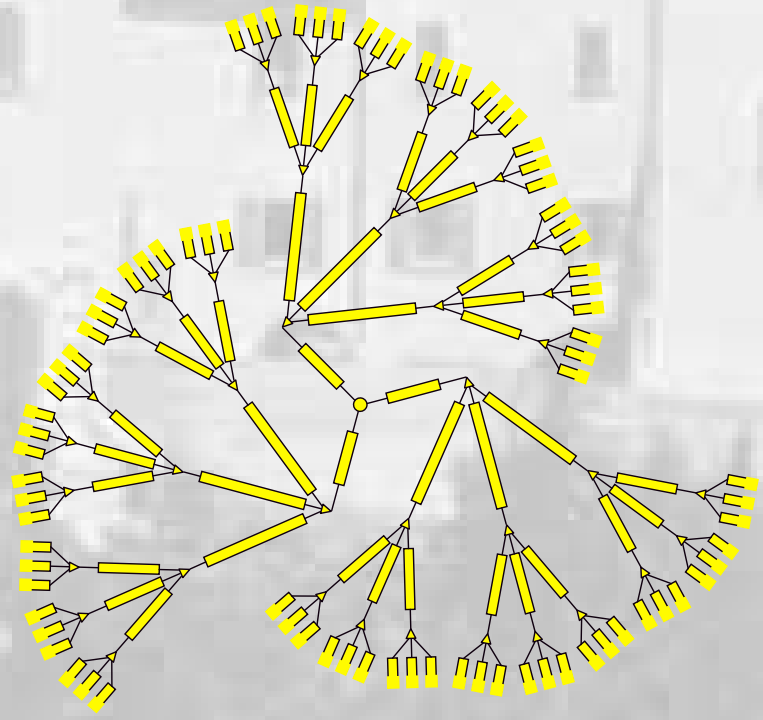
Cache Oblivious Sorting



Gerth Stølting Brodal

University of Aarhus

Algorithms and Data Structures, Bertinoro, Forlì, Italy, June 22-28, 2003

– Foundation

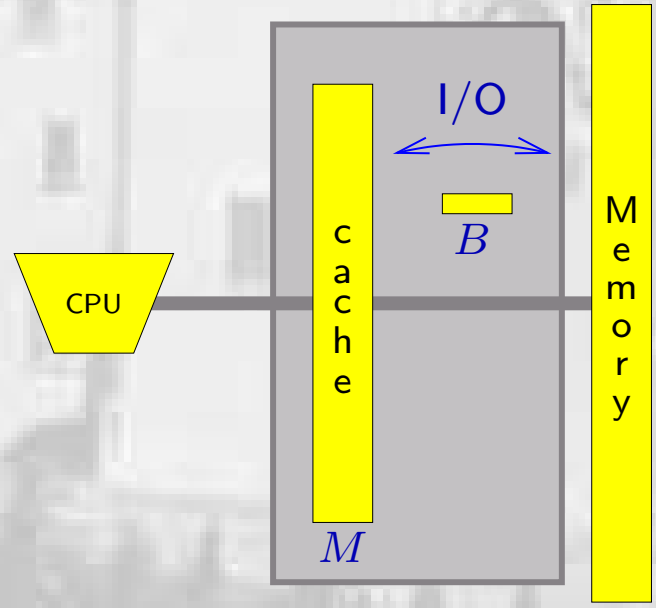
Outline of Talk

- Cache oblivious model
- Sorting problem
- Binary and multiway merge-sort
- Funnel-sort
- Lower bound — tall cache assumption
- Experimental results
- Conclusions

Cache Oblivious Model

Frigo, Leiserson, Prokop, Ramachandran, FOCS'99

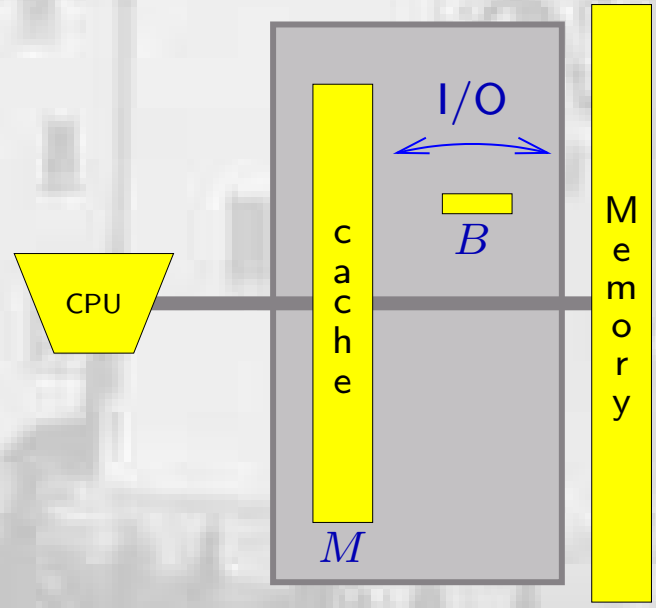
- Program in the RAM model
- Analyze in the I/O model for arbitrary B and M



Cache Oblivious Model

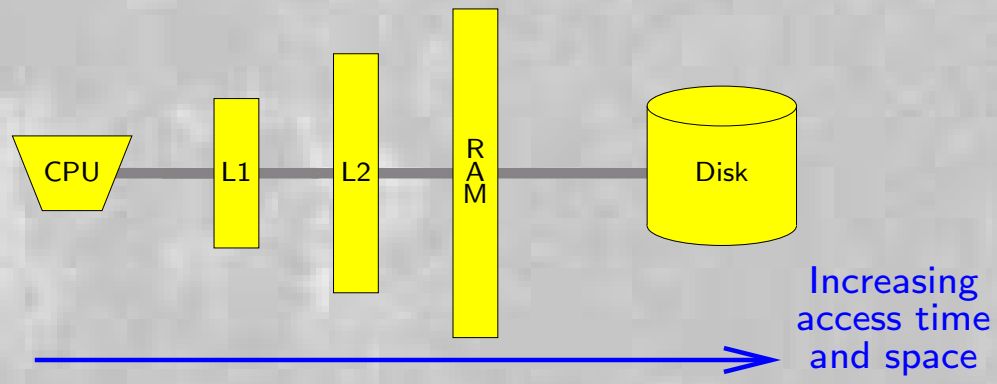
Frigo, Leiserson, Prokop, Ramachandran, FOCS'99

- Program in the RAM model
- Analyze in the I/O model for arbitrary B and M



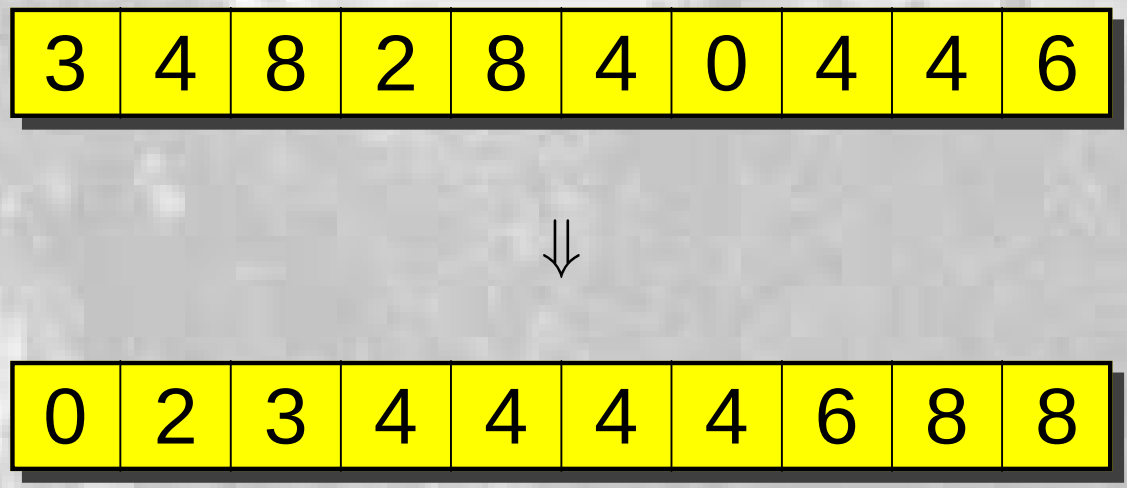
Advantages:

- Optimal on arbitrary level $\Rightarrow$ optimal on all levels
- Portability

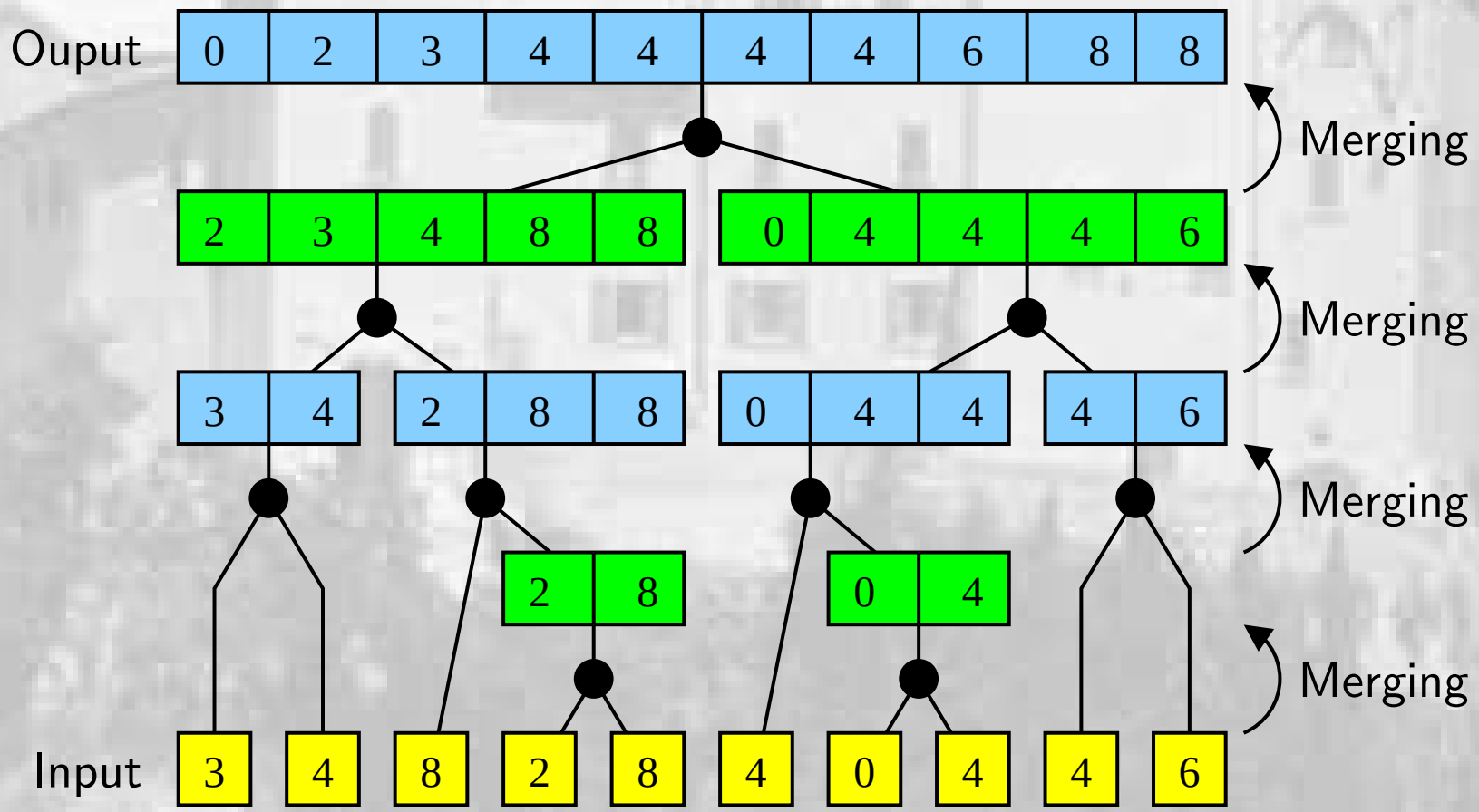


Sorting Problem

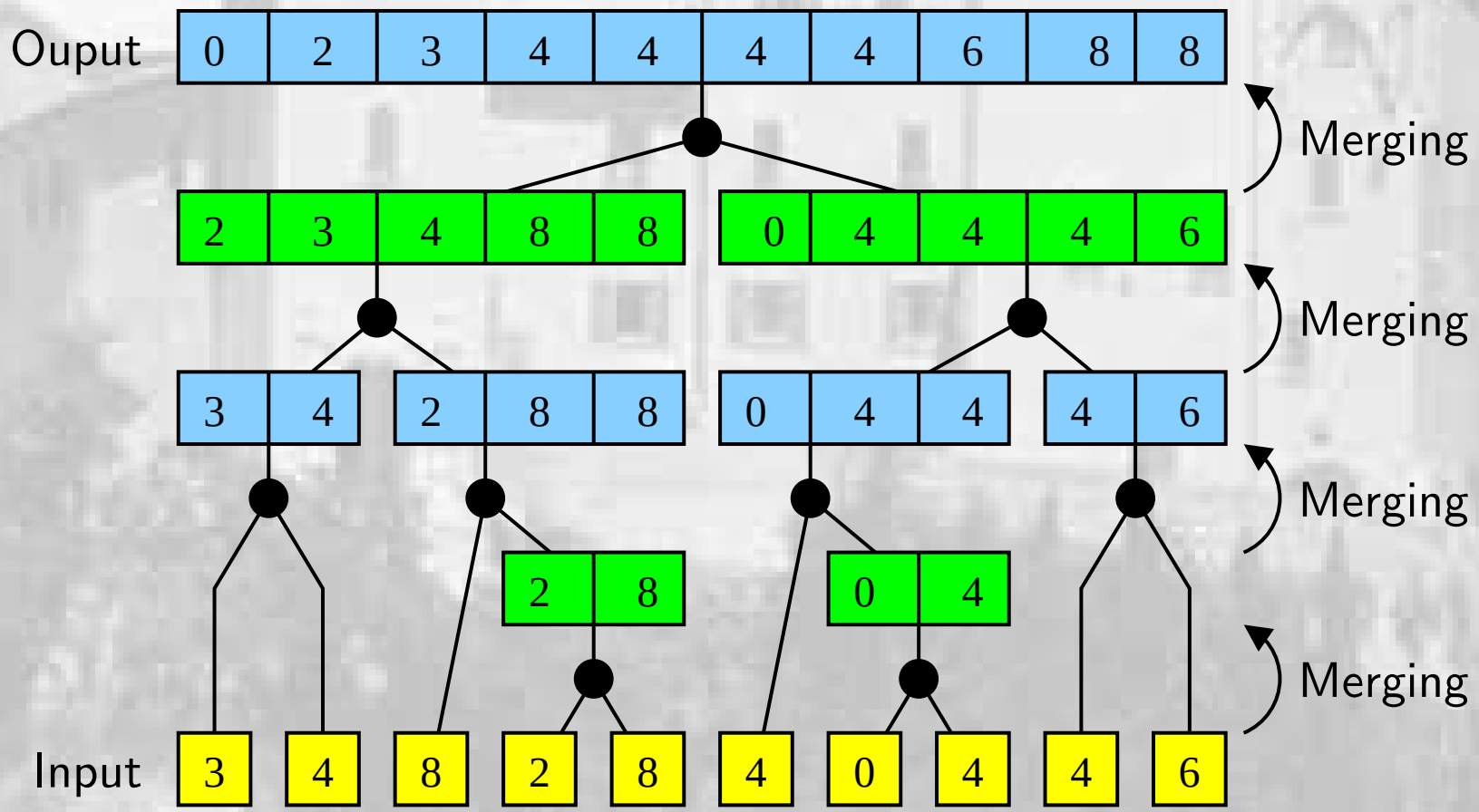
- Input : array containing $x_1, \ldots, x_N$
- Output : array with $x_1, \ldots, x_N$ in sorted order
- Elements can be compared and copied



Binary Merge-Sort



Binary Merge-Sort



- Recursive; two arrays; size $O(M)$ internally in cache
- $O(N \log N)$ comparisons
- $O\left(\frac{N}{B} \log_2 \frac{N}{M}\right)$ I/Os

Merge-Sort

Degree	I/O
2	$O\left(\frac{N}{B}\log_2 \frac{N}{M}\right)$
d $(d \leq \frac{M}{B} - 1)$	$O\left(\frac{N}{B}\log_d \frac{N}{M}\right)$
$\Theta\left(\frac{M}{B}\right)$	$O\left(\frac{N}{B}\log_{M/B} \frac{N}{M}\right) = O(\mathrm{Sort}_{M,B}(N))$

Aggarwal and Vitter 1988

Funnel-Sort

2 $(M \geq B^{1+\varepsilon})$	$O(\frac{1}{\varepsilon}\,\mathrm{Sort}_{M,B}(N))$

Frigo, Leiserson, Prokop and Ramachandran 1999

Brodal and Fagerberg 2002

Outline of Talk

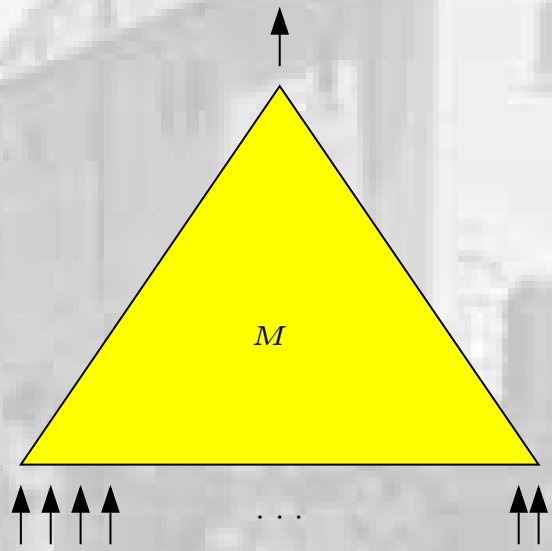
- Cache oblivious model
- Sorting problem
- Binary and multiway merge-sort
- Funnel-sort
- Lower bound — tall cache assumption
- Experimental results
- Conclusions

Funnel-Sort

k-merger

Frigo et al., FOCS'99

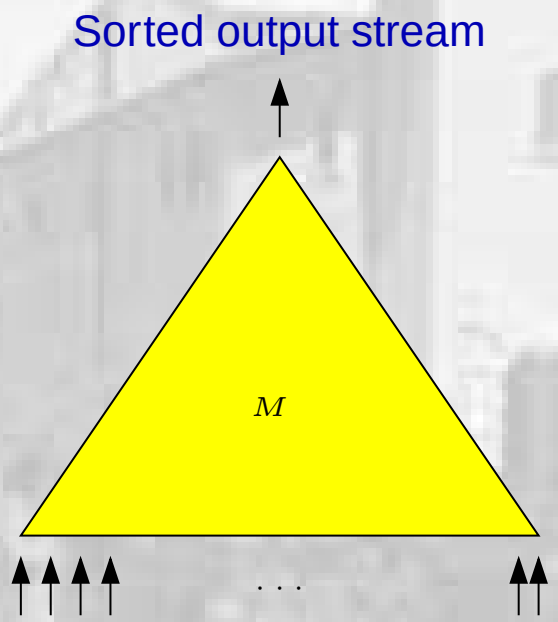
Sorted output stream



k sorted input streams

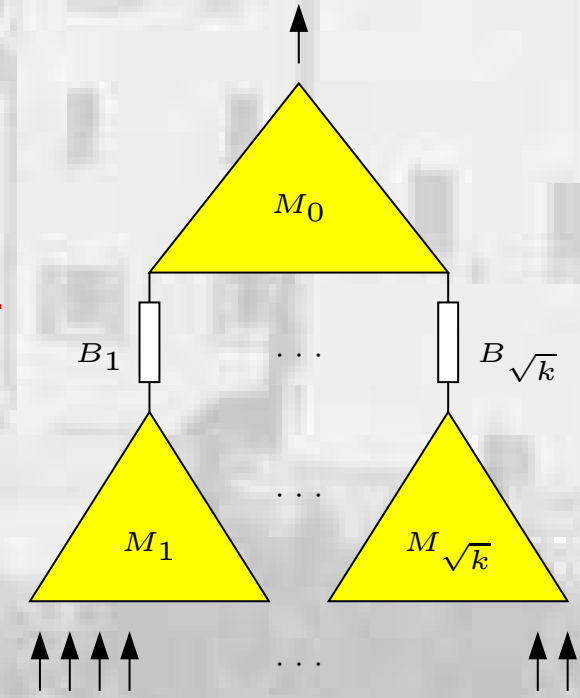
k-merger

Frigo et al., FOCS'99



Recursive def.

=

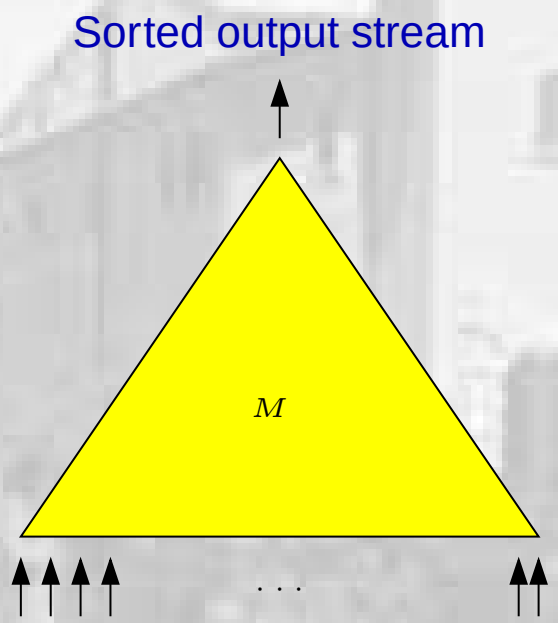


← $k^{1/2}$-mergers

← buffers of size $k^{3/2}$

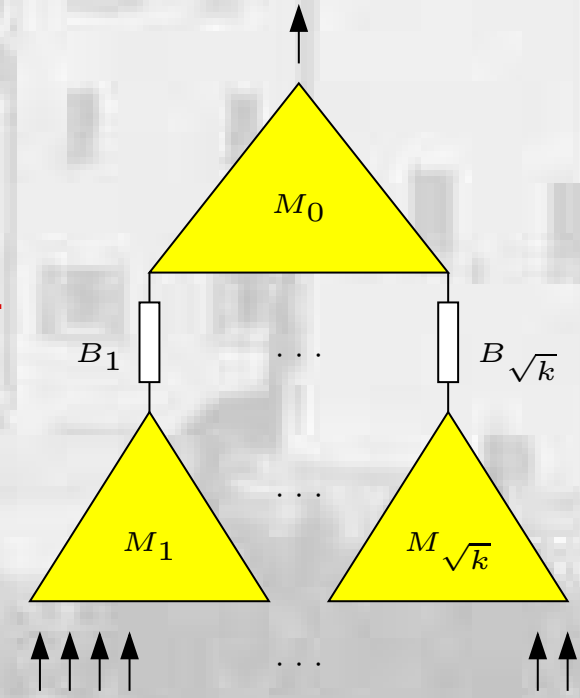
k-merger

Frigo et al., FOCS'99



Recursive def.

=



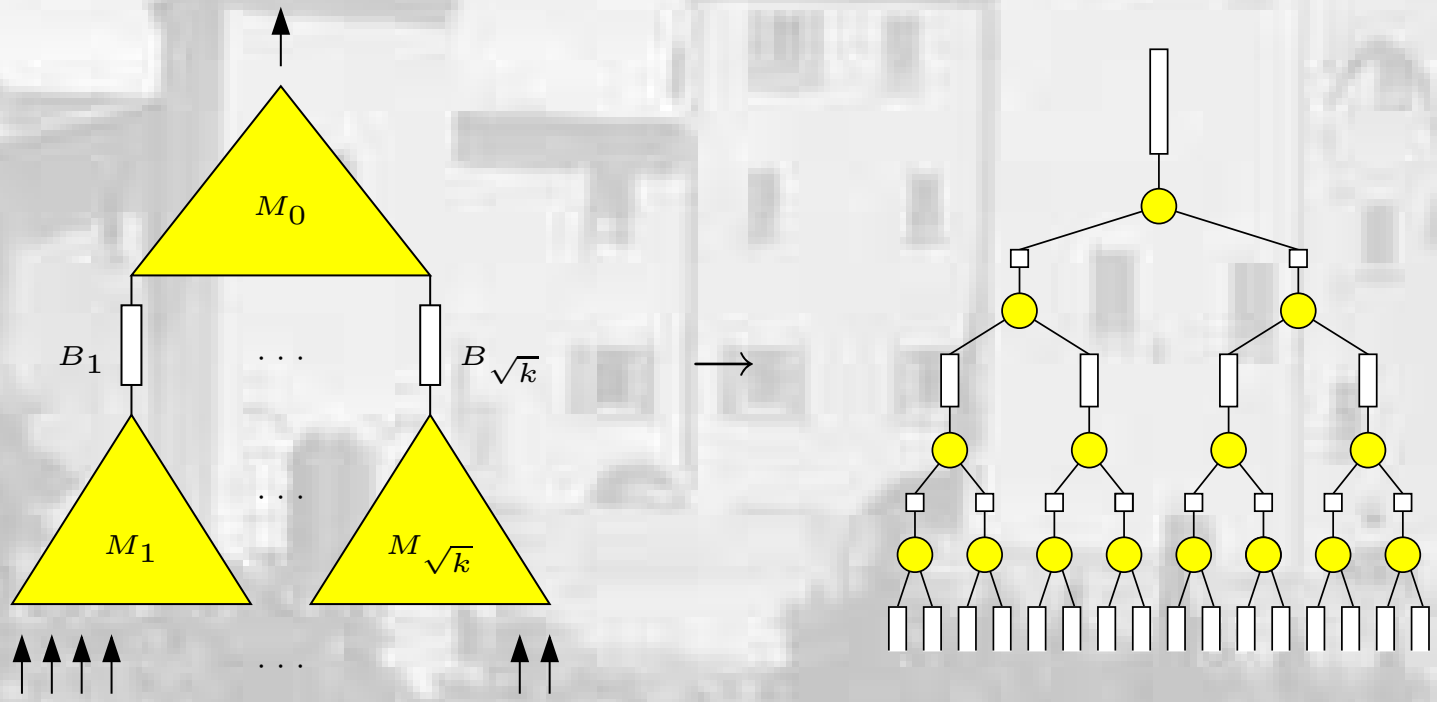
← $k^{1/2}$-mergers

← buffers of size $k^{3/2}$

Recursive Layout

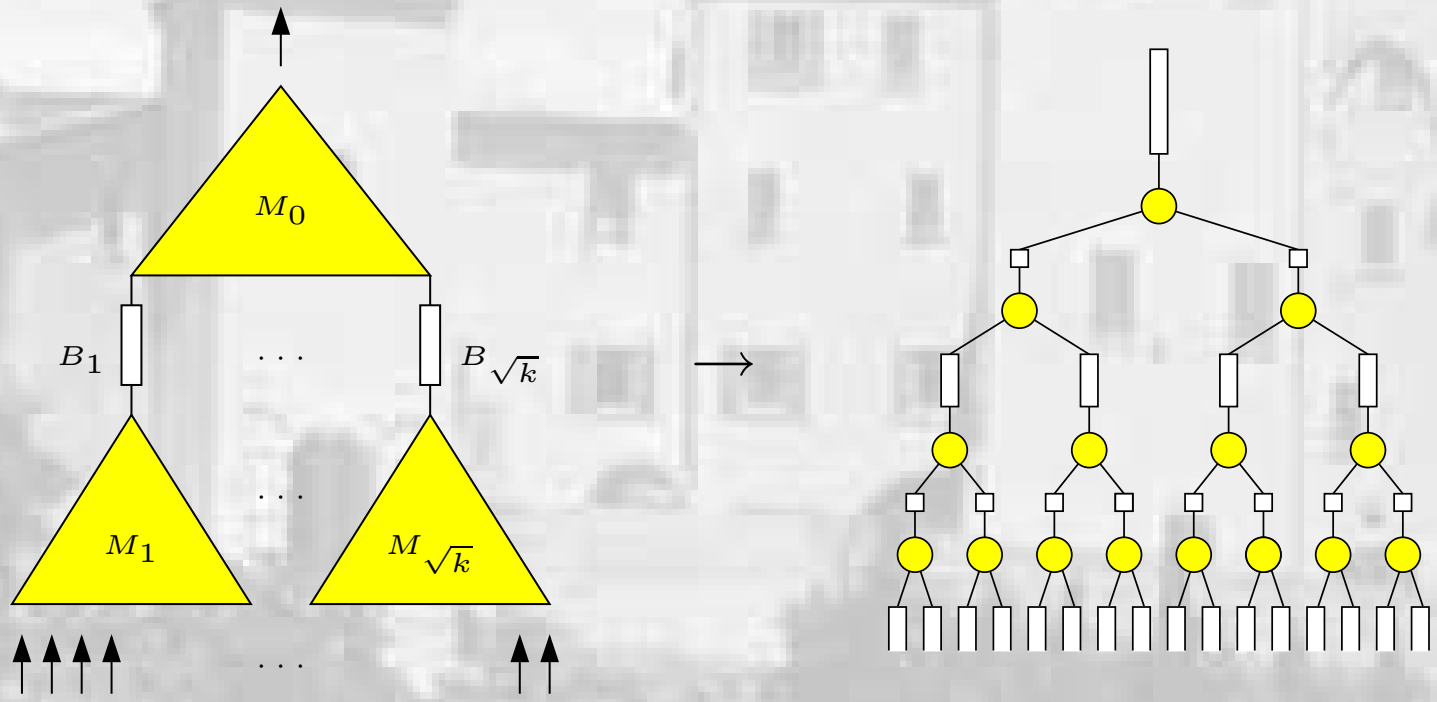
Lazy k-merger

Brodal and Fagerberg 2002



Lazy k-merger

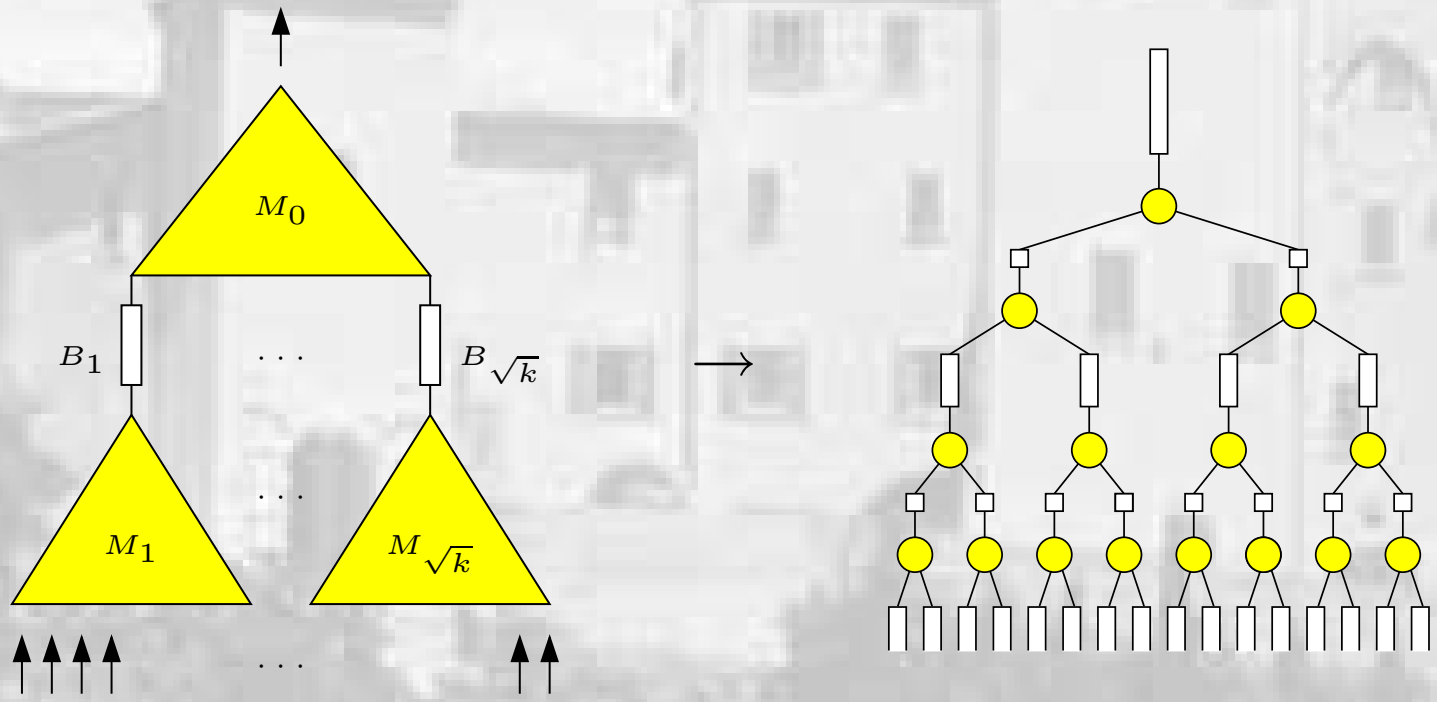
Brodal and Fagerberg 2002



Procedure **Fill**(v)
  while out-buffer not full
    if left in-buffer empty
      Fill(left child)
    if right in-buffer empty
      Fill(right child)
    perform one merge step

Lazy k-merger

Brodal and Fagerberg 2002



Procedure **Fill**(v)
  while out-buffer not full
    if left in-buffer empty
      Fill(left child)
    if right in-buffer empty
      Fill(right child)
    perform one merge step

Lemma

If $M \geq B^2$ and output buffer has size k^3 then $O(\frac{k^3}{B}\log_M(k^3) + k)$ I/Os are done during an invocation of **Fill**(root)

Funnel-Sort

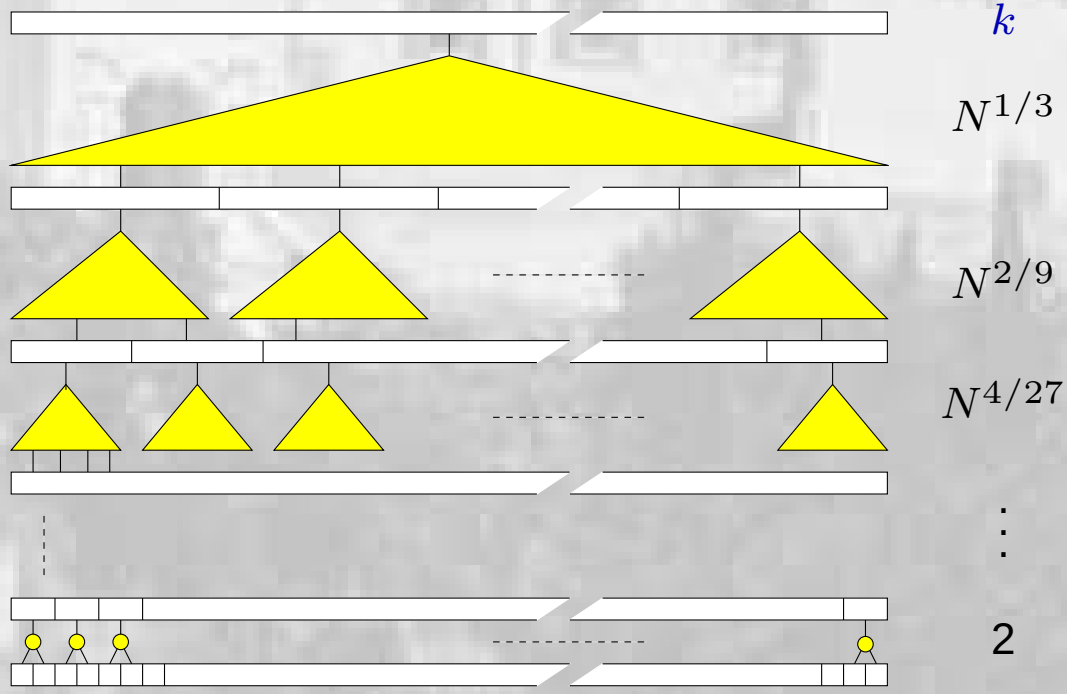
Brodal and Fagerberg 2002

Frigo, Leiserson, Prokop and Ramachandran 1999

Divide input in $N^{1/3}$ segments of size $N^{2/3}$

Recursively **MergeSort** each segment

Merge sorted segments by an $N^{1/3}$-merger



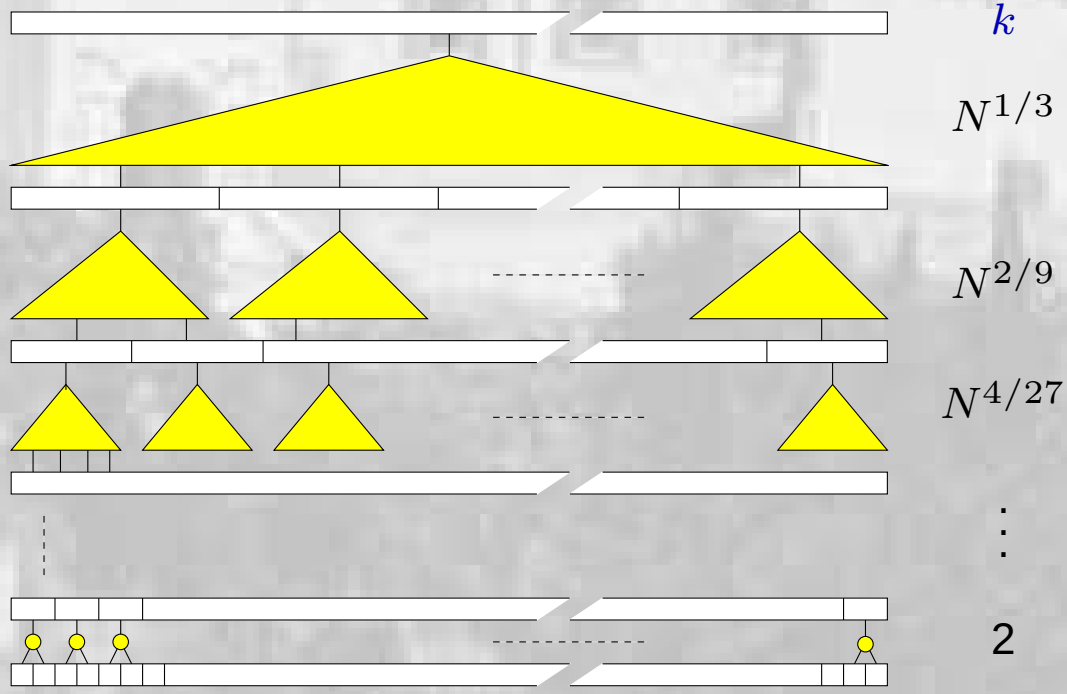
Funnel-Sort

Brodal and Fagerberg 2002
Frigo, Leiserson, Prokop and Ramachandran 1999

Divide input in $N^{1/3}$ segments of size $N^{2/3}$

Recursively **MergeSort** each segment

Merge sorted segments by an $N^{1/3}$-merger



Theorem Funnel-Sort performs $O(\mathrm{Sort}_{M,B}(N))$ I/Os for $M \geq B^2$

Outline of Talk

- Cache oblivious model
- Sorting problem
- Binary and multiway merge-sort
- Funnel-sort
- Lower bound — tall cache assumption
- Experimental results
- Conclusions

Lower Bound

Brodal and Fagerberg 2003

	Block Size	Memory	I/Os
Machine 1	B_1	M	t_1
Machine 2	B_2	M	t_2

One algorithm, two machines, $B_1 \leq B_2$

Trade-off

$$8t_1B_1 + 3t_1B_1 \log \frac{8Mt_2}{t_1B_1} \geq N \log \frac{N}{M} - 1.45N$$

Lower Bound

	Assumption	I/Os	
Lazy Funnel-sort	$B \leq M^{1-\varepsilon}$	$(a)\ B_2 = M^{1-\varepsilon}$:	$\mathrm{Sort}_{B_2,M}(N)$
		$(b)\ B_1 = 1$:	$\mathrm{Sort}_{B_1,M}(N) \cdot \frac{1}{\varepsilon}$
Binary Merge-sort	$B \leq M/2$	$(a)\ B_2 = M/2$:	$\mathrm{Sort}_{B_2,M}(N)$
		$(b)\ B_1 = 1$:	$\mathrm{Sort}_{B_1,M}(N) \cdot \log M$

Corollary $(a) \Rightarrow (b)$

Fake Proof

Goal:

$$8t_1B_1 + 3t_1B_1 \log \frac{8Mt_2}{t_1B_1} \geq N \log \frac{N}{M} - 1.45N$$

Merging sorted lists X and Y takes $\approx |X| \log \frac{|Y|}{|X|}$ comparisons

In total t_1B_1 elements touched $\Rightarrow$ t_1B_1/t_2 elements touched on average per B_2-I/O $\Rightarrow$ effective B_2 is t_1B_1/t_2

Comparisons gained per B_2-I/O:

$$t_1B_1/t_2 \cdot \log \frac{M}{t_1B_1/t_2}$$

Hence:

$$t_1B_1 \cdot \log \frac{Mt_2}{t_1B_1} \geq N \log N - 1.45N$$

Fake Proof

Goal:

$$8t_1B_1 + 3t_1B_1 \log \frac{8Mt_2}{t_1B_1} \geq N \log \frac{N}{M} - 1.45N$$

Merging sorted lists X and Y takes $\approx |X| \log \frac{|Y|}{|X|}$ comparisons

In total t_1B_1 elements touched $\Rightarrow$ t_1B_1/t_2 elements touched on average per B_2-I/O $\Rightarrow$ effective B_2 is t_1B_1/t_2

Comparisons gained per B_2-I/O:

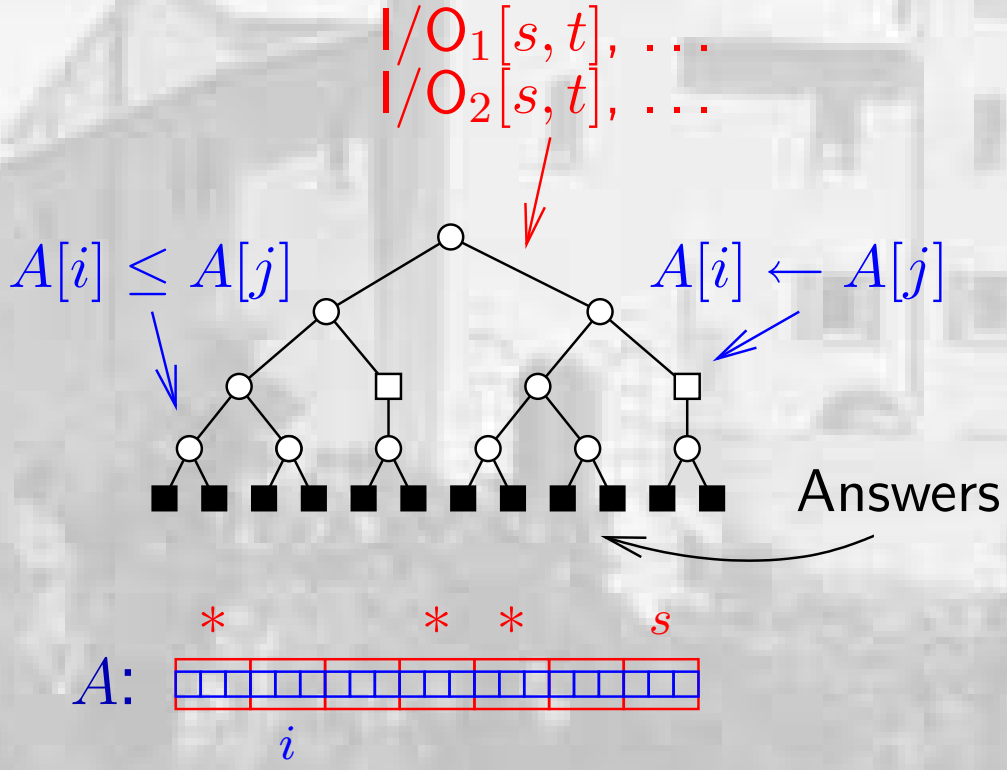
$$t_1B_1/t_2 \cdot \log \frac{M}{t_1B_1/t_2}$$

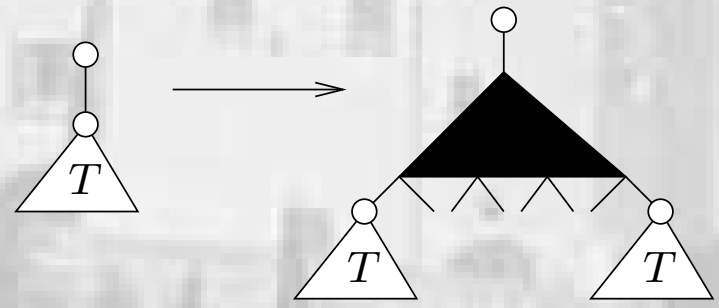
One problem : Online choice

Hence:

$$t_1B_1 \cdot \log \frac{Mt_2}{t_1B_1} \geq N \log N - 1.45N$$

Ideas from Real Proof





$$8t_1B_1 + 3t_1B_1 \log \frac{8Mt_2}{B_1t_1} \geq \text{height} \geq N \log \frac{N}{M} - 1.45N$$

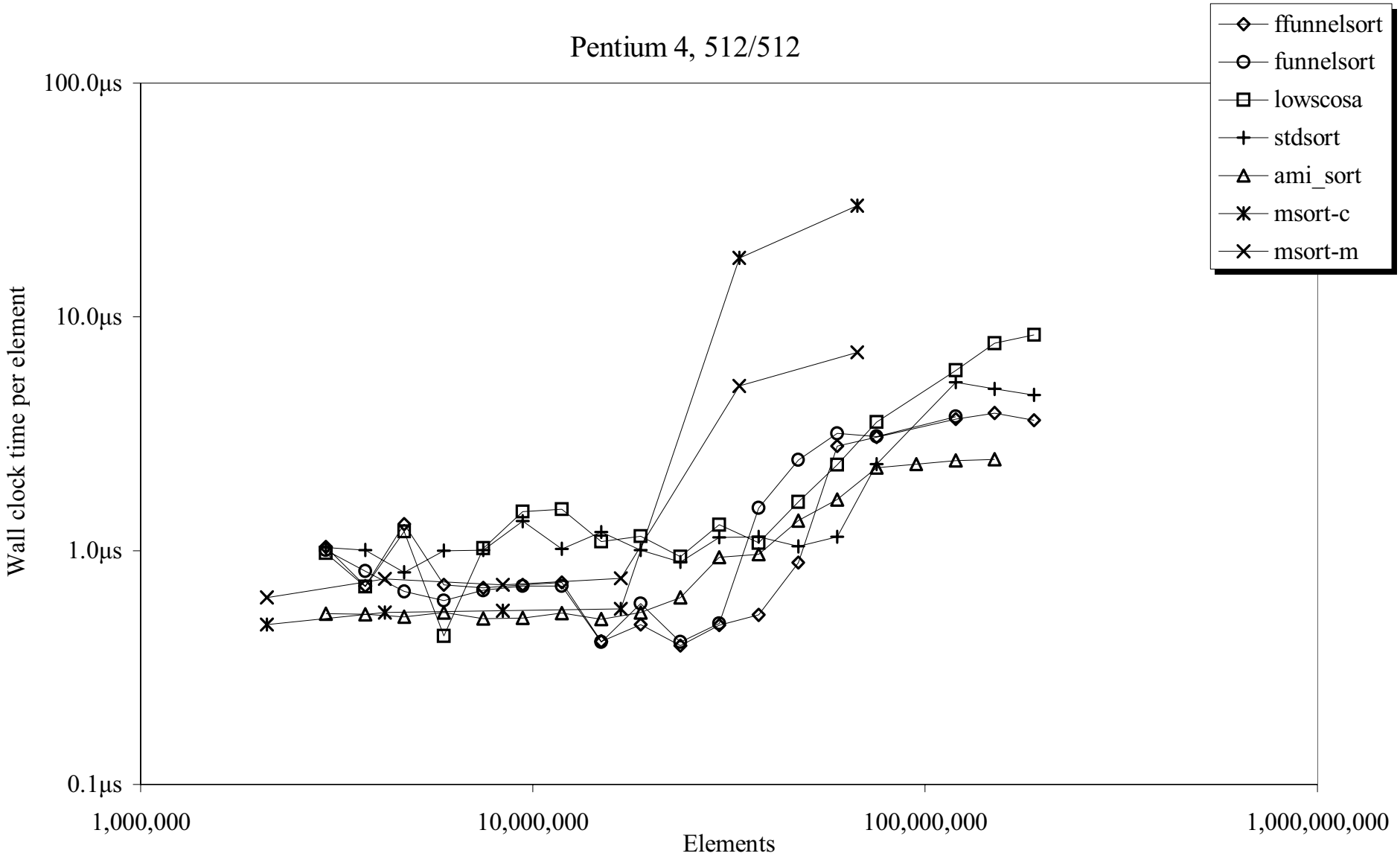
Outline of Talk

- Cache oblivious model
- Sorting problem
- Binary and multiway merge-sort
- Funnel-sort
- Lower bound — tall cache assumption
- Experimental results
- Conclusions

Hardware

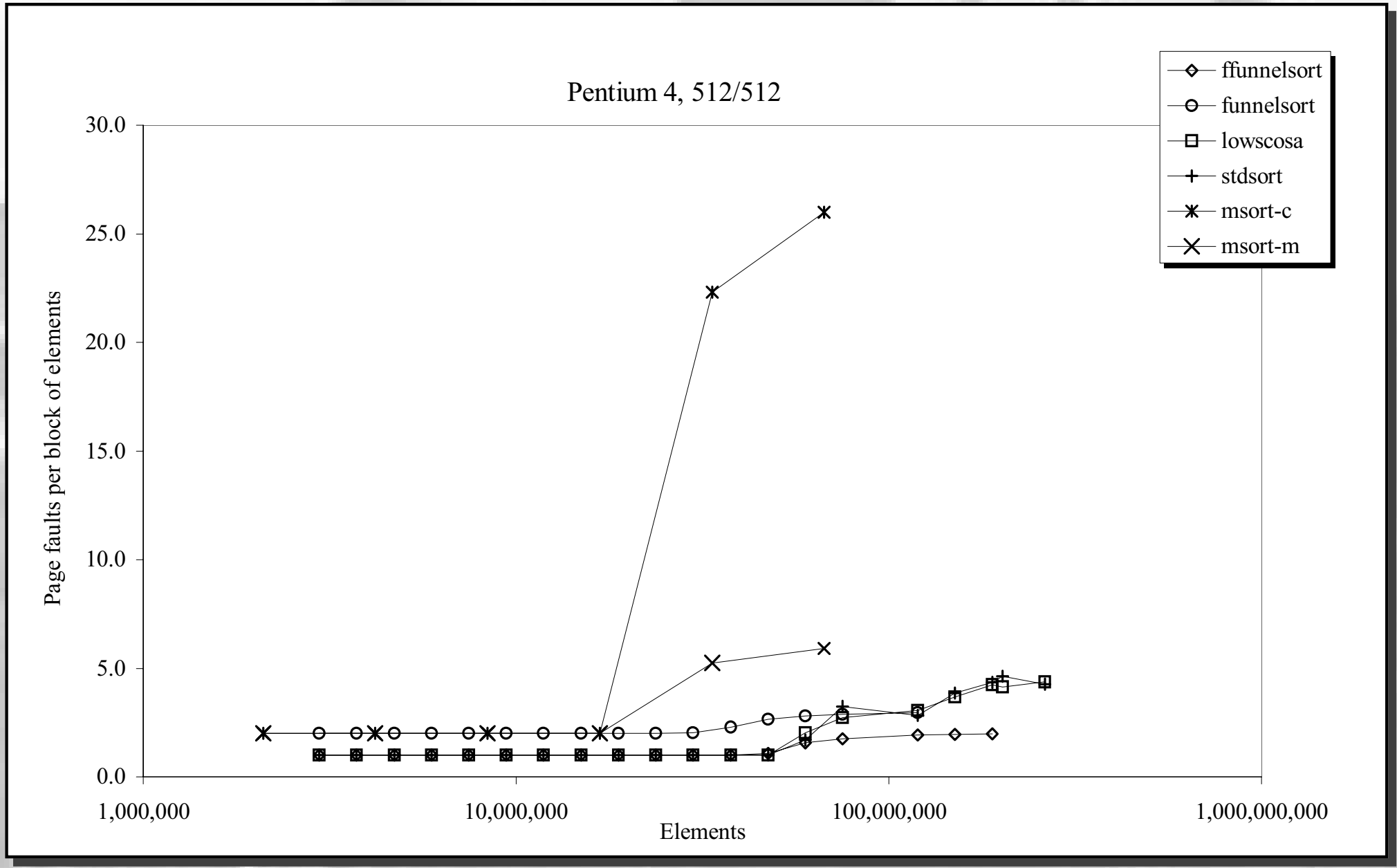
Processor type	Pentium 4	Pentium 3	MIPS 10000
Workstation	Dell PC	Delta PC	SGI Octane
Operating system	GNU/Linux Kernel version 2.4.18	GNU/Linux Kernel version 2.4.18	IRIX version 6.5
Clock rate	2400 MHz	800 MHz	175 MHz
Address space	32 bit	32 bit	64 bit
Integer pipeline stages	20	12	6
L1 data cache size	8 KB	16 KB	32 KB
L1 line size	128 Bytes	32 Bytes	32 Bytes
L1 associativity	4 way	4 way	2 way
L2 cache size	512 KB	256 KB	1024 KB
L2 line size	128 Bytes	32 Bytes	32 Bytes
L2 associativity	8 way	4 way	2 way
TLB entries	128	64	64
TLB associativity	Full	4 way	64 way
TLB miss handler	Hardware	Hardware	Software
Main memory	512 MB	256 MB	128 MB

Wall Clock



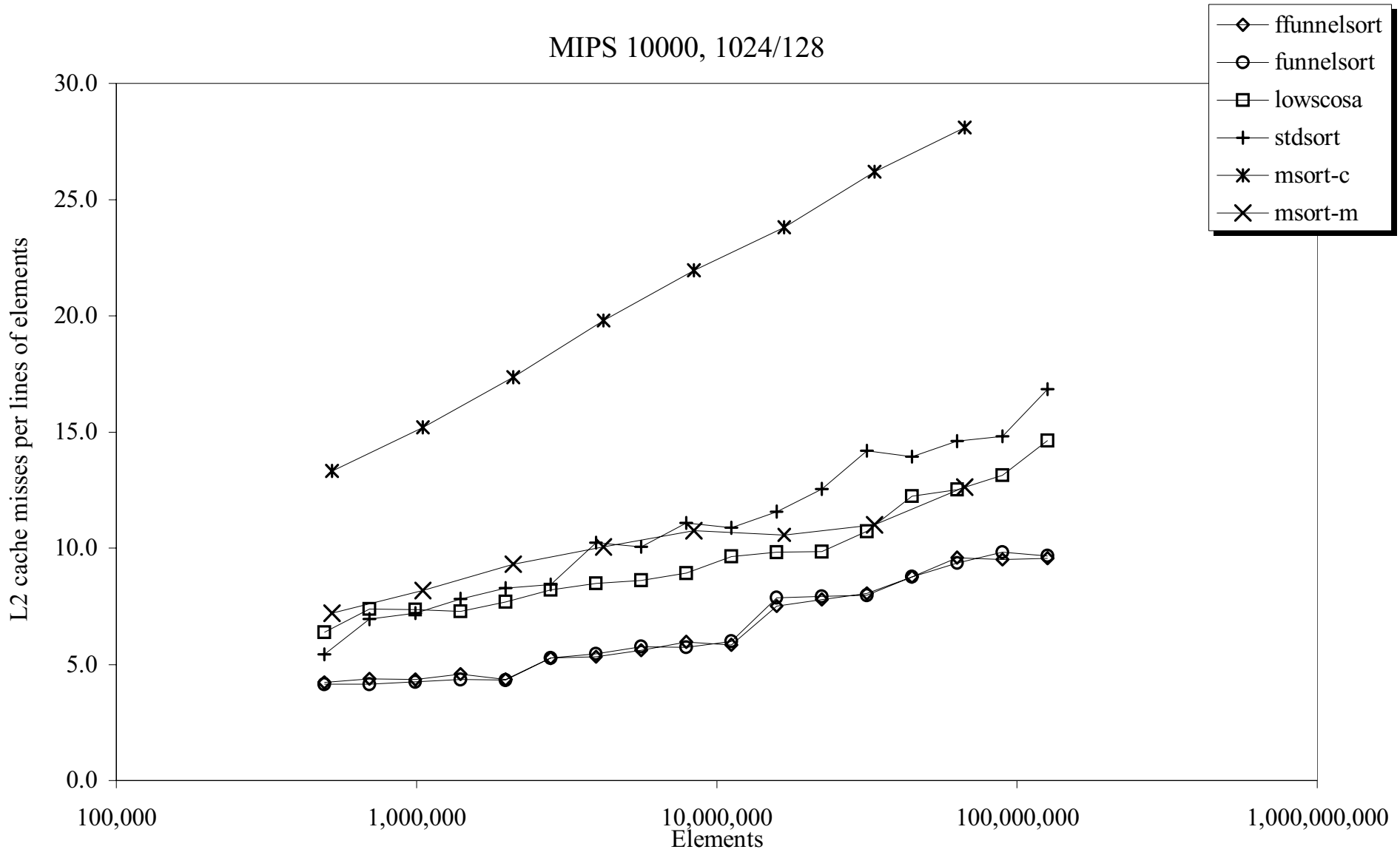
Kristoffer Vinther 2003

Page Faults



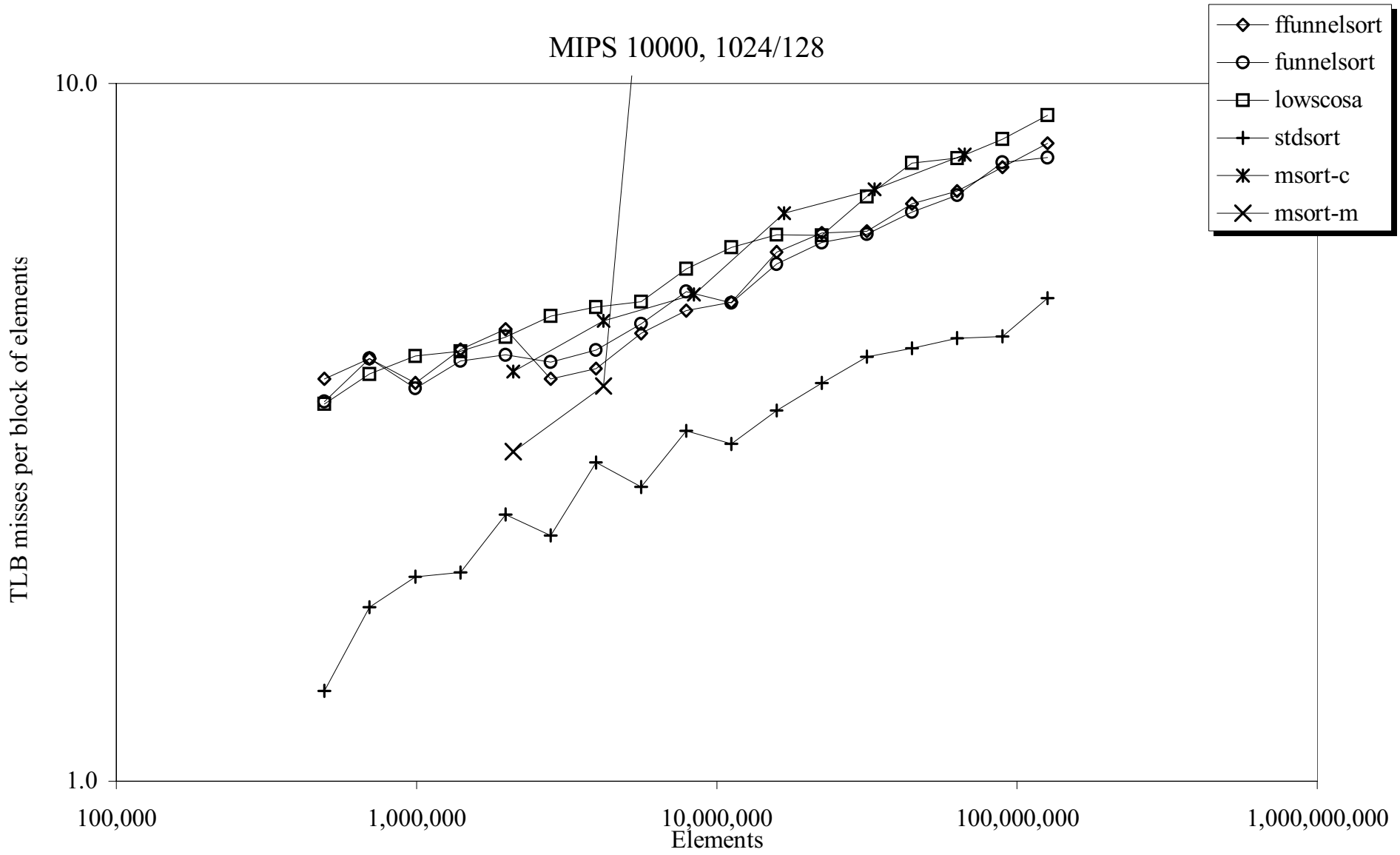
Kristoffer Vinther 2003

Cache Misses



Kristoffer Vinther 2003

TLB Misses



Kristoffer Vinther 2003

Outline of Talk

- Cache oblivious model
- Sorting problem
- Binary and multiway merge-sort
- Funnel-sort
- Lower bound — tall cache assumption
- Experimental results
- Conclusions

Conclusions

Cache oblivious sorting

- is possible
- requires a tall cache assumption $M \geq B^{1+\varepsilon}$
- comparable performance with cache aware algorithms

Future work

- more experimental justification for the cache oblivious model
- limitations of the model — time space trade-offs ?
- tool-box for cache oblivious algorithms