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

<table>
<thead>
<tr>
<th></th>
<th>Quicksort</th>
<th>BlockQuicksort</th>
<th>SSSS</th>
<th>ISSSSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decoupled control and data flow</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Conditional branches</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Data transfers / element</td>
<td>$\approx \log_2 n$</td>
<td>$\approx \log_2 n$</td>
<td>$\approx \log_k n$</td>
<td>$\approx \log_k n$</td>
</tr>
<tr>
<td>Additional space</td>
<td>$O(1)$</td>
<td>$O(b)$</td>
<td>$O(n)$</td>
<td>$O(kb)$</td>
</tr>
<tr>
<td>Parallelization</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
</tbody>
</table>
BlockQuicksort

Goals

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

Goals

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

$b$ Elements
BlockQuicksort

Goals

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

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

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

![Diagram of BlockQuicksort with Pivot and b Elements]
BlockQuicksort

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

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

Input

Local classification

Block permutation
In-place Super Scalar Samplesort

Input

Local classification

Block permutation

Cleanup
Local classification

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

Flush buffer block if charged
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
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
In-place Super Scalar Samplesort

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

![Diagram showing block permutation and buffer swapping]
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

Swap buffers  

![Diagram showing block permutation and two buffers to swap blocks]
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

Swap buffers
In-place Super Scalar Samplesort

Block permutation

- Invariant
- Permutated blocks \([b_i, w_i]\)
- Unpermuted blocks \([w_i, r_i]\)
- Empty Blocks \((r_i, b_{i+1})\)

- Two buffers to swap blocks
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

\[
\begin{array}{c}
\text{Permuted blocks} \ [b_i, w_i) \\
\text{Unpermuted blocks} \ [w_i, r_i] \\
\text{Empty Blocks} \ (r_i, b_{i+1}) \\
\end{array}
\]
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
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

Swap buffers

\[
\begin{align*}
&b_1, w_1 & r_1 & b_2 & r_2, w_2 & b_3 & r_3, w_3, r_4, b_4 & w_4 & b_5 \\
& \text{Permuted blocks} & \text{Unpermuted blocks} & \text{Empty Blocks} & \text{Two buffers to swap blocks}
\end{align*}
\]
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

Swap buffers

```
\begin{align*}
& b_1 w_1 & r_1 & b_2 & r_2 & w_2 & b_3 & r_3 & w_3 & r_4 & b_4 & w_4 & b_5 \\
& \text{blue} & \text{red} & \text{blue} & \text{red} & \text{blue} & \text{red} & \text{red} & \text{blue} & \text{red} & \text{blue} & \text{red} & \text{red} & \text{blue} \\
\end{align*}
```
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

![Diagram of block permutation]

Swap buffers

- \(b_1\)
- \(w_1, r_1\)
- \(b_2\)
- \(r_2\)
- \(b_3, w_2, r_3\)
- \(b_4, w_3, r_4\)
- \(w_4\)
- \(b_5\)
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

![Diagram of block permutation with elements and buffer swap]

Swap buffers

\[\ldots\ b_1 \quad w_1 r_1 \quad b_2 \quad r_2 \quad \cdots \quad b_4 w_3 r_4 \quad \cdots \quad b_5 \\]
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

\[
\begin{array}{cccccccccccc}
  b_1 & w_1 & r_1 & b_2 & r_2 & \ldots & b_n & w_n & r_n & b_{n+1} & \ldots & w_k & b_5 \\
\end{array}
\]

Swap buffers

\[
\begin{array}{cccccccccccc}
  \text{Swap buffers} & & & & & & & & & & & & \\
\end{array}
\]
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

Swap buffers

Diagram: [Image of block permutation diagram with placeholders for buffers]
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

![Diagram of block permutation and swap buffers](image)
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
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
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
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

![Diagram showing block permutation and buffer swap](image-url)
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
In-place Super Scalar Samplesort

Sequential on two Xeon E5-2683 v4 16-core processors – Uniform Input
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 $n \leq n_{\text{init}}/t$
In-place Parallel Super Scalar Samplesort

![Graph](image)

One of two Intel Xeon E5-2683 v4 16-core processors – Uniform Input
In-place Parallel Super Scalar Samplesort

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

Running time / $n \log_2 n$ [µs]

Item count $n$
In-place Parallel Super Scalar Samplesort

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

Running time / $n \log_2 n$ [µs]
In-place Parallel Super Scalar Samplesort

Two Intel Xeon E5-2683 v4 16-core processors – RootDup Input
In-place Parallel Super Scalar Samplesort

Two Intel Xeon E5-2683 v4 16-core processors – TwoDup Input
In-place Parallel Super Scalar Samplesort

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

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