JKU
JOHANNES KEPLER
UNIVERSITY LINZ

Performance Evaluation of Multi-Threaded Granular Force Kernels in LIGGGHTS

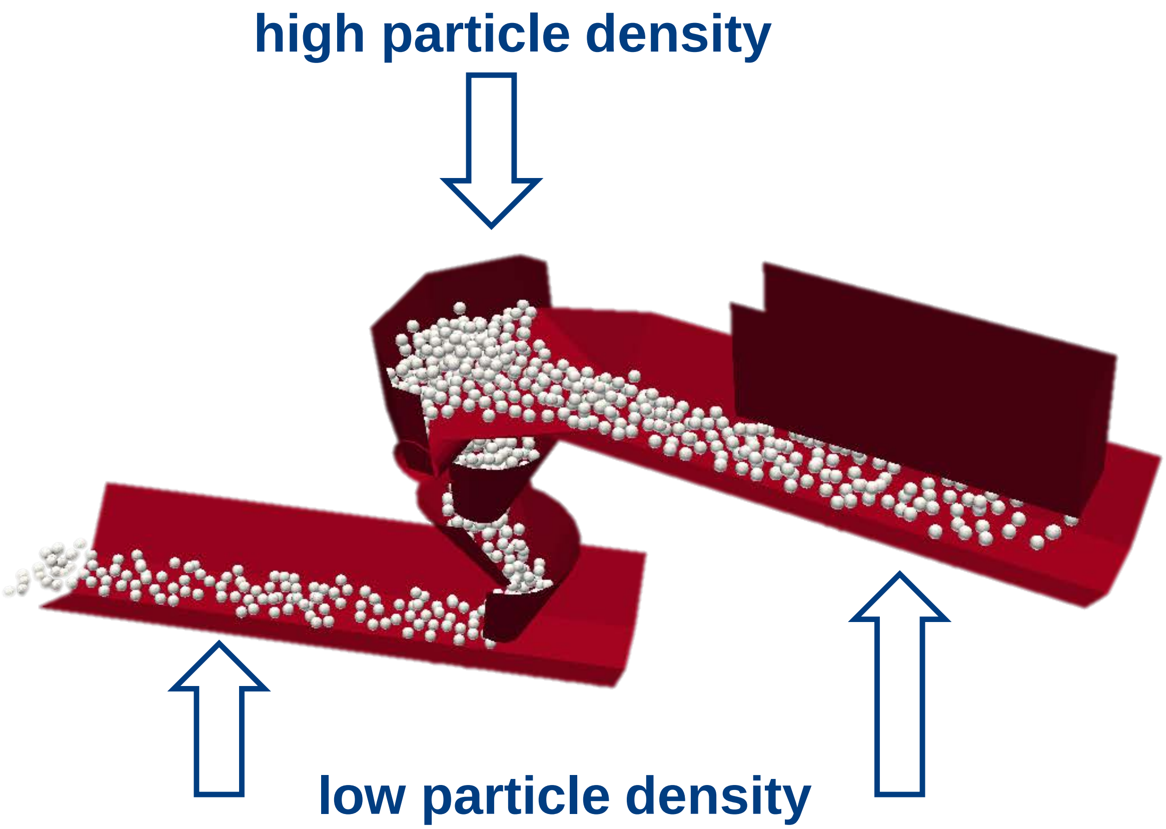
Richard Berger

CFDEM project | CFDEM coupling | LIGGGHTS | DCS Computing GmbH

Why add threading optimizations?

- Domain decomposition not enough for load-balancing

Transfer chute example



Why add threading optimizations?

- Domain decomposition not enough for load-balancing
- Shared memory programming gives you more control
- With MPI you have to rely on individual implementations (OpenMPI, MPICH2)
- More optimization potential with shared memory programming (e.g. cache efficiency)
- A hybrid approach would give us the best of both worlds.

Starting Point: MiniMD

- **LIGGGHTS**
  - Based on LAMMPS
  - ~280,000 LOC
  - Optimizing this code base is hard
- **MiniMD-granular**
  - Based on MiniMD, which is a light-weight benchmark of LAMMPS
  - ~3,800 LOC
  - Makes it much easier to test new ideas and optimize critical parts
- **What was done in OpenMP:**
  - Pair Styles (pair_gran_hooke)
  - Neighbor List
  - Integration
  - Primitive Walls

Atom decomposition
OpenMP static schedule

Force array

Each box represents the force calculated for one particle.

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

Thread 0

Thread 1

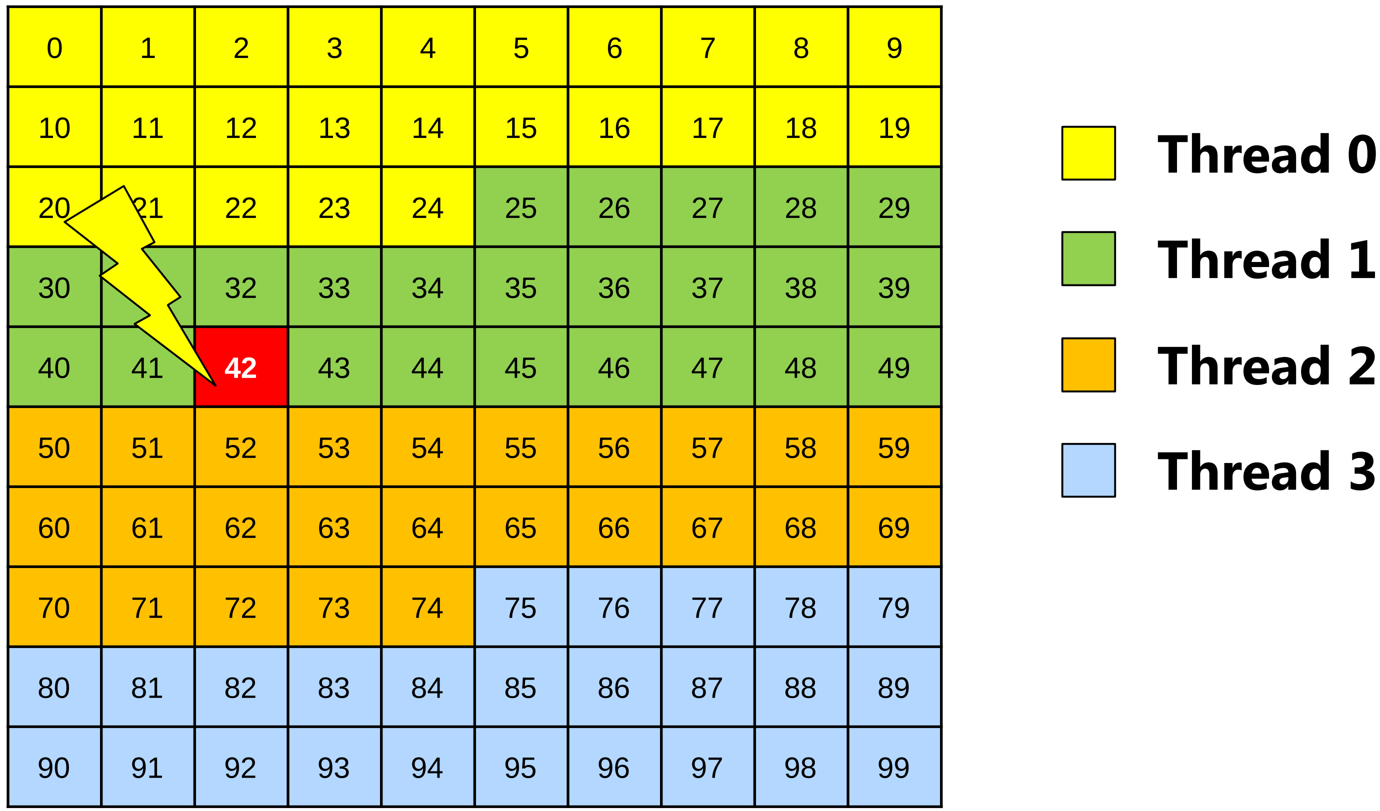
Thread 2

Thread 3

Atom decomposition
Data Races

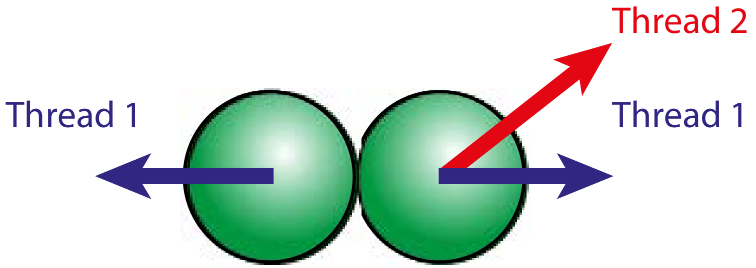
Data Race:
Access the same memory location, at least one thread writes

Write Conflict:
Two threads try to update force of the same particle

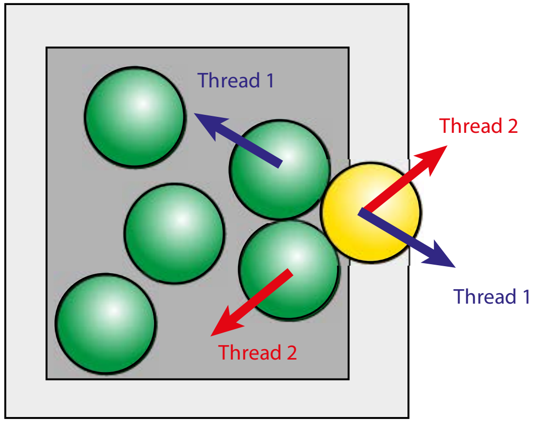


Sources of Data Races

- **Newton's 3rd Law (Actio = Reactio, always used in LIGGGHTS):**
  - o Pair Forces of local particles only computed once, applied to both contact partners

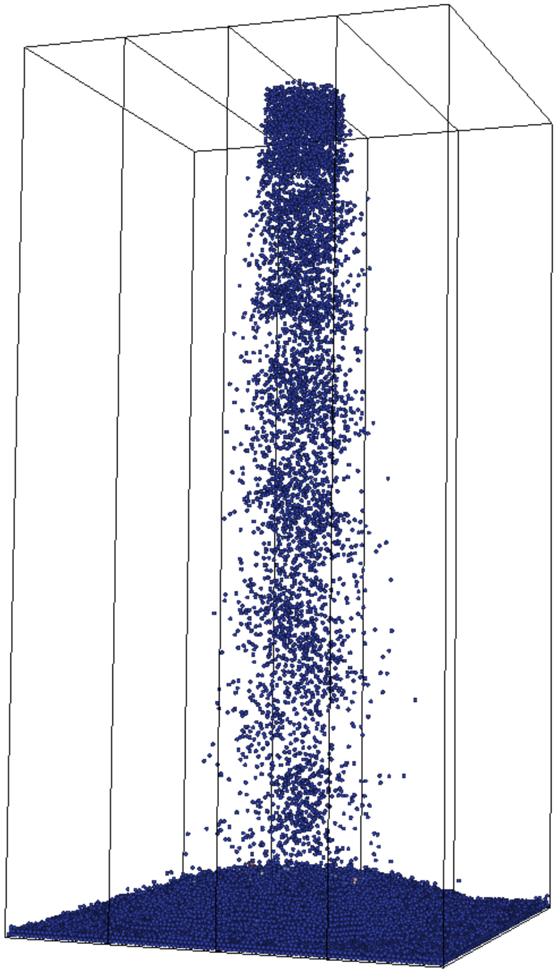


- **Ghost Particles**
  - o Pair Forces are only computed once at Process Boundaries
  - o Multiple threads could try adding contributions to a single ghost particle



- **Global Accumulators:**
  - o Compute (Energy, Virial)

Boxfill example



Load balancing

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

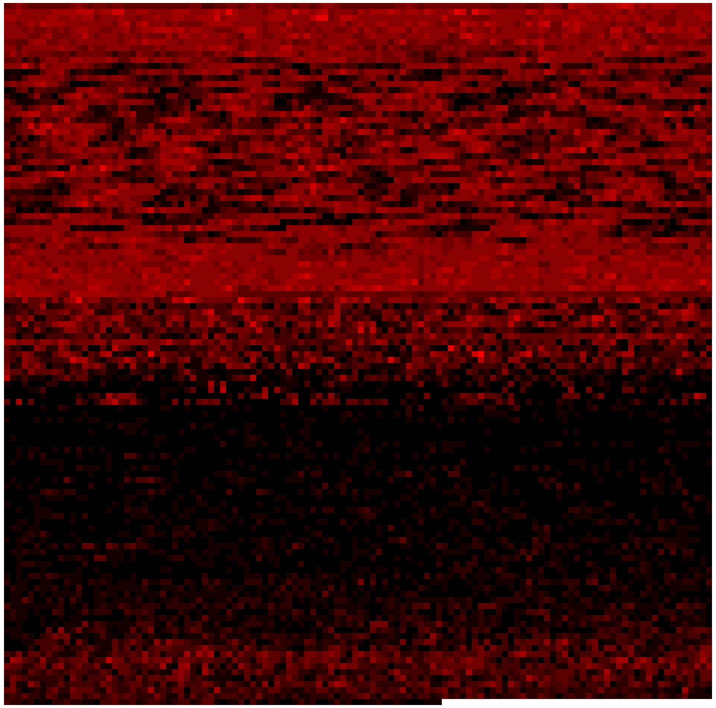
Thread 0

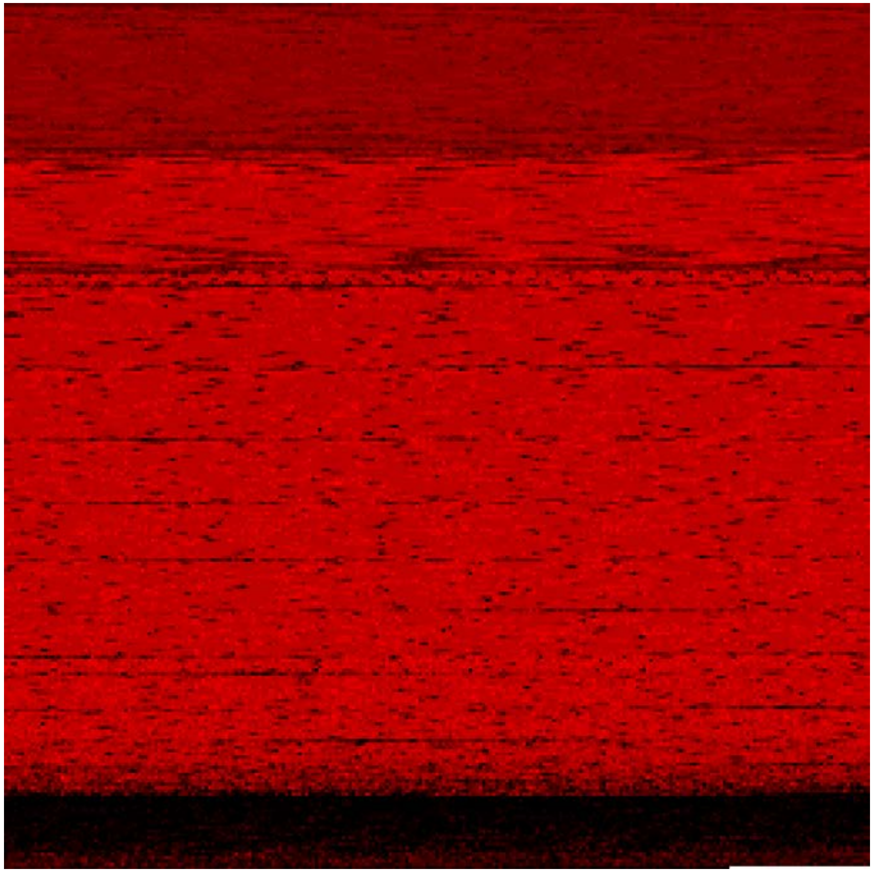
Thread 1

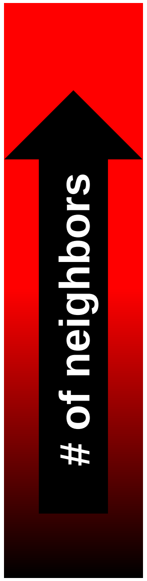
Thread 2

Thread 3

Load balancing
Visualization of the workload (serial run)

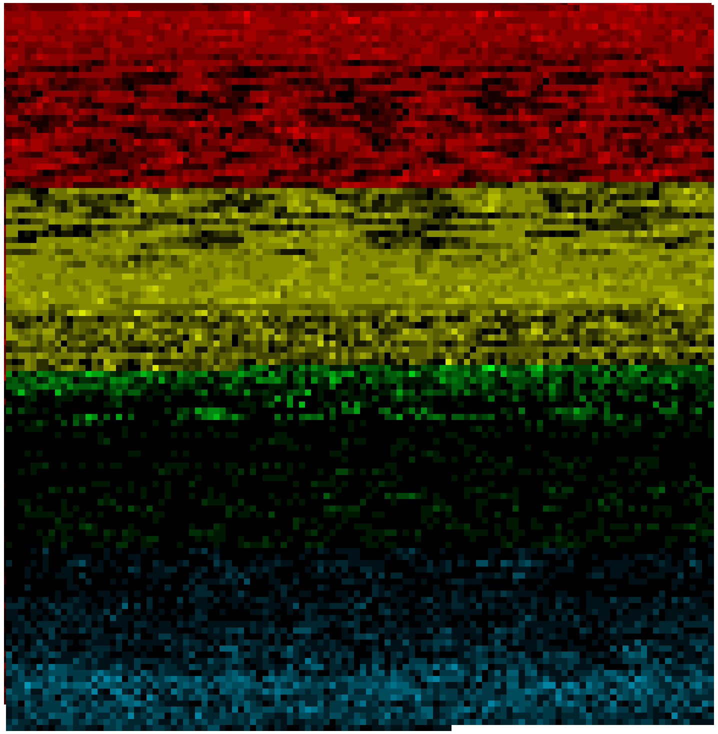




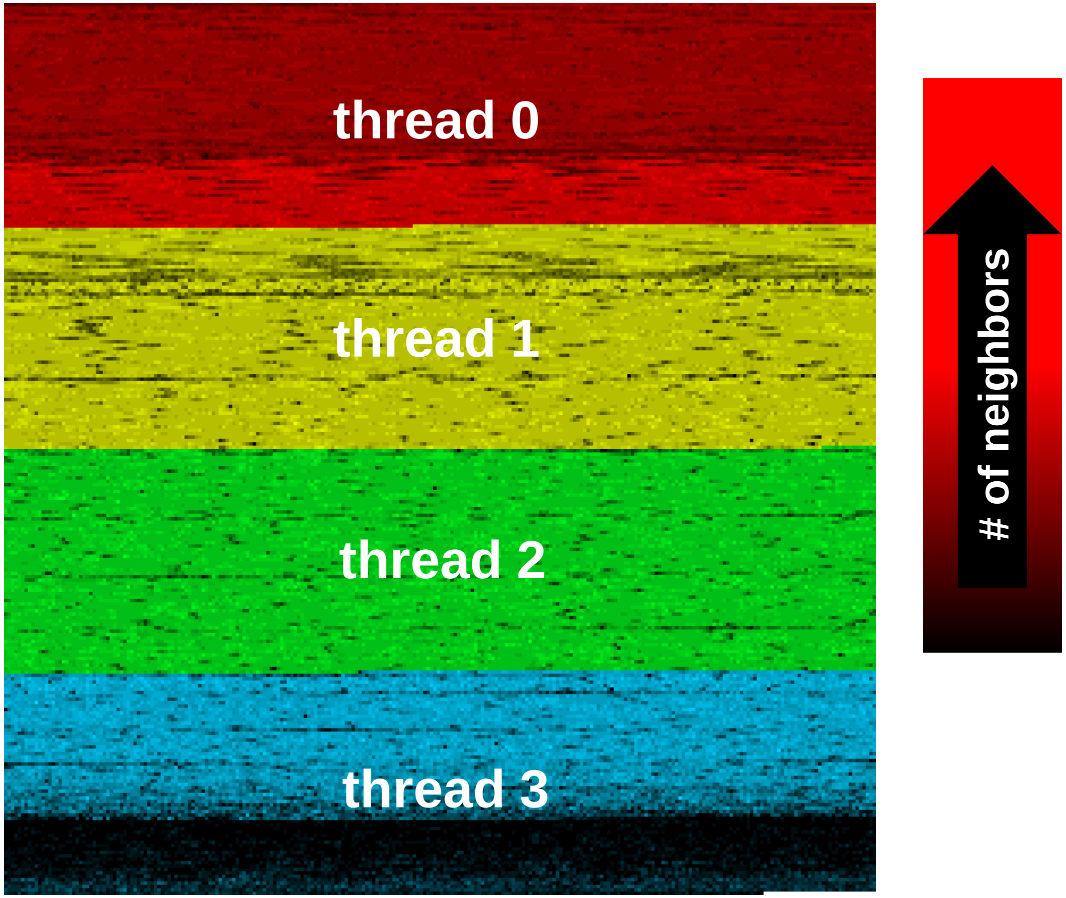


13,000 particles **67,000 particles**

Load balancing
Visualization of the workload (OpenMP run)

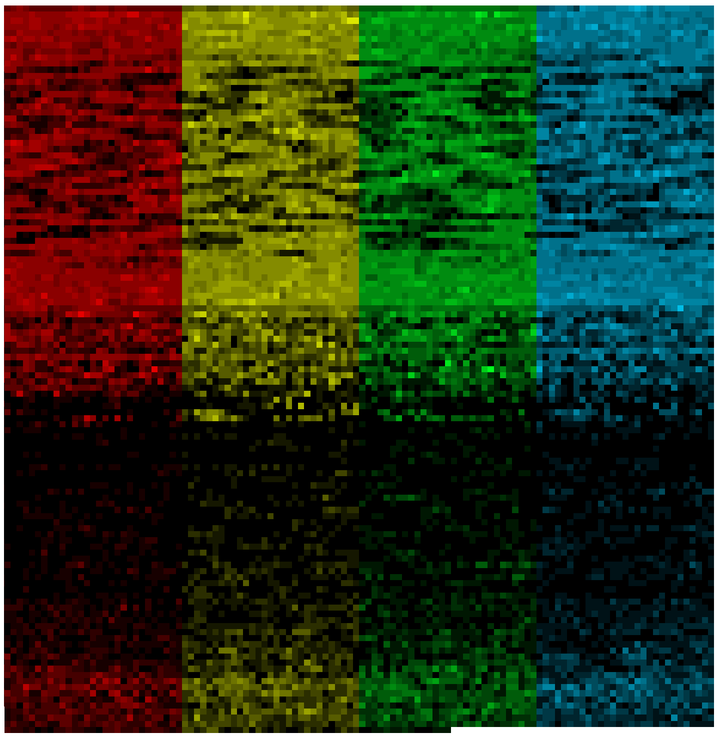


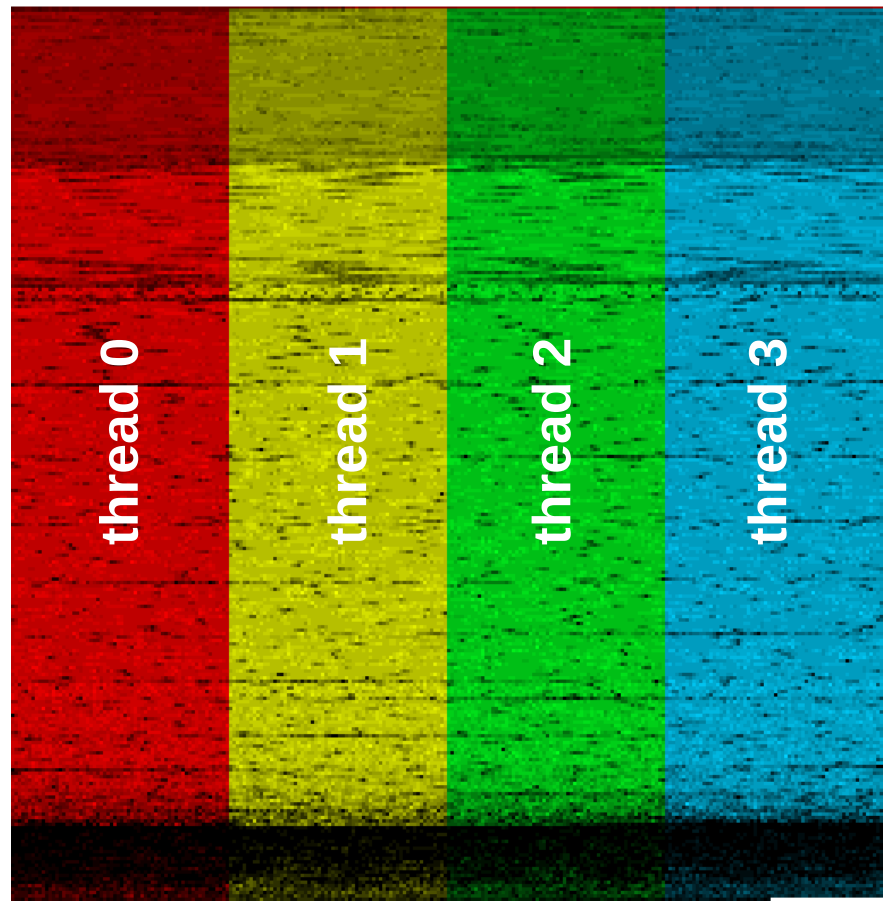
13,000 particles

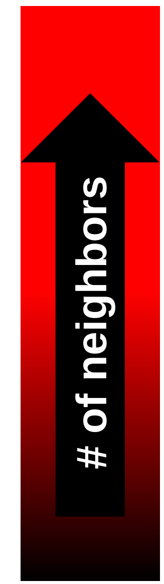


67,000 particles

Load balancing
Optimized Access Pattern



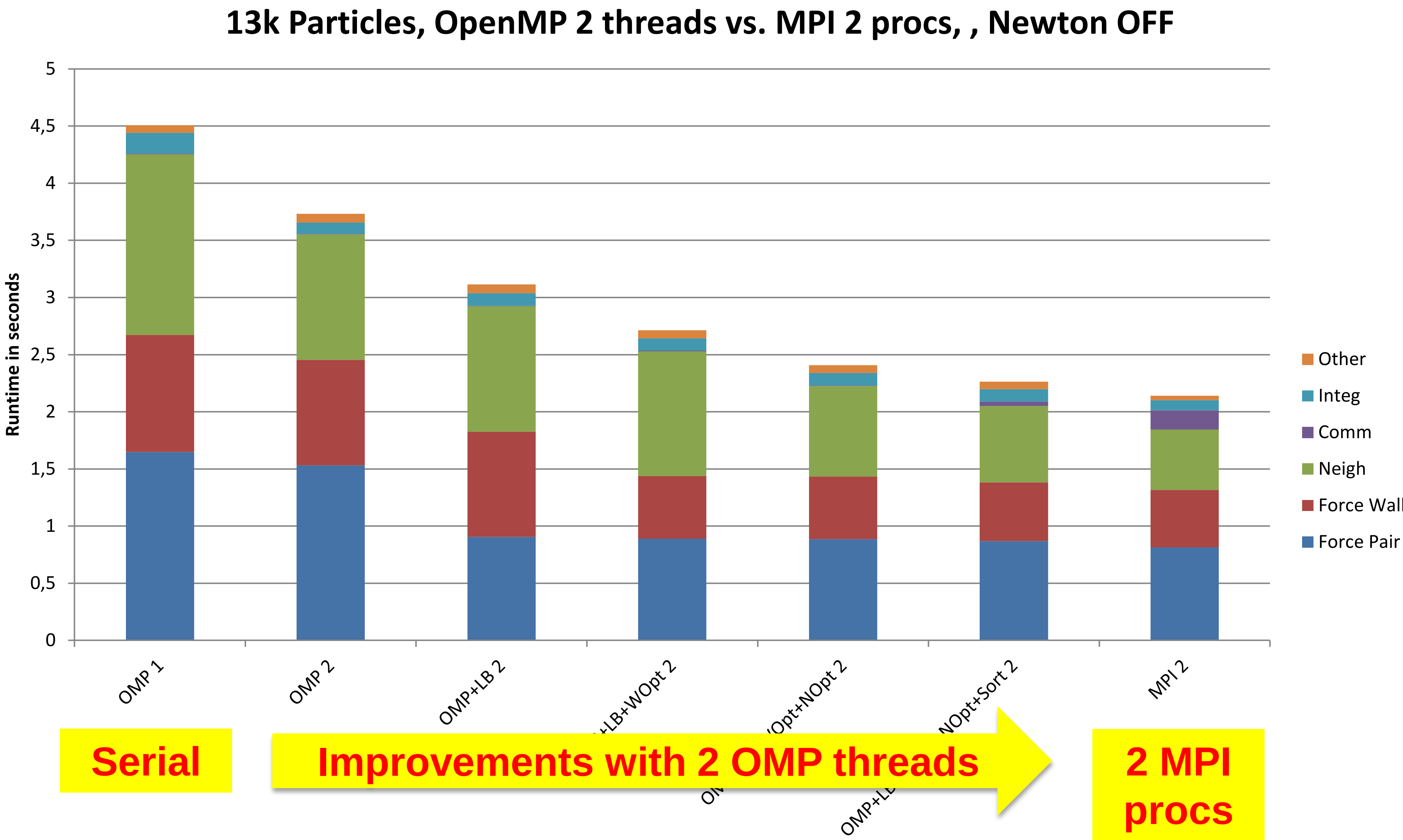




13,000 particles

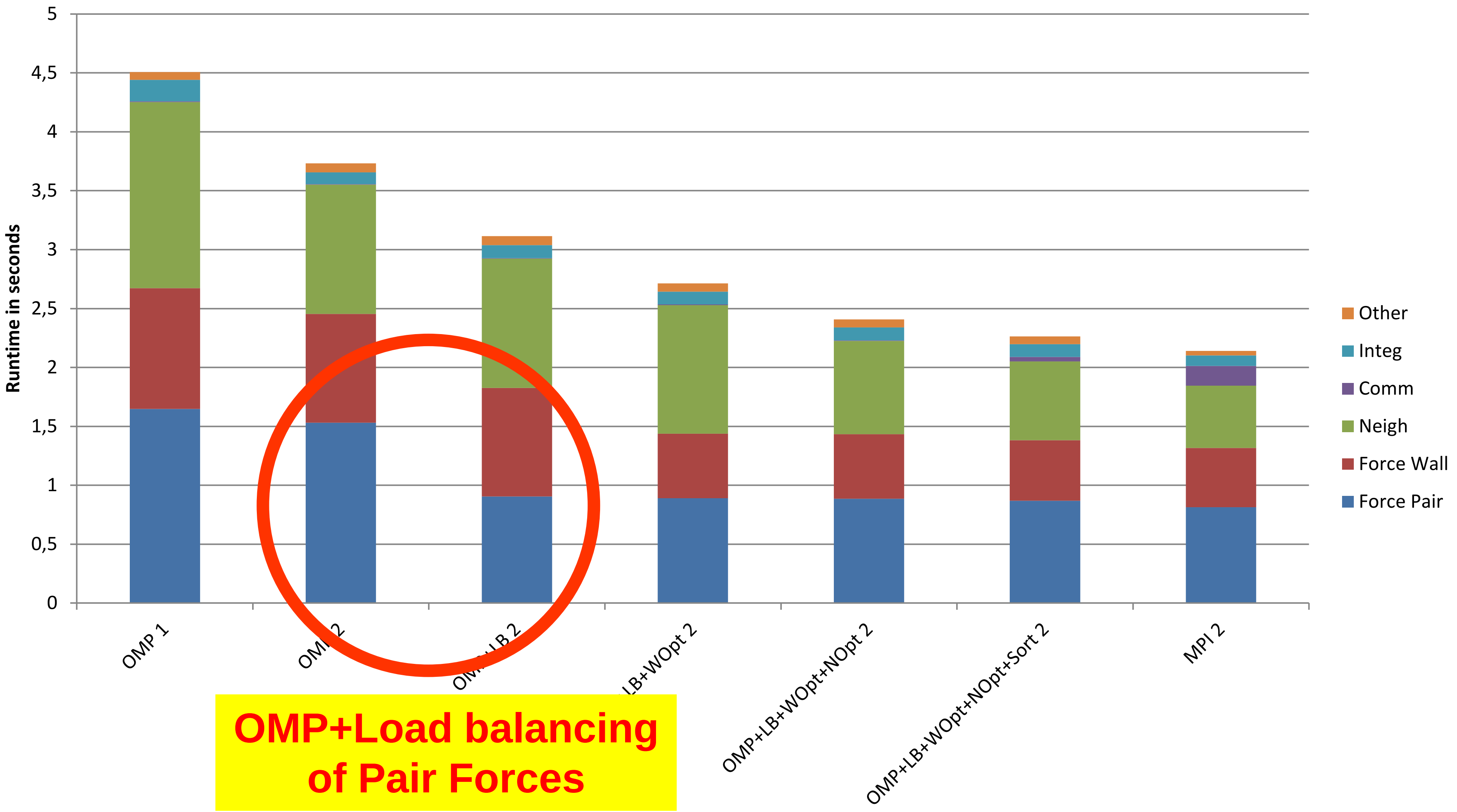
67,000 particles

OpenMP Results (miniMD-granular)
Newton 3rd law not used



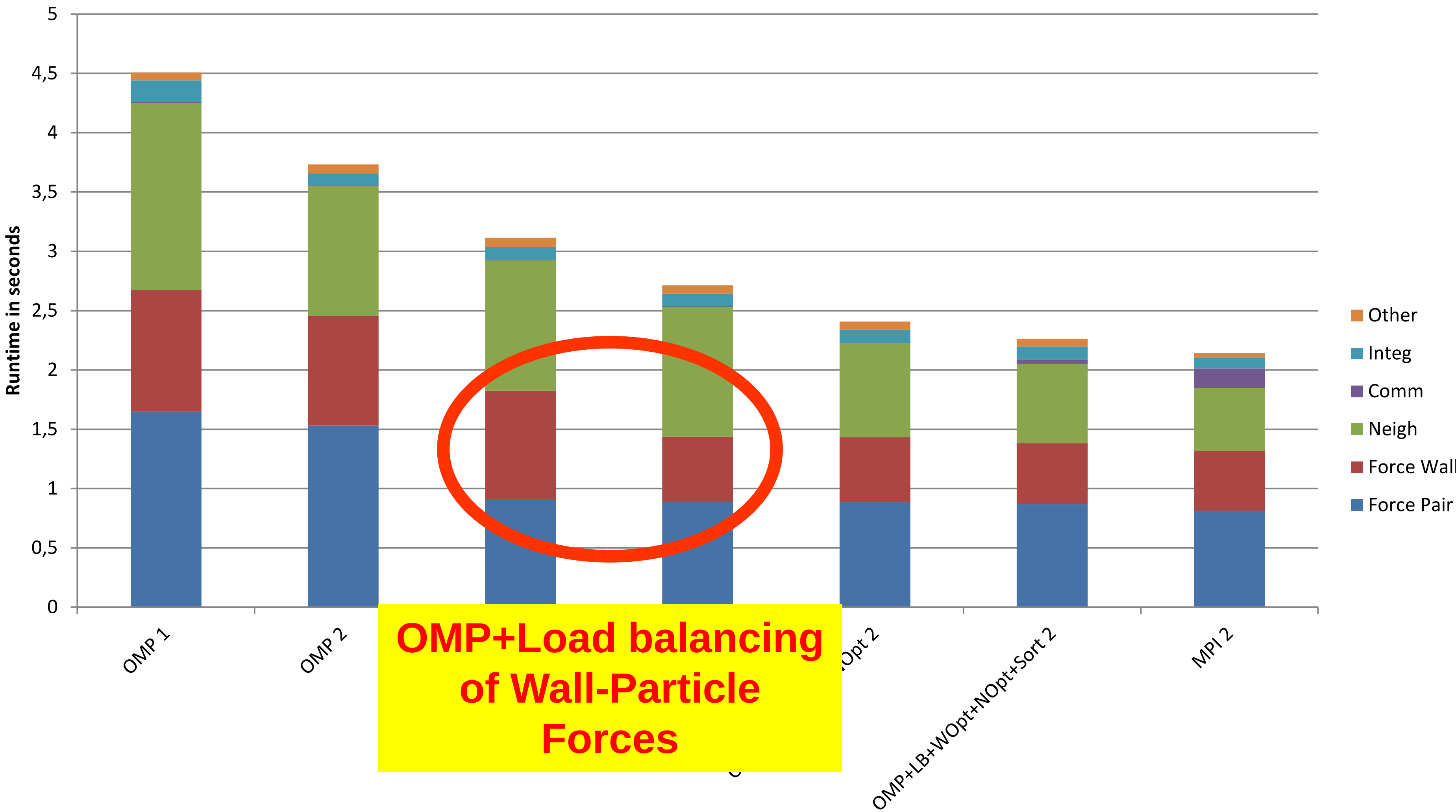
OpenMP Results (miniMD-granular)
Newton 3rd law not used

13k Particles, OpenMP 2 threads vs. MPI 2 procs, , Newton OFF

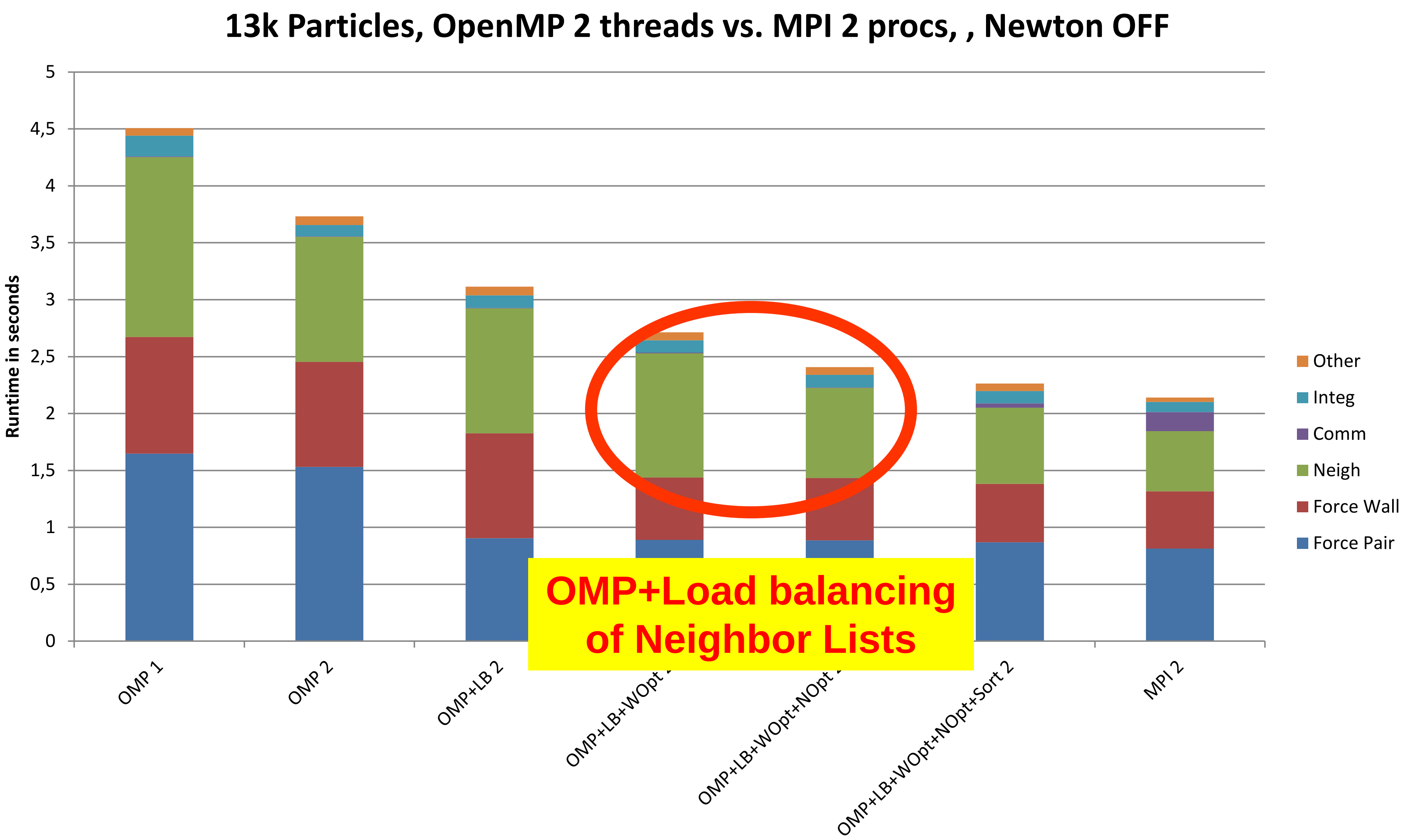


OpenMP Results (miniMD-granular)
Newton 3rd law not used

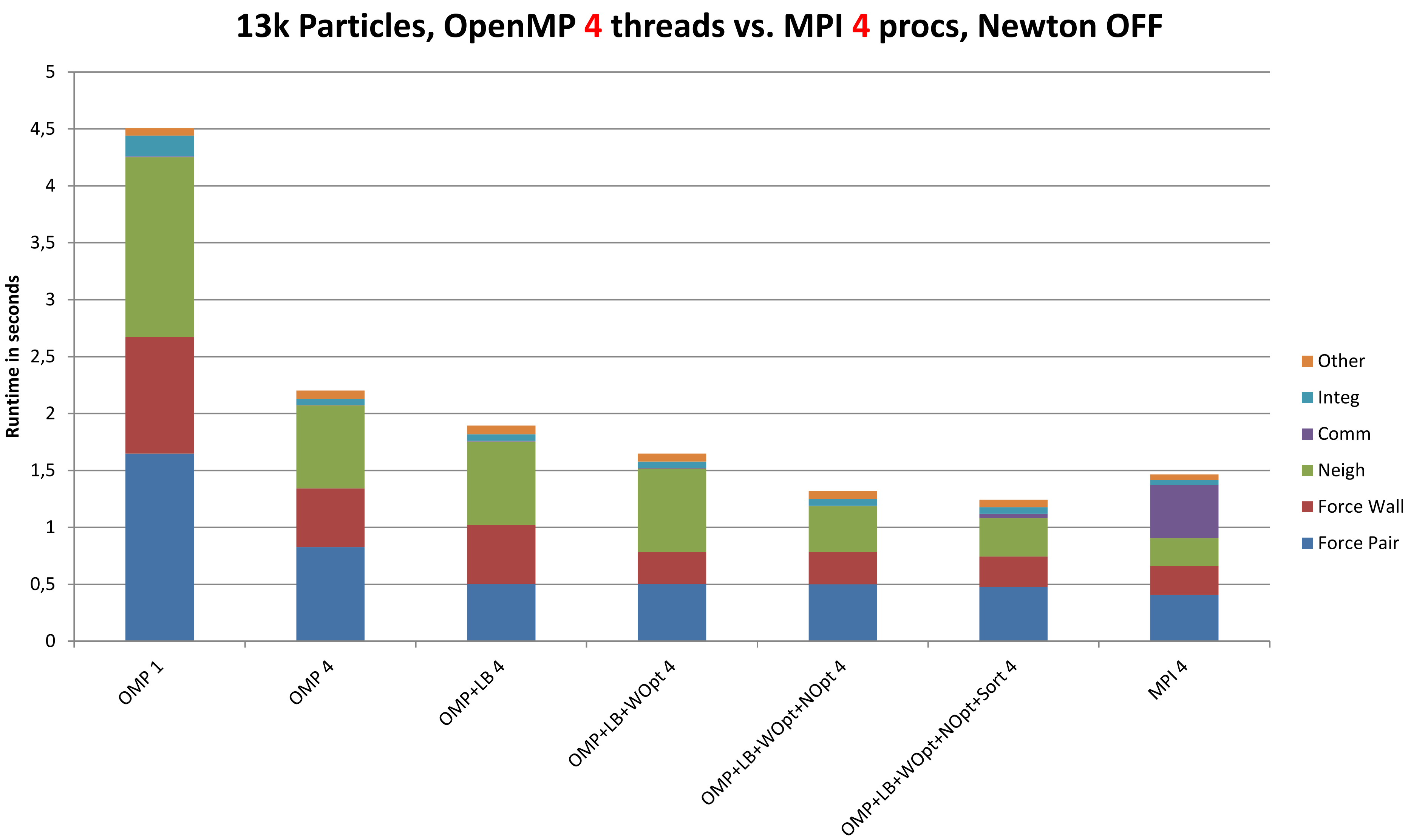
13k Particles, OpenMP 2 threads vs. MPI 2 procs, , Newton OFF



OpenMP Results (miniMD-granular)
Newton 3rd law not used

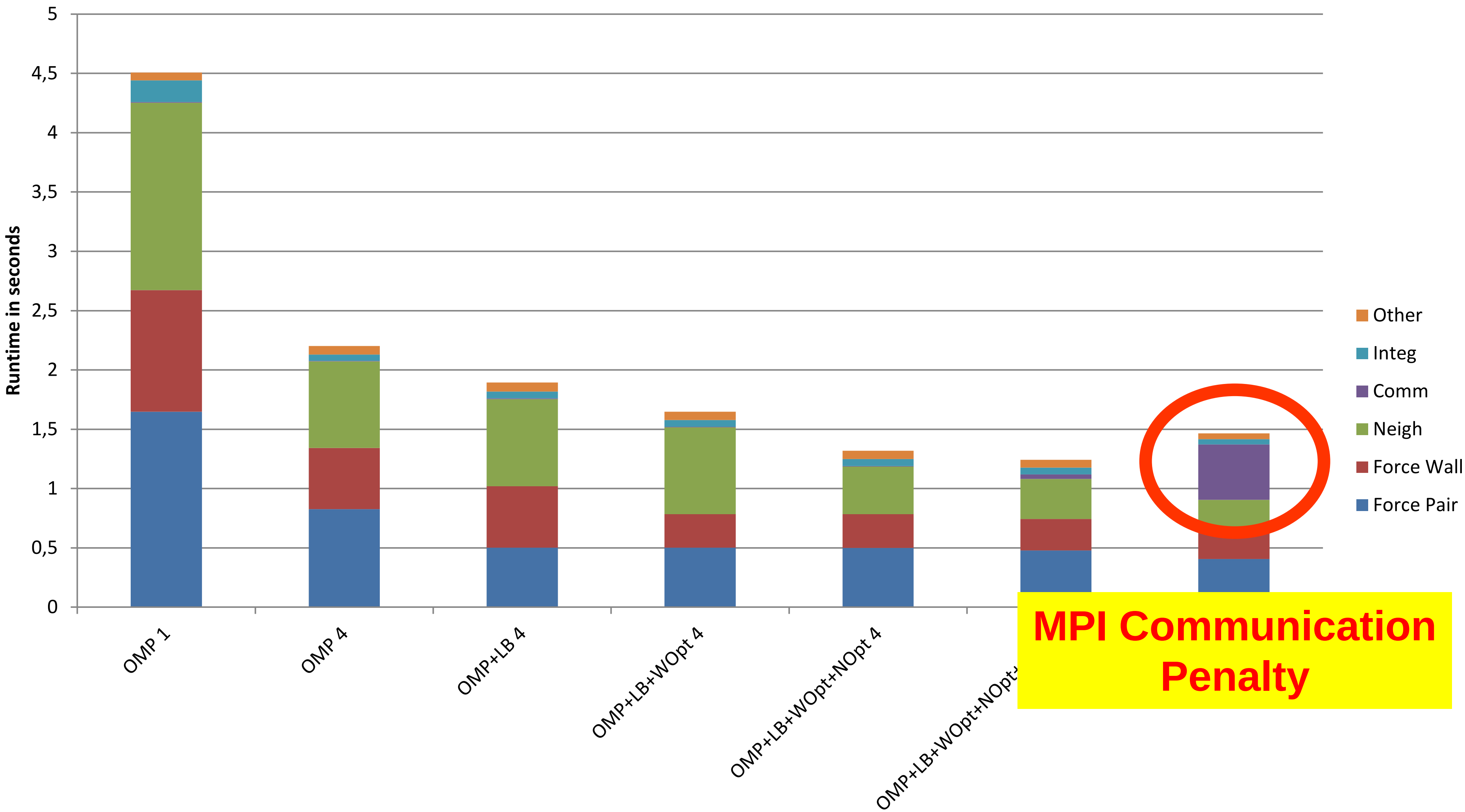


OpenMP Results (miniMD-granular)
Newton 3rd law not used



OpenMP Results (miniMD-granular)
Newton 3rd law not used

13k Particles, OpenMP 4 threads vs. MPI 4 procs, Newton OFF

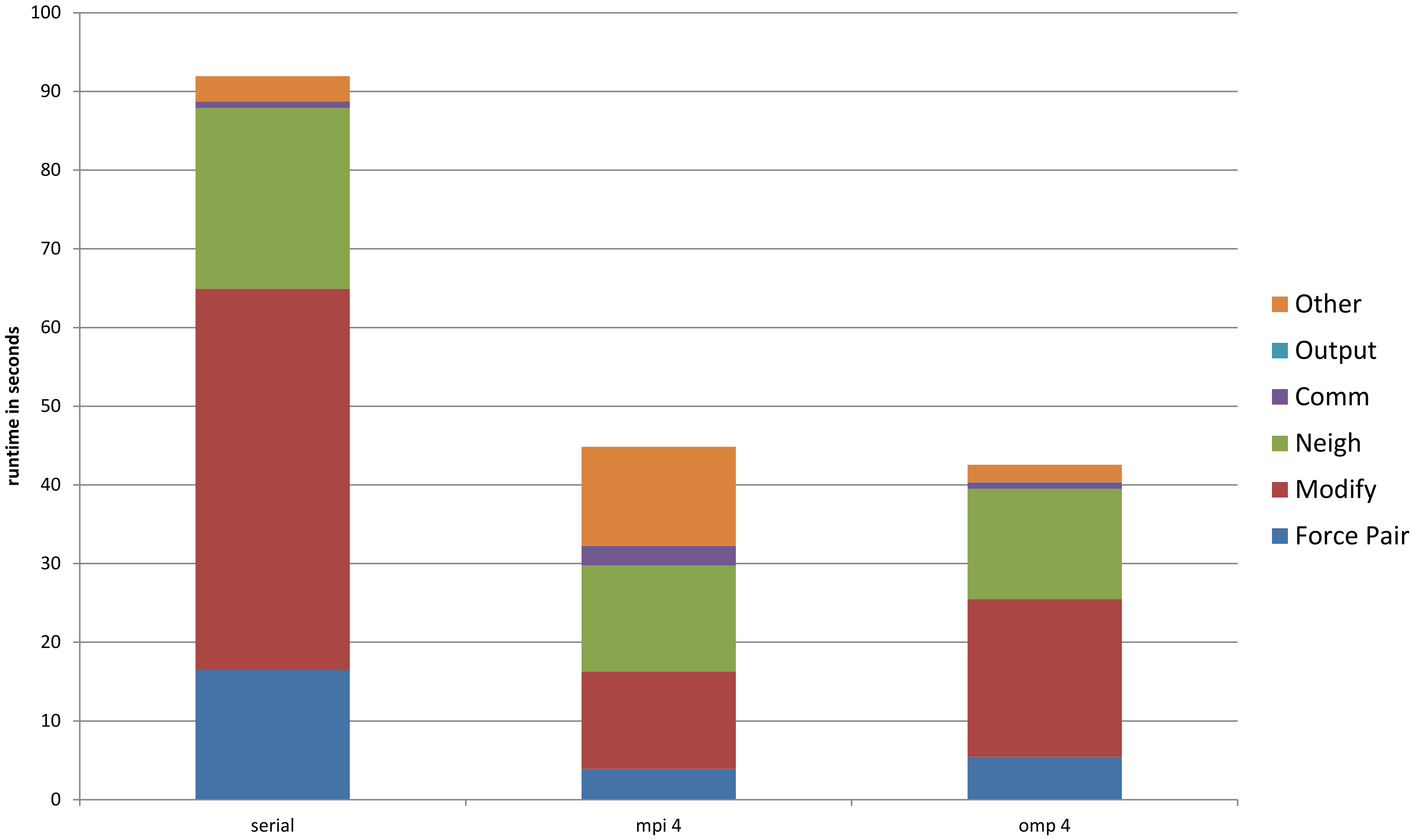


MiniMD -> LIGGGHTS

- MiniMD was a good start
- But threading optimizations in LIGGGHTS require more effort
- LAMMPS has OpenMP support (by Axel Kohlmeyer), uses Array Reduction
- In its current form the only way to add OpenMP support to LIGGGHTS is by code duplication
- Custom Locks instead of Array Reduction
- New features were added to allow detailed timings
- Load balancing

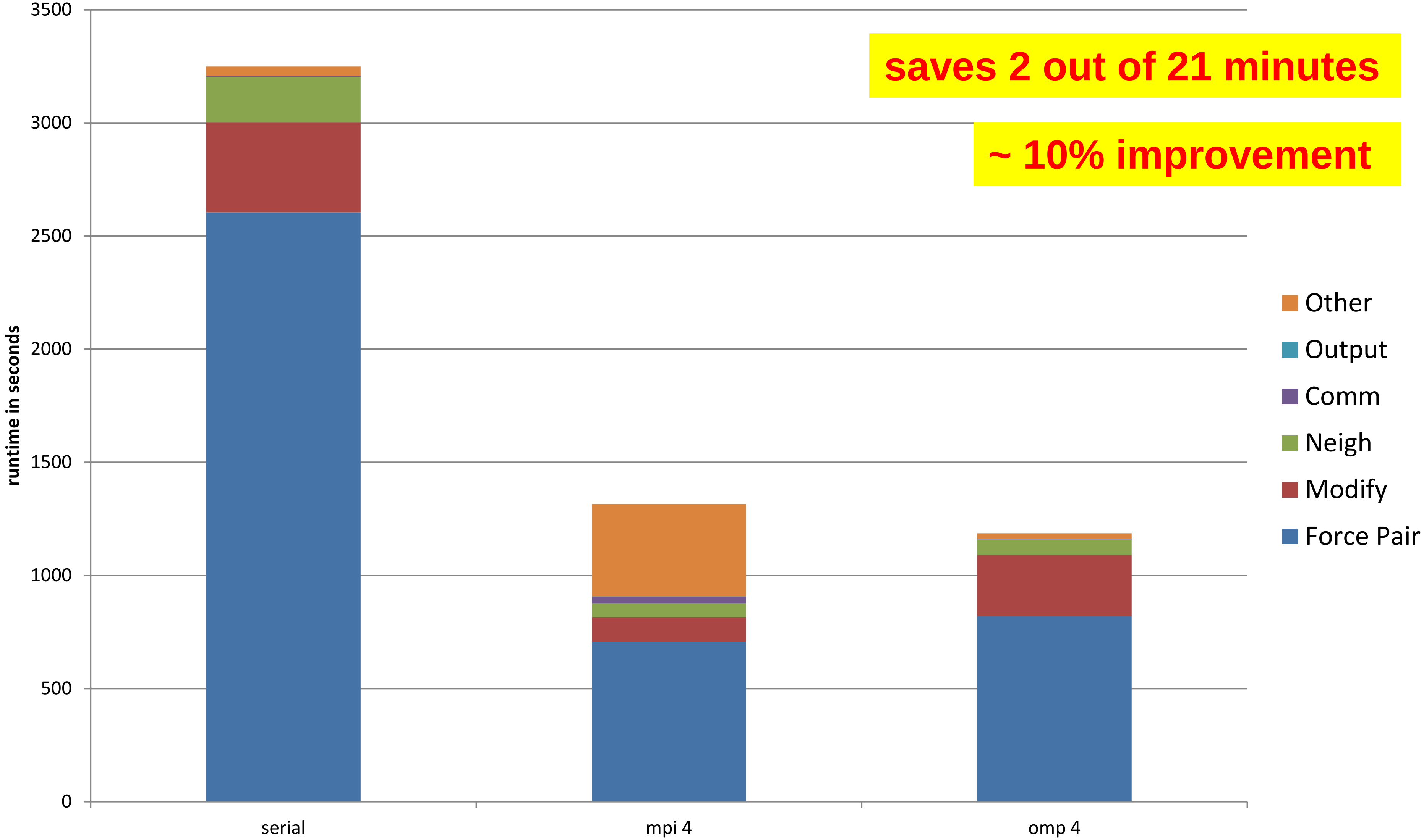
LIGGGHTS Results

Testcase 1 – 13k particles, MPI 4 vs OpenMP 4



LIGGGHTS Results

Testcase 1 – 67k particles, MPI 4 vs OpenMP 4



Outlook

- Currently working on LIGGGHTS 3.x
- OpenMP support should be much simpler
- Bringing OpenMP to more code paths (e.g. insertion of particles)
- Reaching feature parity
- Performance evaluation on bigger testcases from industrial partners

JKU
JOHANNES KEPLER
UNIVERSITY LINZ

Thank you for your attention!