Multi-level Data Translocation for Faster Processing of Scattered Data on Shared-Memory Computers

Dimitris Floros* Alexandros-Stavros Iliopoulos† Nikos Pitsianis*† Xiaobai Sun†

*ECE, Aristotle University of Thessaloniki †CS, Duke University

Problem description

- Points $\mathcal{X} = \{x_i\}_{i=1}^{n}$ initially stored in order $a(\mathcal{X})$ Access order $b(\mathcal{X})$ far different from $a(\mathcal{X})$
  - inter-procedure or inter-operation
- Indirect indexing $a(\mathcal{X}) \mapsto b(\mathcal{X})$
  - invokes irregular memory access patterns
  - computation becomes acutely memory bounded
  - difficult to parallelize
- Common solutions: reordering operations, data, or both
  - loop transformations: splitting, fusion, skewing, distribution
  - strip-mining, tiling and permutation
- Common solutions are challenged by scattered data
- **Fast data translocation:** Physical data relocation $\Pi : a(\mathcal{X}) \mapsto b(\mathcal{X})$

Objective

Improve performance of operations on scattered data
– optimal data locality for minimal memory access latency
– maximal utilization of parallel resources & scheduling schemes

Applications

- Scattered data samples acquired/generated in various applications
  - 3D scans, magnetic resonance imaging
  - Molecular/celestial dynamics simulations
  - Integral imaging, augmented or virtual reality
  - Graph embedding
- Processing typically involves calculation of all-point interactions
  - direct evaluation too expensive, $O(n^2)$ operations
  - approximation/compression techniques: $O(n \log n)$ or $O(n)$
    - $\frac{\text{arithmetic operations}}{\text{memory operations}}$ becomes very small

    computation becomes acutely memory bounded

Contact

Dimitris Floros

Methodology

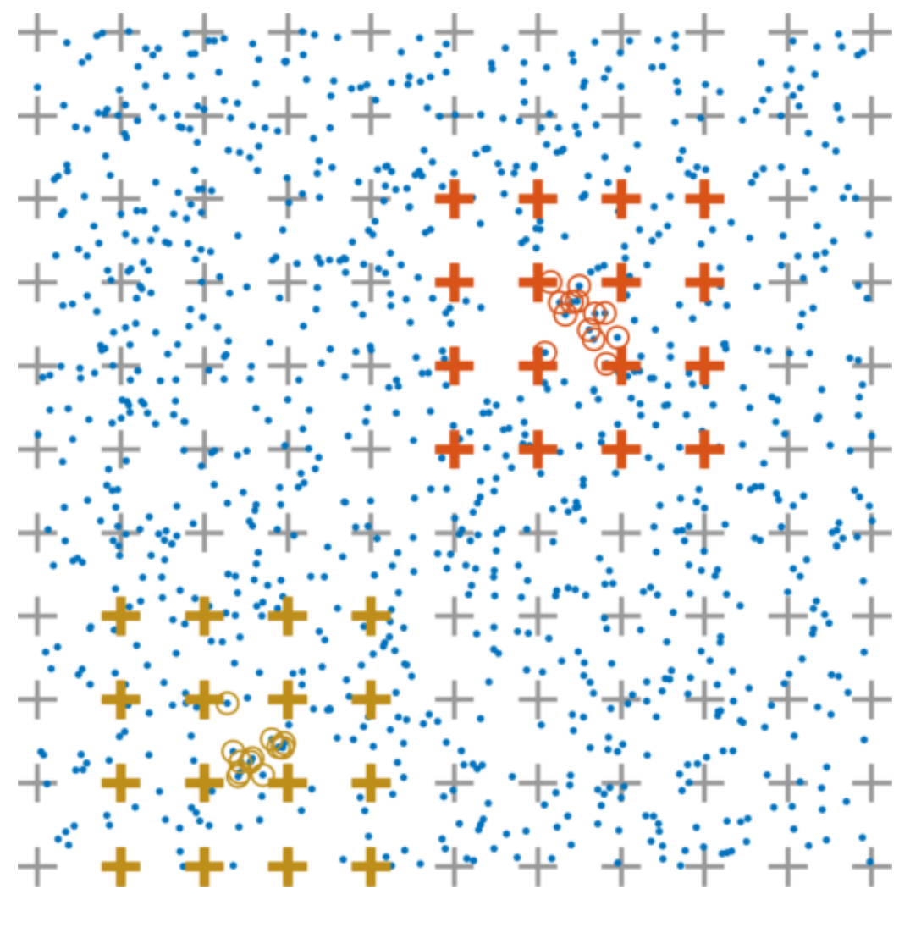
Demo case: scatter-grid translations

Data translation between scattered data points and regular points on an auxiliary grid (externally specified or internally determined)

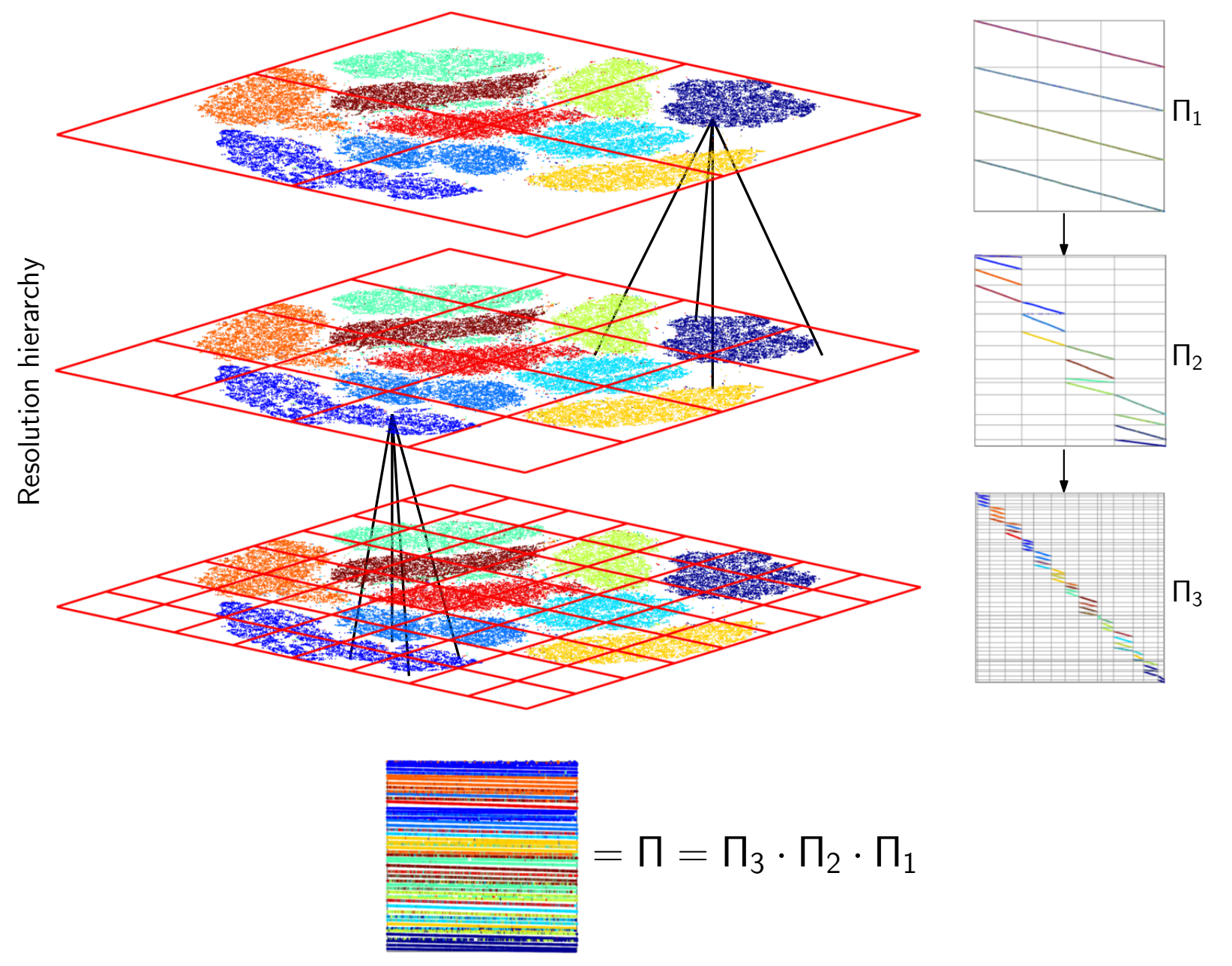
Scattered interactions are decomposed into

- local translations between scattered and grid points (S2G & G2S)
- global interactions among the equispaced grid points (G2G)

This poster focuses on local translations S2G and G2S



The local window support is $w = 4 \times 4$ grid points

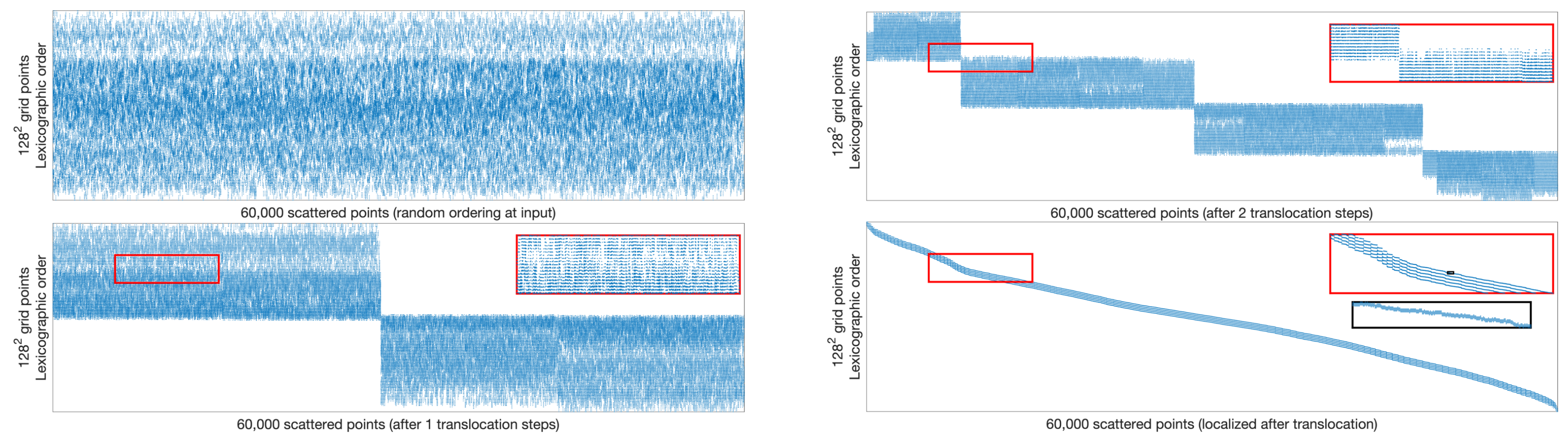


Multi-level data translocation

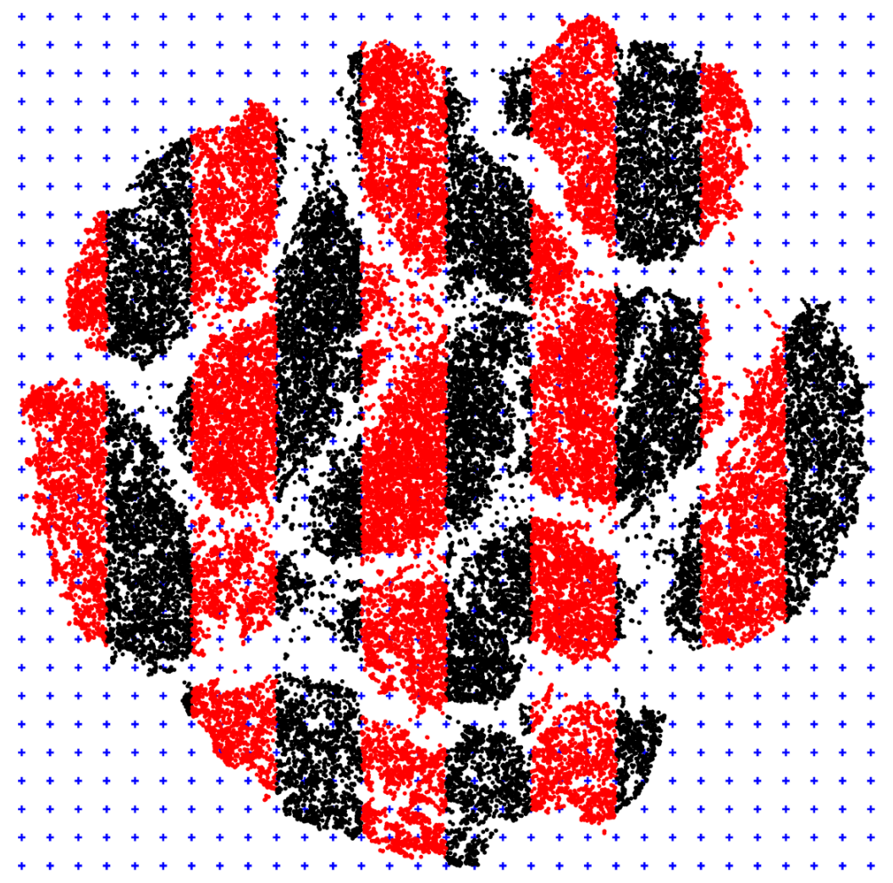
- Hierarchical binning (coarser to finer grids)
  - adhere to memory hierarchy
  - utilize memory bandwidth
  - explore data & task parallelism
- Matrix view
  - block-wise factorization of permutation Π
  - blocks not necessarily of equal size
  - recursion not necessarily uniform in size and depth

Memory access patterns

- Scattered points are translocated prior to S2G and G2S translations
  - points residing in the same grid cell are placed together



Local translation matrix, at each translocation step, with 960,000 coefficients between 128×128 regular grid points (rows) and 60,000 scattered points (columns). The local window support is $w = 4 \times 4$ grid points.



Red-black non-overlapping partition of a 2D grid, with 30×30 grid points (blue) and 60,000 scattered points

Red-black scheduling

Partition grid into non-overlapping regions (red-black)

- Data coherence
  - No write conflicts
  - No data racing
- Minimal synchronization barriers
- Maximal use of parallel resources

Architecture specification

CPU	Clock (GHz)	Cores	L1 (KiB) per-core	L2 (KiB) per-core	L3 (MiB) shared	RAM (GiB)	BW (GiB/s)
Intel Core i7-4558U	2.80	2	64	256	4	8	10.5
Intel Core i7-6700	3.40	4	32	256	8	32	19.9
AMD Ryzen 1900X	3.80	8	96	512	16	64	31.6

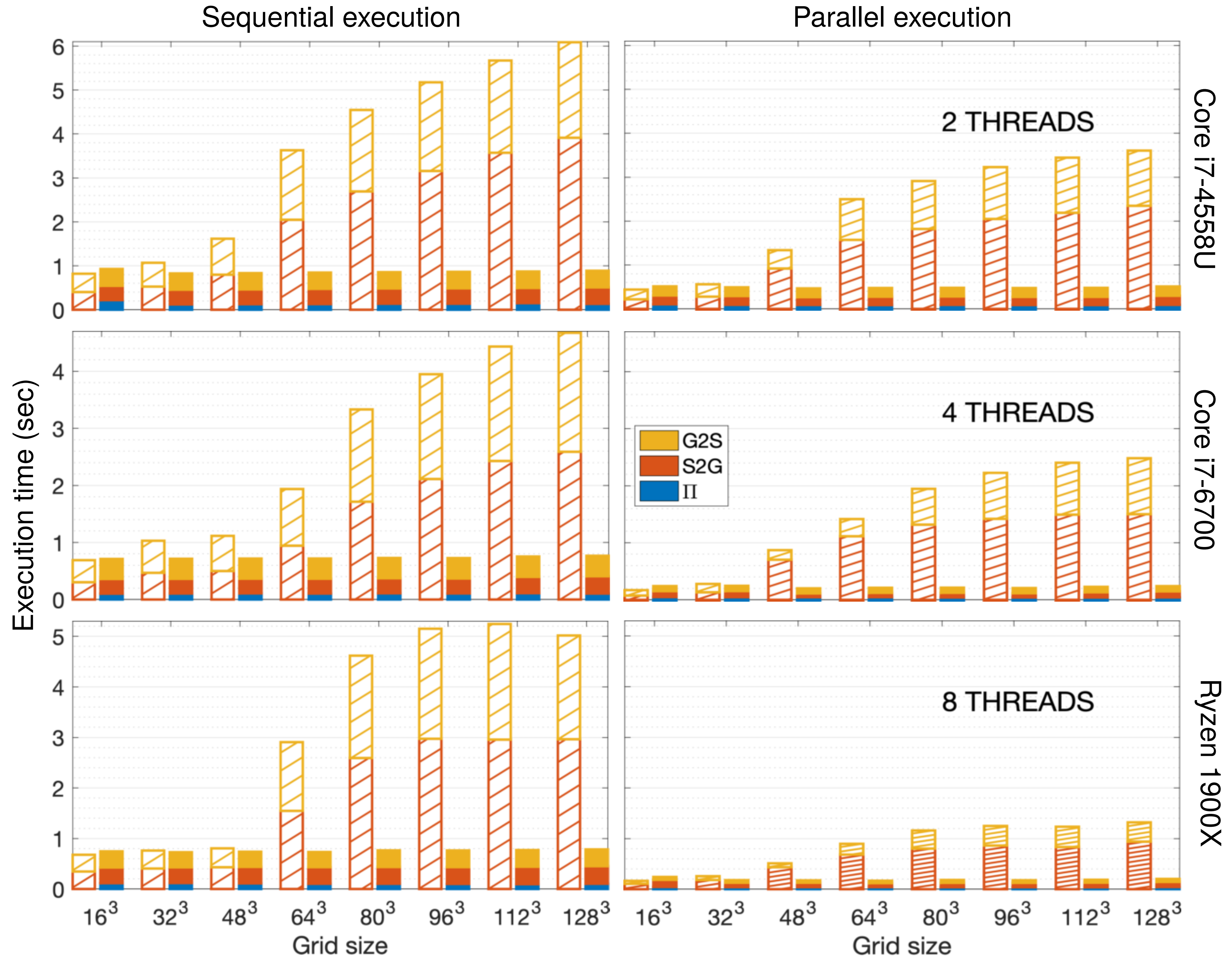
Memory bandwidth measured with the parallel `STREAM copy` benchmark.

Performance results

Performance of S2G & G2S on 3D dataset of $n = 2{,}097{,}152$ scattered points drawn randomly following a uniform distribution over $[0, 1)^3$

Speedup at $128 \times 128 \times 128$ grid

computer	Core i7-4558U		Core i7-6700		Ryzen 1900X	
#thread	1	2	1	4	1	8
without translocation	1.0	1.7	1.0	1.9	1.0	3.8
with translocation	6.9	11.9	6.1	19.5	6.4	24.6
ratio	6.9	7.0	6.1	10.3	6.4	6.5



The execution time with data translocation is shown in solid colored bars
The data translocation overhead, denoted as Π (blue bar), is well paid-off

References

[1] J. Barnes and P. Hut. *Nature*, 1986.
[2] L. Greengard and V. Rokhlin. *J Comput Phys*, 1987.
[3] G. C. Linderman *et al*. *Nat Methods*, 2019.
[4] N. Pitsianis *et al*. In *IEEE HPEC*, 2019.
[5] H. S. Stone. *IEEE Trans Comput*, C-20, 1971.
[6] X. Sun and N. Pitsianis. *SIAM Rev.*, 2001.
[7] L. van der Maaten. *JMLR*, 2014.
[8] L. van der Maaten and G. Hinton. *JMLR*, 2008.
[9] Y. Zhang *et al*. *J Circuit Syst Comp*, 2012.

Acknowledgments

We thank SPPEXA, the DFG priority program for software for Exascale computing, for providing travel support for the conference and workshops. The equipment used in the experiments was partially funded by the ERA.NET Plus RUS initiative and the GSRT.

COLOC, EURO-PAR 2019