`sort`, `stable_sort`

Two Major Options

Parallel Multiway Mergesort

- \+ less “communication” necessary (cache coherence)
- \+ stable variant easy to derive
- – needs twice the space

Parallel Balanced Quicksort

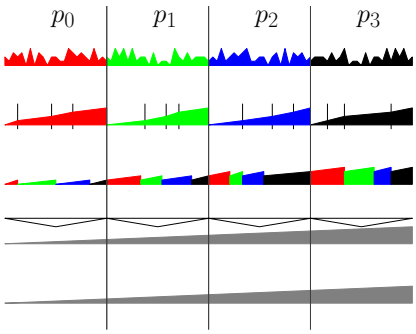
- \+ in-place
- ± dynamic load-balancing due to unequal splitting
- – concurrent access to memory
- – unstable

Both variants implemented in the MCSTL

Parallel Multiway Mergesort

Procedure

1. divide sequence into p parts of equal size
2. in parallel sort the parts locally
3. use parallel p-way merging to compute the final sequence
4. copy result back to original position



Parallel Balanced Quicksort

Procedure

1. split sequence using parallel `partition`, descend parallel recursively with the appropriate number of threads
2. as soon as there is only one processor left per partition: start local sorting “with helping”
3. each processor sorts locally, pushes parts to process later onto explicit dequeue
4. other processors can “partitions” work when out of work

Sorting Performance Results