twenty-five comparators is optimal when sorting nine inputs
(and twenty-nine for ten)

luís cruz-filipe[1] michael codish[2]
michael frank[2] peter schneider-kamp[1]

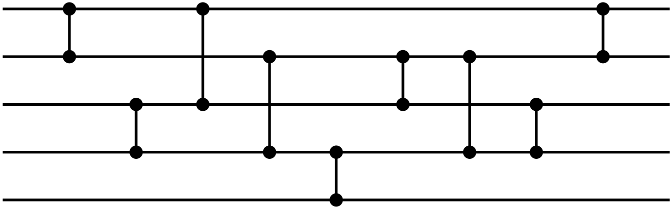
[1]department of mathematics and computer science
university of southern denmark

[2]department of computer science
ben-gurion university of the negev, israel

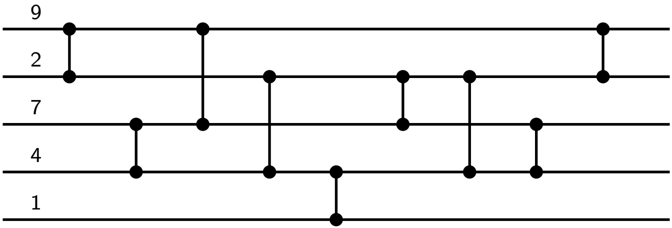
ictai 2014
november 10th, 2014

outline

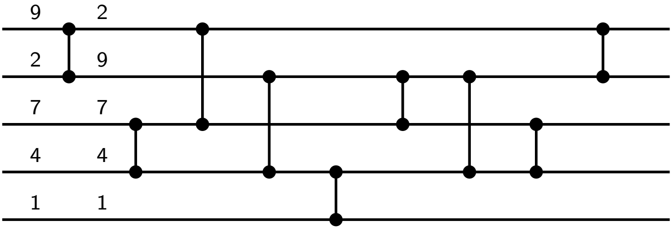
a sorting network



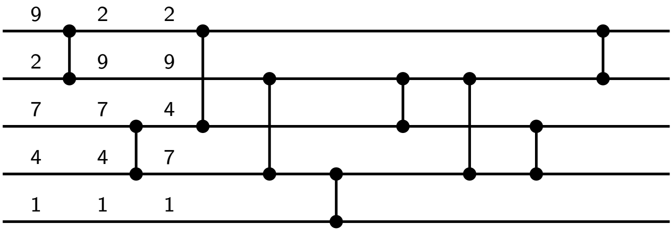
a sorting network



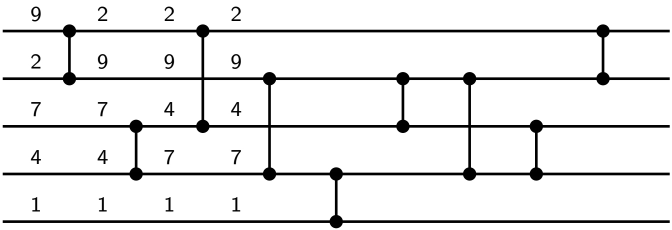
a sorting network



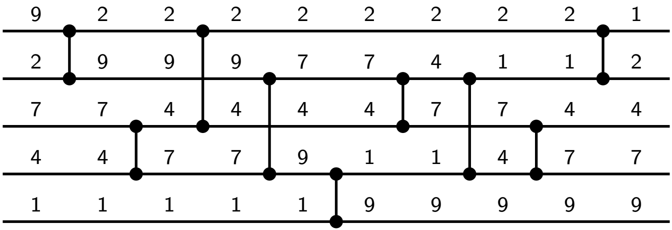
a sorting network



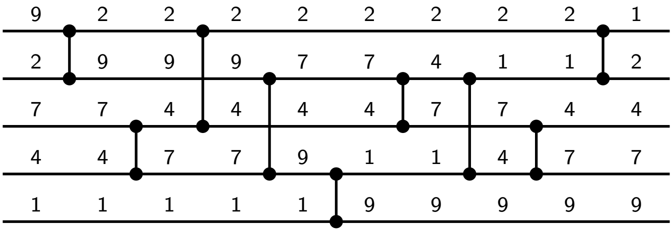
a sorting network



a sorting network

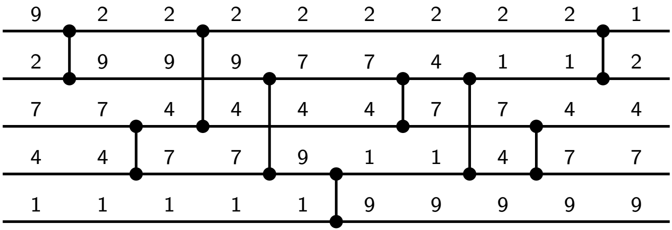


a sorting network



size this net has 5 *channels* and 9 *comparators*

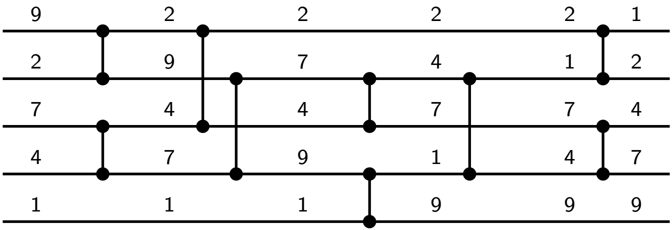
a sorting network



size this net has 5 *channels* and 9 *comparators*

some of the comparisons may be performed in parallel

a sorting network

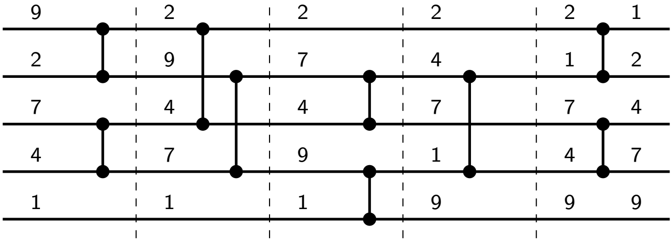


size

this net has 5 *channels* and 9 *comparators*

some of the comparisons may be performed in parallel

a sorting network

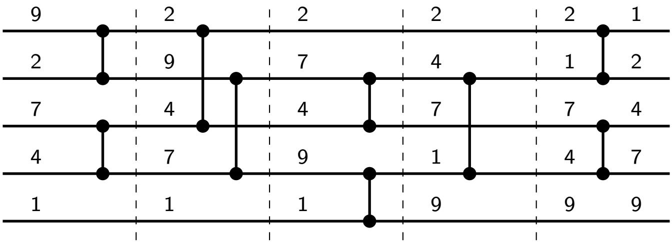


size

this net has 5 *channels* and 9 *comparators*

some of the comparisons may be performed in parallel

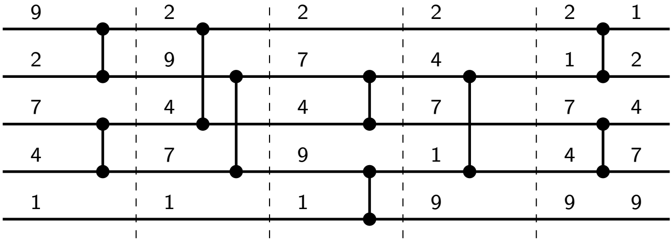
a sorting network



size this net has 5 *channels* and 9 *comparators*

depth this net has 5 *layers*

a sorting network



size this net has 5 *channels* and 9 *comparators*

depth this net has 5 *layers*

more info see d.e. knuth, *the art of computer programming*, vol. 3

the optimization problems

the optimal size problem — what is the minimal number of *comparators* on a sorting network on n channels (s_n)?

the optimal depth problem — what is the minimal number of *layers* on a sorting network on n channels (t_n)?

the optimization problems

the optimal size problem

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

the optimal depth problem

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

knuth 1973

n	1	2	3	4	5	6	7	8	9	10
s_n	0	1	3	5	9	12	16	19	25 23	29 27
t_n	0	1	3	3	5	5	6	6	7 6	7 6

n	11	12	13	14	15	16
s_n	35 31	39 35	45 39	51 43	56 47	60 51
t_n	8 6	8 6	9 6	9 6	9 6	9 6

the optimization problems

the optimal size problem

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

the optimal depth problem

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

parberry 1991

n	1	2	3	4	5	6	7	8	9	10
s_n	0	1	3	5	9	12	16	19	25 23	29 27
t_n	0	1	3	3	5	5	6	6	**7**	**7**

n	11	12	13	14	15	16
s_n	35 31	39 35	45 39	51 43	56 47	60 51
t_n	8 **7**	8 **7**	9 **7**	9 **7**	9 **7**	9 **7**

the optimization problems

the optimal size problem

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

the optimal depth problem

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

n	1	2	3	4	5	6	7	8	9	10
s_n	0	1	3	5	9	12	16	19	25 23	29 27
t_n	0	1	3	3	5	5	6	6	7	7

bundala & závodný 2013

n	11	12	13	14	15	16
s_n	35 31	39 35	45 39	51 43	56 47	60 51
t_n	**8**	**8**	**9**	**9**	**9**	**9**

the optimization problems

the optimal size problem

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

the optimal depth problem

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

our contribution

n	1	2	3	4	5	6	7	8	9	10
s_n	0	1	3	5	9	12	16	19	**25**	**29**
t_n	0	1	3	3	5	5	6	6	7	7

n	11	12	13	14	15	16
s_n	35 **33**	39 **37**	45 **41**	51 **45**	56 **49**	60 **53**
t_n	8	8	9	9	9	9

an exponential explosion

parberry 1991

- exploration of symmetries $\rightsquigarrow$ fixed first layer
- exhaustive search (200 hours of computation)

an exponential explosion

parberry 1991

- exploration of symmetries $\rightsquigarrow$ fixed first layer
- exhaustive search (200 hours of computation)

bundala & závodný 2013

- reduced set of two-layer prefixes
- intensive sat-solving

an exponential explosion

parberry 1991

- exploration of symmetries $\rightsquigarrow$ fixed first layer
- exhaustive search (200 hours of computation)

bundala & závodný 2013

- reduced set of two-layer prefixes
- intensive sat-solving

however... techniques not directly applicable to the size problem

an exponential explosion

parberry 1991

- exploration of symmetries $\rightsquigarrow$ fixed first layer
- exhaustive search (200 hours of computation)

bundala & závodný 2013

- reduced set of two-layer prefixes
- intensive sat-solving

however... techniques not directly applicable to the size problem

9 channels (depth) 2620 possibilities for each layer
$\rightsquigarrow 2620^6 \approx 3.2 \times 10^{20}$ 6-layer networks

9 channels (size) 36 possibilities for each comparator
$\rightsquigarrow 36^{24} \approx 2.2 \times 10^{37}$ 24-comparator networks

outline

sorting networks in a nutshell

encoding the size problem in sat

conclusions & future work

comparator networks

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

output — $C(\vec{x})$ denotes the *output* of C on $\vec{x} = x_1 \dots x_n$

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

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

comparator networks

comparator network	a *comparator network* C on n channels is a sequence of *comparators* (i, j) with $1 \leq i < j \leq n$
output	$C(\vec{x})$ denotes the *output* of C on $\vec{x} = x_1 \ldots x_n$
binary outputs	the set of binary outputs of C is $\text{outputs}(C) = \{C(\vec{x}) \mid x \in \{0,1\}^n\}$
sorting network	a comparator network C is a *sorting network* if $C(\vec{x})$ is sorted for every input $\vec{x}$
0–1 lemma (knuth 1973)	C is a sorting network on n channels iff C sorts all inputs in $\{0,1\}^n$

comparator networks

comparator network	a *comparator network* C on n channels is a sequence of *comparators* (i, j) with $1 \leq i < j \leq n$
output	$C(\vec{x})$ denotes the *output* of C on $\vec{x} = x_1 \ldots x_n$
binary outputs	the set of binary outputs of C is $\text{outputs}(C) = \{C(\vec{x}) \mid x \in \{0, 1\}^n\}$
sorting network	a comparator network C is a *sorting network* if $C(\vec{x})$ is sorted for every input $\vec{x}$
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)

sat encoding

$$\texttt{Network} = \langle \texttt{c}(\texttt{I}_1, \texttt{J}_1), \ldots, \texttt{c}(\texttt{I}_k, \texttt{J}_k) \rangle$$

$$\texttt{valid}_{n,k}(\texttt{Network}) = \bigwedge_{i=1}^{k} \begin{array}{l} \texttt{new\_int}(\texttt{I}_i, 1, n) \\ \wedge\, \texttt{new\_int}(\texttt{J}_i, 1, n) \\ \wedge\, \texttt{int\_lt}(\texttt{I}_i, \texttt{J}_i) \end{array}$$

sat encoding

$$\texttt{Network} = \langle \texttt{c}(\texttt{I}_1, \texttt{J}_1), \ldots, \texttt{c}(\texttt{I}_k, \texttt{J}_k) \rangle$$

$\texttt{valid}_{n,k}(\texttt{Network})$

sat encoding

$$\texttt{Network} = \langle \texttt{c}(\texttt{I}_1, \texttt{J}_1), \ldots, \texttt{c}(\texttt{I}_k, \texttt{J}_k) \rangle$$

$$\texttt{valid}_{n,k}(\texttt{Network})$$

$$\varphi_{\texttt{I},\texttt{J}}(\vec{x}, \vec{y}) = \bigwedge_{1 \leq i < j \leq n} \begin{array}{l} \texttt{int\_eq}(\texttt{I}, i) \land \texttt{int\_eq}(\texttt{J}, j) \\ \rightarrow ((y_i \leftrightarrow x_i \land x_j) \land (y_j \leftrightarrow x_i \lor x_j)) \end{array}$$

$$\psi_{\texttt{I},\texttt{J}}(\vec{x}, \vec{y}) = \bigwedge_{i=1}^{n} \begin{array}{l} \neg\texttt{int\_eq}(\texttt{I}, i) \land \neg\texttt{int\_eq}(\texttt{J}, i) \\ \rightarrow (y_i \leftrightarrow x_i) \end{array}$$

sat encoding

$$\texttt{Network} = \langle \texttt{c}(\texttt{I}_1, \texttt{J}_1), \ldots, \texttt{c}(\texttt{I}_k, \texttt{J}_k) \rangle$$

$$\texttt{valid}_{n,k}(\texttt{Network})$$

$$\varphi_{\texttt{I},\texttt{J}}(\vec{x}, \vec{y}) = \bigwedge_{1 \le i < j \le n} \begin{array}{l} \texttt{int\_eq}(\texttt{I}, i) \land \texttt{int\_eq}(\texttt{J}, j) \\ \to ((y_i \leftrightarrow x_i \land x_j) \land (y_j \leftrightarrow x_i \lor x_j)) \end{array}$$

$$\psi_{\texttt{I},\texttt{J}}(\vec{x}, \vec{y}) = \bigwedge_{i=1}^{n} \begin{array}{l} \neg\texttt{int\_eq}(\texttt{I}, i) \land \neg\texttt{int\_eq}(\texttt{J}, i) \\ \to (y_i \leftrightarrow x_i) \end{array}$$

$$\texttt{sorts}_{n,k}(\texttt{Network}, \vec{b}) = \bigwedge_{i=1}^{k} \begin{array}{l} \varphi_{\texttt{I}_i,\texttt{J}_i}(\vec{x}_{i-1}, x_i) \\ \land \psi_{\texttt{I}_i,\texttt{J}_i}(\vec{x}_{i-1}, \vec{x}_i) \end{array}$$

$$\vec{x}_0 = \vec{b}, \qquad \vec{x}_k = \mathsf{sort}(\vec{b})$$

sat encoding

$$\mathtt{Network} = \langle \mathtt{c}(\mathtt{I}_1, \mathtt{J}_1), \ldots, \mathtt{c}(\mathtt{I}_k, \mathtt{J}_k) \rangle$$

$$\mathtt{valid}_{n,k}(\mathtt{Network}) \qquad \mathtt{sorts}_{n,k}(\mathtt{Network}, \vec{b})$$

sat encoding

$$\texttt{Network} = \langle \texttt{c}(\texttt{I}_1, \texttt{J}_1), \ldots, \texttt{c}(\texttt{I}_k, \texttt{J}_k) \rangle$$

$$\texttt{valid}_{n,k}(\texttt{Network}) \qquad \texttt{sorts}_{n,k}(\texttt{Network}, \vec{b})$$

$$\texttt{sorter}_{n,k}(\texttt{Network}) = \texttt{valid}_{n,k}(\texttt{Network}) \wedge \bigwedge_{\vec{b} \in \{0,1\}^n} \texttt{sorts}_{n,k}(\texttt{Network}, \vec{b})$$

sat encoding

$$\texttt{Network} = \langle \texttt{c}(\texttt{I}_1, \texttt{J}_1), \ldots, \texttt{c}(\texttt{I}_k, \texttt{J}_k) \rangle$$

$$\texttt{valid}_{n,k}(\texttt{Network}) \qquad \texttt{sorts}_{n,k}(\texttt{Network}, \vec{b})$$

$$\texttt{sorter}_{n,k}(\texttt{Network}) = \texttt{valid}_{n,k}(\texttt{Network}) \wedge \bigwedge_{\vec{b} \in \{0,1\}^n} \texttt{sorts}_{n,k}(\texttt{Network}, \vec{b})$$

this is compiled with the bee constraint compiler into a cnf formula $\Psi(n, k)$

theorem $\Psi(n, k)$ is satisfiable iff there is a sorting network on n channels with k comparators

practical evaluation

	optimal sorting networks (sat)				
n	k	bee	#clauses	#vars	sat
4	5	0.18	1916	486	0.01
5	9	1.03	10159	2550	0.03
6	12	4.55	35035	8433	2.45
7	16	21.68	114579	26803	16.70
8	19	82.93	321445	73331	∞
9	25	452.55	977559	219950	∞

	smaller networks (unsat)				
n	k	bee	#clauses	#vars	sat
4	4	0.15	1480	356	0.01
5	8	0.90	8963	2221	1.27
6	11	3.99	32007	7657	242.02
7	15	19.04	107227	25000	∞
8	18	73.34	304145	69221	∞
9	24	406.67	937773	210715	∞

(times in seconds, timeout = 1 week)

divide and conquer

main idea divide the “big” sat problem into smaller problems

divide and conquer

main idea divide the "big" sat problem into smaller problems

consider all possible choices for the first ℓ comparators in

$$\texttt{Network} = \langle \texttt{c}(\texttt{I}_1, \texttt{J}_1), \ldots, \texttt{c}(\texttt{I}_k, \texttt{J}_k) \rangle$$

divide and conquer

main idea divide the “big” sat problem into smaller problems

consider all possible choices for the first ℓ comparators in

$$\texttt{Network} = \langle \texttt{c}(\texttt{I}_1, \texttt{J}_1), \ldots, \texttt{c}(\texttt{I}_k, \texttt{J}_k) \rangle$$

find “minimal” set $\mathcal{F}$ of choices for $\texttt{I}_1$, $\texttt{J}_1$,..., $\texttt{I}_\ell$, $\texttt{J}_\ell$ such that

$$\Psi_{n,k} \text{ is satisfiable iff } \bigvee_{f \in \mathcal{F}} \Psi_{n,k,f} \text{ is satisfiable}$$

where $f = \langle \texttt{I}_1, \texttt{J}_1, \ldots, \texttt{I}_\ell, \texttt{J}_\ell \rangle$

divide and conquer

main idea divide the “big” sat problem into smaller problems

consider all possible choices for the first ℓ comparators in

$$\mathtt{Network} = \langle \mathtt{c}(\mathtt{I}_1, \mathtt{J}_1), \ldots, \mathtt{c}(\mathtt{I}_k, \mathtt{J}_k) \rangle$$

find “minimal” set $\mathcal{F}$ of choices for $\mathtt{I}_1$, $\mathtt{J}_1$,..., $\mathtt{I}_\ell$, $\mathtt{J}_\ell$ such that

$$\Psi_{n,k} \text{ is satisfiable iff } \bigvee_{f \in \mathcal{F}} \Psi_{n,k,f} \text{ is satisfiable}$$

where $f = \langle \mathtt{I}_1, \mathtt{J}_1, \ldots, \mathtt{I}_\ell, \mathtt{J}_\ell \rangle$

reduce the size of $\mathcal{F}$ using symmetry-breaking techniques

breaking symmetry i/ii

output lemma (parberry 1991)

- C and C' are comparator networks
- $\text{outputs}(C) \subseteq \text{outputs}(C')$

if $C'; N$ is a sorting network, then so is $C; N$

output lemma (parberry 1991)

- C and C' are comparator networks
- $\text{outputs}(C) \subseteq \text{outputs}(C')$

if $C'; N$ is a sorting network, then so is $C; N$

$$\begin{array}{ccccc} \{0,1\}^n & \xrightarrow{C} & X & & \\ & & \cap\!| & & \\ \{0,1\}^n & \xrightarrow{C'} & X' & \xrightarrow{N} & S \end{array}$$

breaking symmetry i/ii

output lemma (parberry 1991)

- C and C' are comparator networks
- $\text{outputs}(C) \subseteq \text{outputs}(C')$

if $C'; N$ is a sorting network, then so is $C; N$

$$\begin{array}{ccccc} \{0,1\}^n & \xrightarrow{C} & X & \xrightarrow{N} & S \\ & & \cap\!\!| & & \\ \{0,1\}^n & \xrightarrow{C'} & X' & \xrightarrow{N} & S \end{array}$$

breaking symmetry ii/ii

permuted output lemma (bundala & Závodný 2013)

- C and C' comparator networks of depth 2
- $\pi(\text{outputs}(C)) \subseteq \text{outputs}(C')$ for some permutation π
- C' can be extended to a sorting network

then C can also be extended to a sorting network of depth 2

breaking symmetry ii/ii

permuted output lemma (bundala & Závodný 2013)

- C and C' comparator networks of depth 2
- $\pi(\text{outputs}(C)) \subseteq \text{outputs}(C')$ for some permutation π
- C' can be extended to a sorting network

then C can also be extended to a sorting network of depth 2

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

breaking symmetry ii/ii

permuted output lemma (bundala & Závodný 2013)

- C and C' comparator networks of depth 2
- $\pi(\text{outputs}(C)) \subseteq \text{outputs}(C')$ for some permutation π
- C' can be extended to a sorting network

then C can also be extended to a sorting network of depth 2

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

breaking symmetry ii/ii

permuted output lemma (bundala & Závodný 2013)

- C and C' comparator networks **of depth 2**
- $\pi(\text{outputs}(C)) \subseteq \text{outputs}(C')$ for some permutation π
- C' can be extended to a sorting network

then C can also be extended to a sorting network of **depth 2**

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

breaking symmetry ii/ii

permuted output lemma (generalized)

- C and C' comparator networks **of equal size**
- $\pi(\text{outputs}(C)) \subseteq \text{outputs}(C')$ for some permutation π
- C' can be extended to a sorting network

then C can also be extended to a sorting network of **the same size**

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

breaking symmetry ii/ii

permuted output lemma (generalized)

- C and C' comparator networks of equal size
- $\pi(\text{outputs}(C)) \subseteq \text{outputs}(C')$ for some permutation π
- C' can be extended to a sorting network

then C can also be extended to a sorting network of the same size

subsumption $C \preceq C'$ when

$$\pi(\text{outputs}(C)) \subseteq \text{outputs}(C')$$

for some permutation π

the generate-and-prune approach

init set $R_0^n = \{\emptyset\}$ and $k = 0$

repeat until $k > 1$ and $|R_k^n| = 1$

- *generate* construct N_{k+1}^n by extending each net in R_k^n by one comparator in all possible ways
- *prune* construct R_{k+1}^n from N_{k+1}^n by keeping only one element of each minimal equivalence class w.r.t. the transitive closure of $\preceq$
- *step* increase k

the generate-and-prune approach

init set $R_0^n = \{\emptyset\}$ and $k = 0$

repeat until $k > 1$ and $|R_k^n| = 1$

- *generate* construct N_{k+1}^n by extending each net in R_k^n by one comparator in all possible ways
- *prune* construct R_{k+1}^n from N_{k+1}^n by keeping only one element of each minimal equivalence class w.r.t. the transitive closure of $\preceq$
- *step* increase k

termination condition if C is a sorting network on n channels of size k, then $|R_k^n| = 1$

optimizations

- only generate networks when the extra comparator does something
- prove and implement criteria for when subsumption will fail
- restrict the search space of possible permutations
- optimize data structures
- parallelize to 288 nodes

some numerology

n	s_n	largest $\lvert N_k^n \rvert$	largest $\lvert R_k^n \rvert$	execution time
3	3	2	2	$\sim$0
4	5	12	4	$\sim$0
5	9	65	11	$\sim$0
6	12	380	53	2 sec
7	16	7,438	678	2 min
8	19	253,243	16,095	6 hours
9	25	18,420,674	914,444	16 years

some numerology

n	s_n	largest $\|N_k^n\|$	largest $\|R_k^n\|$	execution time
3	3	2	2	$\sim$0
4	5	12	4	$\sim$0
5	9	65	11	$\sim$0
6	12	380	53	2 sec
7	16	7,438	678	2 min
8	19	253,243	16,095	6 hours
9	25	18,420,674	914,444	16 years

parallel runtime for $n = 9$: 3 weeks

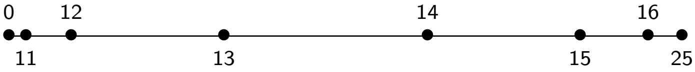
some numerology

n	s_n	largest $\|N_k^n\|$	largest $\|R_k^n\|$	execution time
3	3	2	2	∼0
4	5	12	4	∼0
5	9	65	11	∼0
6	12	380	53	2 sec
7	16	7,438	678	2 min
8	19	253,243	16,095	6 hours
9	25	18,420,674	914,444	16 years

parallel runtime for $n = 9$: 3 weeks

the hard part

going "over the peak" consumes most execution time



collaboration is the key

collaboration is the key

find "minimal" set $\mathcal{F}$ of choices for $\mathtt{I}_1$, $\mathtt{J}_1$,..., $\mathtt{I}_\ell$, $\mathtt{J}_\ell$ such that

$$\Psi_{n,k} \text{ is satisfiable iff } \bigvee_{f \in \mathcal{F}} \Psi_{n,k,f} \text{ is satisfiable}$$

where $f = \langle \mathtt{I}_1, \mathtt{J}_1, \ldots, \mathtt{I}_\ell, \mathtt{J}_\ell \rangle$

collaboration is the key

find "minimal" set $\mathcal{F}$ of choices for $\mathtt{I}_1$, $\mathtt{J}_1$,. . . , $\mathtt{I}_\ell$, $\mathtt{J}_\ell$ such that

$$\Psi_{n,k} \text{ is satisfiable iff } \bigvee_{f \in \mathcal{F}} \Psi_{n,k,f} \text{ is satisfiable}$$

where $f = \langle \mathtt{I}_1, \mathtt{J}_1, \ldots, \mathtt{I}_\ell, \mathtt{J}_\ell \rangle$

taking $\mathcal{F} = R^9_{11}$ gives 188,730 *independent* problems that can be solved in parallel – which is much faster than letting the original program terminate

collaboration is the key

find "minimal" set $\mathcal{F}$ of choices for $\mathtt{I}_1$, $\mathtt{J}_1$,. . . , $\mathtt{I}_\ell$, $\mathtt{J}_\ell$ such that

$$\Psi_{n,k} \text{ is satisfiable iff } \bigvee_{f \in \mathcal{F}} \Psi_{n,k,f} \text{ is satisfiable}$$

where $f = \langle \mathtt{I}_1, \mathtt{J}_1, \ldots, \mathtt{I}_\ell, \mathtt{J}_\ell \rangle$

taking $\mathcal{F} = R^9_{11}$ gives 188,730 *independent* problems that can be solved in parallel – which is much faster than letting the original program terminate

different values of k give different total running times

outline

results & future work

- exact values of s_9 and s_{10}
- technique may be adapted to settle higher values which are still unknown
- algorithms may be useful for *finding* smaller-than-currently-known networks
- further theoretical results may help proving optimality of best known upper bounds

thank you!