N-body memory layout exploration

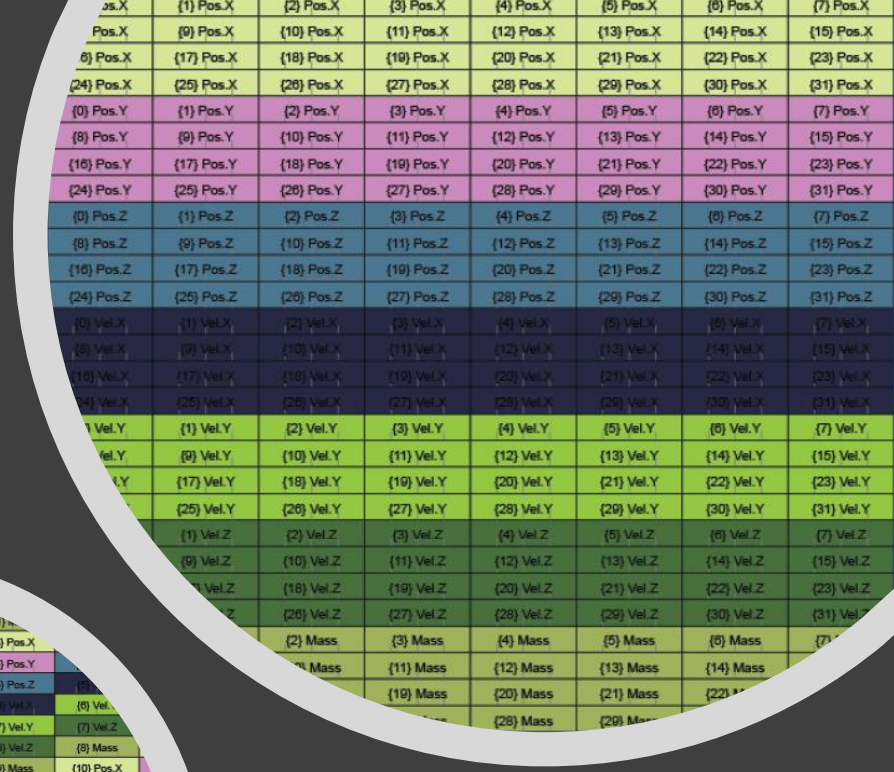
Bernhard Manfred Gruber
CERN, CASUS

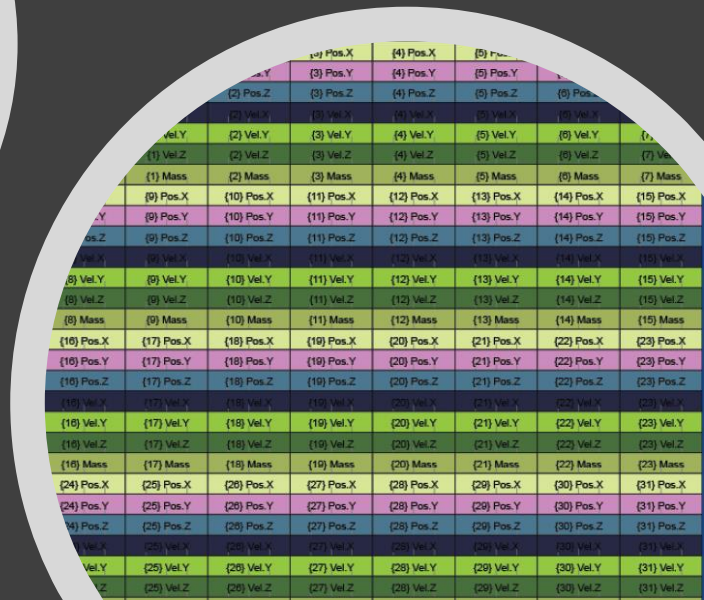
N-body simulations

- Used to simulate interaction of many bodies
  - Galaxies, stars, atoms, electrons, proteins, …
  - Gravity, electromagnetic forces, van der Waals forces, …
- https://www.youtube.com/results?search_query=nbody+simulation
- N^2 computational complexity
  - There are faster methods and acceleration data structures
    - E.g. using Octrees and approx. groups of bodies
    - But they still use the N^2 base algorithm inside

Objective

- Given 3 n-body CPU implementations
  - ~430 LOCs C++
  - with different memory layouts
    - AoS (Array of Structs)
    - SoA (Structs of Arrays)
    - AoSoA (Array of Structs of Arrays)
- Find the fastest possible CUDA implementation
  - by finding the fastest memory layout for global and shared memory
  - without functional changes





Tasks

1. Start by cloning: https://github.com/bernhardmgruber/nbody_memory_layout_student_project
2. Compile/run sample CPU code. Compare runtime of different memory layouts
3. Inspect disassembly and understand CPU vectorization and access pattern: https://godbolt.org/z/1jqo3z
4. Write CUDA versions with various memory layouts in global memory and compare
5. Use a profiler to find explanations for the runtime differences
6. Write a faster CUDA variant using shared memory for caching
7. Find the best memory layout combination for global/shared memory