# Displaying the Power of Heterogeneous Computing on GPUs

Emily Craig, Carina Salcedo, Elizabeth Wang
Lawrence Livermore National Laboratory

## Goal

Demonstrate speedups gained by incorporating a GPU in a way that can be used for demonstrations to the general public

## Introduction

Central Processing Unit (CPU) – standard processor used in computers

- Designed to perform a small number of tasks at a time as quickly as possible
- Reaching a limit in performance speed due to power and heat limitations

Graphics Processing Unit (GPU) – massively parallel processor developed for rendering graphics

- Designed to perform a vast number of tasks concurrently
- Now being used in conjunction with CPUs to accelerate many types of computationally expensive tasks

We chose to compare the speed of computation with and without a GPU accelerator for three applications in different fields of science

## Applications- LAMMPS

Large-scale Atomic/Molecular Massively Parallel Simulator, a molecular dynamics simulation

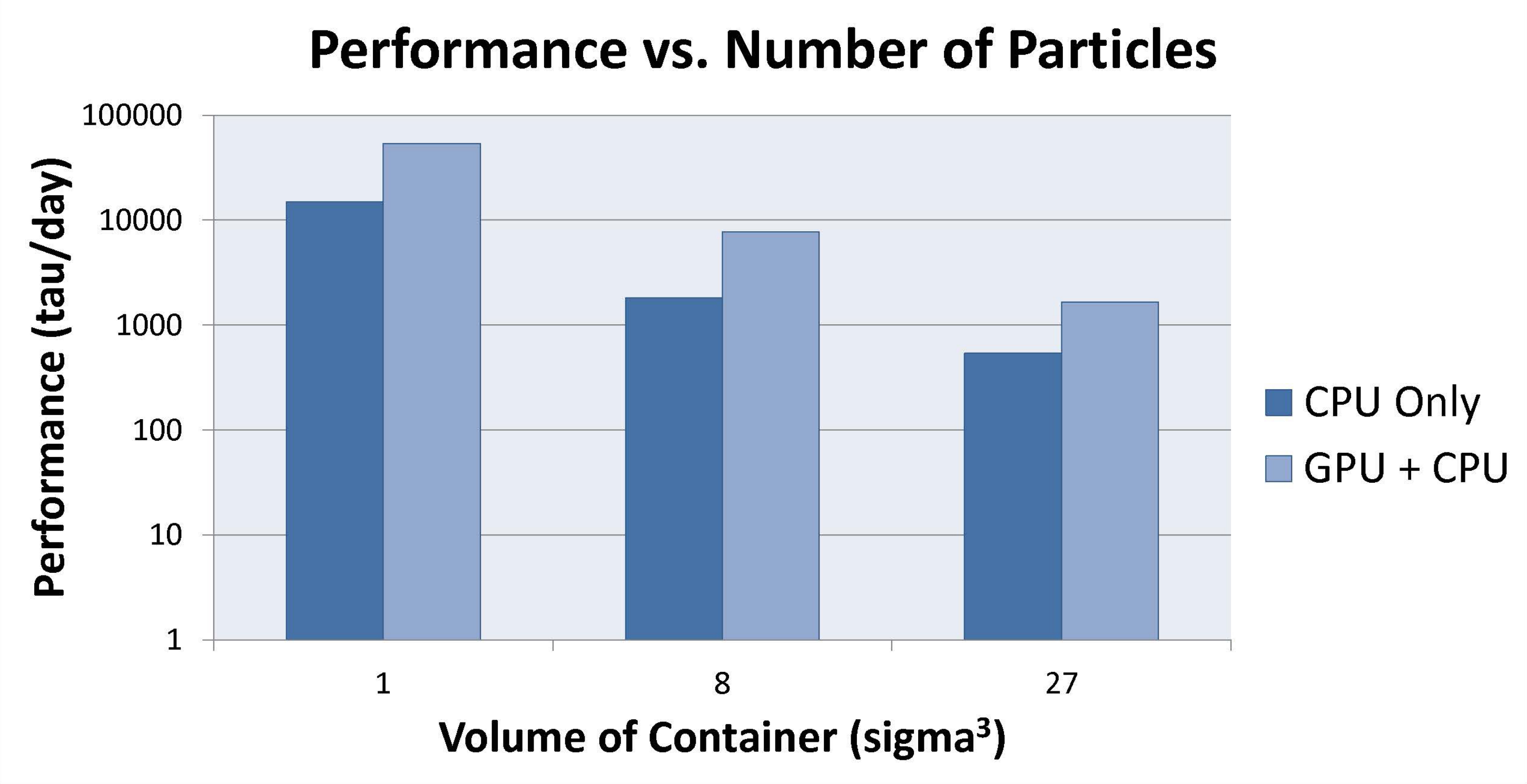
- Can be used to model atoms or simulate particle movement
- Generates a fixed number of particles inside a cube and animates them, tracking where they are and calculating their new positions
- Used Lennard-Jones simulation

**Challenges**

- Compiling application for the GPU
- NVIDIA Jetson Board is a non-standard hardware and software environment

**Results**

- The GPU ran 3 to 4.3 times faster than the CPU alone
- Drastic decrease in performance speed due to very limited memory
- External memory limitations caused crash when container dimensions increased to four



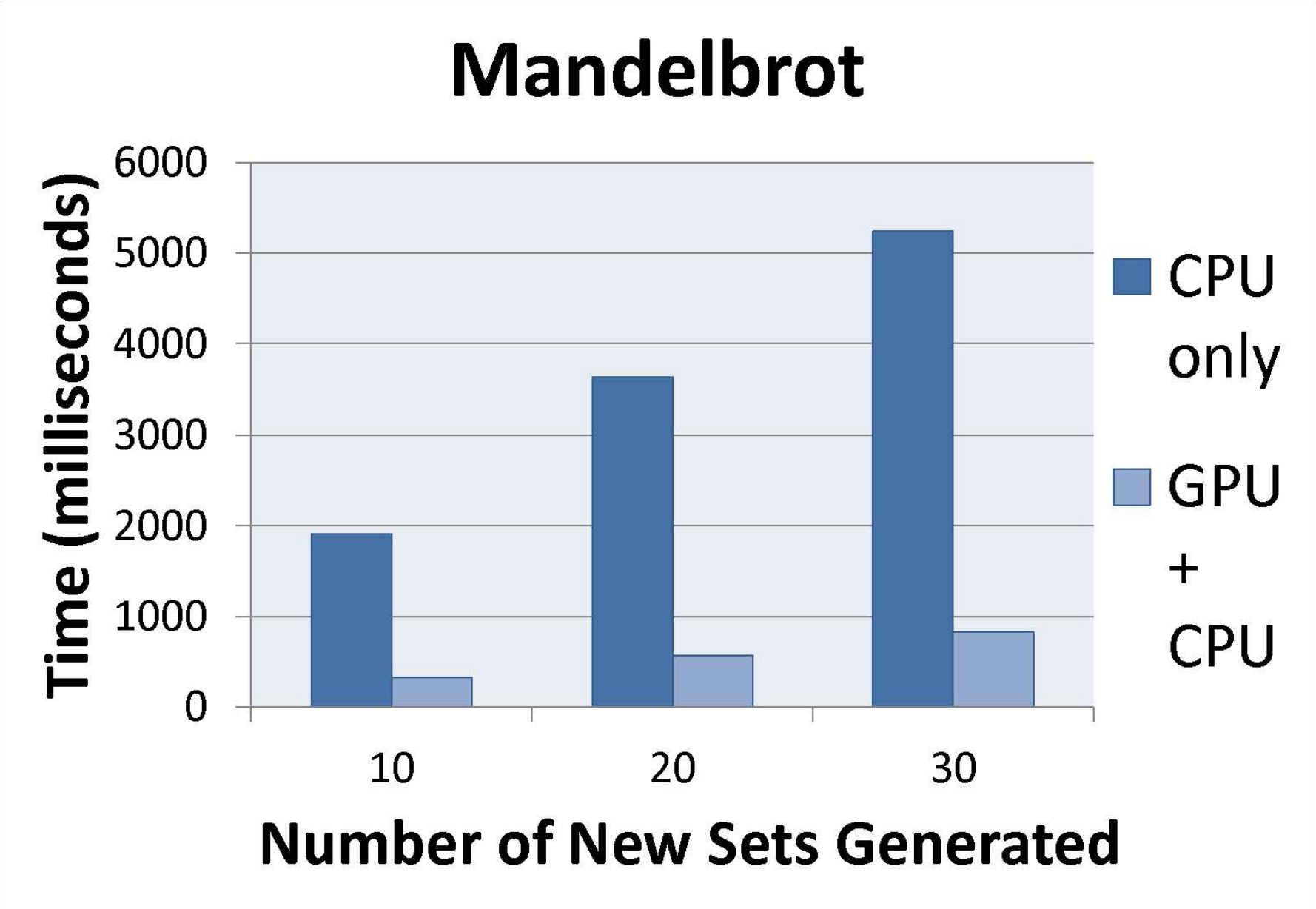
## Applications – Mandelbrot Set

Set of complex numbers c for which the function $z_{n+1}=z_n^2+c$ does not diverge when iterated from $z_0=0$

- Coloration based on divergence rate lends itself to visualization
- Prebuilt demo on our Jetson Boards with a visual simulation

**Results**

- In each run, the GPU was about 6 times faster than the CPU



## Applications – John the Ripper

Password cracking program

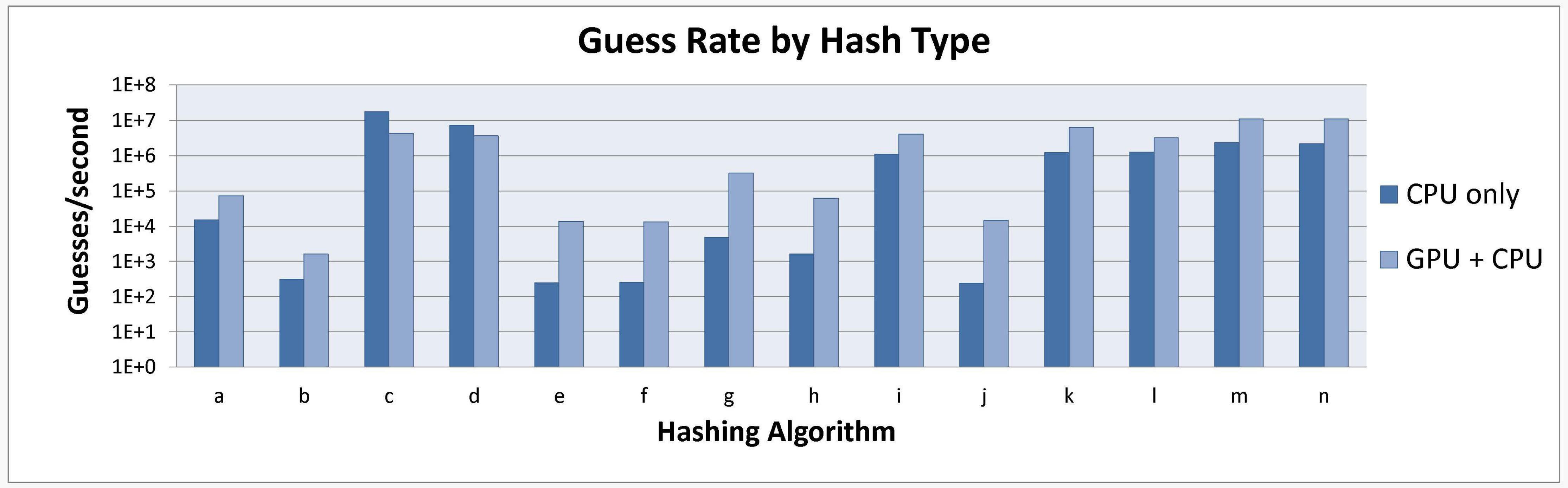
- Recognizes a variety of different types of hashing algorithms
- First uses a word list to guess, then guesses by brute force

**Challenges**

- Better support for OpenCL than CUDA, but our GPU was incompatible with OpenCL
- Supports cracking fewer hashing algorithms with GPU than on CPU alone

**Results**

- For 12/14 hashing algorithms, the GPU code was at least twice as fast as the CPU code
  - In four cases, the GPU was more than 50 times faster
- For the remaining two hashing algorithms, the CPU was 2-4 times faster than the GPU



## Next Steps

- Install and tune GUIs for LAMMPS and John the Ripper
- Design and implement a GUI that incorporates all three applications

Acknowledgements: David Fox, Joseph Han, Pam Hamilton

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-POST-698611