Karlsruhe Institute of Technology

In-place (Parallel) Super Scalar Samplesort

Algorithm Engineering Lecture · June 13, 2017

Michael Axtmann, Sascha Witt, Daniel Ferizovic, Peter Sanders

Institute of Theoretical Informatics · Algorithmics Group

Overview

	Quicksort	BlockQuicksort	SSSS	ISSSS
Decoupled control and data flow	no	yes	yes	yes
Conditional branches	yes	no	no	no
Data transfers / element	$\approx \log_2 n$	$\approx \log_2 n$	$\approx \log_k n$	$\approx \log_k n$
Additional space	$O(1)$	$O(b)$	$O(n)$	$O(kb)$
Parallelization	yes	no	no	yes

Institute of Theoretical Informatics
Algorithmics Group

BlockQuicksort

Goals

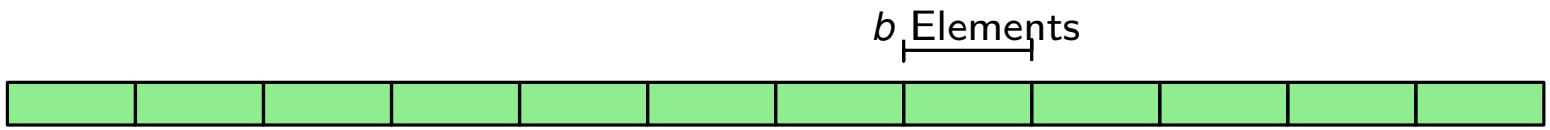
- Partially decoupling control flow from data flow
- Avoid conditional branches
- In-place: $O(b)$ additional space

Institute of Theoretical Informatics
Algorithmics Group

BlockQuicksort

Goals

- Partially decoupling control flow from data flow
- Avoid conditional branches
- In-place: $O(b)$ additional space

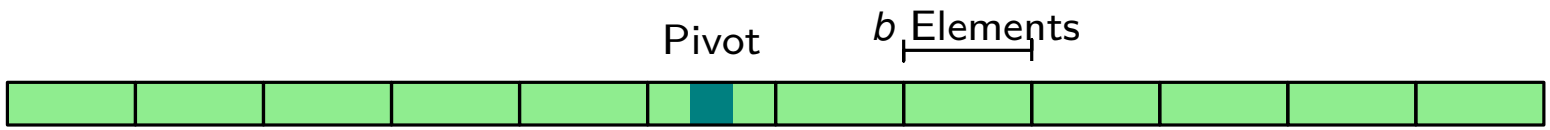


Institute of Theoretical Informatics
Algorithmics Group

BlockQuicksort

Goals

- Partially decoupling control flow from data flow
- Avoid conditional branches
- In-place: $O(b)$ additional space

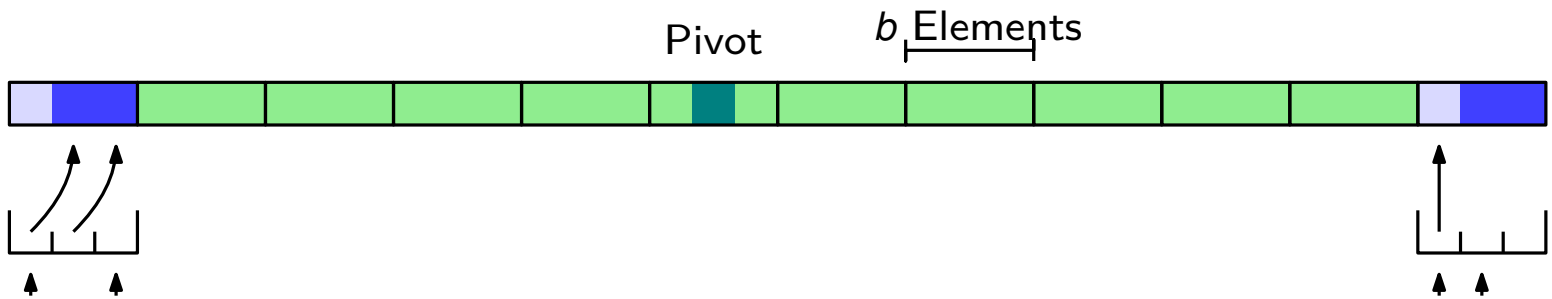


Institute of Theoretical Informatics
Algorithmics Group

BlockQuicksort

Goals

- Partially decoupling control flow from data flow
- Avoid conditional branches
- In-place: $O(b)$ additional space

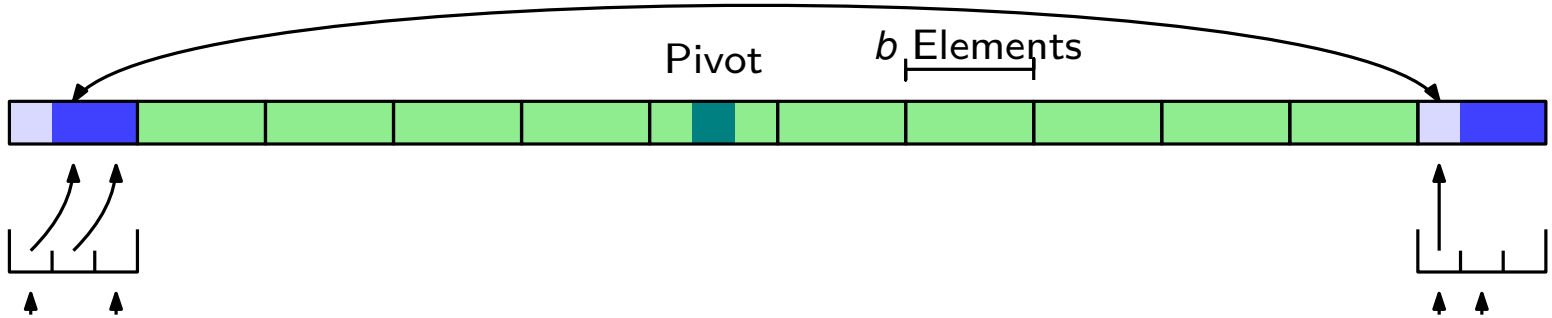


Institute of Theoretical Informatics
Algorithmics Group

BlockQuicksort

Goals

- Partially decoupling control flow from data flow
- Avoid conditional branches
- In-place: $O(b)$ additional space

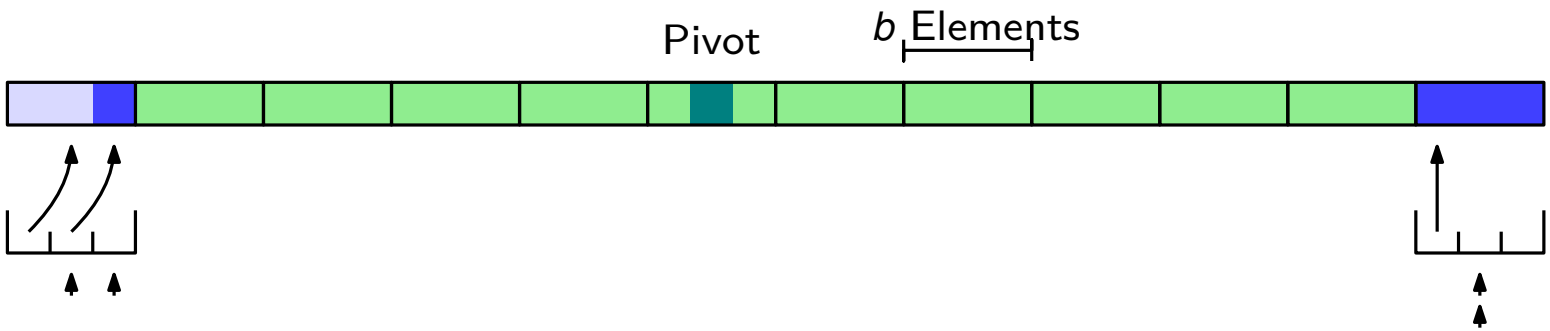


Institute of Theoretical Informatics
Algorithmics Group

BlockQuicksort

Goals

- Partially decoupling control flow from data flow
- Avoid conditional branches
- In-place: $O(b)$ additional space

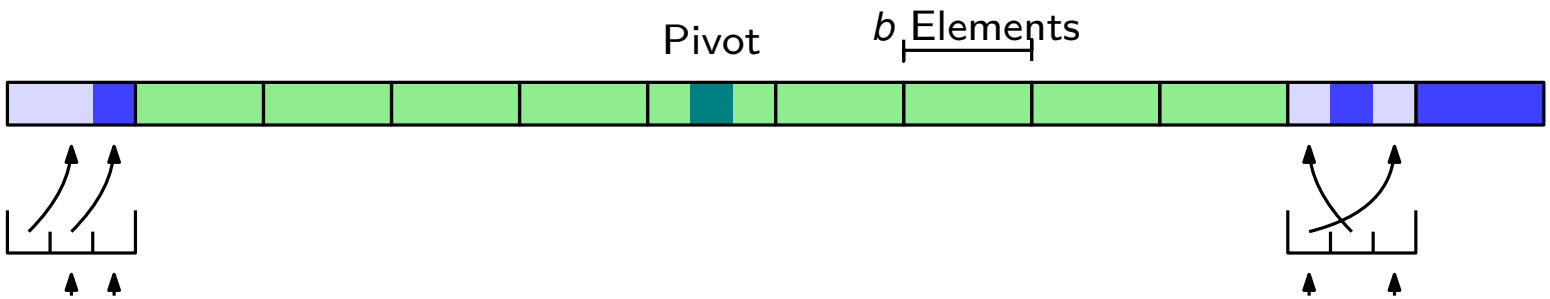


Institute of Theoretical Informatics
Algorithmics Group

BlockQuicksort

Goals

- Partially decoupling control flow from data flow
- Avoid conditional branches
- In-place: $O(b)$ additional space

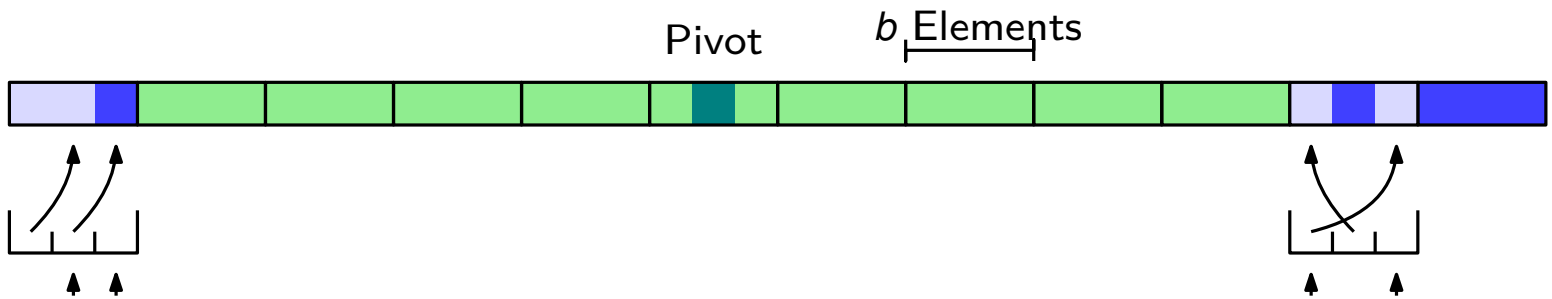


Institute of Theoretical Informatics
Algorithmics Group

BlockQuicksort

Goals

- Partially decoupling control flow from data flow
- Avoid conditional branches
- In-place: $O(b)$ additional space



Drawbacks

- $O\left(\frac{n}{b}\log_2\frac{n}{n_0}\right)$ block transfers

Institute of Theoretical Informatics
Algorithmics Group

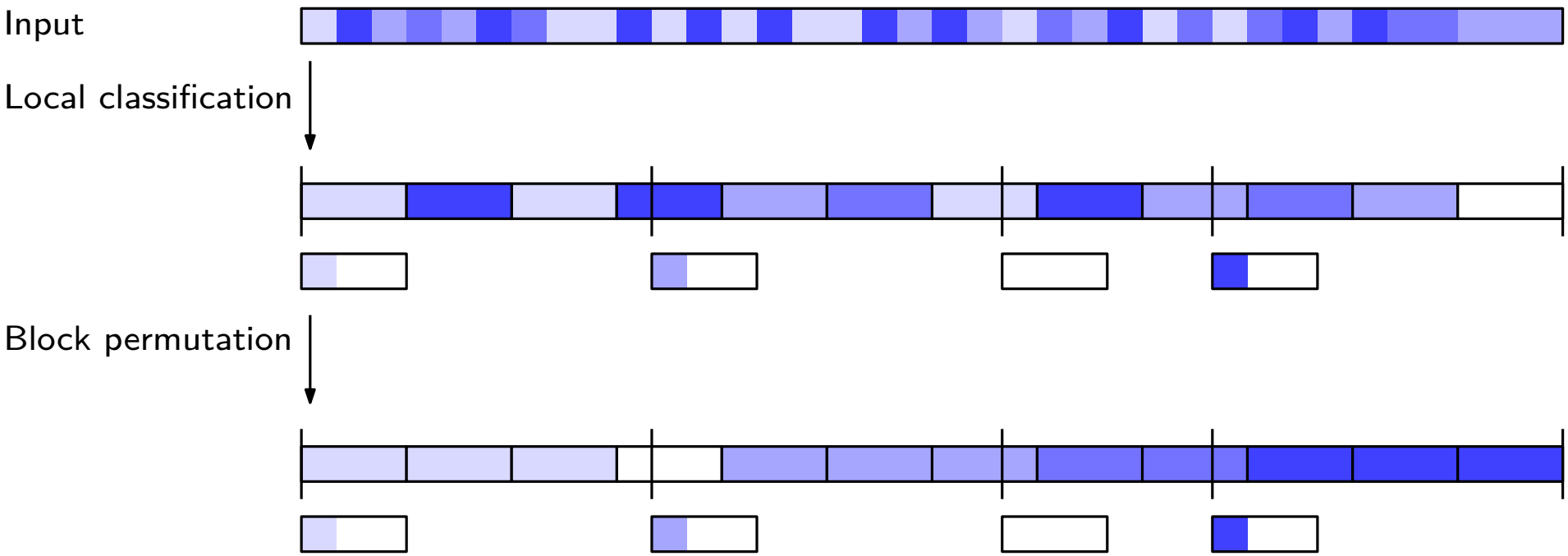
In-place Super Scalar Samplesort

Goals

- Partially decoupling control flow from data flow
- Avoid conditional branches
- k-way distribution
- Cache/IO-efficient
  - $O\left(\frac{n}{tb}\log_k \frac{n}{n_0}\right)$ block transfers
- In-place: $O(kb)$ additional space
- Easy to parallelize

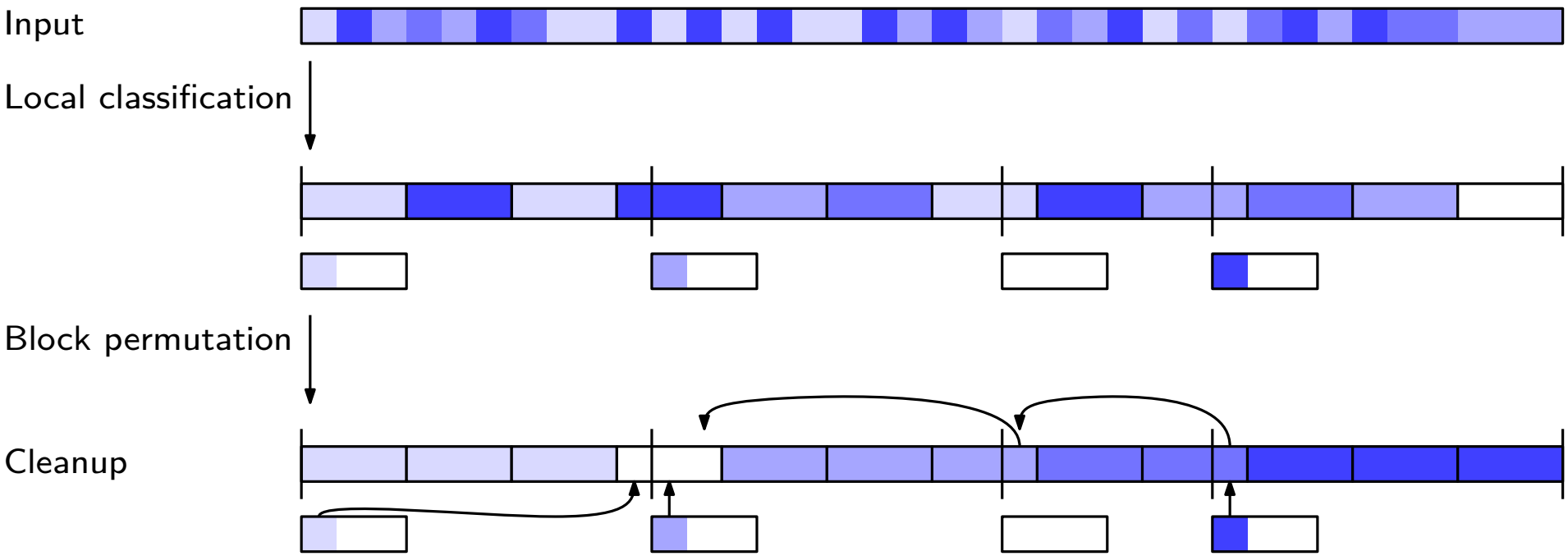
Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort



Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

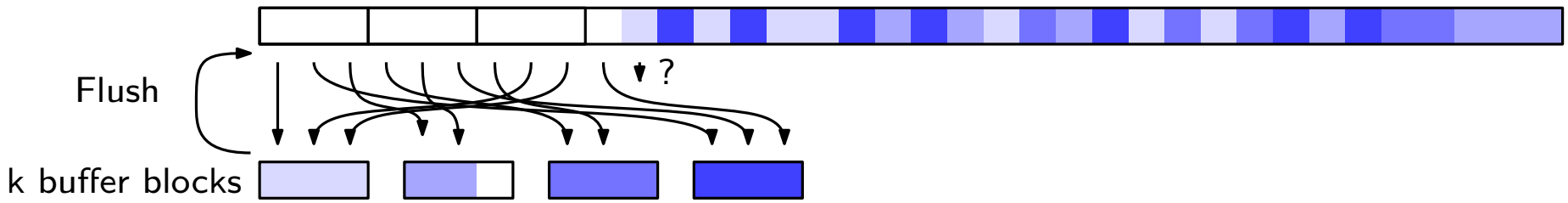


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Local classification

- k-way decision tree without branches
- k buffer blocks of size B
- Flush buffer block if charged



Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

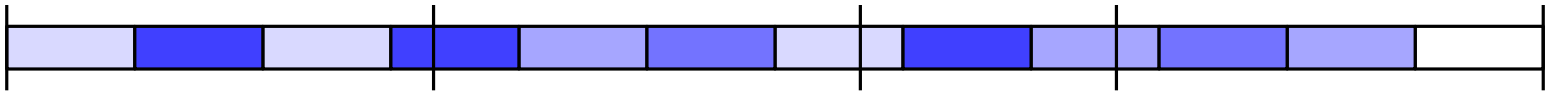
- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

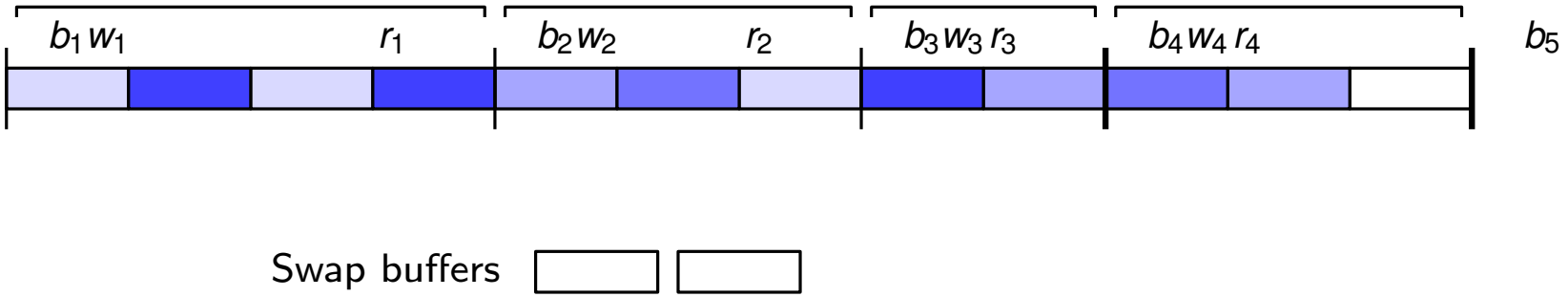


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

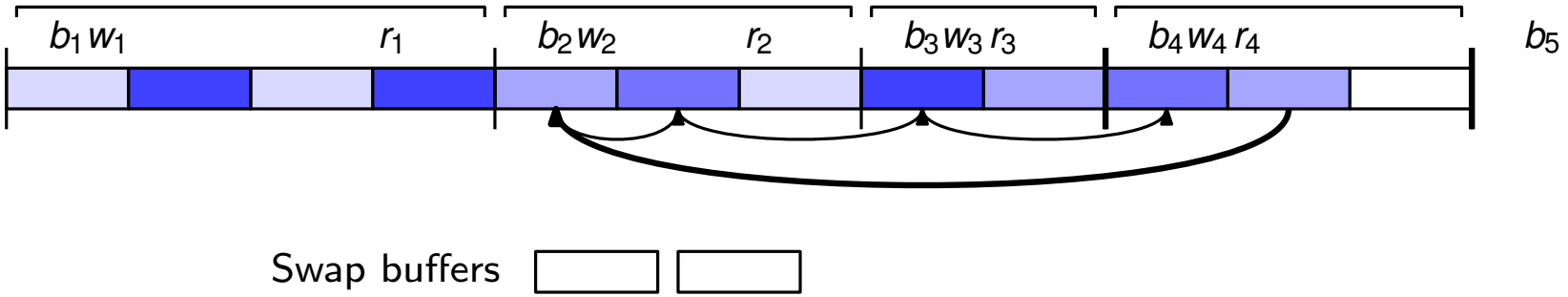


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

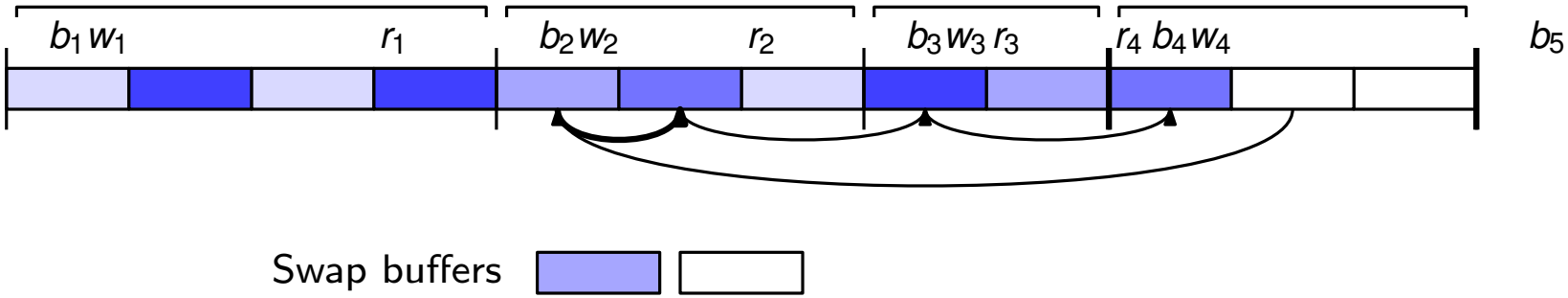


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

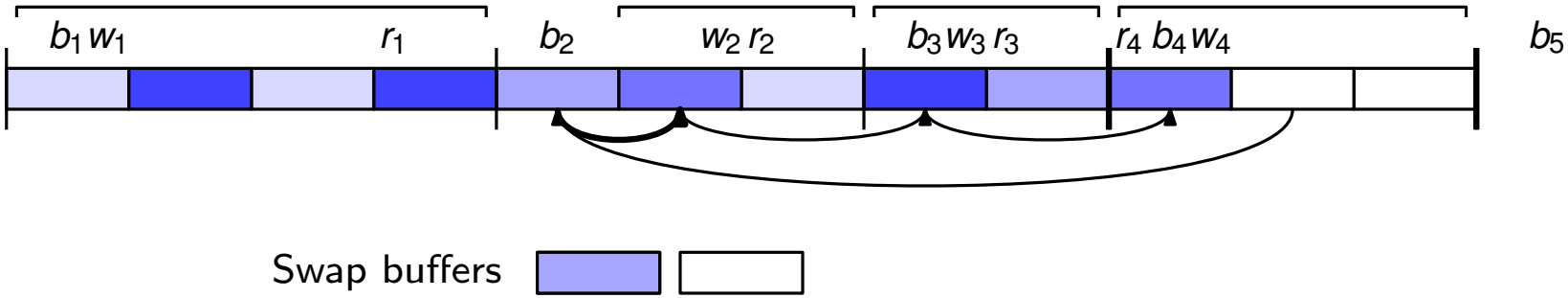


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

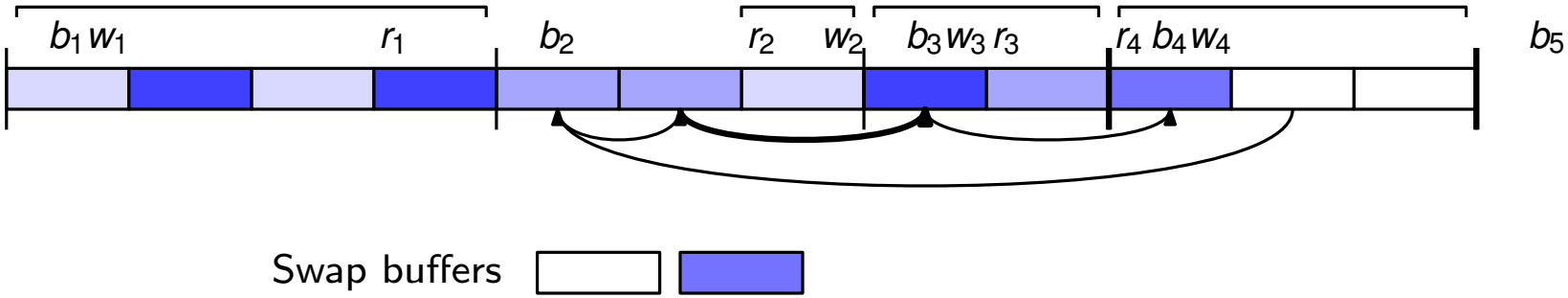


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

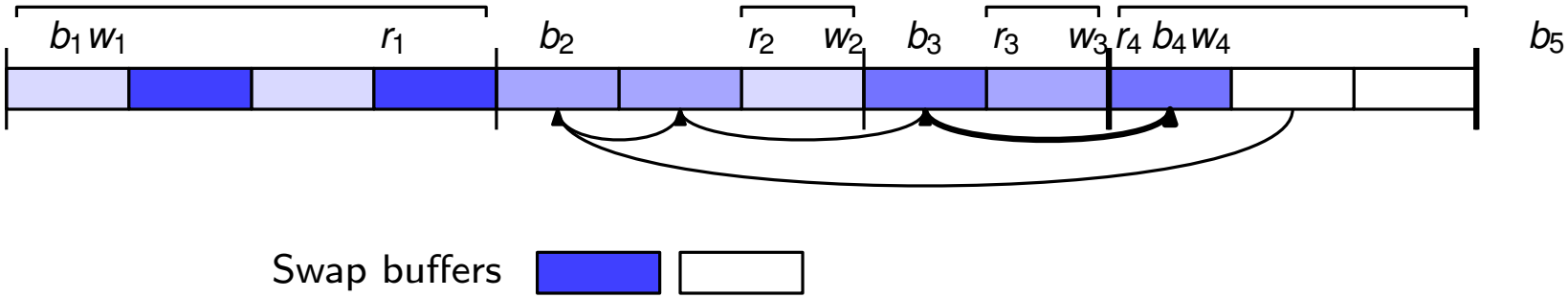


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

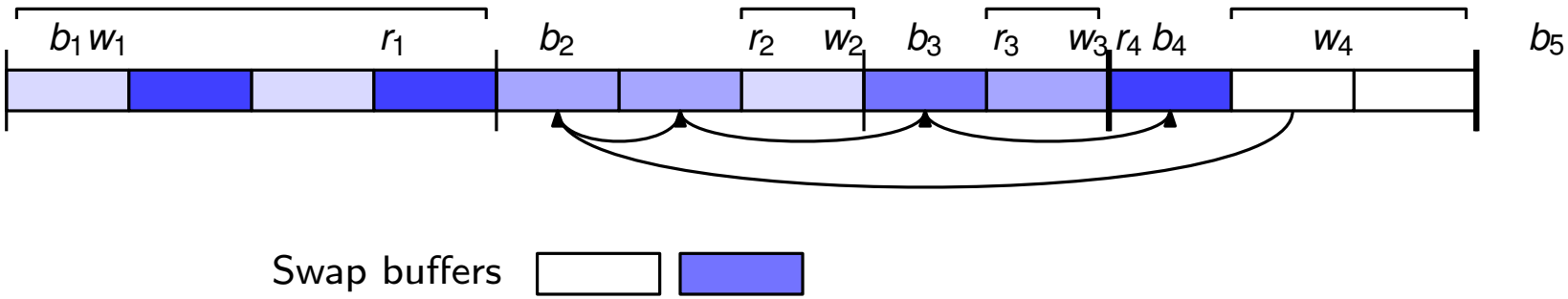


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

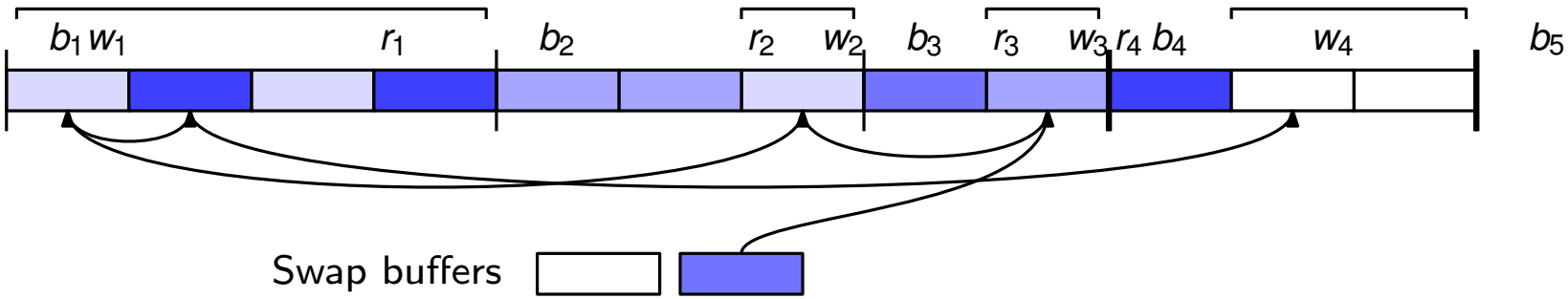


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

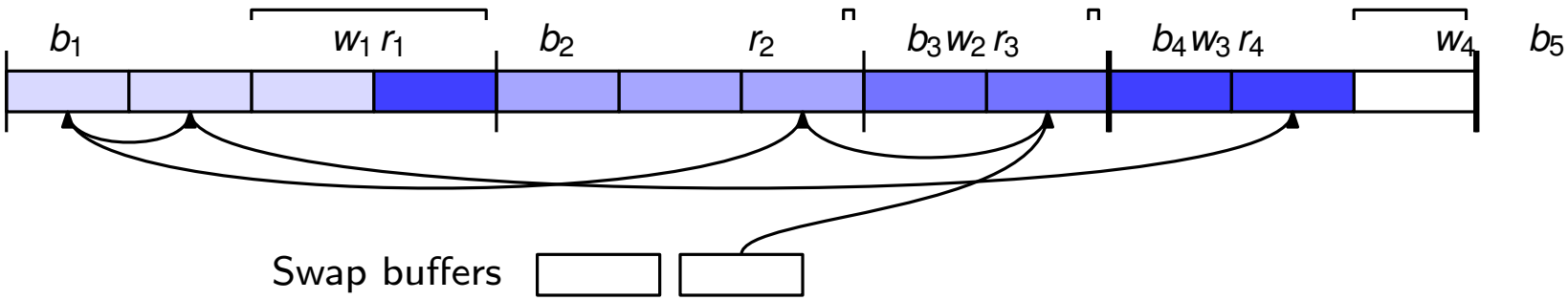


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

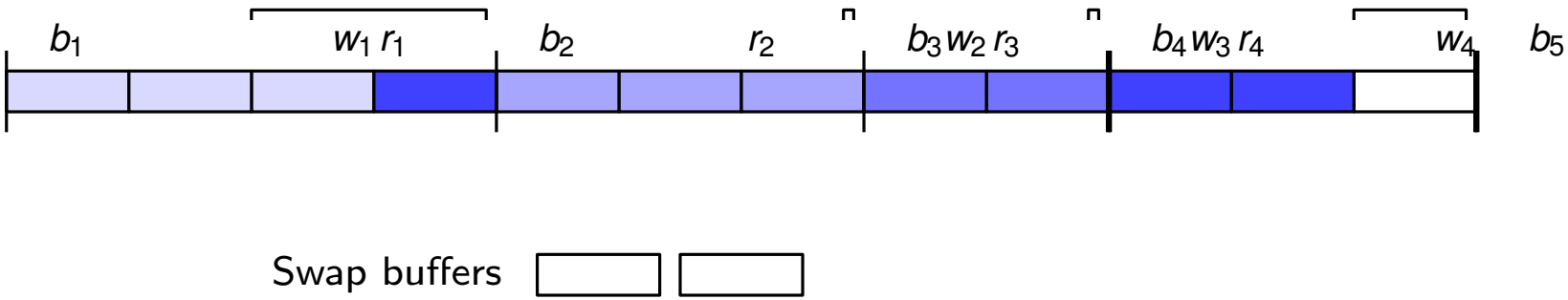


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

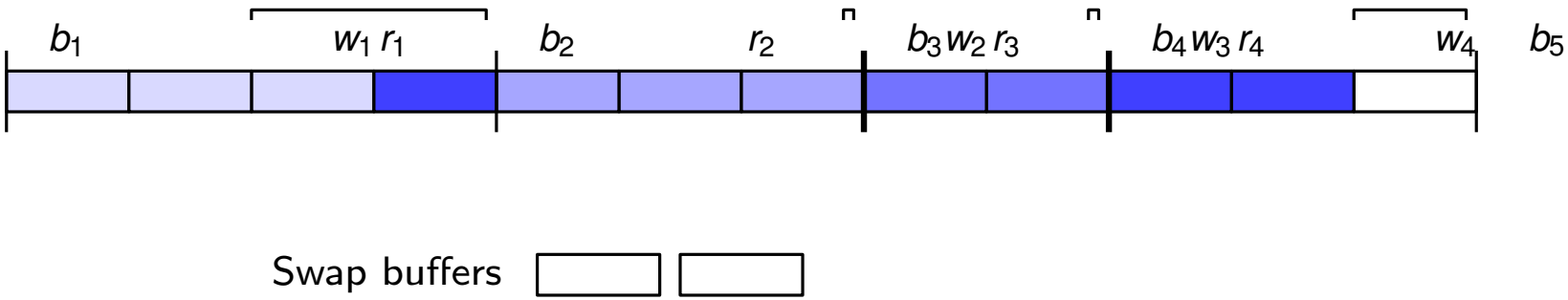


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

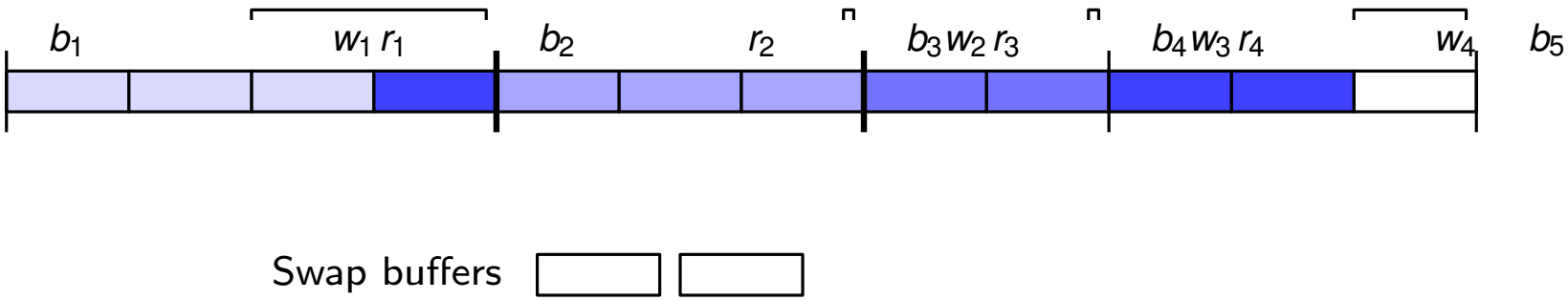


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

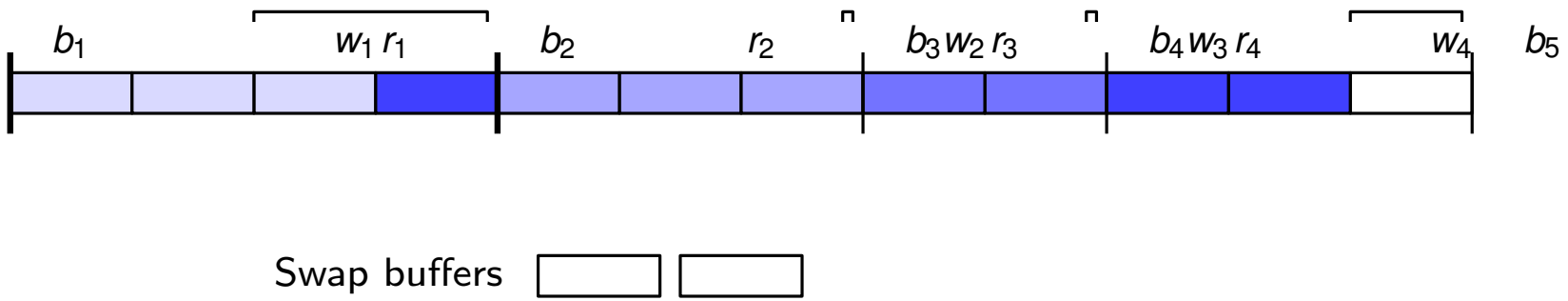


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

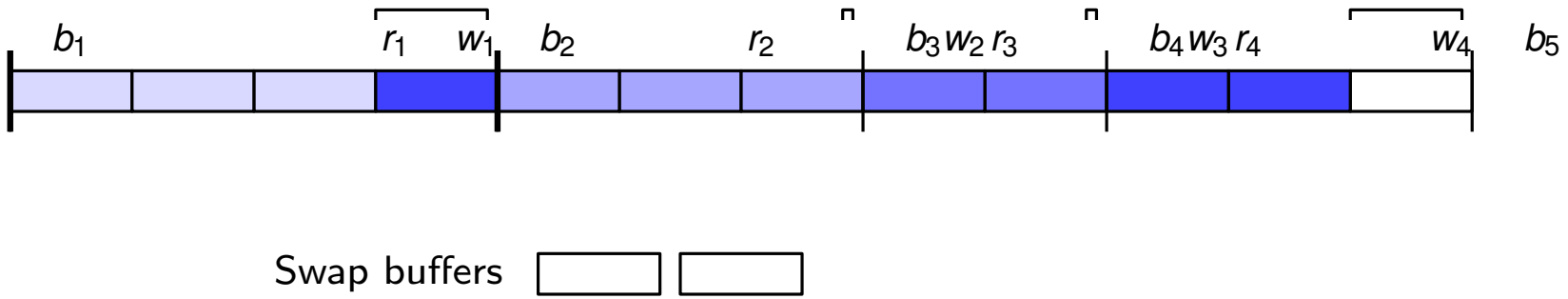


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

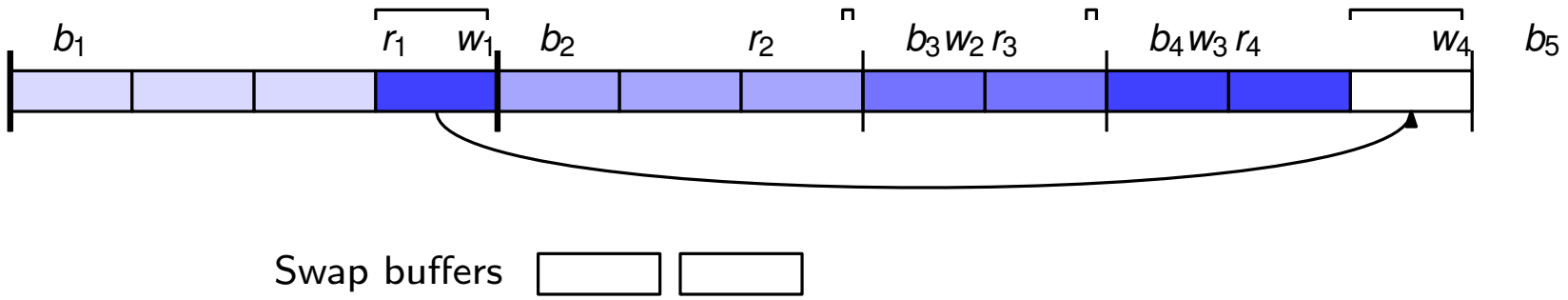


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

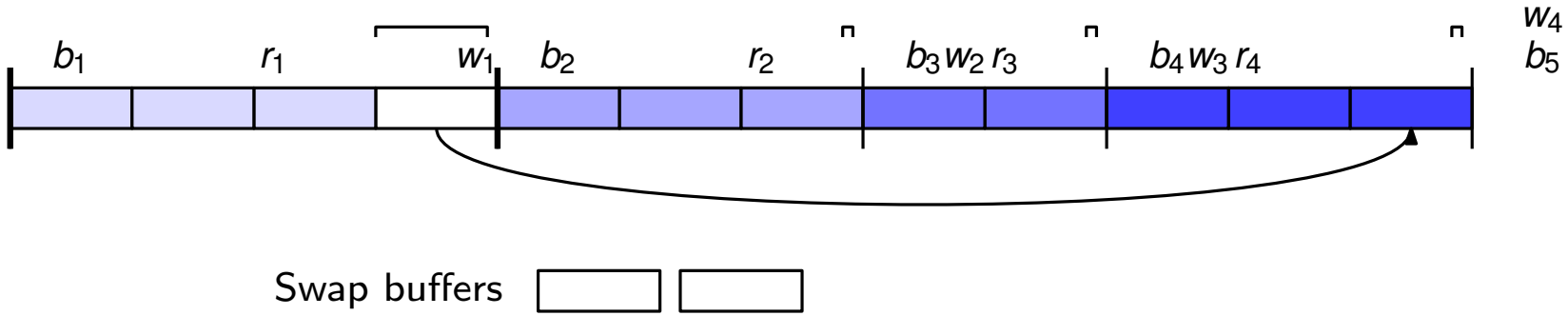


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks

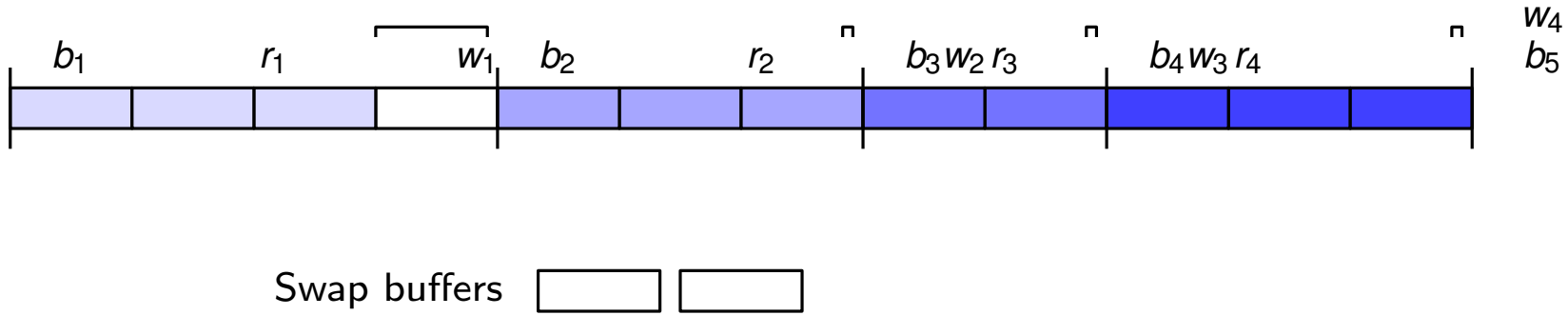


Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort

Block permutation

- Invariant
  - Permuted blocks $[b_i, w_i)$
  - Unpermuted blocks $[w_i, r_i]$
  - Empty Blocks (r_i, b_{i+1})
- Two buffers to swap blocks



Institute of Theoretical Informatics
Algorithmics Group

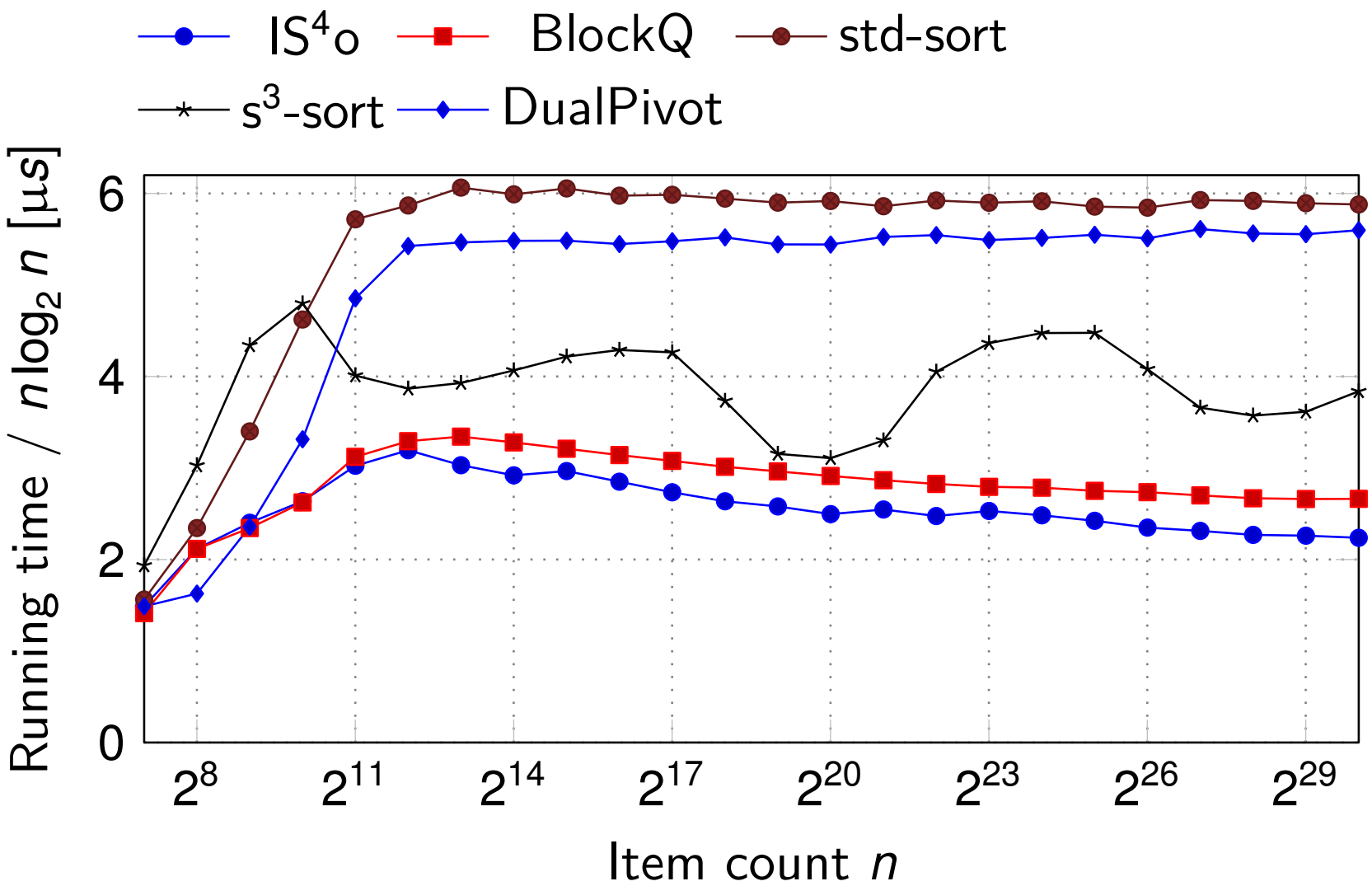
In-place Super Scalar Samplesort

Discussion

- Why blocks
  - Just $n + n/b$ classifications per level
  - Reduced TLB misses
    - Local classification: buffers on same page
    - Write and read whole blocks
  - Better prefetching and less cache misses

Institute of Theoretical Informatics
Algorithmics Group

In-place Super Scalar Samplesort



Sequential on two Xeon E5-2683 v4 16-core processors – Uniform Input

Institute of Theoretical Informatics
Algorithmics Group

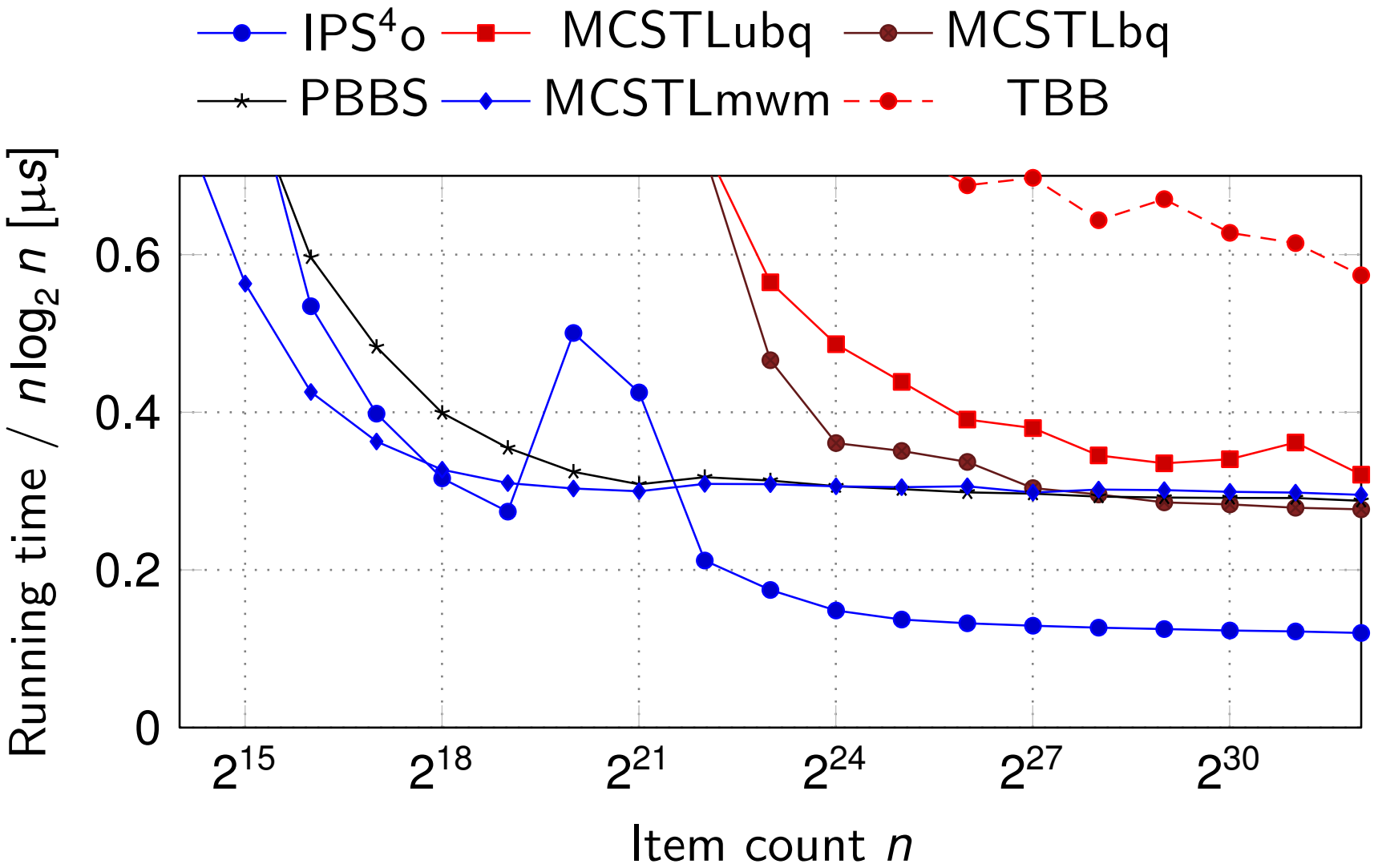
In-place Parallel Super Scalar Samplesort

Shared-memory parallelism with t threads

- Local classification: divide input into stripes – one for each thread
- Block permutation: fetch blocks atomically
  - atomic read pointers and end pointers
    - Access with fetch-and-add operations
  - Blocks of size $\Omega(t)$ to avoid starvation
- Buffers for each thread
- Call sequential subroutines in parallel if $n \leq n_{\text{init}}/t$

Institute of Theoretical Informatics
Algorithmics Group

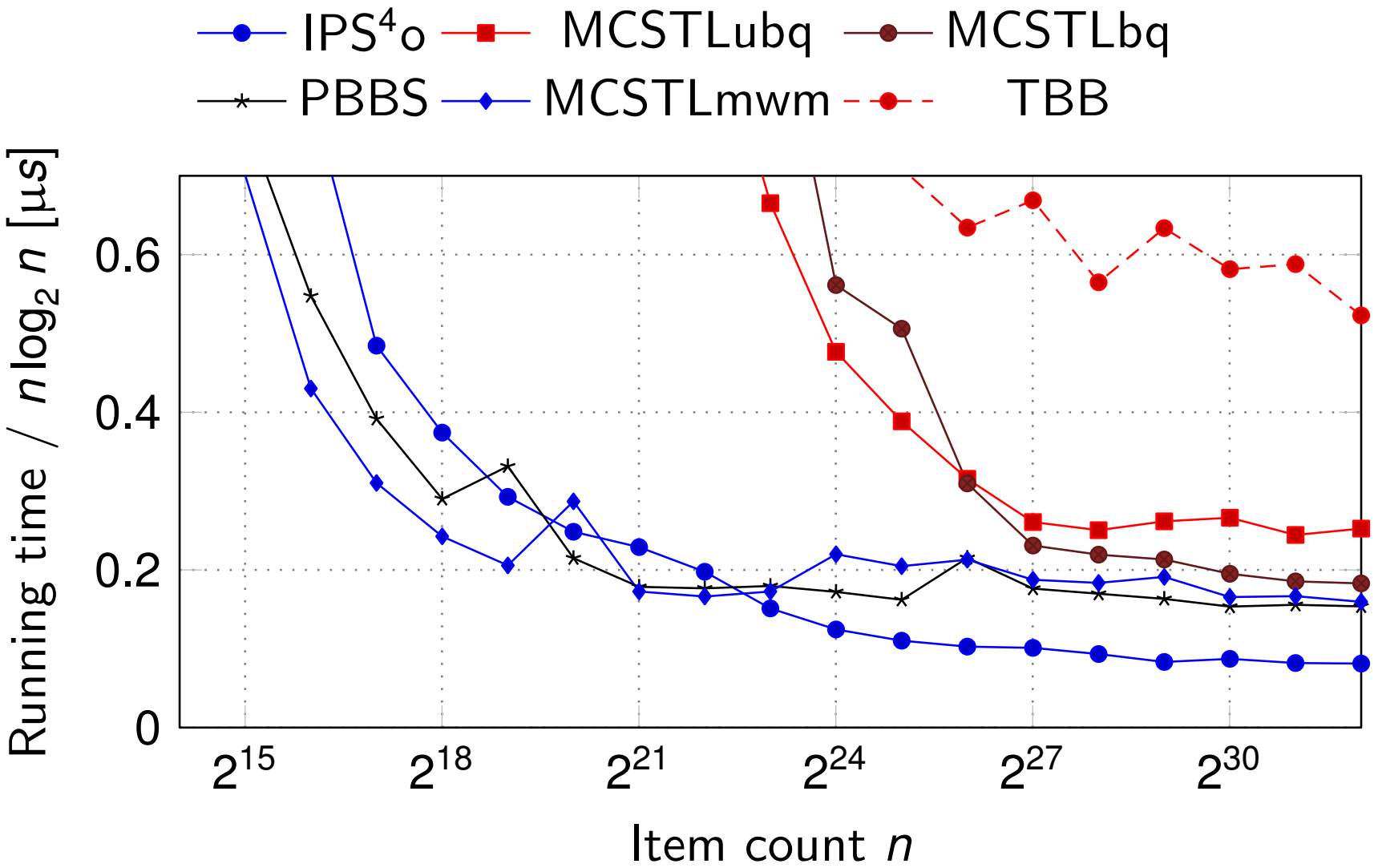
In-place Parallel Super Scalar Samplesort



One of two Intel Xeon E5-2683 v4 16-core processors – Uniform Input

Institute of Theoretical Informatics
Algorithmics Group

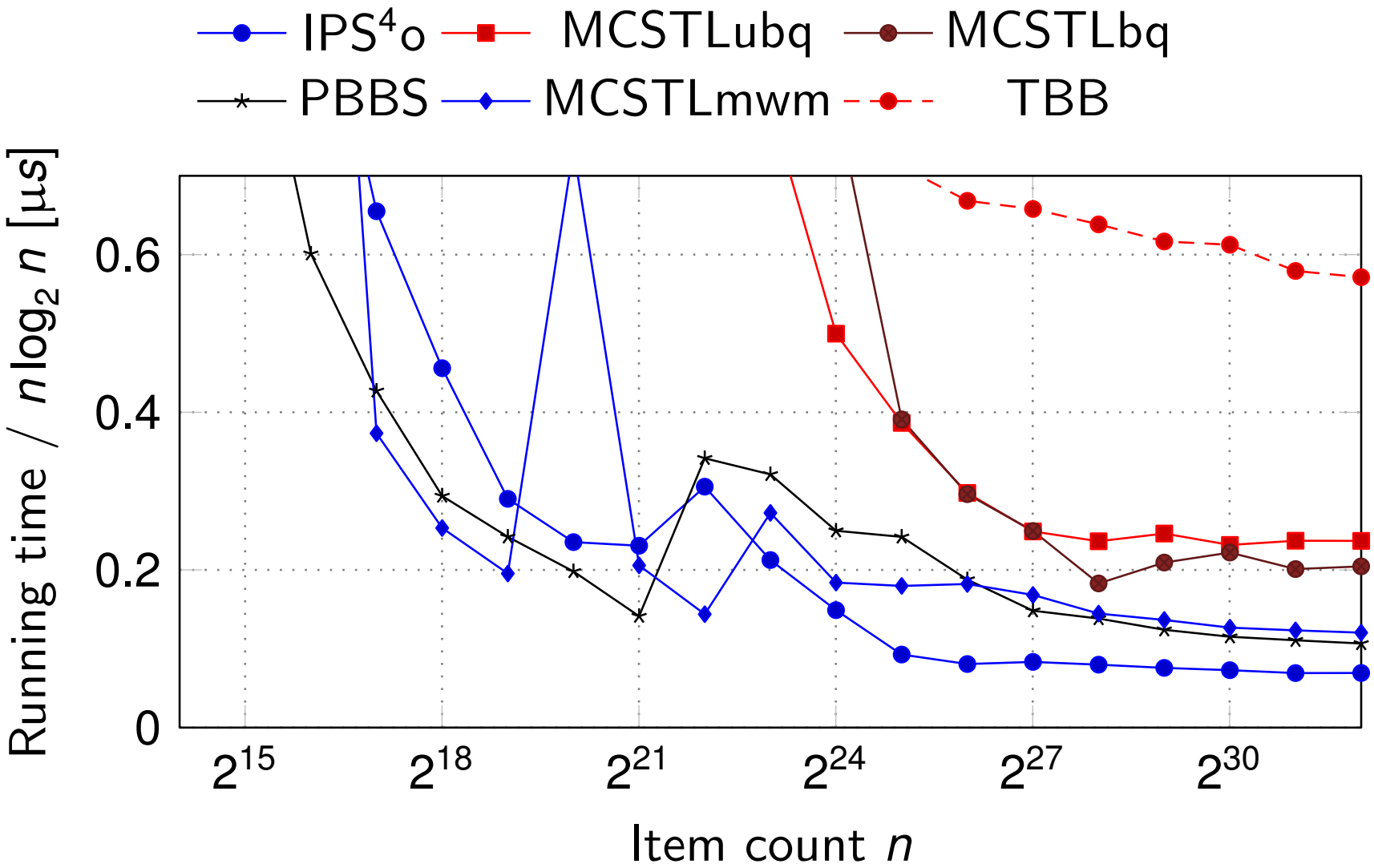
In-place Parallel Super Scalar Samplesort



Two Intel Xeon E5-2683 v4 16-core processors – Uniform Input

Institute of Theoretical Informatics
Algorithmics Group

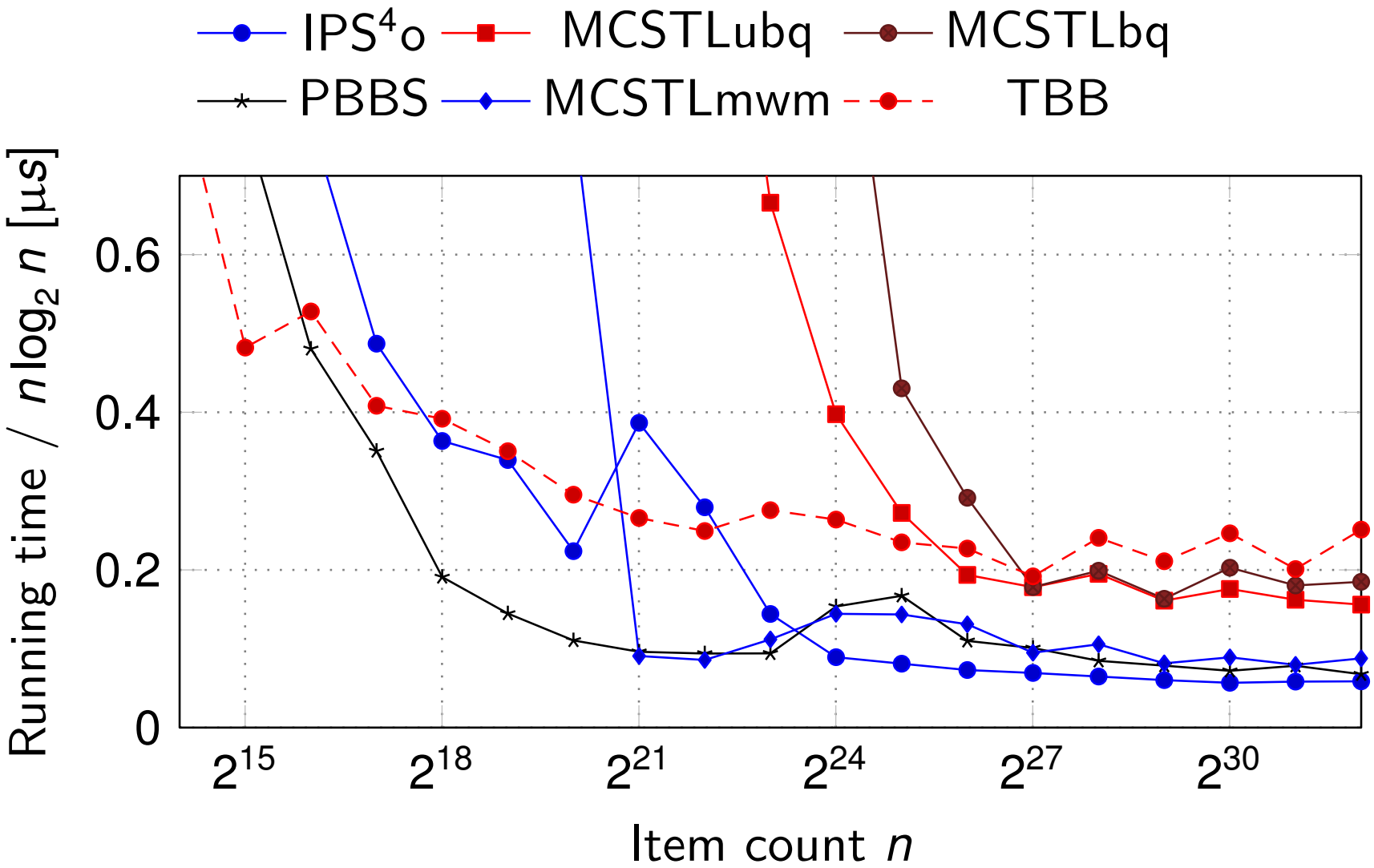
In-place Parallel Super Scalar Samplesort



Two Intel Xeon E5-2683 v4 16-core processors – Exponential Input

Institute of Theoretical Informatics
Algorithmics Group

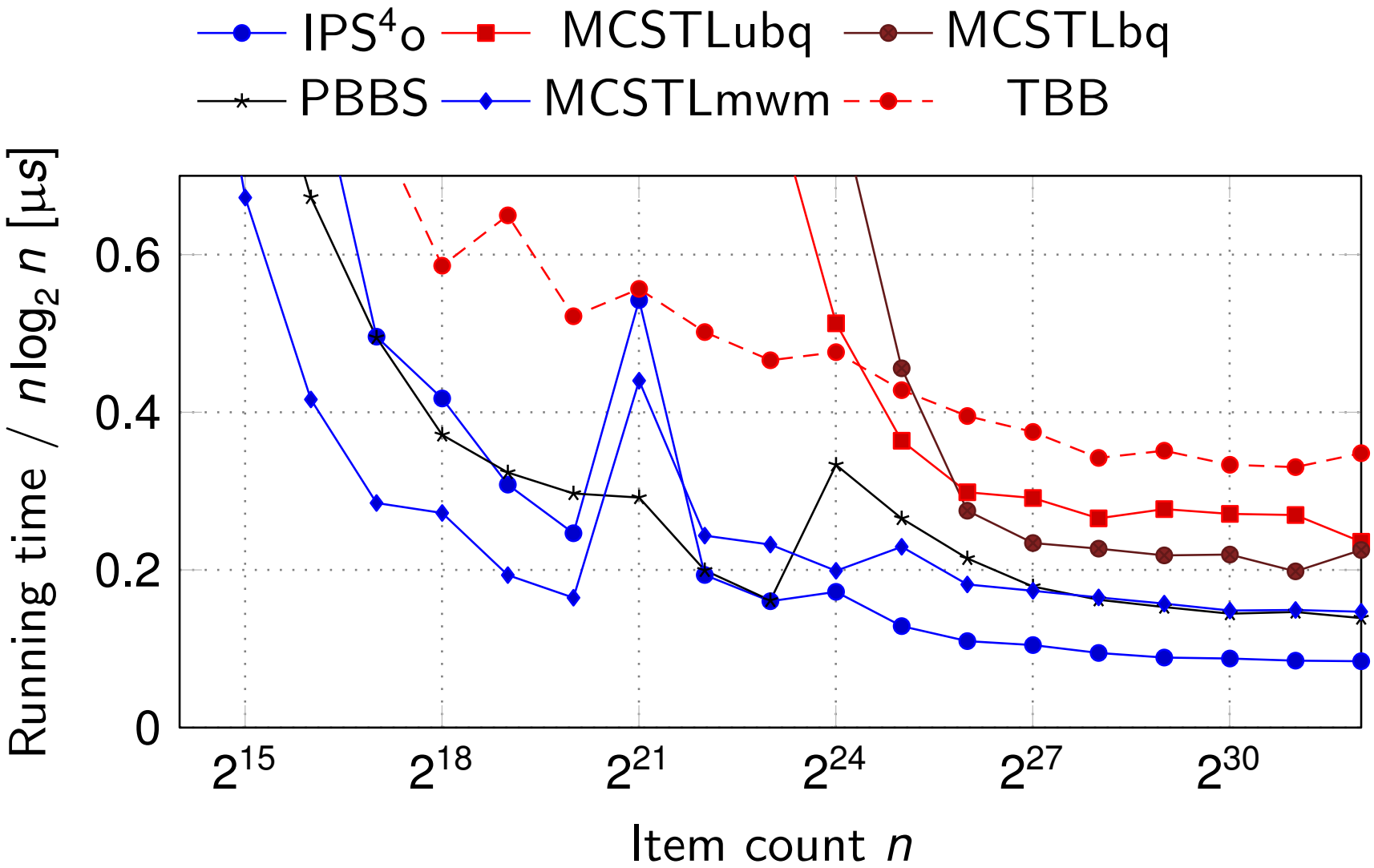
In-place Parallel Super Scalar Samplesort



Two Intel Xeon E5-2683 v4 16-core processors – RootDup Input

Institute of Theoretical Informatics
Algorithmics Group

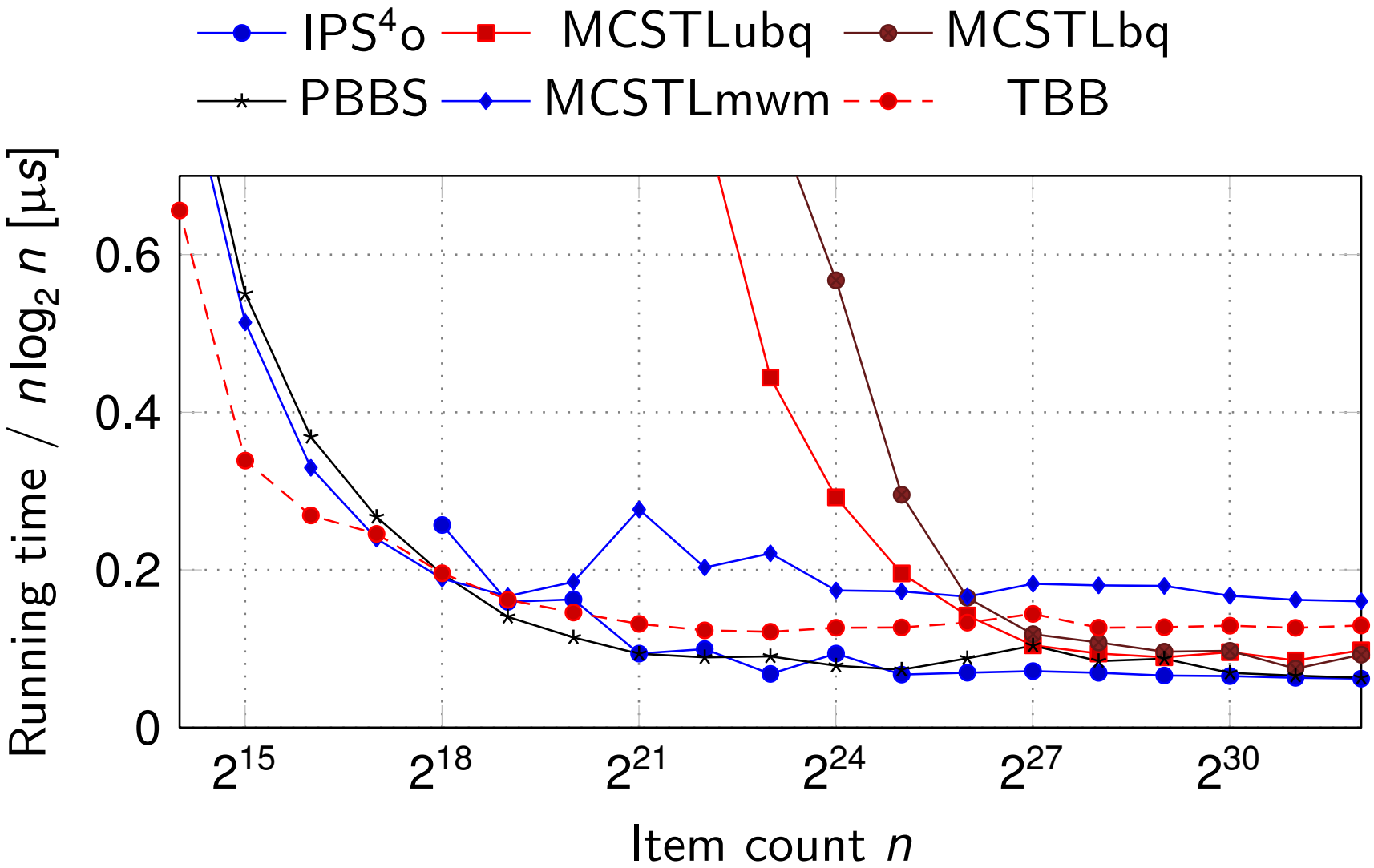
In-place Parallel Super Scalar Samplesort



Two Intel Xeon E5-2683 v4 16-core processors – TwoDup Input

Institute of Theoretical Informatics
Algorithmics Group

In-place Parallel Super Scalar Samplesort



Two Intel Xeon E5-2683 v4 16-core processors – AlmostSorted Input

Institute of Theoretical Informatics
Algorithmics Group

BlockQuicksort

Goals

- Partially decoupling control flow from data flow
- Avoid conditional branches
- In-place

Institute of Theoretical Informatics
Algorithmics Group