The Quest for Optimal Sorting Networks

L. Cruz-Filipe[1]   M. Codish[2]   P. Schneider-Kamp[1]

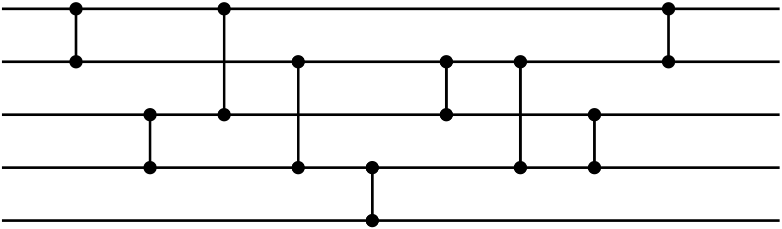
[1]Dept. Mathematics and Computer Science, Univ. Southern Denmark (Denmark)

[2]Ben-Gurion University of the Negev (Israel)

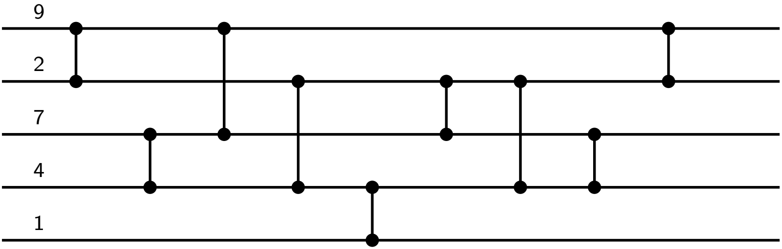
IMADA Colloquium
August 19th, 2014

Outline

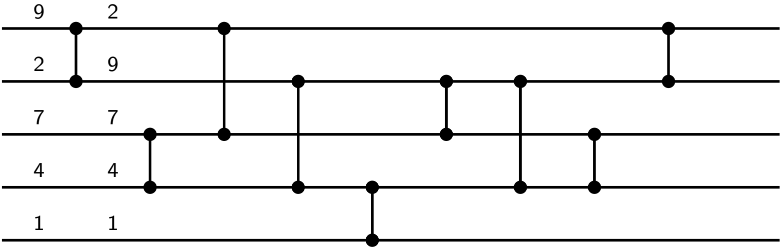
A sorting network



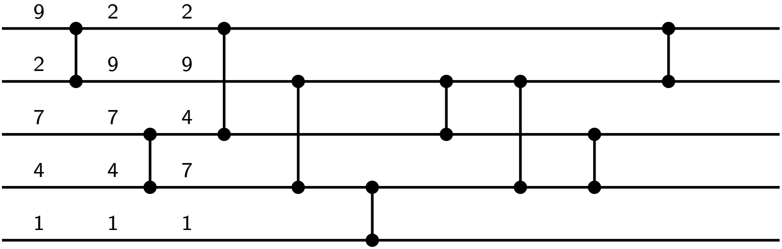
A sorting network



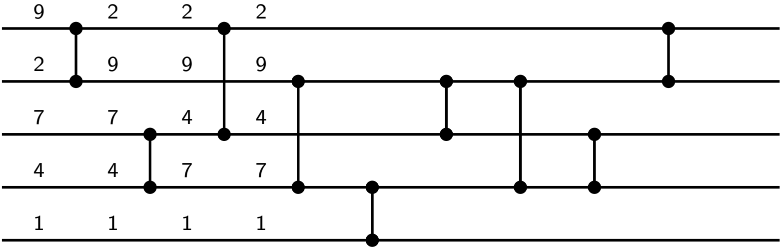
A sorting network



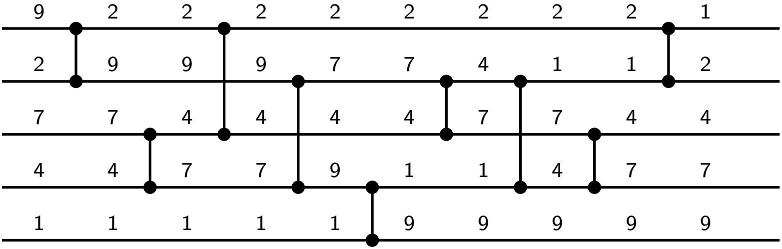
A sorting network



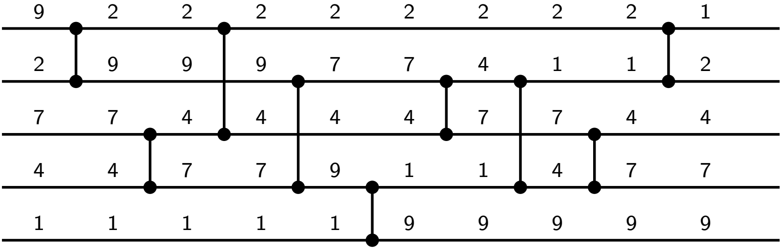
A sorting network



A sorting network



A sorting network



Size

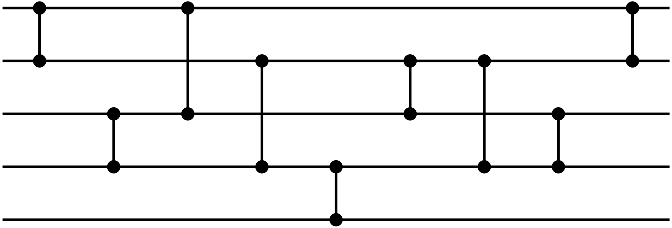
This net has 5 *channels* and 9 *comparators*.

A sorting network

Some of the comparisons may be performed in parallel:

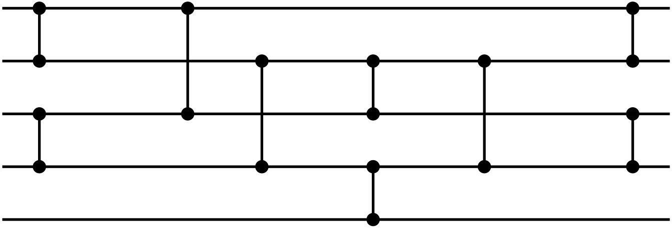
A sorting network

Some of the comparisons may be performed in parallel:



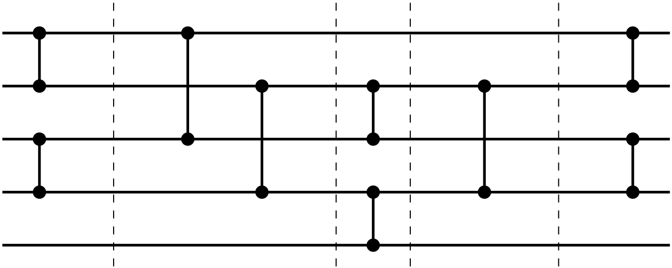
A sorting network

Some of the comparisons may be performed in parallel:



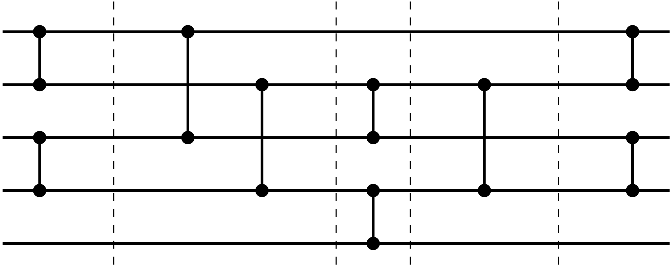
A sorting network

Some of the comparisons may be performed in parallel:



A sorting network

Some of the comparisons may be performed in parallel:

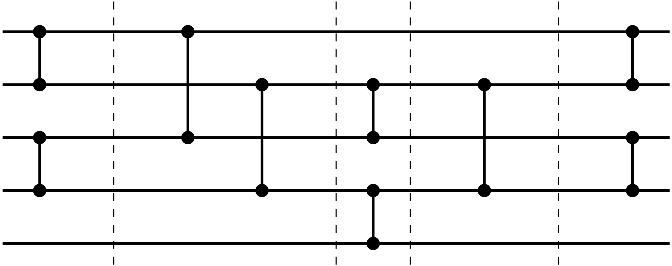


Depth

This net has 5 *layers*.

A sorting network

Some of the comparisons may be performed in parallel:



Depth

This net has 5 *layers*.

See Donald E. Knuth, *The Art of Computer Programming*, vol. 3 for more details

The optimization problems

The size problem

What is the minimal number of *comparators* on a sorting network on n channels (S_n)?

The depth problem

What is the minimal number of *layers* on a sorting network on n channels (T_n)?

The optimization problems

The size problem

What is the minimal number of *comparators* on a sorting network on n channels (S_n)?

The depth problem

What is the minimal number of *layers* on a sorting network on n channels (T_n)?

Knuth 1973

n	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
T_n	3	3	5	5	6	6	7 6	7 6	8 6	8 6	9 6	9 6	9 6	9 6	11 6

The optimization problems

The size problem

What is the minimal number of *comparators* on a sorting network on n channels (S_n)?

The depth problem

What is the minimal number of *layers* on a sorting network on n channels (T_n)?

Parberry 1991

n	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
T_n	3	3	5	5	6	6	**7**	**7**	8 **7**	8 **7**	9 **7**	9 **7**	9 **7**	9 **7**	11 **7**

The optimization problems

The size problem

What is the minimal number of *comparators* on a sorting network on n channels (S_n)?

The depth problem

What is the minimal number of *layers* on a sorting network on n channels (T_n)?

Bundala & Závodný 2013

n	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
T_n	3	3	5	5	6	6	7	7	**8**	**8**	**9**	**9**	**9**	**9**	11 **9**

An exponential explosion

- Upper bounds obtained by concrete examples (1960s)
- Lower bounds obtained by mathematical arguments
- HUGE number of nets

An exponential explosion

- Upper bounds obtained by concrete examples (1960s)
- Lower bounds obtained by mathematical arguments
- HUGE number of nets
- Parberry (1991)
  - exploration of symmetries
  - fixed first layer
  - 200 hours of computation

An exponential explosion

- Upper bounds obtained by concrete examples (1960s)
- Lower bounds obtained by mathematical arguments
- HUGE number of nets
- Parberry (1991)
- Bundala & Závodný (2013)
  - exploration of symmetries
  - reduced set of two-layer prefixes
  - intensive SAT-solving

An exponential explosion

- Upper bounds obtained by concrete examples (1960s)
- Lower bounds obtained by mathematical arguments
- HUGE number of nets
- Parberry (1991)
- Bundala & Závodný (2013)
- These techniques do not scale for T_{17}
  $\approx 211 \times 10^6$ possibilities for each layer when $n = 17$
- These techniques are not directly applicable to the size problem
  36 possibilities for each comparator when $n = 9$, so
  $36^{24} \approx 2.2 \times 10^{37}$ 24-comparator nets

Outline

Comparator networks

A *comparator network* C on n wires is a sequence of *comparators* (i, j) with $1 \leq i < j \leq n$.

The *output* of C on a sequence $\vec{x} = x_1 \dots x_n$ is denoted $C(\vec{x})$.

The set of binary outputs of C is
$\text{outputs}(C) = \{C(\vec{x}) \mid x \in \{0, 1\}^n\}$.

A comparator network C is a *sorting network* if $C(\vec{x})$ is sorted for every input $\vec{x}$.

Well-known results

0–1 lemma (Knuth 1973)

C is a sorting network on n channels iff C sorts all inputs in $\{0,1\}^n$.

Well-known results

0–1 lemma (Knuth 1973)

C is a sorting network on n channels iff C sorts all inputs in $\{0, 1\}^n$.

Proof

The direct implication is straightforward. For the converse, consider all sequences with k or $k + 1$ zeros. If C sorts them all, then it must always place the $(k + 1)$-th smallest element of its input in the right place.

Well-known results

0–1 lemma (Knuth 1973)

C is a sorting network on n channels iff C sorts all inputs in $\{0, 1\}^n$.

"C is a sorting network on n channels" is co-NP (complete).

Well-known results

0–1 lemma (Knuth 1973)

C is a sorting network on n channels iff C sorts all inputs in $\{0,1\}^n$.

Output lemma (Parberry 1991)

Let C and C' be comparator networks such that $\text{outputs}(C) \subseteq \text{outputs}(C')$. If $C';N$ is a sorting network, then so is $C;N$.

Well-known results

0–1 lemma (Knuth 1973)

C is a sorting network on n channels iff C sorts all inputs in $\{0,1\}^n$.

Output lemma (Parberry 1991)

Let C and C' be comparator networks such that $\mathrm{outputs}(C) \subseteq \mathrm{outputs}(C')$. If $C'; N$ is a sorting network, then so is $C; N$.

Proof

$$\begin{array}{ccccc} \{0,1\}^n & \xrightarrow{C} & X & \xrightarrow{N} & S \\ & & \rotatebox{90}{$\subseteq$} & & \\ \{0,1\}^n & \xrightarrow{C'} & X' & \xrightarrow{N} & S \end{array}$$

Well-known results

0–1 lemma (Knuth 1973)

C is a sorting network on n channels iff C sorts all inputs in $\{0,1\}^n$.

Output lemma (Parberry 1991)

Let C and C' be comparator networks such that $\text{outputs}(C) \subseteq \text{outputs}(C')$. If $C'; N$ is a sorting network, then so is $C; N$.

Corollary 1

There is a minimal-depth sorting network on n channels whose first layer contains $\left\lfloor \frac{n}{2} \right\rfloor$ comparators.

Well-known results

0–1 lemma (Knuth 1973)

C is a sorting network on n channels iff C sorts all inputs in $\{0,1\}^n$.

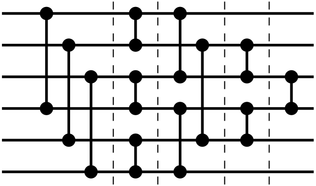
Output lemma (Parberry 1991)

Let C and C' be comparator networks such that $\text{outputs}(C) \subseteq \text{outputs}(C')$. If $C';N$ is a sorting network, then so is $C;N$.

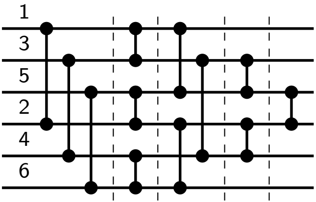
Corollary 2

There is a minimal-depth sorting network on n channels whose first layer F_n contains the comparators $(1,2)$, $(3,4)$, $(5,6)$, &c.

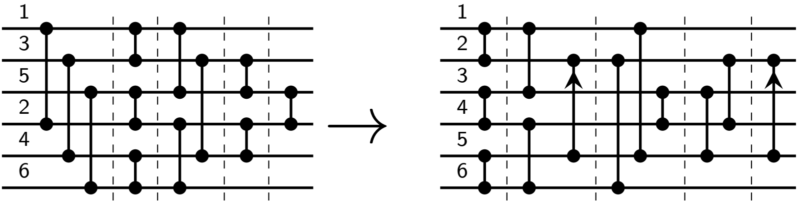
Normalization



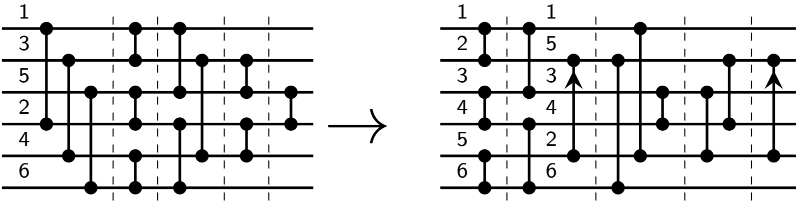
Normalization



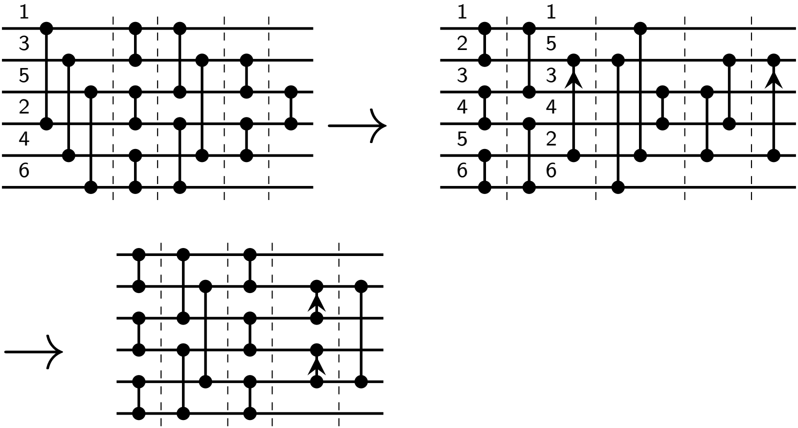
Normalization



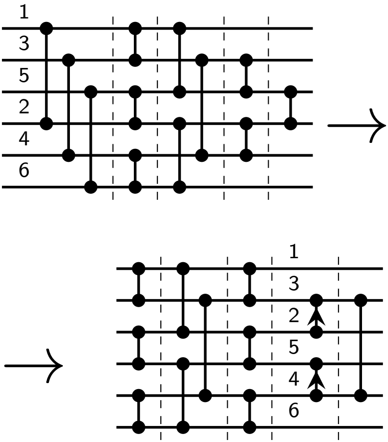
Normalization



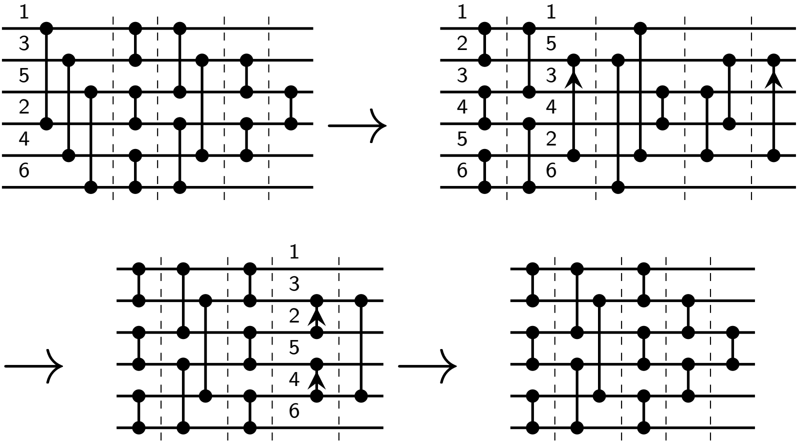
Normalization



Normalization



Normalization



Equivalence

Equivalence of networks

Two standard networks C and C' are equivalent if there is a permutation π of $1..n$ such that C' can be obtained from C by renumbering its wires according to π and normalizing the result.

Permutations (Bundala & Závodný 2013)

Permuted output lemma

- C and C' are two-layer standard comparator networks;
- π is a permutation of $1..n$ mapping outputs(C) into outputs(C');
- C' can be extended to a sorting network;

then C can also be extended to a standard sorting network of the same depth.

Permutations (Bundala & Závodný 2013)

Permuted output lemma

- C and C' are two-layer standard comparator networks;
- π is a permutation of $1..n$ mapping outputs(C) into outputs(C');
- C' can be extended to a sorting network;

then C can also be extended to a standard sorting network of the same depth.

Proof

$$\begin{array}{ccccc} \{0,1\}^n & \xrightarrow{C} & X & \xrightarrow{\pi^{-1}(N)} & \pi^{-1}(S) \\ & & \downarrow \pi & & \\ \{0,1\}^n & \xrightarrow{C'} & X' & \xrightarrow{N} & S \end{array}$$

Finding the value of T_{13}

Saturation

Saturation is a syntactic criterion for two-layer networks. Bundala & Závodný prove that it is enough to consider saturated networks.

Finding the value of T_{13}

Saturation

Saturation is a syntactic criterion for two-layer networks. Bundala & Závodný prove that it is enough to consider saturated networks.

Reflection

(Vertical) reflection of a comparator network produces a "dual" net: if input $\vec{x}$ goes to $\vec{y}$, then $x^{\vec{D}}$ goes to $y^{\vec{D}}$. A two-layer network can be extended to a sorting network of depth d iff the same holds for its reflection.

Finding the value of T_{13}

The strategy

1. Generate all saturated two-layer networks with first layer F_{13}.
2. Remove equivalent nets.
3. Remove nets subsumed by others.
4. Remove reflected nets.
5. Use a SAT-solver to find out if the remaining nets can be extended to a sorting network.

Finding the value of T_{13}

The strategy

1. Generate all saturated two-layer networks with first layer F_{13}.
2. Remove equivalent nets.
3. Remove nets subsumed by others.
4. Remove reflected nets.
5. Use a SAT-solver to find out if the remaining nets can be extended to a sorting network.

n	5	6	7	8	9	10	11	12	13
$\lvert G_n \rvert$	26	76	232	764	2620	9496	35696	140152	568504
$\lvert S_n \rvert$	10	51	74	513	700	6345	8174	93255	113008
$\lvert G_n/\approx \rvert$	18	28	74	101	295	350	1134	1236	4288
$\lvert S_n/\approx \rvert$	8		29		100		341		1155
red.	6	6	14	15	37	27	88	70	212
$\lvert R_n \rvert$	4	5	8	12	22	21	28	50	118

And for higher n?

This approach does not scale.

Computing equivalence of nets is very expensive (and not working correctly).

Checking output subsumption is even worse (there are 2^n outputs, 2^{2^n} possible sets of outputs, and $n!$ permutations).

Furthermore, $T_{13} = T_{14} = T_{15} = T_{16}$.

To go beyond these values, we need different techniques.

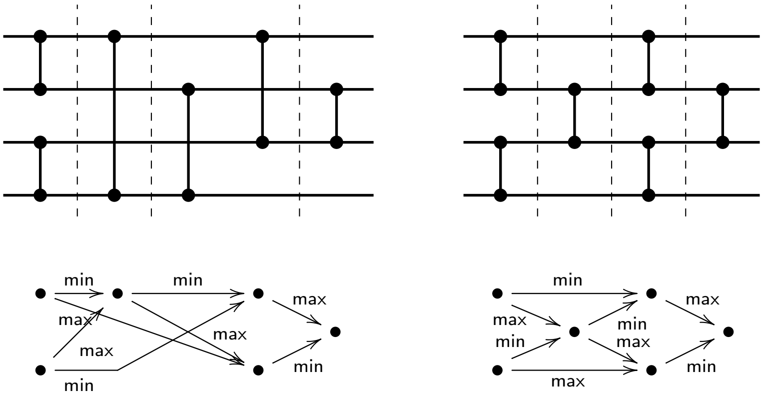
Outline

The general idea

Comparator networks can be represented by labeled graphs (Choi & Moon 2002), where each comparator is a node and there is an edge from v_i to v_j labeled min (max) if the minimum (maximum) output of v_i is an input to v_j.

The general idea

Comparator networks can be represented by labeled graphs (Choi & Moon 2002), where each comparator is a node and there is an edge from v_i to v_j labeled min (max) if the minimum (maximum) output of v_i is an input to v_j.



The general idea

Comparator networks can be represented by labeled graphs (Choi & Moon 2002), where each comparator is a node and there is an edge from v_i to v_j labeled min (max) if the minimum (maximum) output of v_i is an input to v_j.

Equivalence theorem

$C \approx C'$ iff $\mathcal{G}_C \approx \mathcal{G}_{C'}$.

The general idea

Comparator networks can be represented by labeled graphs (Choi & Moon 2002), where each comparator is a node and there is an edge from v_i to v_j labeled min (max) if the minimum (maximum) output of v_i is an input to v_j.

Equivalence theorem

$C \approx C'$ iff $\mathcal{G}_C \approx \mathcal{G}_{C'}$.

But we want to bypass graphs altogether. Can we find a way to generate a set $\mathcal{N}$ of nets such that:

- for every two-layer comparator network C with first layer F_n there is $C' \in \mathcal{N}$ such that $C \approx C'$;
- for every $C, C' \in \mathcal{N}$, $C \not\approx C'$;

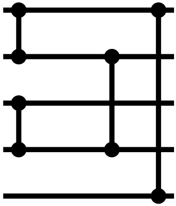
in an efficient way?

Word representation for two-layer networks

Idea: represent two layer-networks by words, corresponding to paths in their graphs. (But avoid graphs altogether.)

Word representation for two-layer networks

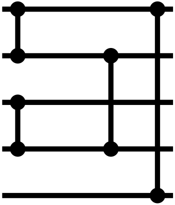
Idea: represent two layer-networks by words, corresponding to paths in their graphs. (But avoid graphs altogether.)



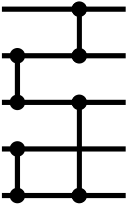
Head word:
01221

Word representation for two-layer networks

Idea: represent two layer-networks by words, corresponding to paths in their graphs. (But avoid graphs altogether.)

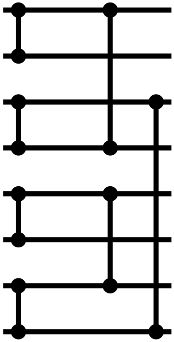


Head word:
01221



Word representation for two-layer networks

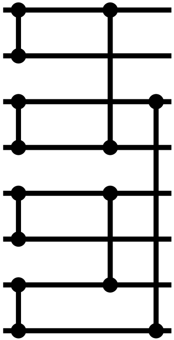
Idea: represent two layer-networks by words, corresponding to paths in their graphs. (But avoid graphs altogether.)



Stick word:
21121212
21212112

Word representation for two-layer networks

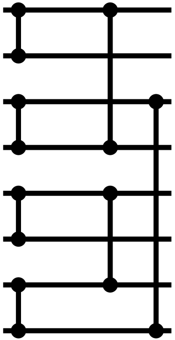
Idea: represent two layer-networks by words, corresponding to paths in their graphs. (But avoid graphs altogether.)



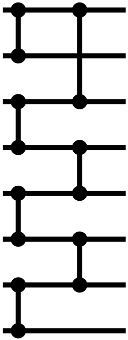
Stick word:
21121212
21212112

Word representation for two-layer networks

Idea: represent two layer-networks by words, corresponding to paths in their graphs. (But avoid graphs altogether.)

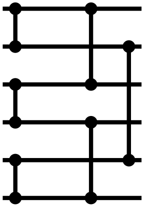


Stick word:
21121212
21212112



Word representation for two-layer networks

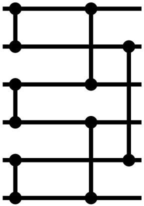
Idea: represent two layer-networks by words, corresponding to paths in their graphs. (But avoid graphs altogether.)



Cycle word:
121221
122121
122112

Word representation for two-layer networks

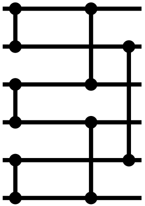
Idea: represent two layer-networks by words, corresponding to paths in their graphs. (But avoid graphs altogether.)



Cycle word:
121221
122121
122112

Word representation for two-layer networks

Idea: represent two layer-networks by words, corresponding to paths in their graphs. (But avoid graphs altogether.)

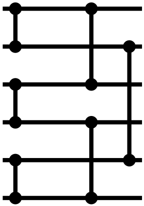


Cycle word:

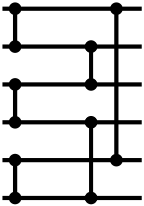
121221
122121
122112

Word representation for two-layer networks

Idea: represent two layer-networks by words, corresponding to paths in their graphs. (But avoid graphs altogether.)



Cycle word:
121221
122121
122112



Word representation for two-layer networks

Idea: represent two layer-networks by words, corresponding to paths in their graphs. (But avoid graphs altogether.)

Every net generates a unique word, and every well-formed word generates a unique net. The functions net-to-word and word-to-net form an adjunction.

A regular language for words

$$
\begin{aligned}
\text{Word} &::= \text{Head} \mid \text{Stick} \mid \text{Cycle} \\
\text{Head} &::= 0(12+21)^* \\
\text{Stick} &::= (12+21)^+ \\
\text{Cycle} &::= 12(12+21)^*(1+2)
\end{aligned}
$$

A regular language for words

$$
\begin{aligned}
\text{Word} &::= \text{Head} \mid \text{Stick} \mid \text{Cycle} \\
\text{Head} &::= 0(12+21)^* \\
\text{Stick} &::= (12+21)^+ \\
\text{Cycle} &::= 12(12+21)^*(1+2)
\end{aligned}
$$

Generating all words and filtering to obtain only the lexicographically smallest is very easy for the relevant values of n.

A regular language for words

$$\begin{aligned} \text{Word} &::= \text{Head} \mid \text{Stick} \mid \text{Cycle} \\ \text{Head} &::= 0(12+21)^* \\ \text{Stick} &::= (12+21)^+ \\ \text{Cycle} &::= 12(12+21)^*(1+2) \end{aligned}$$

Generating all words and filtering to obtain only the lexicographically smallest is very easy for the relevant values of n. Two-layer comparator networks can be represented by multi-sets of words. By choosing a canonical representation of multi-sets, we can easily generate exactly one representative for all two-layer networks with first layer F_n modulo equivalence.

Saturation, revisited [I]

Saturation

Saturation is a syntactic criterion for two-layer networks. Bundala & Závodný prove that it is enough to consider saturated networks.

We want a semantic characterization of saturation that is optimal.

Saturation, revisited [I]

Saturation

Saturation is a syntactic criterion for two-layer networks. Bundala & Závodný prove that it is enough to consider saturated networks.

We want a semantic characterization of saturation that is optimal.

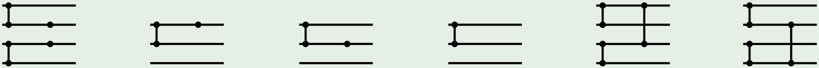
Saturation (better)

A comparator network C is *redundant* if there exists a network C' obtained from C by removing a comparator such that $\text{outputs}(C') = \text{outputs}(C)$.
A network C is *saturated* if it is non-redundant and every network C' obtained by adding a comparator to the last layer of C satisfies $\text{outputs}(C') \not\subseteq \text{outputs}(C)$.

Saturation, revisited [II]

Saturation theorem

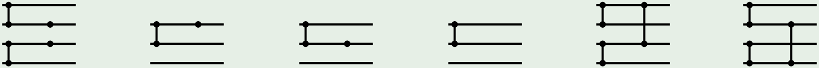
Let C be a two-layer network. Then C is saturated iff C contains none of the following two-layer patterns.



Saturation, revisited [II]

Saturation theorem

Let C be a two-layer network. Then C is saturated iff C contains none of the following two-layer patterns.



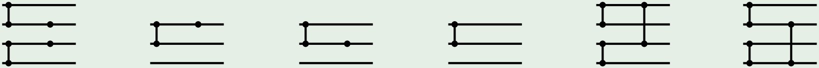
This is easily encoded in words.

$$\begin{aligned} \text{Word} &::= \text{Head} \mid \text{Stick} \mid \text{Cycle} \\ \text{Head} &::= 0(12+21)^* \\ \text{Stick} &::= (12+21)^+ \\ \text{Cycle} &::= 12(12+21)^*(1+2) \end{aligned}$$

Saturation, revisited [II]

Saturation theorem

Let C be a two-layer network. Then C is saturated iff C contains none of the following two-layer patterns.



This is easily encoded in words.

$$\begin{aligned} \text{Word} &::= \text{Head} \mid \text{Stick} \mid \text{Cycle} \\ \text{Head} &::= 0 \mid \text{eHead} \mid \text{oHead} \\ \text{eHead} &::= 0(12+21)^*12 \\ \text{oHead} &::= 0(12+21)^*21 \end{aligned}$$

$$\begin{aligned} \text{Stick} &::= 12 \mid \text{eStick} \mid \text{oStick} \\ \text{oStick} &::= 12(12+21)^+21 \\ \text{eStick} &::= 21(12+21)^+12 \\ \text{Cycle} &::= 12(12+21)^*(1+2) \end{aligned}$$

Saturation, revisited [III]

Multi-sets of words corresponding to saturated nets do not contain eWords and oWords.

We can again generate all saturated networks very efficiently for n up to 40.

Saturation, revisited [III]

Multi-sets of words corresponding to saturated nets do not contain eWords and oWords.

We can again generate all saturated networks very efficiently for n up to 40.

A similar technique eliminates nets that are (equivalent to) reflections of others.

This is more expensive because the test for cycles takes time proportional to the number of wires. However, the smallest asymmetric cycle has 12 wires, so for practical purposes this is not a problem.

Some numerology

n	5	6	7	8	9	10	11	12	13
$\|G_n\|$	26	76	232	764	2,620	9,496	35,696	140,152	568,504
$\|S_n\|$	10	28	70	230	676	2,456	7,916	31,374	109,856
$\|R(G_n)\|$	16	20	52	61	165	152	482	414	1,378
$\|R(S_n)\|$	6	6	14	15	37	27	88	70	212
$\|R_n\|$	4	5	8	12	22	21	28	50	117

n	14	15	16	17	18
$\|G_n\|$	2,390,480	10,349,536	46,206,736	211,799,312	997,313,824
$\|S_n\|$	467,716	1,759,422	7,968,204	31,922,840	152,664,200
$\|R(G_n)\|$	1,024	3,780	2,627	10,187	6,422
$\|R(S_n)\|$	136	494	323	1,149	651
$\|R_n\|$	94	262	211	609	411

n	19	20	25	30	35	40
$\|R(S_n)\|$	2,632	1,478	30,312	64,168	1,604,790	2,792,966
$\|R_n\|$	1,367	894	15,469	34,486	806,710	1,429,836

Outline

Results

- Efficient generation of two-layer prefixes for comparator networks
- Representation can capture different important semantic properties
- Identified relevant sets of networks for open cases
- Bottleneck is now processing each relevant network