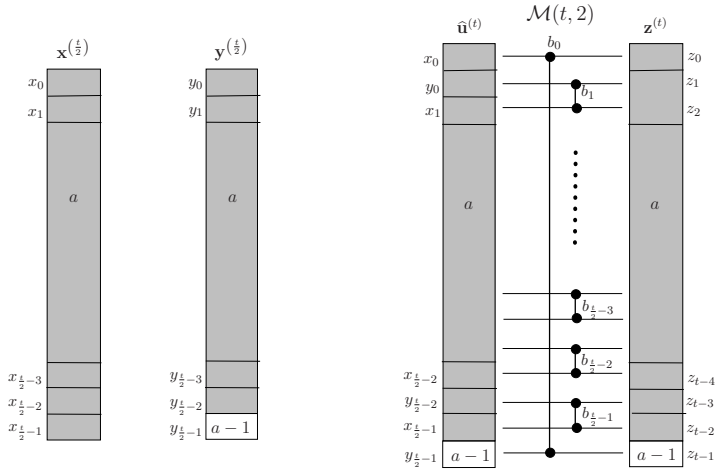


Figure 6: The case $a = b$ and $k < t/2$

(a): $k = l = \frac{t}{2}$



(b): $k = l + 1 = \frac{t}{2}$



(c): $k = l + 2 = \frac{t}{2}$

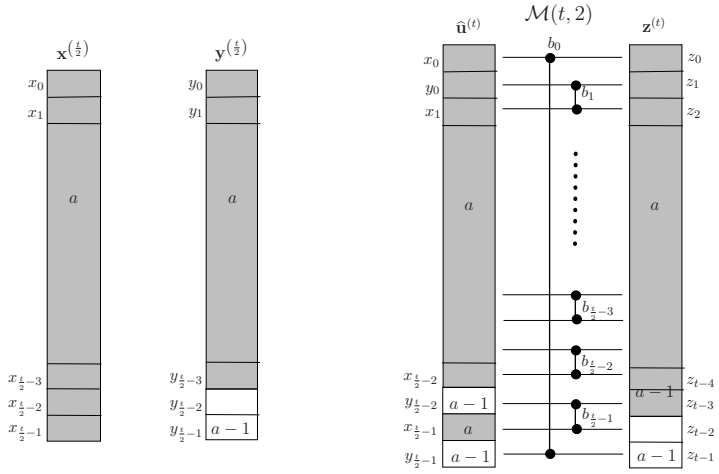


Figure 7: The case $a = b$ and $k = t/2$

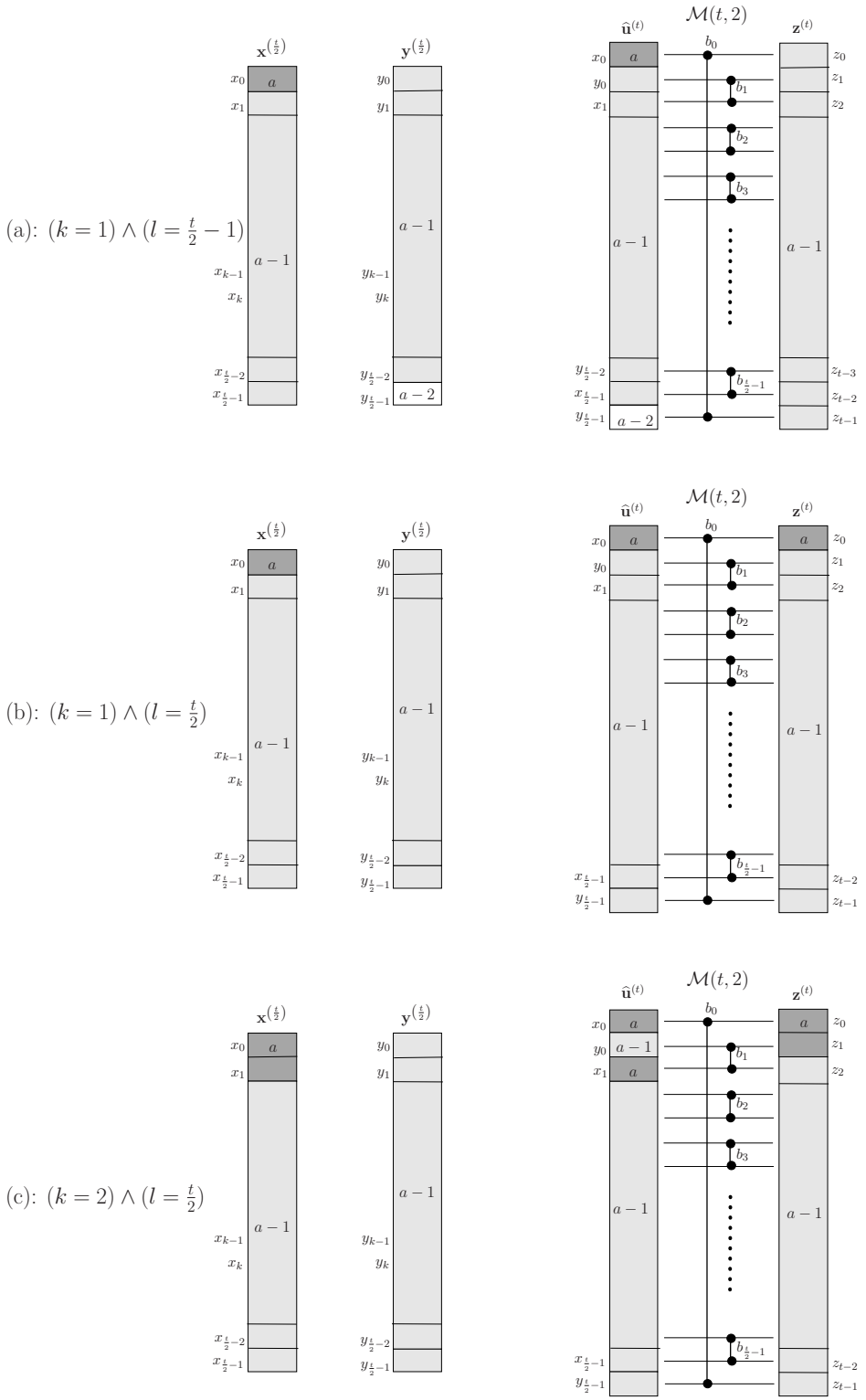


Figure 8: The case $a = b + 1$

which implies that

$$0 \leq \sum \left(\mathbf{g}^{(\frac{t}{2})} \right) - \sum \left(\mathbf{h}^{(\frac{t}{2})} \right) \leq 2.$$

Since each of $\mathbf{g}^{(t/2)}$ and $\mathbf{h}^{(t/2)}$ is step, and $0 \leq \sum(\mathbf{g}^{(t/2)}) - \sum(\mathbf{h}^{(t/2)}) \leq 2$, the output sequence $\mathbf{z}^{(t)}$ is also step, as needed. ∎

4 Counting Network

We present the construction of counting network $\mathcal{C}(w,t)$, that has input width w and output width t, where $w = 2^k$, $t = p2^l$, $p \geq 1$, and $1 \leq k \leq l$ (such assignments to parameters w and t will be called *valid*).

4.1 Construction of $\mathcal{C}(w,t)$

In the construction of $\mathcal{C}(w,t)$, we will use as a building block the *ladder* network $\mathcal{L}(w)$ (see Figure 9). The network $\mathcal{L}(w)$ is a balancing network of input and output width w, that consists of a single layer of $w/2$ copies of the $(2,2)$-balancer, denoted $b_0, \ldots, b_{\frac{w}{2}-1}$. Consider a balancer b_i, where $0 \leq i \leq w/2 - 1$. The top and bottom input wires of balancer b_i are connected to input wires i and $i + w/2$, respectively, of $\mathcal{L}(w)$ (corresponds to elements i and $i + w/2$ of the input sequence of $\mathcal{L}(w)$). The top and bottom output wires of balancer b_i are connected to output wires i and $i + w/2$, respectively, of $\mathcal{L}(w)$ (corresponds to elements i and $i + w/2$ of the output sequence of $\mathcal{L}(w)$).

The construction of $\mathcal{C}(w,t)$ is by recursion on w, where t takes arbitrary valid values.

- *Recursive basis,* $w = 2$. The network $\mathcal{C}(2,t)$ is just a $(2,t)$-balancer.
- *Recursive step,* $w > 2$. For the inductive case, suppose that we have constructed the network $\mathcal{C}(w/2,t')$, for any valid value t'. We will show how to construct the network $\mathcal{C}(w,t)$. The network $\mathcal{C}(w,t)$ is constructed in two sub-steps as follows (see Figure 9).
  - *Sub-step 1.* Let $\mathbf{x}^{(w)}$ and $\mathbf{y}^{(t)}$ denote the input and output sequences, respectively, of $\mathcal{C}(w,t)$. We take a copy of the ladder network $\mathcal{L}(w)$, and two copies of $\mathcal{C}(w/2,t/2)$, denoted as $C_0(w/2,t/2)$ and $C_1(w/2,t/2)$, Let $\mathbf{e}^{(w/2)}$ and $\mathbf{g}^{(t/2)}$ be the input and output sequences, respectively, of $\mathcal{C}_0(w/2,t/2)$; let $\mathbf{f}^{(w/2)}$ and $\mathbf{h}^{(t/2)}$ be the input and output sequences, respectively, of $\mathcal{C}_1(w/2,t/2)$. The input sequence $\mathbf{x}^{(w)}$ is directly-connected to the input sequence of $\mathcal{L}(w)$. The input sequence $\mathbf{e}^{(w/2)}$ is directly connected to the first half of the output sequence of $\mathcal{L}(w)$, while input sequence $\mathbf{f}^{(w/2)}$, is directly-connected to the second half of the output sequence of $\mathcal{L}(w)$.
  - *Sub-step 2.* Take now a copy of the merging network $\mathcal{M}(t,w/2)$ (described in Section 3). The first input sequence of network $\mathcal{M}(t,w/2)$ is directly-connected to the output sequence $\mathbf{g}^{(t/2)}$, while the second input sequence of $\mathcal{M}(t,w/2)$,

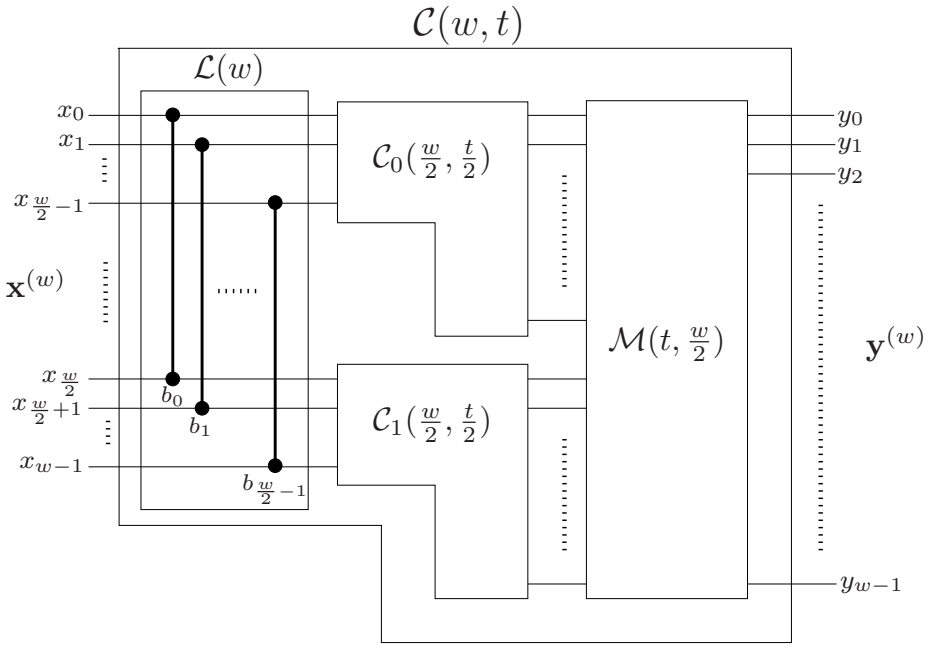


Figure 9: The counting network $\mathcal{C}(w,t)$

is directly connected to $\mathbf{h}^{(t/2)}$. The output sequence of $\mathcal{M}(t,w/2)$ is directly-connected to the output sequence $\mathbf{y}^{(t)}$ of network $\mathcal{C}(w,t)$.

Next, we calculate the depth of network $\mathcal{C}(w,t)$.

Theorem 4.1 *depth*$(\mathcal{C}(w,t)) = (\lg^2 w + \lg w)/2$.

Proof: By the recursive construction of $\mathcal{C}(w,t)$, $\text{depth}(\mathcal{C}(2,t)) = 1$, and

$$\begin{aligned}
\text{depth}\left(\mathcal{C}(w,t)\right) &= 1 + \text{depth}\left(\mathcal{C}\left(\frac{w}{2},\frac{t}{2}\right)\right) + \text{depth}\left(\mathcal{M}\left(t,\frac{w}{2}\right)\right) \\
&= 1 + \text{depth}\left(\mathcal{C}\left(\frac{w}{2},\frac{t}{2}\right)\right) + \lg\frac{w}{2} \\
&\quad \text{(by Lemma 3.1)} \\
&= 1 + \left(1 + \text{depth}\left(\mathcal{C}\left(\frac{w}{4},\frac{t}{4}\right)\right) + \text{depth}\left(\mathcal{M}\left(\frac{t}{2},\frac{w}{4}\right)\right)\right) + \lg\frac{w}{2} \\
&= 1 + \left(1 + \text{depth}\left(\mathcal{C}\left(\frac{w}{4},\frac{t}{4}\right)\right) + \lg\frac{w}{4}\right) + \lg\frac{w}{2} \\
&\quad \text{(by Lemma 3.1)} \\
&= 2 + \text{depth}\left(\mathcal{C}\left(\frac{w}{2^2},\frac{t}{2^2}\right)\right) + \sum_{i=1}^{2}\lg\frac{w}{2^i} \\
&= \ldots
\end{aligned}$$

$$
\begin{aligned}
&= k + \text{depth}\left(\mathcal{C}\left(\frac{w}{2^k}, \frac{t}{2^k}\right)\right) + \sum_{i=1}^{k} \lg \frac{w}{2^i} \\
&= k + \text{depth}\left(\mathcal{C}\left(\frac{w}{2^k}, \frac{t}{2^k}\right)\right) + k \lg w - \sum_{i=1}^{k} \lg 2^i \\
&= k + \text{depth}\left(\mathcal{C}\left(\frac{w}{2^k}, \frac{t}{2^k}\right)\right) + k \lg w - \sum_{i=1}^{k} i \\
&= k + \text{depth}\left(\mathcal{C}\left(\frac{w}{2^k}, \frac{t}{2^k}\right)\right) + k \lg w - \frac{k^2}{2} - \frac{k}{2} \\
&= \ldots \\
&= \lg w - 1 + \text{depth}\left(\mathcal{C}\left(\frac{w}{2^{\lg w - 1}}, \frac{t}{2^{\lg w - 1}}\right)\right) + (\lg w - 1) \lg w \\
&\quad - \frac{(\lg w - 1)^2}{2} - \frac{\lg w - 1}{2} \\
&= \lg w - 1 + \text{depth}\left(\mathcal{C}\left(2, \frac{2t}{w}\right)\right) + \lg^2 w - \lg w \\
&\quad - \frac{\lg^2 w - 2 \lg w + 1}{2} - \frac{\lg w - 1}{2} \\
&= -1 + 1 + \lg^2 w - \frac{\lg^2 w - \lg w}{2} \\
&\quad \left(\text{since depth}\left(\mathcal{C}\left(2, \tfrac{2t}{w}\right)\right) = 1\right) \\
&= \frac{\lg^2 w + \lg w}{2},
\end{aligned}
$$

as needed. ∎

4.2 Correctness of $\mathcal{C}(w, t)$

We show the correctness of network $\mathcal{C}(w, t)$.

Theorem 4.2 *$\mathcal{C}(w, t)$ is a counting network.*

Proof: Suppose that $\mathcal{C}(w, t)$ is quiescent. We prove that for any values on the input sequence $\mathbf{x}^{(w)}$ to network $\mathcal{C}(w, t)$, the output sequence $\mathbf{y}^{(t)}$ is step. We will prove the correctness of $\mathcal{C}(w, t)$ by induction on w.

For the basis case $w = 2$, the network $\mathcal{C}(2, t)$ is just a $(2, t)$-balancer, which is a counting network by definition.

For $w > 2$, we will assume that the network $\mathcal{C}(w/2, t')$ is counting (induction hypothesis), for any valid t', and we will show that the network $\mathcal{C}(w, t)$ is counting too, for any valid t.

Consider the construction of $\mathcal{C}(w, t)$. By the induction hypothesis, we have that the respective outputs $\mathbf{g}^{(t/2)}$ and $\mathbf{h}^{(t/2)}$ of $\mathcal{C}_0(w/2, t/2)$ and $\mathcal{C}_1(w/2, t/2)$, are step. These sequences are inputs of the network $\mathcal{M}(t, w/2)$, whose output is the sequence $\mathbf{y}^{(t)}$. Since, by Lemma

3.3, $\mathcal{M}(t, w/2)$ is a difference merging network, the sequence $\mathbf{y}^{(t)}$ is step if $0 \leq \sum(\mathbf{g}^{(t/2)}) - \sum(\mathbf{h}^{(t/2)}) \leq w/2$; next we prove that this property holds.

By the sum preservation property of networks $\mathcal{C}_0$ and $\mathcal{C}_1$,

$$\sum\left(\mathbf{e}^{\left(\frac{w}{2}\right)}\right) = \sum\left(\mathbf{g}^{\left(\frac{t}{2}\right)}\right),$$

and

$$\sum\left(\mathbf{f}^{\left(\frac{w}{2}\right)}\right) = \sum\left(\mathbf{h}^{\left(\frac{t}{2}\right)}\right).$$

Thus, we only need to prove that $0 \leq \sum(\mathbf{e}^{(w/2)}) - \sum(\mathbf{f}^{(w/2)}) \leq w/2$.

The sequences $\mathbf{e}^{(w/2)}$ and $\mathbf{f}^{(w/2)}$ are connected to the outputs of the $(2,2)$-balancers $b_0, \ldots b_{w/2-1}$ of the ladder network $\mathcal{L}(w)$, so that the first output wire of b_i is connected to e_i and the second to f_i, for all $0 \leq i \leq w/2-1$. Since the outputs of balancer b_i have the step property for any input sequence $\mathbf{x}^{(w)}$, $0 \leq e_i - f_i \leq 1$, for all i, where $0 \leq i < w/2$. By summing these inequalities for all the $w/2$ balancers, we obtain that

$$0 \leq \left(e_0 + \cdots + e_{\frac{w}{2}-1}\right) - \left(f_0 + \cdots + f_{\frac{w}{2}-1}\right) \leq \frac{w}{2}$$

which implies that

$$0 \leq \sum\left(\mathbf{e}^{\left(\frac{w}{2}\right)}\right) - \sum\left(\mathbf{f}^{\left(\frac{w}{2}\right)}\right) \leq w/2.$$

Consequently, the sequence $\mathbf{y}^{(t)}$ is step, as needed. ∎

5 Butterfly Network

We will see in Section 6 that the first $\lg w$ layers of network $\mathcal{C}(w,t)$ are isomorphic to a special network that we call *butterfly*. We give two isomorphic descriptions of the butterfly, the *forward-butterfly*, and the *backward-butterfly*. We will actually see in Section 6 that the first layers $\mathcal{C}(w,t)$ are isomorphic to a backward-butterfly. However, it is easier to analyze the contention in the forward-butterfly.

5.1 Forward-Butterfly

Here we describe the forward-butterfly, denoted $\mathcal{D}(w)$, which is a regular network of width $w = 2^k$, where $0 \leq k$.

Network $\mathcal{D}(w)$ is constructed recursively on k (see Figure 10).

- *Recursive basis* $w = 1$. $\mathcal{D}(w)$ is simply a wire.
- *Recursive step* $w > 1$. Suppose we have constructed $\mathcal{D}(w/2)$. $\mathcal{D}(w)$ is constructed by taking two copies of $\mathcal{D}(w/2)$, which we denote $\mathcal{D}_0(w/2)$ and $\mathcal{D}_1(w/2)$, and the ladder network $\mathcal{L}(w)$ (which was described in Section 4.1). The output sequence of $\mathcal{D}_0$ is directly-connected to the first half of the input sequence of $\mathcal{L}(w)$, while the output

sequence of $\mathcal{D}_0$ is directly-connected to the second half of the input sequence of $\mathcal{L}(w)$. The input sequence of $\mathcal{D}(w)$ is the concatenation of the input sequences of $\mathcal{D}_0$ and $\mathcal{D}_1$, while the output sequence of $\mathcal{D}w$ is the output sequence of $\mathcal{L}(w)$.

From the recursive construction of the forward-butterfly network, we immediately have:

Lemma 5.1 *depth*$(\mathcal{D}(w)) = \lg w$.

We next show that the forward-butterfly is $\lg w$-smoothing.

Lemma 5.2 $\mathcal{D}(w)$ *is* $\lg w$*-smoothing.*

Proof: Suppose that $\mathcal{D}(w)$ is quiescent. The proof is by induction of w.

For the basis case, where $w = 1$, the network is a wire that trivially has the 0-smooth property, and hence the 1-smooth property.

Assume that the claim holds for $w/2$. For the induction step, we will show that it holds also for w.

Recall the recursive construction of $\mathcal{D}(w)$. By the induction hypothesis, each of $\mathcal{D}_0(w/2)$ and $\mathcal{D}_1(w/2)$ is $\lg(w/2)$-smooth. Let c_0 and d_0 be the minimum and maximum values, respectively, of the output sequence of $\mathcal{D}_0$. Similarly, define c_1 and d_1 for $\mathcal{D}_1$. By the $\lg(w/2)$-smooth property we have $d_0 - c_0 \leq \lg(w/2)$, and $d_1 - c_1 \leq \lg(w/2)$.

Each balancer b_i at the final layer ($\mathcal{L}(w)$) of $\mathcal{D}(w)$ receives one input from $\mathcal{D}_0$ and the other from $\mathcal{D}_1$. The balancer will output the minimum value when it receives the minimum values in its inputs which are c_0 and c_1. In this case, the minimum value on the output of the balancer will appear on the bottom output wire and is given by $c = \lceil (c_0 + c_1 - 1)/2 \rceil$. Similarly, the maximum value is $d = \lceil (d_0 + d_1)/2 \rceil$. Hence,

$$\begin{aligned} d &= \left\lceil \frac{d_0 + d_1}{2} \right\rceil \\ &\leq \left\lceil \frac{c_0 + c_1 + 2\lg \frac{w}{2}}{2} \right\rceil \\ &= \left\lceil \frac{c_0 + c_1}{2} \right\rceil + \lg \frac{w}{2}. \end{aligned}$$

Clearly,

$$\left\lceil \frac{c_0 + c_1}{2} \right\rceil - \left\lceil \frac{c_0 + c_1 - 1}{2} \right\rceil \leq 1.$$

which implies that

$$c \geq \left\lceil \frac{c_0 + c_1}{2} \right\rceil - 1.$$

Therefore,

$$\begin{aligned} d - c &\leq \left\lceil \frac{c_0 + c_1}{2} \right\rceil + \lg \frac{w}{2} - \left\lceil \frac{c_0 + c_1}{2} \right\rceil + 1 \\ &= \lg \frac{w}{2} + 1 \\ &= \lg w, \end{aligned}$$

as needed. ∎

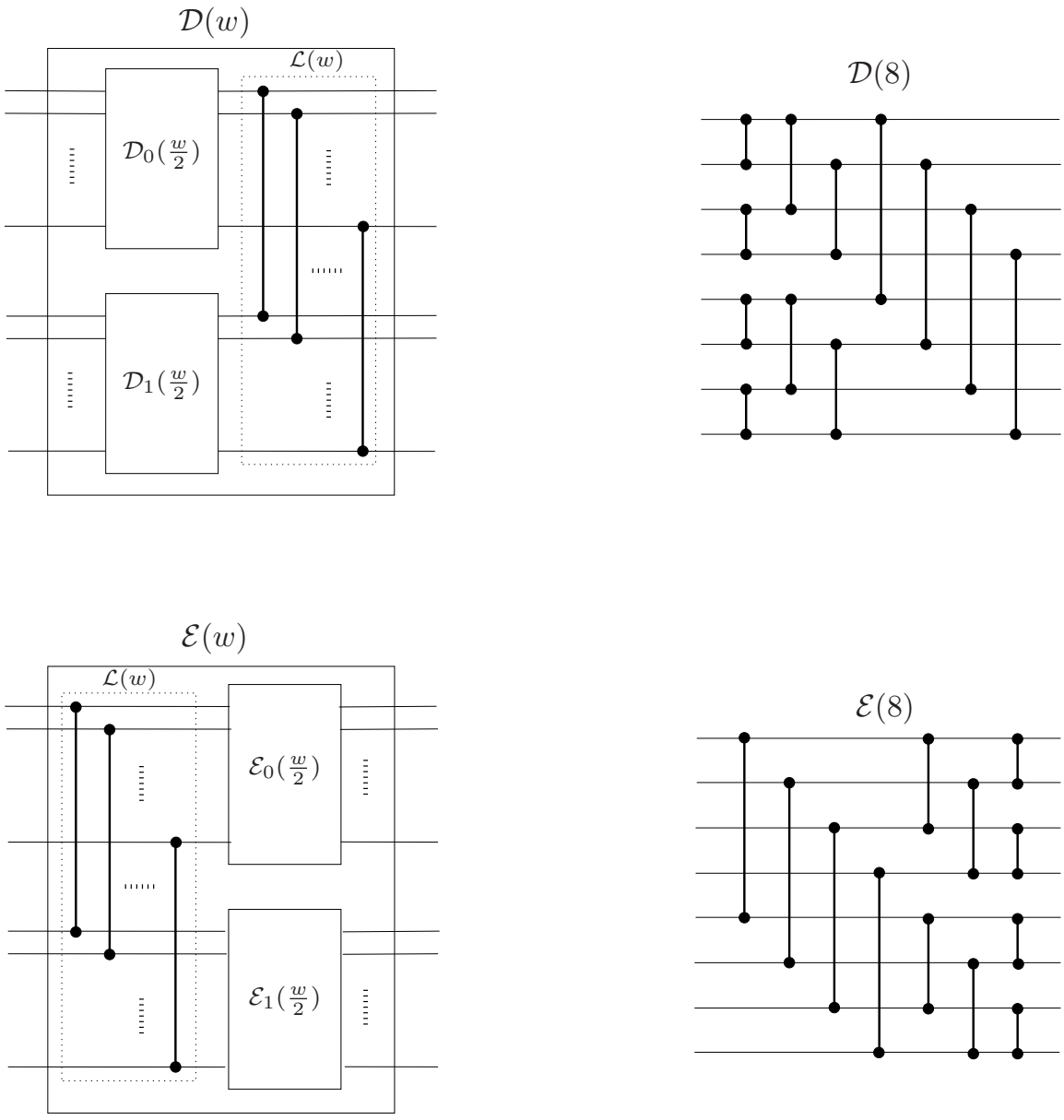


Figure 10: Top: the forward-butterfly $\mathcal{D}(w)$, and instance $\mathcal{D}(8)$; Bottom: the backward-butterfly $\mathcal{E}(w)$, and instance $\mathcal{E}(8)$

5.2 Backward-Butterfly

Here, we describe the *backward-butterfly* network, whose construction is similar to the forward-butterfly network, with the only difference that the ladder network appears in the front.

A backward-butterfly network $\mathcal{E}(w)$ is a regular network of width w, where $w = 2^k$ and $0 \leq k$.

The network is constructed recursively on w (see Figure 10).

- *Recursion basis,* $w = 1$. $\mathcal{E}(w)$ is simply a wire.
- *Recursion step,* $w > 2$. Assume that we have constructed $\mathcal{E}(w/2)$. The network $\mathcal{E}(w)$ is constructed by taking two copies of $\mathcal{E}(\frac{w}{2})$, which we denote $\mathcal{E}_0(w/2)$ and $\mathcal{E}_1(w/2)$, and the ladder network $\mathcal{L}(w)$. The input sequence of $\mathcal{E}(w)$ is directly connected to the input sequence of $\mathcal{L}(w)$. The input sequence of $\mathcal{E}_0$ is directly-connected to the first half of the output sequence of $\mathcal{L}(w)$, while the input sequence of $\mathcal{E}_0$ is directly-connected to the second half of the output sequence of $\mathcal{L}(w)$. The first half and second half of the output sequence of $\mathcal{E}(w)$ is directly connected to the output sequences of $\mathcal{E}_0$ and $\mathcal{E}_1$, respectively.

Next, we show that the backward-butterfly and the forward-butterfly are isomorphic.

Lemma 5.3 *The backward-butterfly $\mathcal{E}(w)$ is isomorphic to the forward-butterfly $\mathcal{D}(w)$.*

Proof: We will prove the claim by induction on w. For the induction basis, we consider the cases $w = 1$ and $w = 2$. For $w = 1$, $\mathcal{E}(w)$ is simply a wire which is trivially a forward-butterfly. For $w = 2$, $\mathcal{E}(w)$ is a $(2, 2)$-balancer, which is trivially a forward-butterfly.

Suppose now that the claim holds for all w' such that $1 \leq w' < w$. We will show that the claim holds for w, that is, $\mathcal{E}(w)$ and $\mathcal{D}(w)$ are isomorphic.

We will prove the claim by transforming $\mathcal{E}(w)$ to $\mathcal{D}(w)$ to intermediate isomorphic networks where in the last step of the transformation the resulting network is $\mathcal{D}(w)$. The transformation is depicted in Figure 11.

By construction, $\mathcal{E}(w)$ consists of two identical backward-butterflies $\mathcal{E}_0(w/2)$ and $\mathcal{E}_1(w/2)$, and the ladder network $\mathcal{L}(w)$ (Figure 11.a). By the induction hypothesis, $\mathcal{E}_0(w/2)$ is isomorphic to forward-butterfly $\mathcal{D}_0(w/2)$. As illustrated in Figure 11.b, in the construction of $\mathcal{E}(w)$, we replace $\mathcal{E}_0$ with $\mathcal{D}_0$, by an adding an appropriate input permutation π_{in} and output permutation π_{out} (which are the input and output permutations specified by condition (iii) of the isomorphism which is defined in Section 2.4). Similarly $\mathcal{E}_1(w/2)$ is substituted with $\mathcal{D}_1(w/2)$.

Input wire i of $\mathcal{D}_0$ corresponds to input wire $j = \pi_{\text{in}}^R(i)$ of $\mathcal{E}_0$. Similarly, input wire i of $\mathcal{D}_1$ corresponds to input wire $j = \pi_{\text{in}}^R(i)$ of $\mathcal{E}_1$. Let b_j be the balancer of $\mathcal{L}$ that connects the jth input of $\mathcal{E}_0$ and $\mathcal{E}_1$. We have that b_j connects the ith input wires of $\mathcal{D}_0$ and $\mathcal{D}_1$. Suppose that $\mathcal{L}(w)$ consists of balancers $b_0, \ldots, b_{w/2-1}$. Let $\mathcal{L}'(w)$ be a new ladder network with balancers $b'_0, \ldots, b'_{w/2-1}$, which is isomorphic to $\mathcal{L}(w)$ so that balancer b_j corresponds

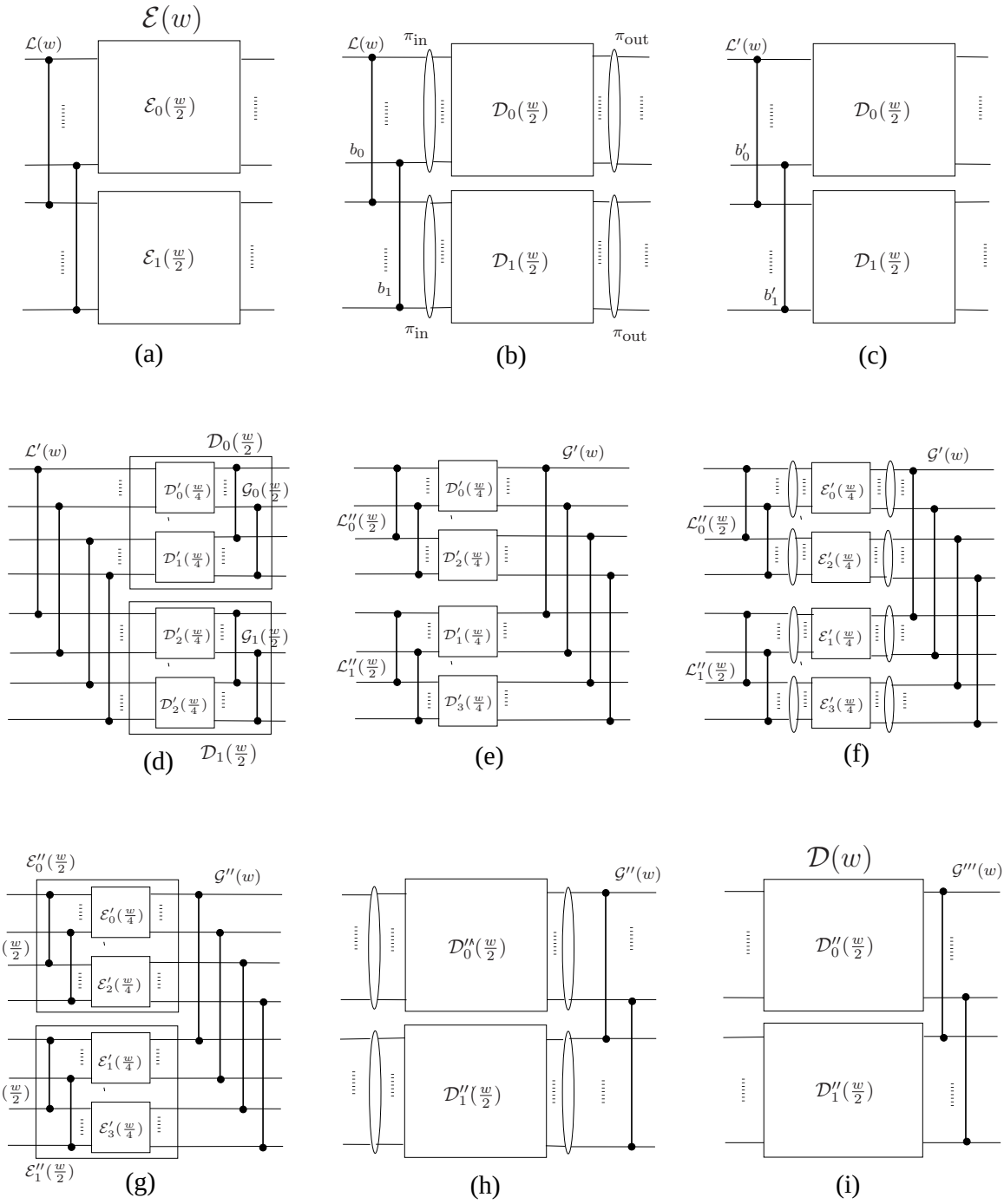


Figure 11: A series of transformations to prove that backward-butterflies are isomorphic to forward-butterflies

to balancer b'_i, where $i = \pi_{\text{in}}(j)$. Replace now the input permutation and the network $\mathcal{L}(w)$, with network $\mathcal{L}'(w)$, and remove also the output permutations π_{out} (Figure 11.c). The new network is isomorphic to $\mathcal{E}(w)$.

Since $w/2 > 1$, from the recursive construction of forward-butterflies, we have that $\mathcal{D}_0(w/2)$ consists of two copies of $\mathcal{C}(w/4)$, denoted $\mathcal{D}'_0(w/4)$ and $\mathcal{D}'_1(w/4)$, whose outputs are connected with a copy of the ladder network $\mathcal{L}(w/2)$, denoted $\mathcal{G}_0(w/4)$ (Figure 11.d). Similarly, $\mathcal{D}_1(w/2)$ consists of two forward-butterflies, $\mathcal{D}'_2(w/4)$ and $\mathcal{D}'_3(w/4)$, and a ladder network $\mathcal{G}_1$.

Next, we exchange the positions of $\mathcal{D}'_2$ and $\mathcal{D}'_3$, as shown in Figure 11.e. This exchange results to the transformation of ladder network $\mathcal{L}'w$ to two canonical layers $\mathcal{L}''_0(w/2)$ and $\mathcal{L}''_1(w/2)$, where $\mathcal{L}'_0$ connects the inputs of $\mathcal{D}'_0(w/4)$ and $\mathcal{D}'_2(w/4)$, while layer $\mathcal{L}''_1$ connects the inputs of $\mathcal{D}'_1(w/4)$ and $\mathcal{D}'_3(w/4)$. Further, the exchange of $\mathcal{D}'_2(w/4)$ and $\mathcal{D}'_3(w/4)$ results to the combination of $\mathcal{G}_0(w/2)$ and $\mathcal{G}_1(w/2)$ to a new ladder network of width w that we denote as $\mathcal{G}'(w)$. The whole new network is isomorphic to $\mathcal{E}(w)$.

By the induction hypothesis, $\mathcal{D}'_0(w/4), \ldots, \mathcal{D}'_4(w/4)$ are isomorphic to some backward-butterflies, say $\mathcal{E}'_0(w/4), \ldots, \mathcal{E}'_0(w/4)$, respectively. We replace each $\mathcal{D}'_i$ with the respective $\mathcal{E}'_i$, using appropriate input and output permutations (Figure 11.f). After we remove the input and output permutations, we obtain the network of Figure 11.g), where input ladder networks $\mathcal{L}''_0$ and $\mathcal{L}''_1$ are translated to $\mathcal{L}'''_0$ and $\mathcal{L}'''_1$, respectively, and the output ladder network $\mathcal{G}'$ is translated to $\mathcal{G}''$.

The combination of $\mathcal{L}'''_0$, $\mathcal{E}'_0$, and $\mathcal{E}'_2$ forms a backward-butterfly that we denote as $\mathcal{E}''_0(w/2)$ (Figure 11.g). Similarly, the combination of $\mathcal{L}'''_1$, $\mathcal{E}'_1$, and $\mathcal{E}'_3$ form a backward-butterfly $\mathcal{E}''_1(w/2)$. By the induction hypothesis, $\mathcal{E}''_0(w/2)$ and $\mathcal{E}''_1(w/2)$ are isomorphic to some forward-butterflies, say $\mathcal{D}''_0(w/2)$ and $\mathcal{D}''_1(w/2)$, respectively. We replace $\mathcal{E}''_0$ with $\mathcal{D}''_0$ and $\mathcal{E}''_1$ with $\mathcal{D}''_1$, respectively, with appropriate input and output permutations (Figure 11.h). After we remove the input and output permutations, we obtain a new output ladder network $\mathcal{G}'''(w)$ (Figure 11.i). The resulting network is a forward-butterfly $\mathcal{D}(w)$ of width w which is isomorphic to $\mathcal{D}(w)$. ∎

6 Contention Analysis

6.1 Methodology

We derive a general formula for computing the amortized contention of any layer of a balancing network. Parts of the following discussion are adapted from [9, Section 3.2].

The contention on $\mathcal{B}$ will be measured as the sum of the contention caused in its layers. Let ℓ be a layer of a balancing network $\mathcal{B}$. Assume that layer ℓ is made of balancers of output width at most q. Denote by w the output width of ℓ. Assume that in any quiescent state of $\mathcal{B}$, the output of ℓ has the k-smooth property. (In [9], ℓ is 1-smooth). Recall that the concurrency of $\mathcal{B}$ be n.

Dwork *et al.* [9] have introduced the following methodology for analyzing and bounding the amortized contention of ℓ:

1. Partition the set of tokens that traverse ℓ into equally sized groups of size w, called *generations*.

2. Calculate the average number of stalls occurring between tokens in different generations.

3. Divide the result by the size of the generations, w.

We will show that as a group, each generation of tokens at layer ℓ causes $O(qn + qkw)$ stalls to other tokens. It then follows that, on the average, a generation receives $O(qn+qkw)$ stalls. since in the amortized analysis the stalls are distributed uniformly among the tokens. Dividing by the number of tokens in a generation, it follows that the average token passing through ℓ receives (or causes) $O(qn/w + qk)$ stalls.

We now continue with the details of the formal proof. Let b be a balancer of ℓ with output width r, where $2 \leq r \leq q$. We say that a token belongs to the *gth generation of tokens arriving at b* if it is one of the $((g-1)r+1)$th, ...,$((g-1)r+r)$th tokens to arrive at b. Note that each generation of b has r tokens. The gth generation of ℓ is the set of gth generation tokens of the balancers at layer ℓ. We say that by time t, the *gth generation has completed its arrival at ℓ* if for each balancer in ℓ, all the tokens of the gth generation have already entered the network by that time. Finally, we say that at time t there are f tokens of the gth generation missing at layer ℓ if by time t exactly $w - f$ tokens of generation g have arrived at ℓ.

The following two results are adaptations of [9, Fact 1] and [9, Fact 2] for the case where ℓ has the k-smooth property (instead of the 1-smooth property that is considered in [9]).

Lemma 6.1 *Let ℓ be in a quiescent state, and let g be the maximum generation such that some balancer b in ℓ has received at least one token of generation g. Then all balancers in ℓ have received at least one token of generation $g - k$.*

Proof: Since balancer b has received a generation g token, the maximum value in the output sequence of b is at least g. Assume, for contradiction, that there is a balancer b' that has received no generation $g - k$ token. The maximum value on the output sequence of b' is at most $g - k - 1$. So, there is an output wire of b and an output wire of b' with difference at least $g - (g - k - 1) = k + 1$. This is a contradiction, since the output sequence of ℓ is k-smooth. ∎

Lemma 6.2 *Let t be the time at which the first g generation token arrives at ℓ. Then the number of tokens of generations strictly less than $g - k$ stuck (entered but not exited) at ℓ plus the number of tokens of generations strictly less than $g - k$ still missing from layer ℓ is at most n.*

Proof: Run the network $\mathcal{B}$ to quiescence from its state at time t. Let g' be the maximum generation such that some balancer in layer ℓ has received at least one generation g' token. Clearly $g' \geq g$. By Lemma 6.1, every balancer has received at least one token from generation $g' - k \geq g - k$. Thus, the claim follows since at most n tokens (the maximum number of tokens in $\mathcal{B}$ at any time) were involved in moving $\mathcal{B}$ to a quiescent state. ∎

Recall that when a token passes through a balancer, it causes stalls to all tokens that are waiting at the balancer. By *stalls caused at layer ℓ by generation g to generation g',* we refer to stalls incurred by tokens of generation g' when they are waiting at some balancer of layer ℓ and some token of generation g passes. By *stalls caused at layer ℓ between generation g and generation g',* we refer to stalls caused by generation g to generation g' and vice versa.

The following result is an adaptation of [9, Lemma 3.2.4] for the case where ℓ has the k-smooth property (instead of the 1-smooth property that is considered in [9]).

Lemma 6.3 *Consider the g generation passing through layer ℓ. The maximum number of stalls caused between generation g and generations less than or equal to g at this layer is at most $qn + q(k+1)w$.*

Proof: Suppose that the first token of generation g arrives at ℓ at time t. A generation g token can encounter (and hence cause a stall to or be stalled by)

(1) tokens of generation strictly less than $g - k$,

(2) generation $g - k, \ldots, g$ tokens.

By Lemma 6.2, the total number of tokens of generation strictly less than $g - k$ stuck at ℓ or missing from ℓ is at most n. Therefore, the type (1) tokens are at most n. The tokens of type (2) are at most $(k+1)w$, since each generation has w tokens.

The number of stalls occurring between each token of generation g and tokens of generation less than or equal to g are at most the number of tokens of these generations that this token encounters at the balancer is passes through. Each token of generation less than or equal to g can be encountered by at most q tokens of generation g (since q is the maximum balancer width in ℓ). Therefore, we get qn stalls of type (1) and $q(k+1)w$ stalls of type (2), for a total of $qn + q(k+1)w$ stalls. ∎

Since there are w tokens at any generation g, Lemma 6.3 implies that the amortized contention endured by a token at layer ℓ is at most $(qn + q(k+1)w)/w$. Hence:

Corollary 6.4 *The amortized contention of layer ℓ is*

$$\mathit{cont}(l, n) \leq \frac{qn}{w} + q(k+1).$$

6.2 Contention of Butterfly

We now compute the contention of the forward-butterfly network, which will help to compute the contention of our counting network.

Lemma 6.5 *The amortized contention of the forward-butterfly network $\mathcal{D}(w)$ is*

$$\mathit{cont}(\mathcal{D}(w), n) < \frac{2n \lg(2w)}{w} + \lg^2 w + 3 \lg w.$$

Proof: From the construction of the network $\mathcal{D}(w)$ (see Section 5.2) we have that a token first traverses either the subnetwork $\mathcal{D}_0(w/2)$ or the subnetwork $\mathcal{D}_1(w/2)$ and then the layer $\mathcal{L}(w)$. Clearly, the concurrency of each of $\mathcal{D}_0$ and $\mathcal{D}_1$ is $n/2$, since each input wire receives tokens from at most n/w processes (the distributed system assigns each process to a particular input wire so that the processes are uniformly distributed on the input wires). Furthermore, the concurrency of layer $\mathcal{L}(w)$ is n, since all processes traverse this layer. Therefore, the amortized contention incurred by any token is equal to

$$\mathrm{cont}(\mathcal{D}(w), n) = \mathrm{cont}\left(\mathcal{D}\left(\frac{w}{2}\right), \frac{n}{2}\right) + \mathrm{cont}(\mathcal{L}(w), n).$$

By Lemma 5.2, the output sequence of layer $\mathcal{L}(w)$ has the $\lg w$-smooth property. Since layer $\mathcal{L}(w)$ is made of $(2,2)$-balancers, and the width of the layer is w, we have from Corollary 6.4 that

$$\begin{aligned} \mathrm{cont}(l(w), n) &\leq 2\frac{n}{w} + 2(\lg w + 1) \\ &= 2\left(\frac{n}{w} + \lg w + 1\right). \end{aligned}$$

For the basis case, where $w = 1$, the network $\mathcal{D}(1)$ is just a wire which trivially has amortized contention equal to its concurrency.

Denote $k = \lg w$. Then,

$$\begin{aligned} \mathrm{cont}\left(\mathcal{D}(w), n\right) &= \mathrm{cont}\left(\mathcal{D}\left(\frac{w}{2}\right), \frac{n}{2}\right) + \mathrm{cont}\left(\mathcal{L}(w), n\right) \\ &\leq \mathrm{cont}\left(\mathcal{D}\left(\frac{w}{2}\right), \frac{n}{2}\right) + 2\left(\frac{n}{w} + k + 1\right) \\ &= \mathrm{cont}\left(\mathcal{D}\left(\frac{w}{2^2}\right), \frac{n}{2^2}\right) + \mathrm{cont}\left(\mathcal{L}\left(\frac{w}{2}\right), \frac{n}{2}\right) + 2\left(\frac{n}{w} + k + 1\right) \\ &\leq \mathrm{cont}\left(\mathcal{D}\left(\frac{w}{2^2}\right), \frac{n}{2^2}\right) + 2\left(\frac{\frac{n}{2}}{\frac{w}{2}} + (k-1) + 1\right) + 2\left(\frac{n}{w} + k + 1\right) \\ &= \mathrm{cont}\left(\mathcal{D}\left(\frac{w}{2^2}\right), \frac{n}{2^2}\right) + 2\left(2\frac{n}{w} + 2k - 1 + 2\right) \\ &\quad \cdots \\ &\leq \mathrm{cont}\left(\mathcal{D}\left(\frac{w}{2^j}\right), \frac{n}{2^j}\right) + 2\left(j\frac{n}{w} + jk - \sum_{i=1}^{j-1} i + j\right) \\ &= \mathrm{cont}\left(\mathcal{D}\left(\frac{w}{2^j}\right), \frac{n}{2^j}\right) + 2\left(j\frac{n}{w} + jk - \frac{j^2 - 3j}{2}\right) \\ &\quad \cdots \\ &\leq \mathrm{cont}\left(\mathcal{D}\left(\frac{w}{2^k}\right), \frac{n}{2^k}\right) + 2\left(k\frac{n}{w} + k^2 - \frac{k^2 - 3k}{2}\right) \\ &\leq \frac{n}{w} + 2\left(k\frac{n}{w} + k^2 - \frac{k^2 - 3k}{2}\right) \\ &\quad \left(\text{since } \mathrm{cont}\left(\mathcal{D}(1), \tfrac{n}{w}\right) \leq \tfrac{n}{w}\right) \end{aligned}$$

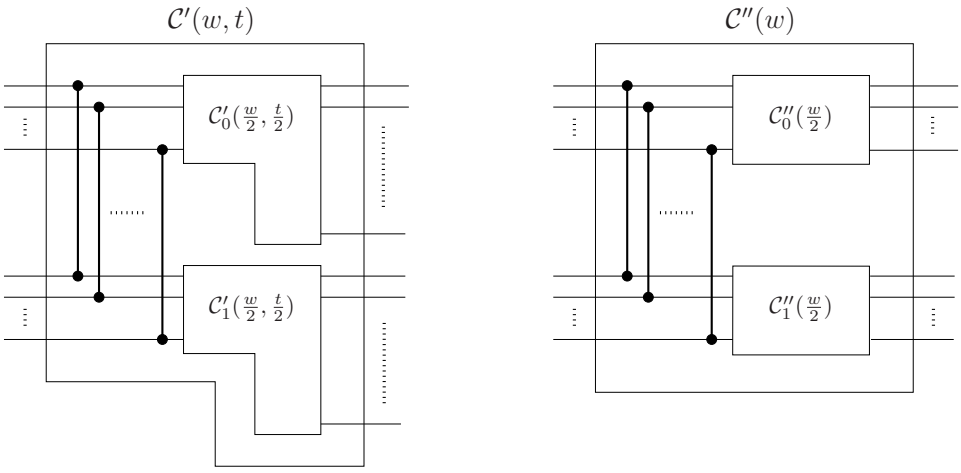


Figure 12: Left: the network $\mathcal{C}'(w,t)$ which consists of the first $\log w$ layers of $\mathcal{C}(w,t)$; Right: the network $\mathcal{C}''(w)$ in which the last layer of $\mathcal{C}''(w,t)$ is replaced with $(2,2)$-balancers

$$\begin{aligned} &= (2k+1)\frac{n}{w} + k^2 + 3k \\ &< \frac{2n\lg(2w)}{w} + \lg^2 w + 3\lg w, \end{aligned}$$

as needed. ∎

6.3 Contention of Counting Network

Here, we compute the amortized contention of the counting network $\mathcal{C}(w,t)$ (described in Section 4.1). Recall from Section 1.3.2, that the unfolded construction of the network $\mathcal{C}(w,t)$ consists of blocks $\mathcal{N}_a$, $\mathcal{N}_b$ and $\mathcal{N}_c$ (Figure 3). Let $\mathcal{N}_{a,b}$ denote the cascade of $\mathcal{N}_a$ and $\mathcal{N}_b$. We will show that $\mathcal{N}_{a,b}$ is isomorphic to a forward-butterfly. This will help to compute the contention of $\mathcal{N}_{a,b}$ and $\mathcal{N}_c$, which will give the contention of $\mathcal{C}(w,t)$.

Let $\mathcal{C}'(w,t)$ be the network $\mathcal{C}(w,t)$ without the difference-merging subnetworks in the recursive construction of $\mathcal{C}(w,t)$, as shown in the left part of Figure 12. The input width of $\mathcal{C}'(w,t)$ is w while its output width t. The construction of $\mathcal{C}'(w,t)$ resembles the recursive construction of a backward-butterfly $\mathcal{E}(w)$. The only difference is that when $w=2$, $\mathcal{C}'(w,t)$ is a $(2,2t/w)$-balancer, instead of a $(2,2)$-balancer in $\mathcal{E}(w)$. Thus, all layers of $\mathcal{C}'(w,t)$, except for the last, consist of $(2,2)$-balancers. Clearly, the depth of $\mathcal{C}'(w,t)$ is $\lg w$.

Since network $\mathcal{C}'(w,t)$ describes how the balancers of the first $\lg w$ layers of $\mathcal{C}(w,t)$ are connected, and since $\mathcal{N}_{a,b}$ consists of the first $\lg w$ layers of $\mathcal{C}(w,t)$, we have that $\mathcal{N}_{a,b}$ is isomorphic to $\mathcal{C}'(w,t)$. We have the following result.

Lemma 6.6 *$N_{a,b}$ is s-smoothing, where,*

$$s = \left\lfloor \frac{w\lg w}{t} \right\rfloor + 2.$$

Proof: Let $\mathcal{C}''(w)$ denote the network that we obtain if we replace each $(2, 2t/w)$-balancer in the last layer of $\mathcal{C}'(w,t)$ with a $(2,2)$-balancer. The recursive construction of $\mathcal{C}''(w)$ is depicted in the right part of Figure 12. Clearly, $\mathcal{C}''(w)$ is a backward-butterfly. From Lemma 5.3, the backward-butterfly is isomorphic to the forward-butterfly. Therefore, from Lemmas 2.8 and 5.2, the $\mathcal{C}''(w)$ is $\log w$-smoothing.

Let $b_0, \ldots, b_{w/2-1}$ be the $(2,2)$-balancers in the last layer of $\mathcal{C}''(w)$. Let $\mathbf{x}_i^{(2)}$ denote the output sequence of balancer b_i. Since $\mathcal{C}''(w)$ is $\lg w$-smoothing, $|\sum(\mathbf{x}_i^{(2)}) - \sum(\mathbf{x}_j^{(2)})| \leq 2\lg w$, for any indices i and j such that $0 \leq i, j < w/2$. The factor 2 in front of the term $2\lg w$ comes from the fact each $(2,2)$-balancer has two output wires.

Now, restore the $(2, 2t/w)$-balancers to the last layer of $\mathcal{C}''(w)$, in order to obtain network $\mathcal{C}'(w,t)$. Denote by $\widehat{b}_0, \ldots, \widehat{b}_{\frac{w}{2}-1}$ these balancers and by $\widehat{\mathbf{x}}_0^{(2t/w)}, \ldots, \widehat{\mathbf{x}}_{\frac{w}{2}-1}^{(2t/w)}$ their respective output sequences. Since for any balancer $\hat{b}_i$ the only difference from b_i is the number of output wires, the total sum of tokens that leave any balancer in both cases is the same. That is, $\sum(\mathbf{x}_i^{(2)}) = \sum(\widehat{\mathbf{x}}_i^{(2t/w)})$. Hence, $|\sum(\widehat{\mathbf{x}}_i^{(2t/w)}) - \sum(\widehat{\mathbf{x}}_j^{(2t/w)})| \leq 2\lg w$ for any two balancers $\hat{b}_i$ and $\hat{b}_j$.

From Lemma 2.2, the maximum values on any two output wires, one wire from balancer $\widehat{b}_i$ and the other from balancer $\widehat{b}_j$, will differ by at most $\lfloor (2\lg w)/(2t/w) \rfloor + 1$. So, the maximum difference between any two output wires is $\lfloor w \lg w/t \rfloor + 2 = s$. Consequently, $\mathcal{C}'(w,t)$ is s-smoothing.

Since $N_{a,b}$ is isomorphic to $\mathcal{C}'(w,t)$, Lemma 2.8 implies that $N_{a,b}$ is s-smoothing. ∎

We are now ready to prove an upper bound on the amortized contention of the network $\mathcal{C}(w,t)$.

Theorem 6.7 *The amortized contention of network $\mathcal{C}(w,t)$ is*

$$\mathit{cont}\,(\mathcal{C}(w,t),n) < \frac{2n\lg(2w)}{w} + \frac{n\lg^2 w}{t} + \frac{w\lg^3 w}{t} + 4\lg^2 w + 3\lg w.$$

Proof: From the construction of $\mathcal{C}(w,t)$,

$$\mathrm{cont}\,(\mathcal{C}(w,t),n) = \mathrm{cont}\,(\mathcal{N}_{a,b},n) + \mathrm{cont}\,(\mathcal{N}_c,n)\,.$$

Recall from the proof of Lemma 6.6, that the network $\mathcal{C}''(w)$ is isomorphic to the forward-butterfly network $\mathcal{E}(w)$, which implies that their amortized contention is the same (since any execution in one network has a corresponding execution in the other network). Thus, from Lemma 6.5,

$$\mathrm{cont}\,(\mathcal{C}''(w),n) < \frac{2n\lg(2w)}{w} + \lg^2 w + 3\lg w.$$

This contention remains the same even when we restore the original $(2, 2t/w)$-balancers, namely, $\mathrm{cont}(\mathcal{C}'(w,t),n) = \mathrm{cont}(\mathcal{C}''(w),n)$, since different outputs widths for a balancer do not change its contention. Since $N_{a,b}$ and $\mathcal{C}'(w)$ are isomorphic, we have $\mathrm{cont}(N_{a,b}) = \mathrm{cont}(\mathcal{C}'(w,t),n) = \mathrm{cont}(\mathcal{C}''(w),n)$.

Now, consider block $\mathcal{N}_c$. The total amortized contention of block $\mathcal{N}_c$ is equal to the contention of a layer multiplied by the number of layers in $\mathcal{N}_c$. The concurrency for every layer of $\mathcal{N}_c$ is n. From Lemma 6.6, $N_{a,b}$ is s-smoothing. Lemma 2.6 implies that each layer of $\mathcal{N}_c$ will be s-smooth. Therefore, Corollary 6.4 implies that each layer of $\mathcal{N}_c$ has amortized contention at most $2n/t + 2(s+1)$. From Theorem 4.1, we have

$$\begin{aligned} \operatorname{depth}(\mathcal{N}(c)) &= \operatorname{depth}(\mathcal{C}(w,t)) - \lg w \\ &= \frac{\lg^2 w - \lg w}{2}. \end{aligned}$$

Therefore, we have:

$$\begin{aligned} \operatorname{cont}(\mathcal{N}_c, n) &\leq \left(2\frac{n}{t} + 2(s+1)\right)\frac{\lg^2 w - \lg w}{2} \\ &\leq \frac{n\lg^2 w}{t} + s\lg^2 w + \lg^2 w \\ &\leq \frac{n\lg^2 w}{t} + \frac{w\lg^3 w}{t} + 3\lg^2 w \end{aligned}$$

By adding the amortized contentions for blocks $\mathcal{N}_{a,b}$ and $\mathcal{N}_c$ we obtain:

$$\begin{aligned} \operatorname{cont}(\mathcal{C}(w,t), n) &= \operatorname{cont}(\mathcal{N}_{a,b}, n) + \operatorname{cont}(\mathcal{N}_c, n) \\ &< \frac{2n\lg(2w)}{w} + \lg^2 w + 3\lg w + \frac{n\lg^2 w}{t} + \frac{w\lg^3 w}{t} + 3\lg^2 w \\ &= \frac{2n\lg(2w)}{w} + \frac{n\lg^2 w}{t} + \frac{w\lg^3 w}{t} + 4\lg^2 w + 3\lg w, \end{aligned}$$

as needed. ■

From Theorem 6.7, we obtain the following two results for the amortized contention of the network $\mathcal{C}(w,t)$ for the cases $t = w$ and $t = w\lg w$.

Corollary 6.8 *For $t = w$,*

$$cont(\mathcal{C}(w,t), n) = O\left(\frac{n\lg^2 w}{w} + \lg^3 w\right).$$

Corollary 6.9 *For $t = w\lg w$,*

$$cont(\mathcal{C}(w,t), n) = O\left(\frac{n\lg w}{w} + \lg^2 w\right).$$

7 Discussion

We presented a counting network construction with w input wires and t output wires, where $w = 2^k$, $t = p2^l$, and $w \leq t$. This is one of a very few known irregular counting networks constructions ([3, 13]), whose output width may be different from its input width. We showed that the irregularity can benefit the amortized contention, by bringing it to lower levels than other networks.

As a byproduct of our analysis, we obtain a novel *sorting network* construction. It is known that from any regular balancing network, we can obtain an isomorphic *comparator network* if we substitute each balancer by a comparator [5]. The isomorphic of such a counting network is a sorting network [5] (for more information on sorting networks see [12]). Our counting network $\mathcal{C}(w, w)$, where the input width is the same with the output width, gives a novel sorting network with depth $O(\lg^2 w)$.

Several interesting questions remain. Is it possible to extend our construction to arbitrary input and output widths, other than multiples of a power of two? It follows from impossibility results in [1, 6] that appropriate sets of balancer types would have to be used for such extension. Using such larger balancers is often expected to cause a reduction in depth (see [7, 8, 10, 11]). What would be a trade-off between depth and contention in this situation? Can the combinatorial techniques in [6] be used to show impossibility results on constructible widths for difference merging networks? We believe that the difference merging network we presented is of independent interest and could be used for other counting and sorting network constructions.

Acknowledgements:

We are indebted to Maurice Herlihy for providing invaluable comments on this paper.

References

[1] E. Aharonson and H. Attiya, "Counting Networks with Arbitrary Fan-Out," *Distributed Computing,* Vol. 8, pp. 163–169, 1995.

[2] W. Aiello, C. Busch, M. Herlihy, M. Mavronicolas, N. Shavit and D. Touitou, "Supporting Increment and Decrement Operations in Balancing Networks," *Chicago Journal of Theoretical Computer Science,* Vol. 2000, Article 4, December 2000.

[3] W. Aiello, R. Venkatesan and M. Yung, "Coins, Weights and Contention in Balancing Networks," *Proceedings of the 13th Annual ACM Symposium on Principles of Distributed Computing,* pp. 193–205, August 1994.

[4] M. Ajtai, J. Komlós and E. Szemerédi, "An $O(n \log n)$ Sorting Network," *Combinatorica,* Vol. 3, pp. 1–19, 1983.

[5] J. Aspnes, M. Herlihy and N. Shavit, "Counting Networks," *Journal of the ACM,* Vol. 41, No. 5, pp. 1020–1048, September 1994.

[6] C. Busch and M. Mavronicolas, "A Combinatorial Treatment of Balancing Networks," *Journal of the ACM,* Vol. 43, No. 5, pp. 749–839, September 1996.

[7] C. Busch, N. Hardavellas and M. Mavronicolas, "Contention in Counting Networks," *Proceedings of the 13th Annual ACM Symposium on Principles of Distributed Computing,* pp. 404, August 1994.

[8] C. Busch and M. Herlihy, "Sorting and Counting Networks of Arbitrary Width and Small Depth," *Theory of Computing Systems,* Vol. 35, No. 2, pp. 99–128, January 2002.

[9] C. Dwork, M. Herlihy and O. Waarts, "Contention in Shared Memory Algorithms," *Journal of the ACM,* Vol. 44, No. 6, pp. 779–805, November 1997.

[10] E. W. Felten, A. LaMarca and R. Ladner, "Building Counting Networks from Larger Balancers," Technical Report 93-04-09, Department of Computer Science and Engineering, University of Washington, April 1993.

[11] N. Hardavellas, D. Karakos and M. Mavronicolas, "Notes on Sorting and Counting Networks," *Proceedings of the 7th International Workshop on Distributed Algorithms (WDAG-93),* Lecture Notes in Computer Science, Vol. 725 (A. Schiper, ed.), Springer-Verlag, pp. 234–248, Lausanne, Switzerland, September 1993.

[12] D. Knuth, *Sorting and Searching*, Volume 3 of *The Art of Computer Programming*, Addison-Wesley, 1973.

[13] N. Shavit and A. Zemach, "Diffracting Trees," *ACM Transactions on Computer Systems,* Vol. 14, No. 4, pp. 385–428, November 1996.

[14] N. Shavit, E. Upfal and A. Zemach, "A Steady State Analysis of Diffracting Trees," *Theory of Computing Systems,* Vol. 31, No. 4, pp. 403–423, 1998.