# Reproduciblity Report for the Georgia Institute of Technology

## Abstract

The goal of this paper is to reproduce the results from *The Vectorization of the Tersoff Multi-Body Potential: An Exercise in Performance Portability*, published by Höhnerbach, Ismail, and Bientinesi. In the paper published by Höhnerbach et al., the authors describe five optimizations to the Tersoff Multi-Body Potential algorithm on LAMMPS, and make several claims about these optimizations. The primary claim made is that the performance operations are portable, meaning not machine-specific. The present study attempts to reproduce these claims on a new CPU architecture, PowerPC, using a two node cluster composed of two Power8 CPUs per node, and four Nvidia P100 GPUs.

## Introduction

In \textit{*The Vectorization of the Tersoff Multi-Body Potential: An Exercise in Performance Portability*}, Höhnerbach et al. describe five optimizations to Tersoff Multi-Body Potential simulations in LAMMPS. For context, the Tersoff Multi-Body Potential simulation is a form of molecular dynamics that differs from pair potentials (which simulate by calculating the force acting on each pair of molecules using scalar distance) by also taking into account the atoms close to either in the pair and the angle between the pair and surrounding atoms. \\

The optimizations included in Höhnerbach et al.'s paper fall into three categories: scalar, vectorization, and optimizations to aid vectorization. For the rest of the paragraph, molecule I and molecule J will be referenced and refer to two molecules whose pair-wise forces are being calculated. The scalar optimization is a restructuring of the algorithm so that the zeta term and its derivatives are computed only one, in the first loop, and the product of zeta and sigma is only calculated in the second loop. The vectorization optimization is in three subcategories: 1.) for vector architectures with short vectors, molecule I is mapped to parallel execution, and molecule J to data parallelism; 2.) for vector widths that exceed the iteration count of j, molecule I is mapped to parallel execution, and molecules I and J to data parallelism; 3.) for GPU, molecule I is mapped to parallel execution and data parallelism, and molecule J to sequential execution. The optimizations that aid vectorization are fast forwarding through the loop that looks through neighbors, until as many lanes as possible can participate in the calculation of a numerical kernel and reducing the amount of lane spinning by filtering the neighbor list in the scalar segment of the program. \\

In addition to optimizations, the author introduces two new modes of operation: single precision and mixed precision. By default, LAMMPS uses double precision when representing numbers. The mixed precision works by having all calculations in single precision, except for the accumulations. It is important to note, however, that despite the claimed portability, mixed precision is not implemented for Power8 systems. The primary benefit of single and mixed precision is that the simulations run quicker. However, this comes at the expense of accuracy. \\

This paper aims to reproduce several claims made about these optimizations. The central claim is the portability of their performance optimizations, meaning that none of these optimizations are machine-specific. Hohnerbach et al. test this claim by running the optimized code on various machines including four Vector ISA's (NEON-ARM, SSE4.2, AVX, AVX2), GPUs, and an Intel Xeon Phi accelerator. Throughout Höhnerbach et al.'s paper, the metric used to quantify performance is simulated time over run time, which has units ns/day. Another claim is that the performance boost is due to scalar and vector improvements, which is tested by single thread runs. In single thread runs on a variety of architectures, the authors find between 1.9x and 6.4x speedups. The authors also claim that the scalar and vector improvements are scalable, which is tested by single node runs, and various cluster configurations of nodes (including GPU). In single node execution on a variety of architectures, the authors find between 2.69x and 5x speedups. The last claim that the authors make is that the single and mixed precision modes of their code have within 0.002 relative total energy for a system of 32000 atoms. This claim is validated by the authors through running one million timestamps for a system of 32000 atoms and verifying that in no single timestamp does the relative total energy exceed the threshold.\\

## Machine Description

The machine used in this paper has two nodes, two CPUs per node, four GPUs per node, and a 256GB DDR4 RAM. Specifically, each CPU is the IBM Power8+, which has 10 cores and 64 threads. Each GPU is the Nvidia P100, which are connected with NVLink. The interconnect between the two nodes is the Mellanox ConnectX-4 InfiniBand EDR, which is the fastest interconnect on the market. The vendors who provided this hardware configuration are Flagship Solutions, TechData, and IBM. The operating system used is CentOS version 7.4.

## Compilation Steps

This section describes the steps we have taken to compile and run the experiments on our system.

We compiled all code using gcc-5.3.0 and Openmpi-2.1 with cuda support.

**Reference Implementation:** master branch of github.com/lammps/lammps

**Optimized Implementation:** master branch of github.com/HPAC/lammps-tersoff-vector

The optimized repository contains a machines directory with several subdirectories like aices-hw_phi, aices-westmere, hpac-arm, lrz-ib_phi. Each subdirectories represents a specific cluster named with the following format "<ORGANIZATION_NAME> - <CLUSTER ARCH>". For our system, we introduced a new machine subdirectory "gt_scc17-ibm_power8."

1. Patch new optimization files into LAMMPS/src: The paper introduced four new files are intel_intrinsics.h, pair_tersoff_intel.cpp, pair_tersoff_intel.h, and vector_math_neon.h
2. Install the MANYBODY, USER-OMP, and USER-INTEL packages into LAMMPS
3. Execute `make power8`: this will invoke the Power8 Makefile found at src/MAKE/OPTIONS/Makefile.power8

This produces a lmp_power8 binary. To run, the binary, we pass the following parameters into LAMMPS "-in DESCRIPTION -pk omp 0 -pk intel 1 mode $MODE -sf intel -v p vanilla."

## Accuracy Measurements

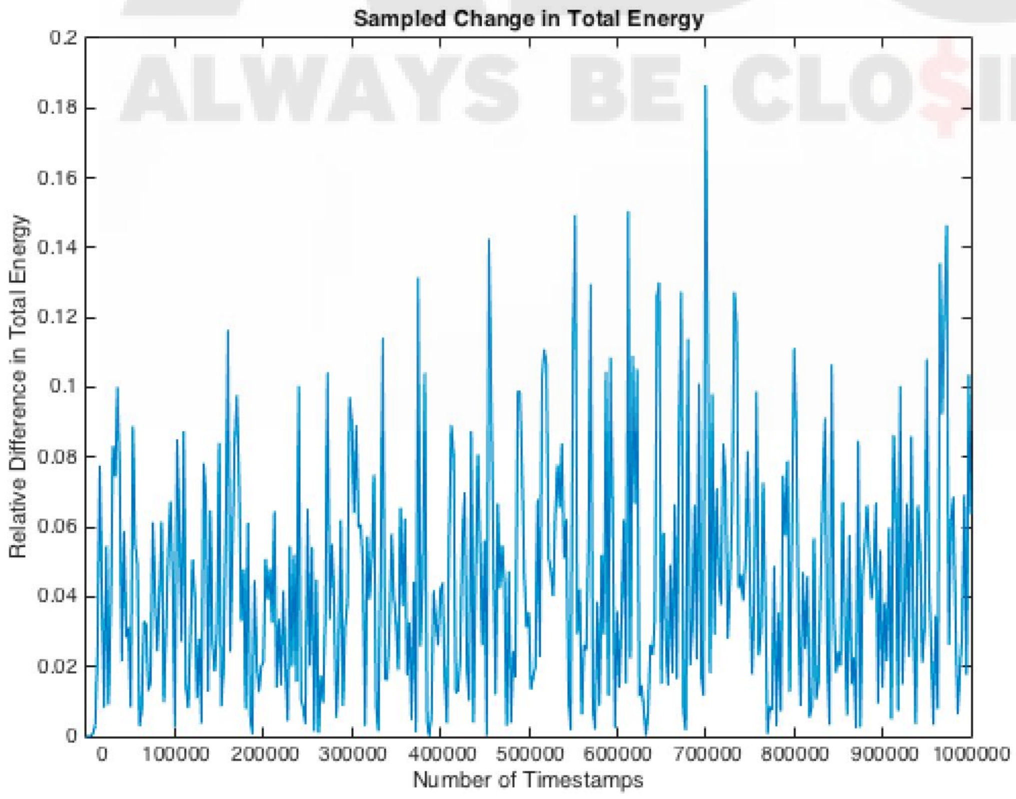

**Figure 0.** Plot of Relative Difference in Total Energy at Every 100 Timesteps, 1 Million Timesteps Total.

One of the important parts of Höhnerbach et al.'s paper was the validation of the single precision solver introduced. The authors set a threshold of 0.002 relative difference in total energy between single and double precision solvers for a system of 32000 atoms. The findings in this paper however indicate a maximum 0.25 relative difference in total energy, which is 125 times larger than the threshold. Thus, the claim that single and mixed precision modes of their code have within 0.002 relative total energy for a system of 32000 atoms could not be reproduced. Validation of the mixed precision solver could not be completed because the authors of the optimizations never ported mixed precision to Power8.

## Benchmarks

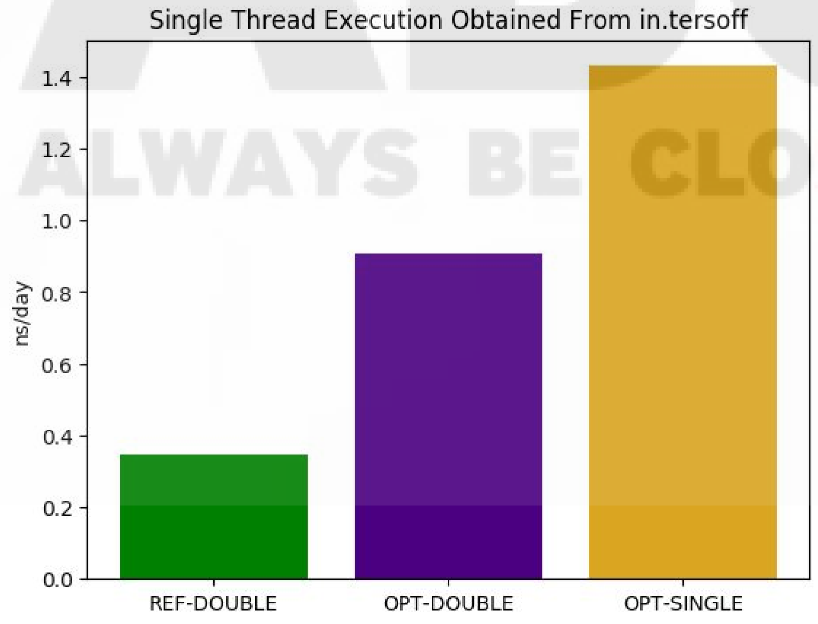

**Figure 1.** Plot of Single Threaded Execution obtained from Input.Porter.

The speedups found in single threaded execution are 2.7x for double precision and 3.9x for single precision, as shown in Figure 1. In the paper by Höhnerbach et al., the speedups found by the authors were at 1.9x for single precision in the worst case and 3.5x for double precision in the worst case. Thus, the claims of scalar and vector optimizations have been reproduced.

## Conclusions

Of the five claims that were sought to be reproduced in this paper, few were reproduced. For the claim that the single and mixed precision modes of their code have within 0.002 relative total energy for a system of 32000 atoms, an explanation for the results not being reproduced is that these optimizations were originally written for Intel architectures, not Power8, so in the processing of porting the code to Power8, the vectorization may have been imperfect. To that point, the authors did not even release a mixed precision port to Power8. The claim that the performance boost is due to scalar and vector improvements was reproduced.