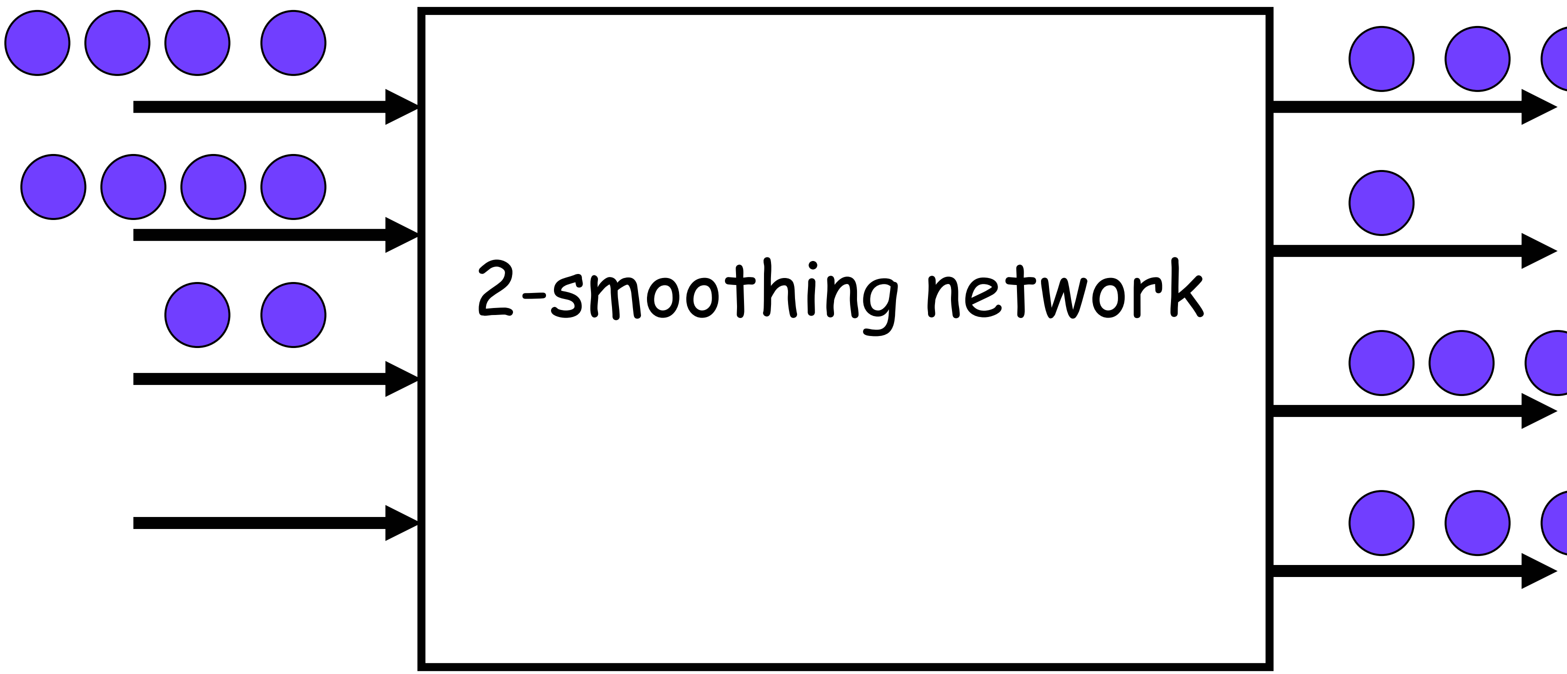
Self Stabilizing Smoothing and Counting

Maurice Herlihy, Brown University
Srikanta Tirthapura, Iowa State University

Overview

- Smoothing and Counting Networks
- Analysis of behavior without proper initialization
  - upper and lower bounds
- Self stabilization of smoothing networks

Smoothing Networks

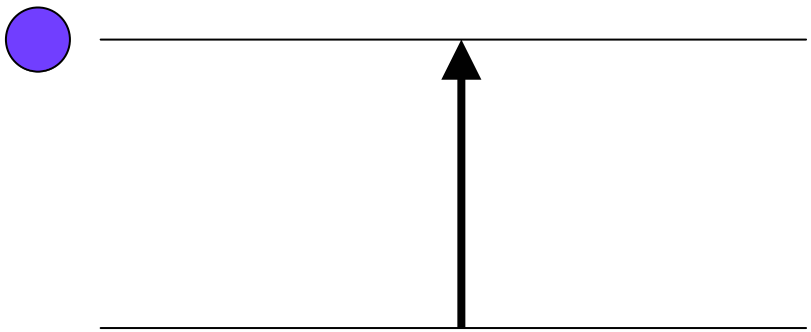


In a k-smoothing Network, the numbers of Tokens on different output wires differ by at most 2

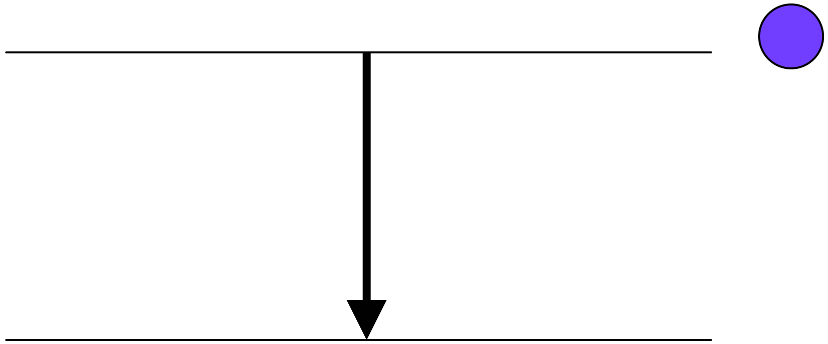
Counting Networks

- 1-smoothing networks with other additional properties
- Aspnes, Herlihy and Shavit in 1991
- Since then, scalable Construction and Properties well studied
- Bitonic and Periodic networks are two popular counting networks

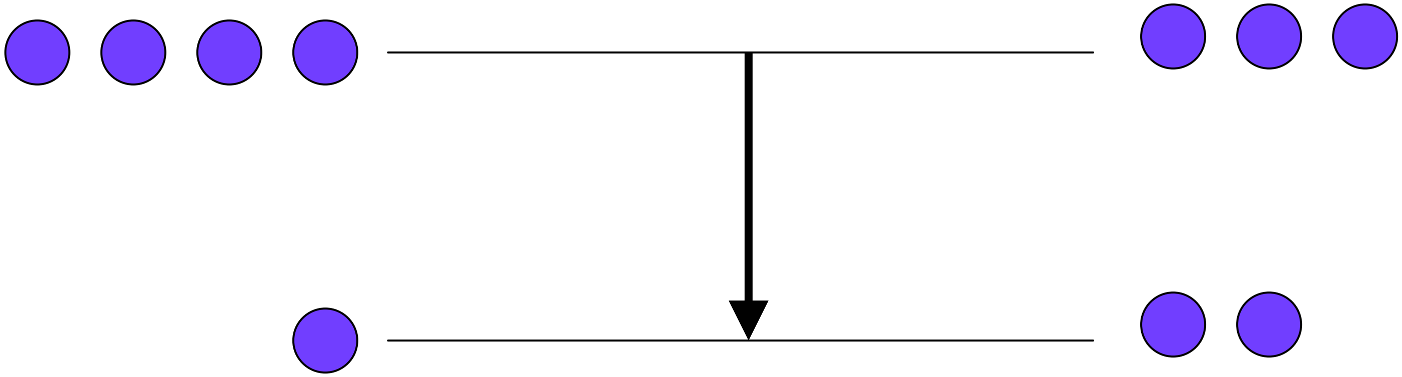
Balancer



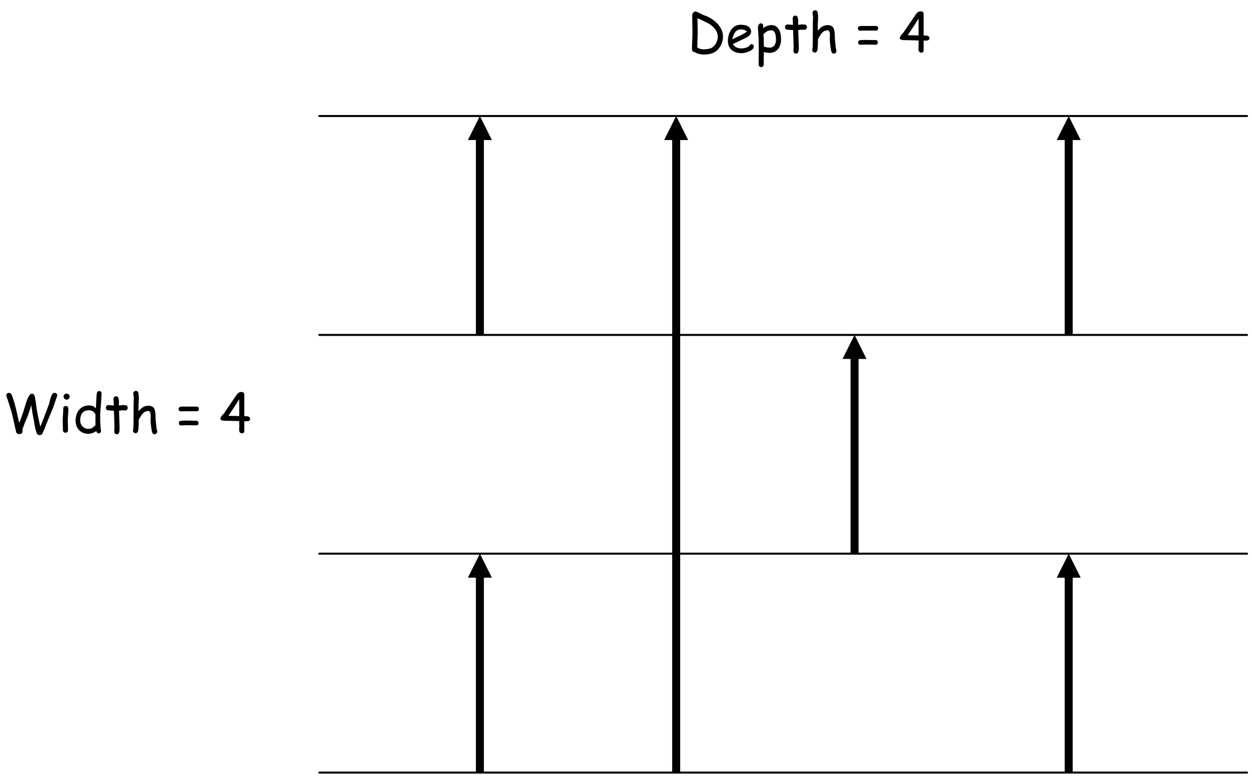
Balancer



Balancer

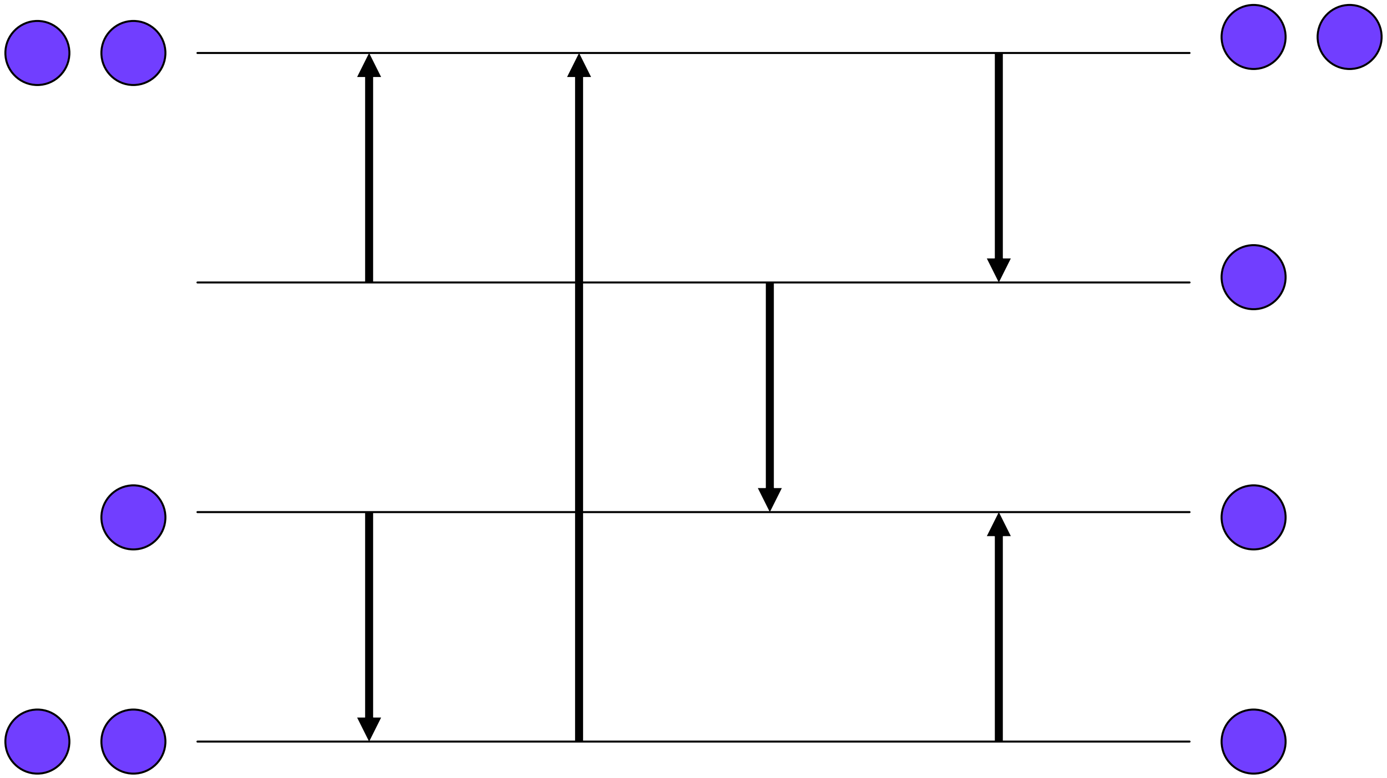


Counting Network



Initial State: All balancers pointing up

1-Smoothing Property



Questions

- How do counting networks perform when initialized incorrectly (or by an adversary)?
- How to recover from illegal states reached during execution?

Motivation

- Initializing to a "correct" global state is hard or may be impossible
  - global reconfiguration expensive
  - network switches reboot
- Step towards building fault tolerant and dynamic smoothing networks

Our Results(1)

Periodic and *Bitonic* Counting Networks:

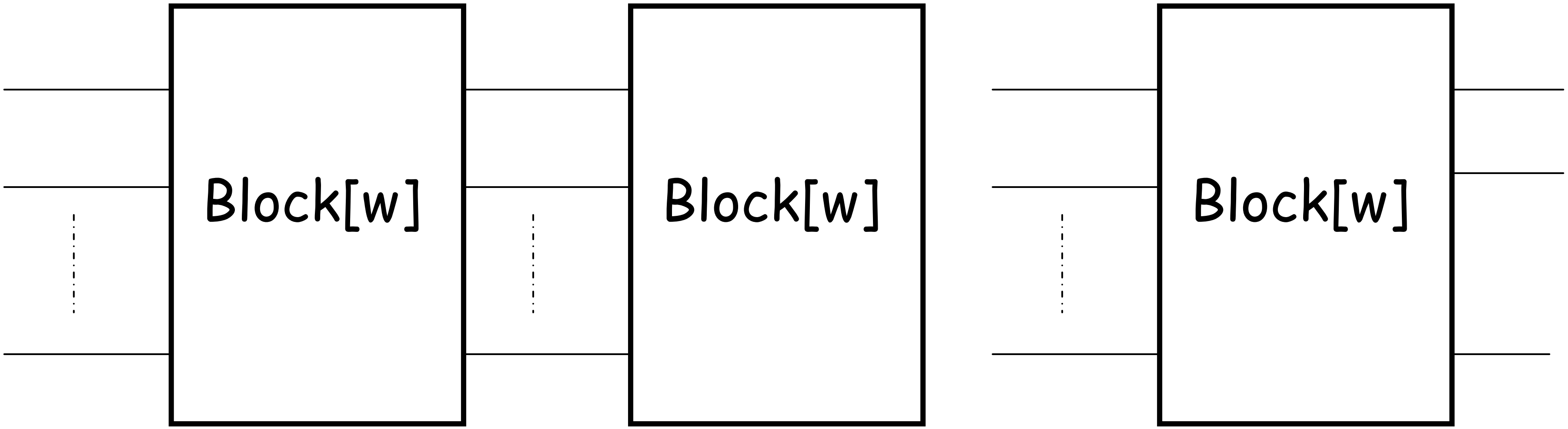
- When started from an arbitrary state, output is $\log w$ smooth (w = width of network)
- Tight lower bound: We demonstrate inputs such that the output is not $\log k$ smooth for any $k < w$

Our Results (2)

Self-stabilization of Balancing Networks

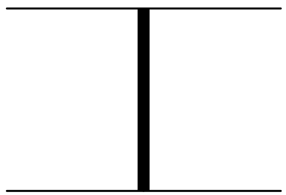
- Add extra state and actions
- If network begins in illegal state, will eventually return to a legal state
- Upper bound on the time till stabilization, and extra space required

Periodic[w] Counting Network

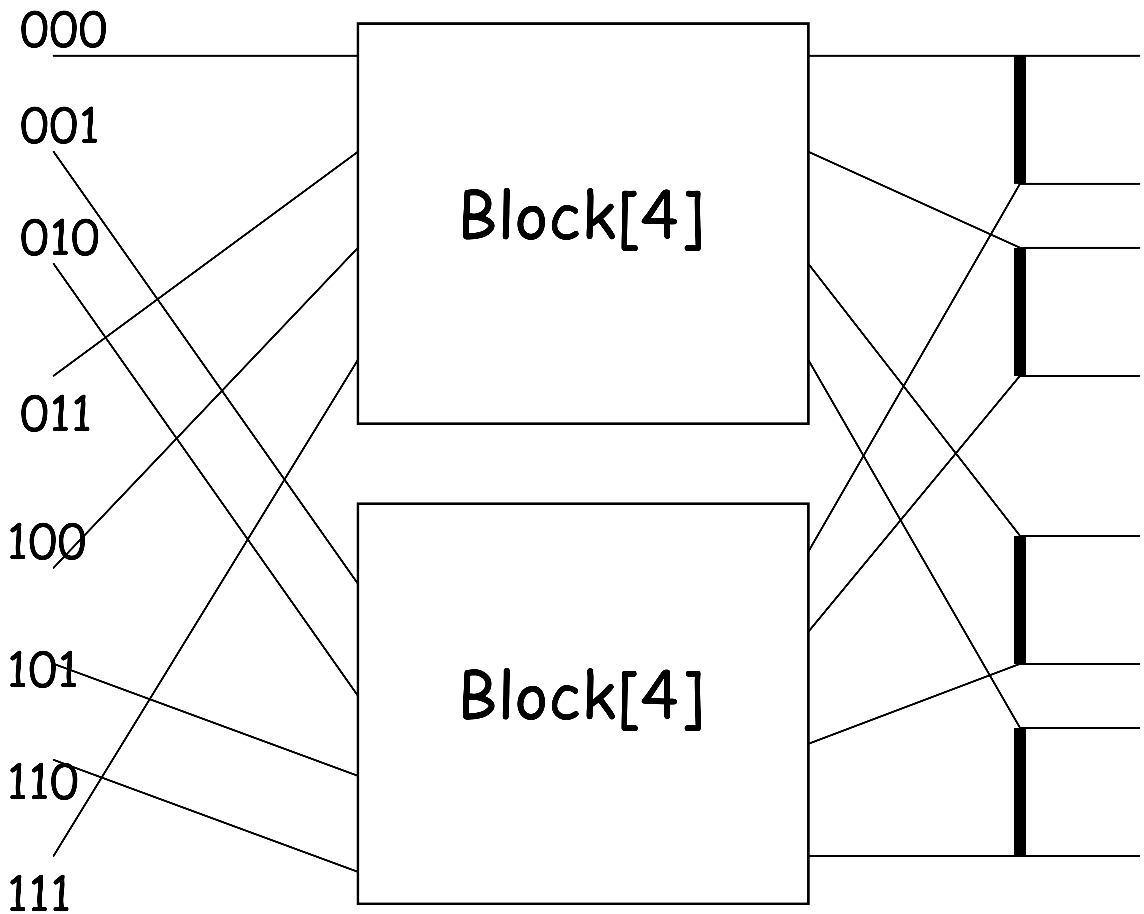


Block Network:

Inductive Definition



Block[2]



Definitions

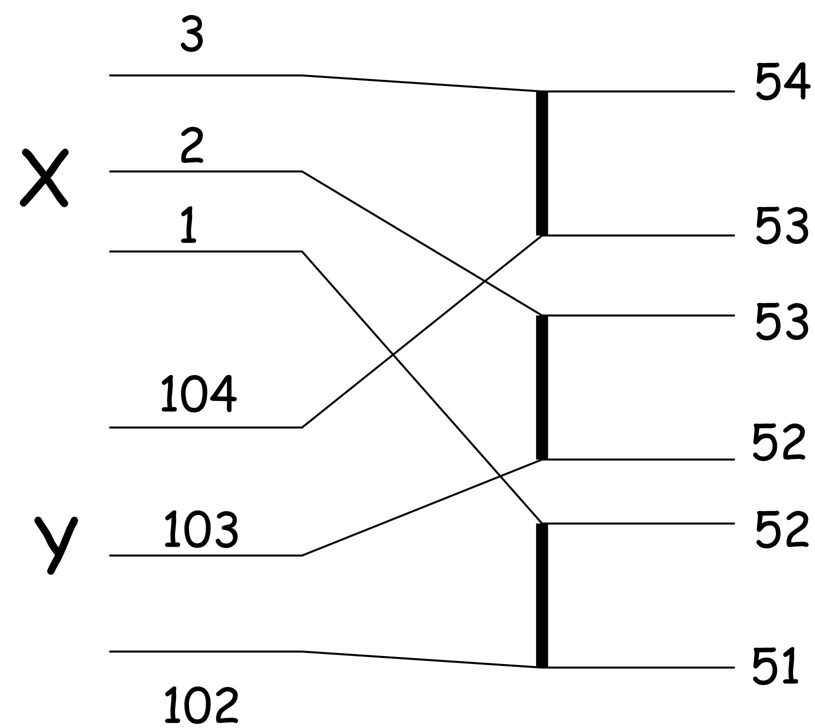
- Sequence $X = x_1 x_2 \ldots x_l$ is *k-smooth* if $| x_i - x_j | \leq k$ for all $i, j < l$

- Matching layer of balancers for sequences X and Y joins x_i and y_i in a one-to-one correspondence

Matching Lemma

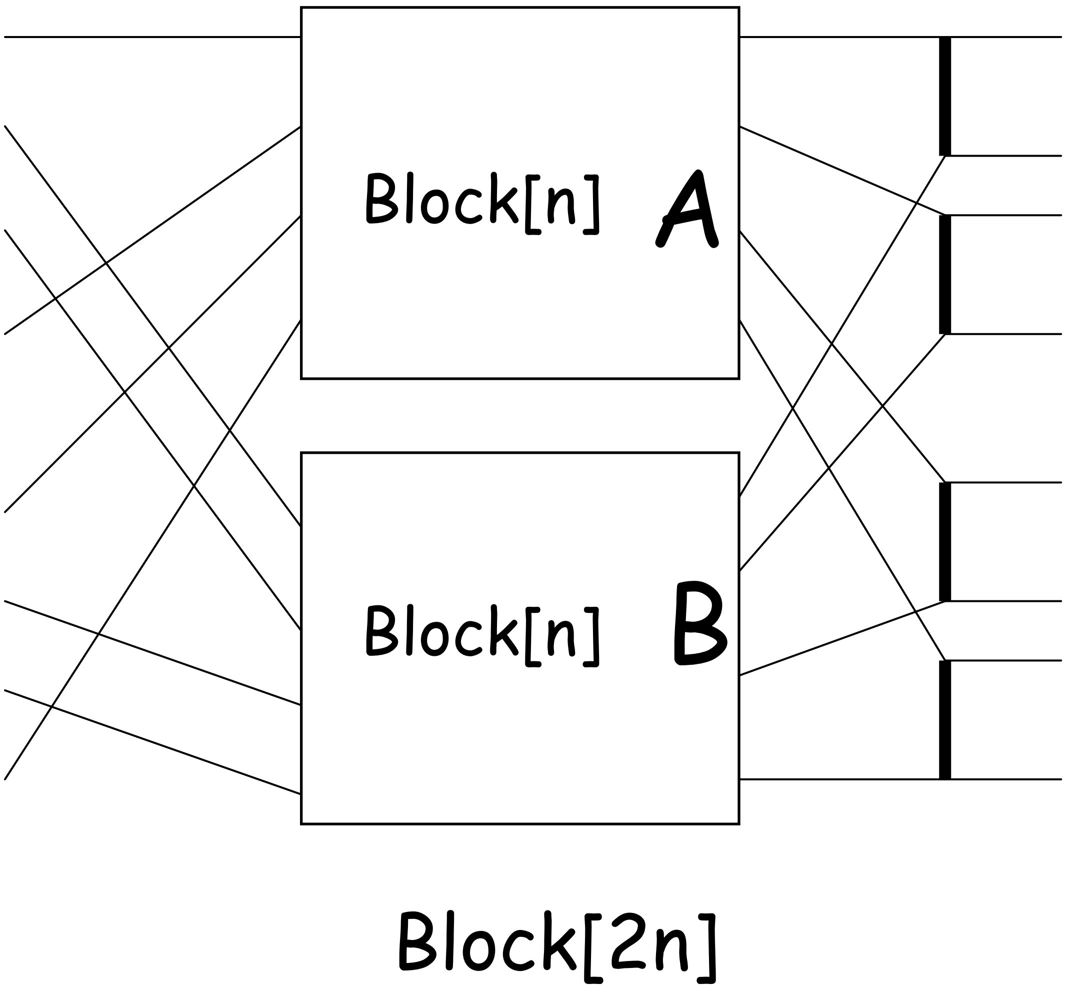
If X and Y are each k-smooth then result of matching X and Y is (k+1)-smooth

Holds irrespective of the orientations of balancers



Block[w] is (log w)-smooth

- Proof by Induction
- Assume Output of Block[n] is log n smooth
- Show that output of Block[2n] is log (2n) smooth

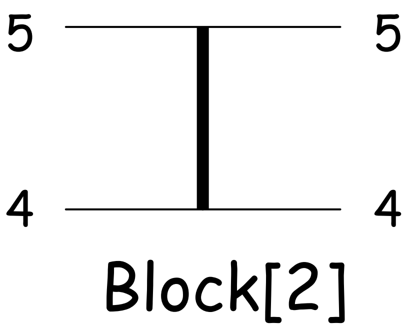


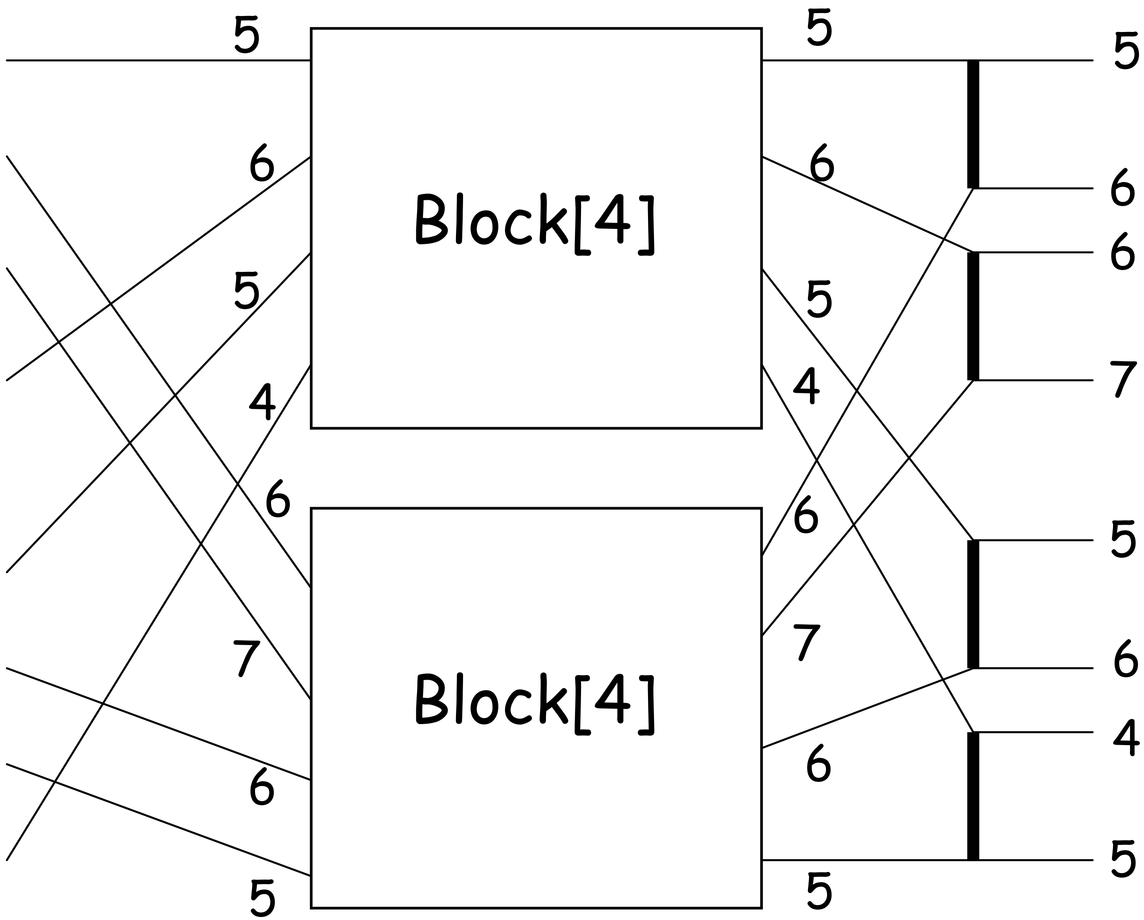
Lower Bound

- Worst Case bound:
  There exist input sequences and initial states such that output of Block[w] is not k-smooth for any k < log w

- Show a *fixed-point* sequence for Block[w] which is not k-smooth for any k < log w

Fixed Point Sequence





Sequence not k-smooth for any k < log (width)

Bitonic Counting Network

Starting from an arbitrary initial state

- Output is always log w smooth, where w=width
- Matching worst case lower bound on smoothness

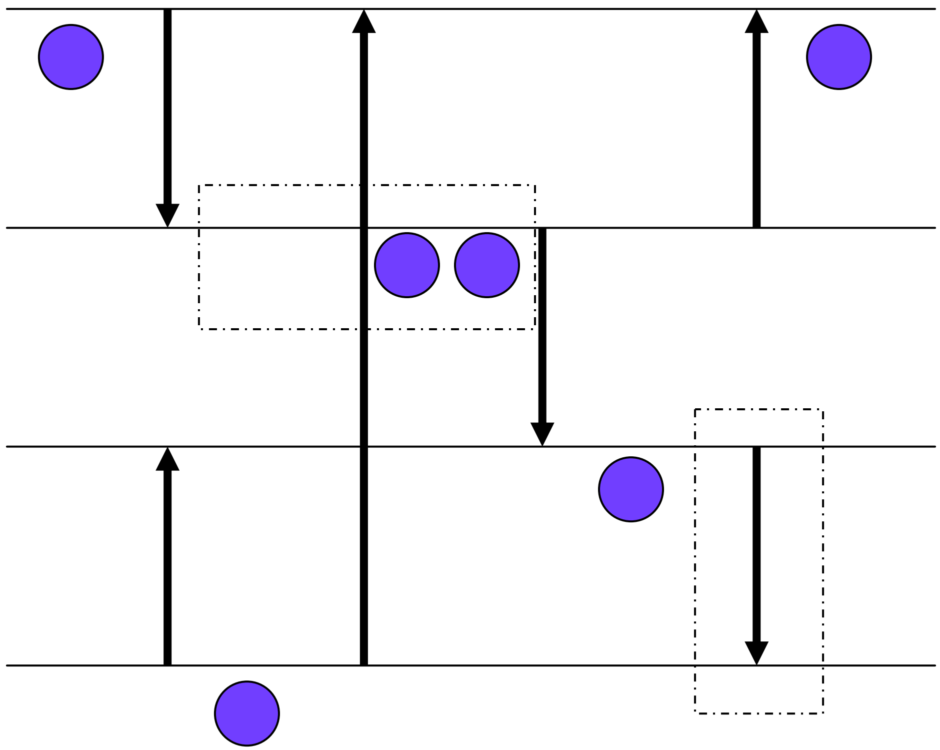
Self Stabilization

- Extra state and actions added to the network
- Self-stabilizing Actions enabled only if network in illegal state otherwise, normal execution

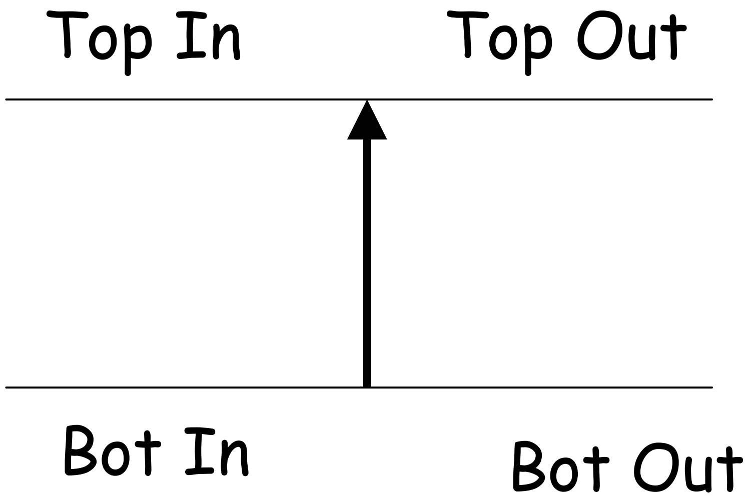
Self Stabilization

- Definition:
  Legal State can be reached in an execution starting from the Correct Initial State
- Natural definition, but hard to use directly, so need alternate characterization
- Local state can be observed easily
- Strategy: Characterize legality in terms of local states

Global vs Local States



Additional State



These counters can be bounded - details in paper

Local States

- Balancer is Legal if
  (1)Top In + Bot In = Top Out + Bot Out
  (2)Toggle State is correct
- Wire is Legal if
  Tokens entering the wire = Tokens leaving the wire + Tokens in Transit

Global Legality in terms of Local

Theorem:

Iff (every wire and every balancer is in legal
local state), then
(the network is in a legal global state)

Now focus on stabilizing the local states
- simpler problem

Space and Time Complexity

- Time to Stabilization = d parallel timesteps where d = depth of network
- Total additional space = $O(wd^2)$
  w = width of network

Issues

- Lazy versus pro-active stabilization
- Transient Behavior till stabilization might differ from “legal” behavior
- Tokens might be unevenly distributed till then

Summary

- Even if bitonic and periodic networks are not initialized, they are log smooth
- If only approximate smoothing is needed, then use (log w) depth uninitialized block network
- Can be converted into 1-smooth behavior by self-stabilization
  - overhead is small and analytically bounded