LAMMPS performance on ARCHER/ARCHER-KNL

Authors, EPCC

Version 0.1, March 17, 2017

1. Introduction

The aim of this short report is to provide some representative performance figures for the LAMMPS application on the ARCHER service as existing in Spring 2017. This encompasses both the main machine (Intel Ivy Bridge), to be referred to as "ARCHER," and the Knights Landing (KNL) testing and development platform, which will be referred to as "ARCHER-KNL." In addition, we provide figures for the slightly more recent Intel Broadwell architecture which will form the basis of the UK Tier 2 system Cirrus, to be referred to as "CIRRUS".

The question of providing comparison of application performance on different architectures has become something of a bone of contention in HPC circles in recent times. As architectures become more heterogeneous, the question of what represents a fair comparison, or a relevant comparison for a given application or benchmark, can become debatable. For a recent discussion, see for example Pennycook *et al.* [1]. LAMMPS is a good example here, as there are a number of different implementations for any given problem, of which we will investigate four. There are also a wide range of benchmark problems one could choose. Here we employ five standard cases used discussed in the LAMMPS documentation [2].

However, to reduce complexity, we restrict ourselves to the consideration of single node performance (one or two sockets). We have also attempted to employ meaningful and comparable metrics of performance in terms of time and energy where possible.

The following sections discuss the scope of the study, hardware and software details, performance metrics and the benchmarks; details of the various configuration parameters used on each platform are relegated to an appendix.

2. Hardware, Software and Benchmarks

We provide here a brief description of the hardware used and some comments on the LAMMPS package. For details of KNL hardware features, the interested reader is referred to, e.g., Jeffers *et al.* [3].

2.1. Hardware

The hardware platforms considered are:

1. ARCHER. This will refer to the main Cray XC30 service machine with nodes containing two 12-core Intel Xeon E5-2697 v2 (Ivy Bridge) chips. Each core is clocked at 2.7 GHz and can support two threads in hardware. Within a node, two sockets provide access to two 12-core chips connected by two QuickPath Interconnect (QPI) links. Each core can issue 4 double precision floating point multiply and add operations per cycle (8 flop/cycle), providing 21.6 GFlop/s per core, or roughly 0.52 TFlop/s per node. The thermal design power rating is 260 W per node.

2. ARCHER-KNL. This refers to the KNL test and development system where each node houses one 64-core Intel Xeon Phi 7210 (Knights Landing) card [4]. Each core is clocked at 1.3 GHz and can support 4 threads in hardware. The KNL cards are configured in quadrant mode with 100% of high-bandwidth MCDRAM used in cache mode [5]. Each core has two vector units which can each issue 16 double precision floating point instructions per cycle (vector length 8 doubles), providing 41.6 Gflop/s per core, or roughly 2.66 TFlop/s per node. The manufacturer's thermal design power rating is 215 W per node (or per socket in this case).

3. CIRRUS. One node has two 18-core Intel Xeon E5 2695 v4 (Broadwell) processors. Each core runs at 2.1 GHz and supports 2 threads in hardware. Each core can issue 4 double precision floating point multiply-add instructions per cycle (vector length 4 doubles), giving 16.8 Gflop/s per core, or about 0.60 TFlop/s per node. Two sockets have thermal design power rating of 290 W.

2.2. Software

LAMMPS stable version dated 17 November 2016 is used as the basis of the benchmarks. A number of different options are available which determine the exact implementation of any given interaction style used at compilation time. The following are relevant for this work:

1. “NO-PACKAGE”: this is the default MPI only implementation;
2. “USER-OMP”: a package offering optimised OpenMP implementations;
3. “KOKKOS”: uses the Kokkos accelerator abstraction framework[6];
4. “USER-INTEL”: a package with implementations optimised for Intel architectures, in particular, with attention to vectorisation.

Each benchmark (where possible) will be run in each of these 4 modes to identify the performance of the various implementations for each platform.

2.3. Benchmarks

Five standard benchmarks from the LAMMPS `./bench` directory are employed. These are referred to as: LJ, FENE-CHAIN, EAM, GRANULAR, and RHODOPSIN. The first is a standard short-range Lennard-Jones (LJ) potential; the second represents a polymer-like chain with a finite extensible nonlinear elastic (FENE) potential; the third uses the embedded atom method (EAM); the fourth is a short-range frictional potential for granular media; the fifth is a full protein example using Rhodopsin, including bonds, angles, dihedrals, constraints etc, along with long-range force computing by the LAMMPS default implementation of a particle mesh approach.

Standard LAMMPS input files where unaltered other than to increase the number of time steps from 100 to 10,000 in all cases. The number of atoms/particles was set to 512,000 in all cases via:

```
# LJ
./lmp_exe -var x 4 -var y 2 -var z 2 < in.lj
# FENE-CHAIN
./lmp_exe -var x 4 -var y 2 -var z 2 < in.chain.scaled
# EAM
./lmp_exe -var x 4 -var y 2 -var z 2 < in.eam
# GRANULAR
./lmp_exe -var x 4 -var y 4 < in.chute.scaled
# RHODOPSIN
./lmp_exe -var x 4 -var y 2 -var z 2 < in.rhodo.scaled
```

2.4. Metrics

The following complementary metrics of performance are employed:

1. The LAMMPS standard output performance measure atoms/core/second. This is an absolute metric of performance for given benchmarks, but is not relevant when comparing different benchmarks on the same platform, and questionable when comparing the same benchmark on two different platforms. We have therefore taken the "Loop time" reported by the LAMMPS standard output and computed LAMMPS time steps per second for the run on a node basis.

2. For ARCHER, energy use per run was collected via Craypat (but not successful in all cases). For ARCHER-KNL, energy use per run is reliably available via the Cray Resource Usage Rhubarb (RUR). Unfortunately, no energy figures were available on CIRRUS at the time of writing. A metric of LAMMPS time step per Joule provides a measure of "true" cost based on energy, again on a node basis.

3. The number of floating point operations for each run was monitored (Craypat) and used to gauge a percentage of peak performance; a full "roofline" analysis not not attempted.

2.5. Protocol

Benchmarks were run in standard batch queue configurations on each machine until four successful timings where obtained for each benchmark/implementation. As all benchmarks are single-node, it is not expected that machine load would influence the times.

3. Results

Results for time steps per second and time steps per Joule are presented in Figure 1 for ARCHER, Figure 2 for ARCHER-KNL, and Figure 3 for CIRRUS. The results for ARCHER do not show significant differences between the implementations, and little difference between one and two hardware threads (with little improvement for two hardware

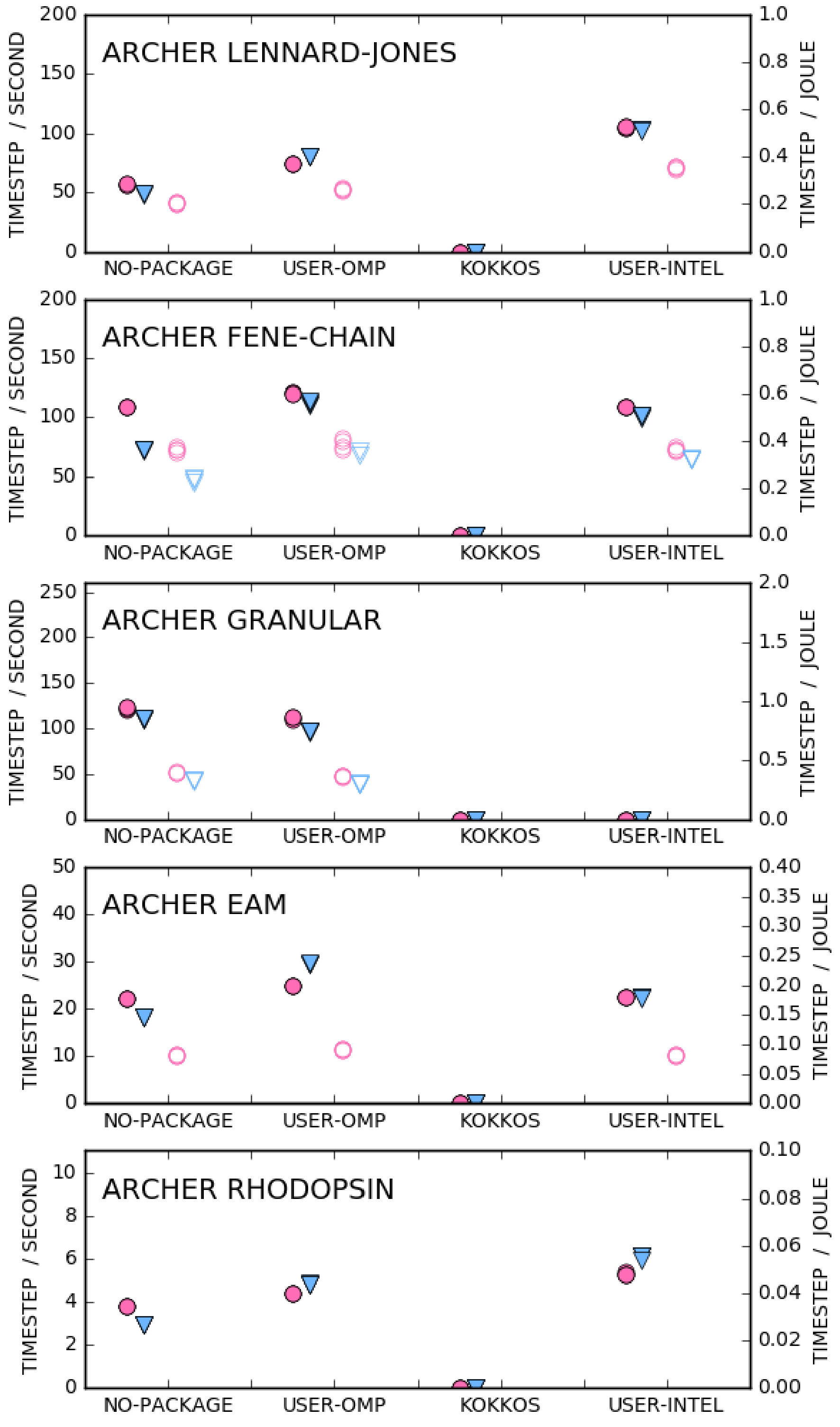

Figure 1: Performance figures for benchmarks run on ARCHER for four different implementations. For each benchmark the filled symbols are time step/second (left-hand scale), and the open symbols are time step/Joule (right-hand scale). All runs are 24 MPI tasks on 24 cores; circles and triangles are 1 and 2 hardware threads per core, respectively. Measurements of zero indicate there is no implementation for the relevant potential. For each measurement, four repeats are present (the symbols overlay each other in most cases indicating any statistical error is no larger than the symbols). All benchmarks are for 512,000 atoms run for 10,000 steps.

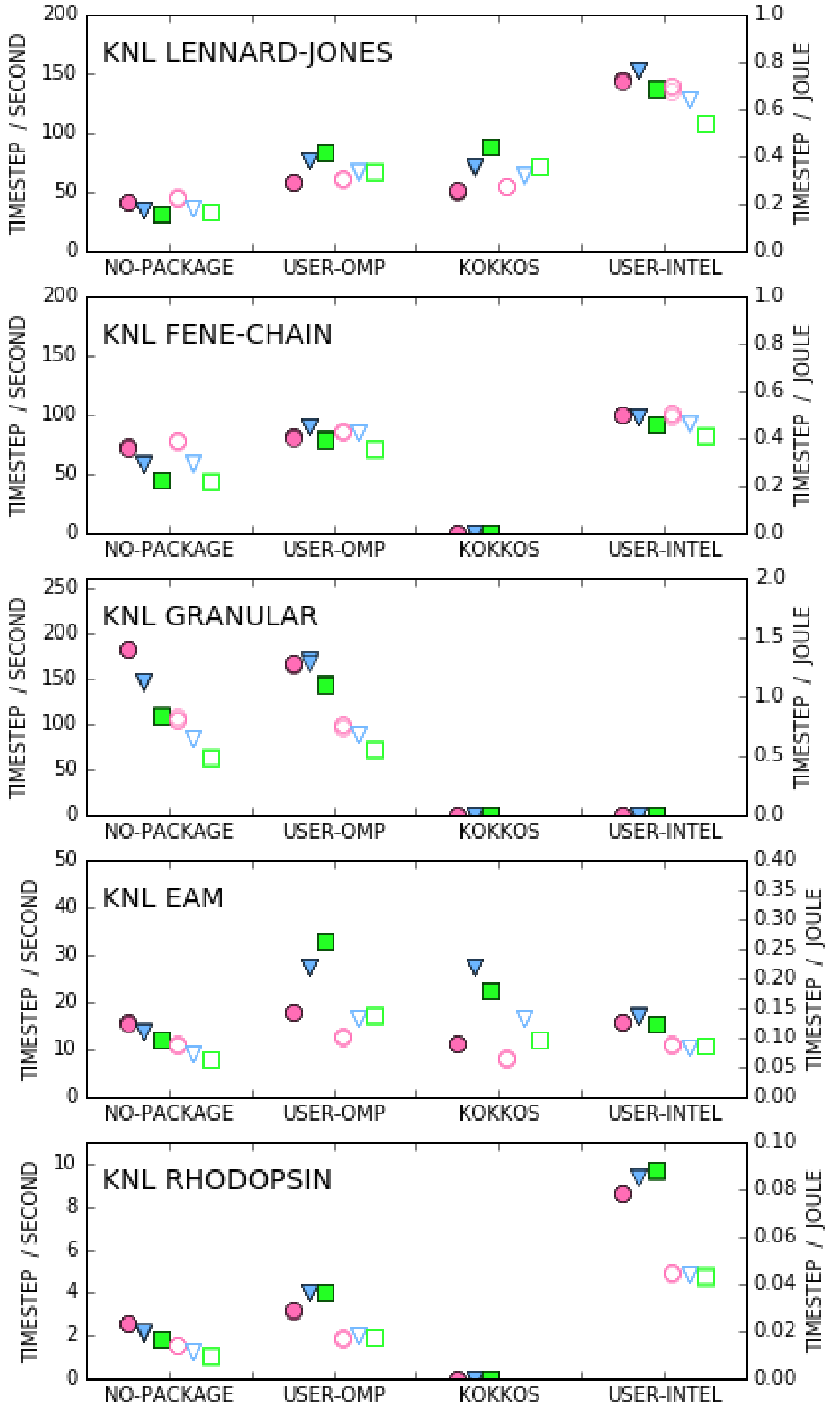

Figure 2: Performance figures for benchmarks run on ARCHER-KNL for four different implementations. For each benchmark the filled symbols are time step/second (left-hand scale), and the open symbols are time step/Joule (right-hand scale). All runs are 64 MPI tasks on 64 cores; circles, triangles, and squares are 1, 2, and 4 hardware threads per core, respectively. Measurements of zero indicate there is no implementation for the relevant potential. For each measurement, four repeats are present (the symbols overlay each other in most cases indicating any statistical error is no larger than the symbols). All benchmarks are for 512,000 atoms run for 10,000 steps.

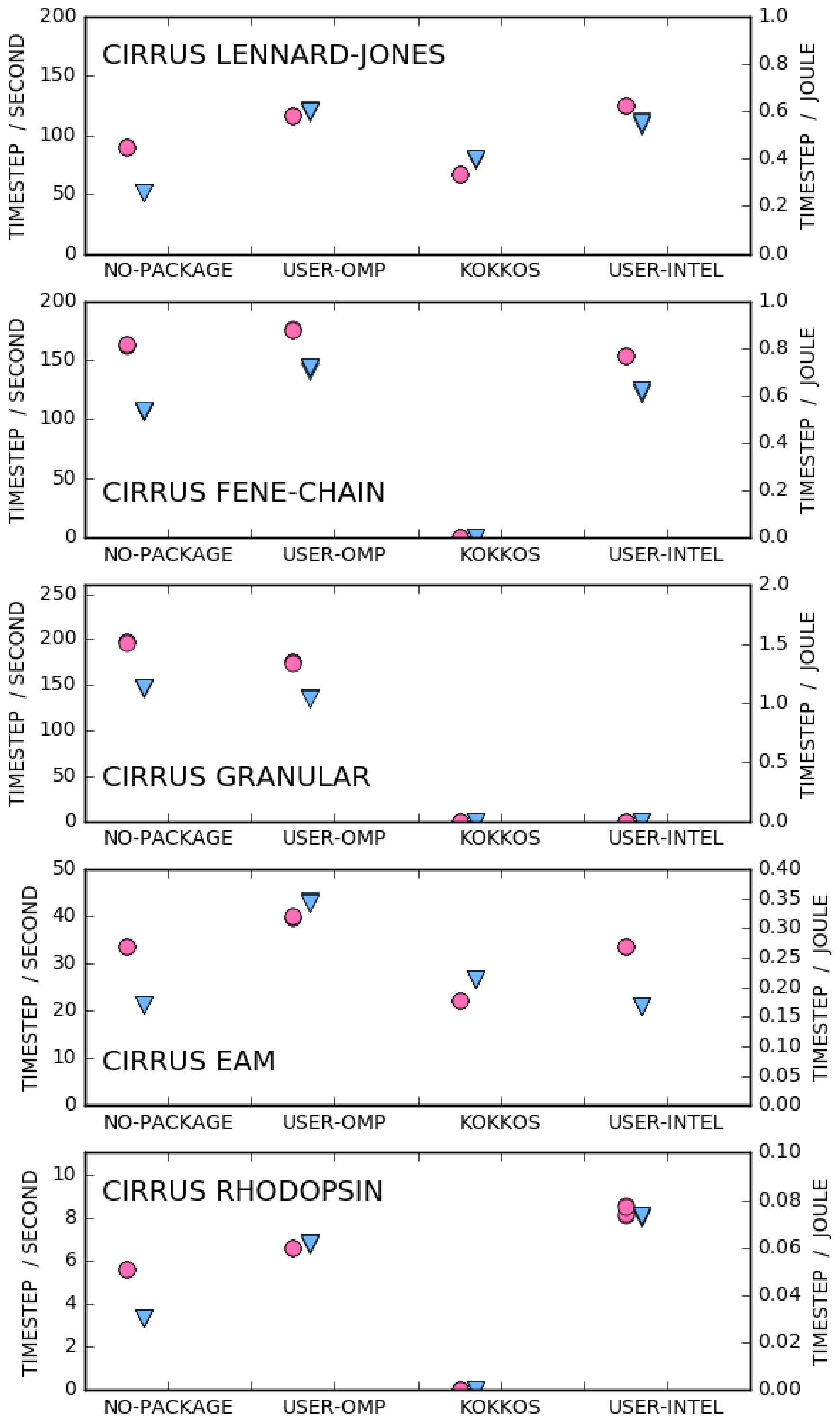

Figure 3: Performance figures for benchmarks run on CIRRUS for four different implementations. For each benchmark the filled symbols are time step/second (left-hand scale), and the open symbols are time step/Joule (right-hand scale). All runs are 36 MPI tasks on 36 cores; circles and triangles are 1 and 2 hardware threads per core, respectively. Measurements of zero indicate there is no implementation for the relevant potential. For each measurement, four repeats are present (the symbols overlay each other in most cases indicating any statistical error is no larger than the symbols). All benchmarks are for 512,000 atoms run for 10,000 steps.

Benchmark	**ARCHER**	**ARCHER-KNL**	**CIRRUS**
LJ	16.5 (U-Intel, 1)	4.7 (U-Intel, 2)	16.7 (U-Intel, 1)
FENE-CHAIN	3.6 (U-OMP, 1)	1.1 (U-Intel, 1)	4.6 (U-OMP, 1)
GRANULAR	2.6 (NO-PAC, 1)	0.8 (NO-PAC, 1)	3.7 (NO-PAC, 1)
EAM	6.2 (U-OMP, 2)	1.6 (U-OMP, 4)	9.3 (U-OMP, 2)
RHODOPSIN	N/A (U-Intel, 2)	N/A (U-Intel, 4)	N/A (U-Intel, 1)

Table 1: Performance as percentage of peak performance (Flop/second) for the implementation with the best time (time steps per second). The best implementation and corresponding number of hardware threads are shown in brackets in each case. Variation in times are sufficiently small that figures are probably good to 0.1 percentage point in all cases. Figures for the number of floating point operations were obtained via Craypat on ARCHER, and it is assumed that the same number of operations is performed on the other architectures. Instrumentation failed for the Rhodopsin benchmark.

threads).

The situation for ARCHER-KNL is more varied. The Lennard-Jones and Rhopodsin benchmarks (which are part of the Intel-specific benchmark suite provided in LAMMPS) show markedly improved performance for the USER-INTEL implementation, where significant attention has been paid to vectorisation. The optimal number of hardware threads depends on the problem.

The CIRRUS results (Figure 3) are again more "flat," in that no particular implementation is favoured, but there is generally improved performance compared with ARCHER, and comparable performance to ARCHER-KNL in most benchmarks.

The performance in terms of percentage of peak performance is presented in Table 1. Both ARCHER and CIRRUS show significantly higher figures than ARCHER-KNL, which we attribute to the inability of any of the benchmarks to take full advantage of vectorisation on the KNL. In contrast, the energy consumption for each benchmark is notably lower than ARCHER (Table 2).

A summary comparison of the time taken for each benchmark is presented in Figure 4. As expected, the more recent hardware shows significant improvement over the

Benchmark	ARCHER	ARCHER-KNL	CIRRUS
LJ	28.1 ± 0.5 (U-Intel, 1)	14.4 ± 0.2 (U-Intel, 1)	N/A
FENE-CHAIN	25.9 ± 1.6 (U-OMP, 1)	19.9 ± 0.3 (U-Intel, 1)	N/A
GRANULAR	25.0 ± 0.2 (No-PACK, 1)	12.3 ± 0.3 (No-PACK, 1)	N/A
EAM	109.6 ± 2.0 (U-OMP, 1))	75.9 ± 0.2 (U-OMP, 4)	N/A
RHODOPSIN	N/A	223.9 ± 2.6 (U-Intel, 1)	N/A

Table 2: Energy use (10^3 Joules) for the implementation with lowest energy use. Energy is reported by Craypat for Archer and Cray RUR for ARCHER-KNL; no energy figures are available for CIRRUS at present. The identity of the best-performing implementation is shown in brackets, but there is little difference between energy use on different numbers of hardware threads.

older ARCHER Ivy Bridge. However, there is no clear favourite for LAMMPS between ARCHER-KNL and CIRRUS.

4. Conclusions

It is possible to draw a number of broad conclusions from the figures obtained.

1. Where available, energy figures suggest time is a reasonable proxy for energy use.

2. The exact disposition of LAMMPS implementation and hardware resources to achieve best performance is strongly dependent on the problem.

3. The percentage of peak performance reached on KNL looks low compared with CIRRUS; we attribute this to the dominance of vectorisation in the KNL peak floating point performance.

We will attempt to add energy figures for the Broadwell platform (CIRRUS) in the near future.

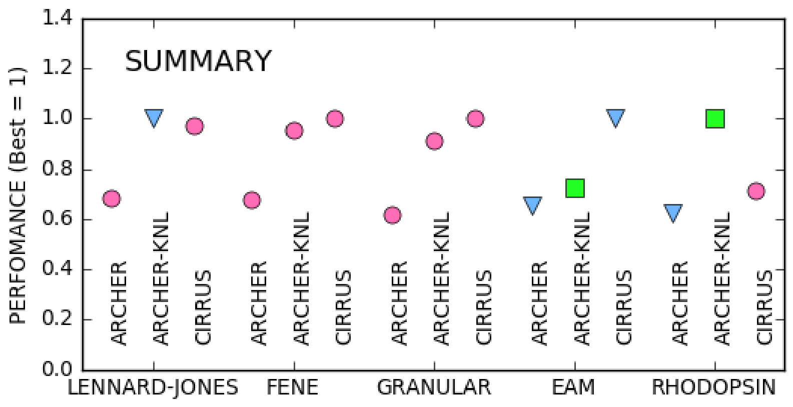

Figure 4: Summary of performance times (based on the recorded loop time) with values normalised so that the best time is unity for each different benchmark. The circles, triangles, and squares represent 1, 2, and four hardware threads per core, respectively (cf Figures 1-3). Statistical errors are no larger than the symbols.

References

[1] S.J. Pennycook, J.D. Sewall, and V.W. Lee, A Metric for Performance Portability, `https://arxiv.org/pdf/1611.07409.pdf` (accessed January 2017).

[2] For details of LAMMPS benchmarks see, e.g., `http://lammps.sandia.gov/bench.html` (accessed March 2017).

[3] J. Jeffers, J. Reinders, and A. Sodani, *Intel Xeon Phi Processor High Performance Programming (Knights Landing Edition)*, Morgan Kaufman Publishers, 2016.

[4] Only one KNL card per node is used as it is not practical to support sufficient memory bandwidth via QPI to make two KNLs per node worthwhile.

[5] The mode of KNL operation refers to the physical path of requests for memory in the event of a cache miss. In quadrant mode, the source of the cache miss and the 'tag directory' which contains the look-up for the physical address of the relevant memory are in different quadrants, but the relevant memory controller is in the same

quadrant as the source of the miss. This is a boot-time configuration, and cannot be changed by the user on the current service.

[6] For a description of KOKKOS as applied on LAMMPS, see, for example, `http://lammps.sandia.gov/doc/accelerate_kokkos.html` (accessed March 2017).

A Configuration Details and Data

A1. Platforms

ARCHER environment:

```
  1) modules/3.2.10.2
  2) eswrap/1.3.3-1.020200.1278.0
  3) switch/1.0-1.0502.57058.1.58.ari
  4) craype-network-aries
  5) craype/2.4.2
  6) pbs/12.2.401.141761
  7) craype-ivybridge
  8) cray-mpich/7.2.6
  9) packages-archer
 10) bolt/0.6
 11) nano/2.2.6
 12) leave_time/1.0.0
 13) quickstart/1.0
 14) ack/2.14
 15) xalt/0.6.0
 16) epcc-tools/6.0
 17) intel/16.0.2.181
 18) cray-libsci/13.2.0
 19) udreg/2.3.2-1.0502.9889.2.20.ari
 20) ugni/6.0-1.0502.10245.9.9.ari
 21) pmi/5.0.7-1.0000.10678.155.25.ari
 22) dmapp/7.0.1-1.0502.10246.8.47.ari
 23) gni-headers/4.0-1.0502.10317.9.2.ari
 24) xpmem/0.1-2.0502.57015.1.15.ari
 25) dvs/2.5_0.9.0-1.0502.1958.2.55.ari
 26) alps/5.2.3-2.0502.9295.14.14.ari
 27) rca/1.0.0-2.0502.57212.2.56.ari
 28) atp/1.8.3
 29) PrgEnv-intel/5.2.56
```

ARCHER-KNL environment:

```
  1) modules/3.2.10.5                   14) lustre-utils/2.3.4-6.74
  2) alps/6.1.6-20.1                    15) Base-opts/2.1.3-2.16
  3) nodestat/2.2-2.40                  16) intel/17.0.0.098
  4) sdb/2.2.1-3.119                    17) craype-mic-knl
  5) udreg/2.3.2-4.6                    18) craype-network-aries
  6) ugni/6.0.12-2.1                    19) craype/2.5.7
  7) gni-headers/5.0.7-3.1              20) cray-mpich/7.4.4
  8) dmapp/7.1.0-12.37                  21) pbs/default
  9) xpmem/0.1-4.5                      22) cray-libsci/16.09.1
 10) llm/20.2.4-3.18                    23) pmi/5.0.10-1.0000.11050.0.0.ari
 11) nodehealth/5.2.0-5.46              24) atp/2.0.3
 12) system-config/2.2.18-3.38          25) PrgEnv-intel/6.0.3
 13) sysadm/2.2.2-3.39
```

BROADWELL environment:

```
  1) mpt/2.14                         4) intel-compilers-17/17.0.2.174
  2) intel-cc-17/17.0.2.174           5) intel-tbb-17/17.0.2.174
  3) intel-fc-17/17.0.2.174
```

A2. Compilation stage

LAMMPS package options are reported from the `./src` directory:

```
$ make package-status | grep YES
Installed YES: package GRANULAR
Installed YES: package KOKKOS
Installed YES: package KSPACE
Installed YES: package MANYBODY
Installed YES: package MOLECULE
Installed YES: package RIGID
Installed YES: package USER-INTEL
Installed YES: package USER-OMP
```

Note that ARCHER with Intel 16 does not support KOKKOS compilation, so this package was switched out.

ARCHER user-defined options:

```
CC =            CC
OPTFLAGS =      -O2 -fp-model fast=2 -no-prec-div -qoverride-limits
CCFLAGS =       -g -qopenmp -restrict -std=c++11 ${OPTFLAGS}
SHFLAGS =       -fPIC
DEPFLAGS =      -M

LINK =          CC
LINKFLAGS =     ${CCFLAGS}
LIB =
SIZE =          size

ARCHIVE =       ar
ARFLAGS =       -rc
SHLIBFLAGS =    -shared
KOKKOS_DEVICES = Serial,OpenMP
```

ARCHER-KNL user-defined options:

```
CC =            CC
OPTFLAGS =      -xMIC-AVX512 -O2 -fp-model fast=2 -no-prec-div -qoverride-limits
CCFLAGS =       -g -fPIC -qopenmp -DLAMMPS_MEMALIGN=64 -qno-offload \
                -fno-alias -ansi-alias -restrict $(OPTFLAGS)
SHFLAGS =       -fPIC
DEPFLAGS =      -M

LINK =          CC
LINKFLAGS =     -g -qopenmp ${OPTFLAGS} -dynamic
LIB =           -ltbbmalloc
SIZE =          size

ARCHIVE =       ar
ARFLAGS =       -rc
SHLIBFLAGS =    -shared
KOKKOS_DEVICES = Serial,OpenMP
KOKKOS_ARCH =    KNL
```

BROADWELL user-defined options:

```
CC =            mpicc -cc=icpc
OPTFLAGS =      -xHost -O2 -fp-model fast=2 -no-prec-div -qoverride-limits
CCFLAGS =       -g -qopenmp -restrict ${OPTFLAGS}
SHFLAGS =       -fPIC
DEPFLAGS =      -M

LINK =          mpicc -cc=icpc
LINKFLAGS =     -g -qopenmp ${OPTFLAGS}
LIB =           -ltbbmalloc
SIZE =          size

ARCHIVE =       ar
ARFLAGS =       -rc
SHLIBFLAGS =    -shared
KOKKOS_DEVICES = Serial,OpenMP
```

A3. Batch scripts and run time

ARCHER:

```
#PBS -l select=1

export OMP_NUM_THREADS=1
aprun -n 24 -N 24 -d 1 -ss -cc numa_node -j 1 ./lmp_exe
# AND
export OMP_NUM_THREADS=2
aprun -n 24 -N 24 -d 2 -ss -cc numa_node -j 2 ./lmp_exe
```

ARCHER-KNL:

```
#PBS -l select=1:aoe=quad_100

export OMP_NUM_THREADS=1
aprun -n 64 -N 64 -d 1 -cc depth -j 1 ./lmp_exe
# AND
export OMP_NUM_THREADS=2
aprun -n 64 -N 64 -d 2 -cc depth -j 2 ./lmp_exe
# AND
export OMP_NUM_THREADS=4
aprun -n 64 -N 64 -d 4 -cc depth -j 4 ./lmp_exe
```

BROADWELL:

```
#PBS -l select=72
#PBS -l place=excl

export OMP_NUM_THREADS=1
mpiexec_mpt -n 36 -ppn 36 omplace -nt 1 ./lmp_exe
# AND
export OMP_NUM_THREADS=2
mpiexec_mpt -n 36 -ppn 36 omplace -nt 2 ./lmp_exe
```

Benchmark (HWT)	ARCHER	ARCHER-KNL	CIRRUS
LJ NO-PACKAGE (1)	176.283, 175.356, 176.353, 175.678	238.015, 240.070, 238.820, 241.483	110.951, 110.848, 110.617, 110.748
LJ NO-PACKAGE (2)	205.977, NaN, 207.254, 205.987	286.369, 288.598, 287.384, 290.796	193.407, 194.306, 193.906, 193.208
LJ NO-PACKAGE (4)	N/A	315.391, 313.224, 315.316, 313.824	N/A
LJ USER-OMP (1)	135.472, 135.380, 135.456, 135.419	170.623, 171.508, 170.819, 172.436	85.865, 86.0513, 85.3087, 85.325
LJ USER-OMP (2)	124.001, NaN, 124.071, 124.065	130.330, 132.002, 131.813, 129.780	83.5803, 81.9257, 83.5712, 82.8491
LJ USER-OMP (4)	N/A	119.379, 119.172, 119.264, 119.743	N/A
LJ KOKKOS (1)	NOT COMPILED	198.752, 196.837, 197.530, 197.664	148.493, 148.729, 148.606, 149.009
LJ KOKKOS (2)		140.161, 142.728, 139.266, 140.482	125.811, 125.533, 123.623, 125.085
LJ KOKKOS (4)		113.557, 112.668, 113.351, 112.788	N/A
LJ USER-INTEL (1)	95.4567, 95.902, 95.0216, 94.3823	68.6706, 69.0314, 68.5154, 69.2849	80.0882, 80.0350, 79.9397, 79.9978
LJ USER-INTEL (2)	96.3187, NaN, 96.2951, 97.7296	65.0921, 65.131, 65.3531, 65.0759	91.4354, 89.1846, 92.3390, 90.7021
LJ USER-INTEL (4)	N/A	72.6577, 72.1196, 72.4640, 72.9232	N/A
FENE NO-PACK (1)	91.7437, 91.9186, 92.1735, 91.6933	137.653, 138.422, 137.274, 138.344	61.5000, 61.4292, 61.1718, 61.3763
FENE NO-PACK (2)	137.514, 140.128, 138.028, 137.042	169.811, 170.226, 172.904, 170.635	93.7986, 93.5287, 93.5070, 93.3188
FENE NO-PACK (4)	N/A	227.053, 226.493, 226.543, 226.733	N/A
FENE USER-OMP (1)	83.0300, 83.0142, 83.3402, 83.2879	123.208, 123.489, 122.831, 124.184	57.0749, 57.0326, 56.7441, 57.0016
FENE USER-OMP (2)	90.3980, 90.0392, 89.0017, 88.1937	110.971, 111.758, 111.874, 112.11	69.3394, 71.4304, 70.1934, 69.1788
FENE USER-OMP (4)	N/A	126.396, 126.336, 126.608, 127.159	N/A
FENE USER-INTEL (1)	91.6503, 91.4008, 91.6964, 91.5761	99.6068, 99.971, 99.5674, 100.414	65.3072, 65.0698, 65.0137, 65.1251
FENE USER-INTEL (2)	99.5556, 100.8140, 98.1205, 98.4971	101.144, 101.493, 101.17, 101.14	81.9251, 80.3174, 81.6068, 80.5603
FENE USER-INTEL (4)	N/A	109.553, 109.477, 109.586, 109.477	N/A
GRANULAR NO-PACK (1)	83.2607, 81.3672, 80.6743, 81.0435	54.8714, 55.0367, 55.0648, 54.99	50.7845, 50.7998, 50.540, 51.1558
GRANULAR NO-PACK (2)	91.9147, 90.7439, 90.5644, 90.5010	67.7399, 68.479, 67.6377, 67.6539	68.4433, 68.2602, 68.026, 68.2341
GRANULAR NO-PACK (4)	N/A	91.9276, 92.8289, 91.4065, 91.8094	N/A
GRANULAR USER-OMP (1)	90.5019, 90.6905, 88.6724, 89.1077	59.9153, 60.2447, 59.7441, 60.1863	57.0850, 57.2138, 56.7818, 57.4721
GRANULAR USER-OMP (2)	104.122, 103.089, 102.753, 103.320	58.5679, 58.7339, 58.5362, 59.7437	73.5212, 73.0385, 73.5672, 73.9319
GRANULAR USER-OMP (4)	N/A	69.2570, 69.2958, 69.6890, 69.5417	N/A
EAM NO-PACK (1)	449.452, 448.653, 449.213, 449.157	637.75, 640.79, 637.827, 643.57	298.003, 297.809, 297.775, 297.581
EAM NO-PACK (2)	546.679, NaN, 546.868, 546.884	726.243, 719.063, 730.118, 729.478	470.862, 471.367, 471.641, 471.913
EAM NO-PACK (4)	N/A	828.215, 832.185, 832.688, 832.078	N/A
EAM USER-OMP (1)	403.275, 404.094, 403.382, 402.961	561.277, 561.500, 561.014, 564.576	252.166, 252.266, 250.500, 250.756
EAM USER-OMP (2)	338.926, NaN, 338.297, 336.626	362.656, 363.892, 364.369, 365.559	230.463, 234.651, 232.424, 233.326
EAM USER-OMP (4)	N/A	302.712, 303.064, 303.743, 304.581	N/A
EAM KOKKOS (1)	NOT COMPILED	890.758, 890.745, 887.885, 891.832	448.313, 448.811, 448.669, 449.255
EAM KOKKOS (2)		362.656, 363.892, 364.369, 365.559	372.656, 375.796, 375.519, 373.146
EAM KOKKOS (4)		441.553, 444.015, 445.617, 447.008	N/A
EAM USER-INTEL (1)	447.079, 447.39, 447.234, 447.69	639.982, 637.74, 638.092, 638.513	297.576, 297.825, 297.459, 297.546
EAM USER-INTEL (2)	446.584, 446.539, 447.119, 448.209	574.616, 580.832, 584.56, 580.859	476.968, 477.646, 478.000, 477.718
EAM USER-INTEL (4)	N/A	645.157, 647.668, 647.991, 650.329	N/A
RHODOPSIN NO-PACK (1)	2640.22, 2646.59, 2647.35, 2656.44	3858.56, 3857.12, 3867.62, 3869.20	1787.55, 1789.57, 1785.49, 1790.01
RHODOPSIN NO-PACK (2)	3387.22, 3387.88, 3409.17, 3391.20	4698.48, 4719.64, 4721.18, 4679.74	2988.94, 3003.11, 2998.05, 2998.78
RHODOPSIN NO-PACK (4)	N/A	5432.57, 5480.41, 5468.26, 5527.57	N/A
RHODOPSIN USER-OMP (1)	2278.39, 2284.58, 2286.04, 2286.76	3163.72, 3168.90, 3191.33, 3151.26	1524.35, 1519.64, 1518.37, 1517.32
RHODOPSIN USER-OMP (2)	2091.02, 2072.99, 2084.90, 2087.00	2483.61, 2487.39, 2499.26, 2510.14	1457.18, 1495.68, 1463.41, 1477.05
RHODOPSIN USER-OMP (4)	N/A	2505.46, 2503.91, 2500.01, 2492.18	N/A
RHODOPSIN USER-INTEL (1)	1860.57, 1909.67, 1909.15, 1913.26	1156.53, 1154.77, 1155.43, 1158.93	1228.49, 1227.61, 1168.41, 1172.75
RHODOPSIN USER-INTEL (2)	1639.70, 1637.84, 1642.28, 1682.03	1058.22, 1060.25, 1058.53, 1062.25	1250.76, 1234.83, 1239.58, 1238.19
RHODOPSIN USER-INTEL (4)	N/A	1030.11, 1029.35, 1033.84, 1027.89	N/A

Table 3: Loop time (all seconds) for 10000 steps reported by the LAMMPS executable; four repeats are quoted for each relevant number of hardware threads employed.

Benchmark (HWT)	ARCHER	ARCHER-KNL	CIRRUS
LJ NO-PACKAGE (1)	48328.0, 49695.0, 49172.0, 48335.0	43711.0, 45485.0, 44651.0, 44256.0	
LJ NO-PACKAGE (2)	NaN	54648.0, 55028.0, 54799.0, 55232.0	
LJ NO-PACKAGE (4)	N/A	60816.0, 61298.0, 61147.0, 59432.0	
LJ USER-OMP (1)	37305.0, 38553.0, 38144.0, 37931.0	32241.0, 33409.0, 32936.0, 32572.0	
LJ USER-OMP (2)	NaN	29630.0, 29990.0, 29790.0, 29359.0	
LJ USER-OMP (4)	N/A	29936.0, 30108.0, 30213.0, 29502.0	
LJ KOKKOS (1)	NOT COMPILED	36294.0, 37095.0, 36807.0, 36167.0	
LJ KOKKOS (2)		30890.0, 31394.0, 30581.0, 30785.0	
LJ KOKKOS (4)		27705.0, 27869.0, 27986.0, 27973.0	
LJ USER-INTEL (1)	27644.0, 28800.0, 28108.0, 27762.0	14197.0, 14734.0, 14502.0, 14315.0	
LJ USER-INTEL (2)		15577.0, 15649.0, 15576.0, 15518.0	
LJ USER-INTEL (4)	N/A	18339.0, 18468.0, 18528.0, 18391.0	
FENE NO-PACK (1)	26548.0, 27684.0, 27514.0, 28646.0	25391.0, 26334.0, 25853.0, 25411.0	
FENE NO-PACK (2)	NaN, 44347.0, 41219.0, 41837.0	33709.0, 33802.0, 34187.0, 33767.0	
FENE NO-PACK (4)	N/A	46529.0, 46924.0, 46397.0, 45393.0	
FENE USER-OMP (1)	26548.0, 27684.0, 24122.0, 25132.0	22912.0, 23735.0, 23381.0, 23143.0	
FENE USER-OMP (2)	NaN, 29742.0, 28101.0, NaN	23602.0, 23749.0, 23719.0, 23746.0	
FENE USER-OMP (4)	N/A	28137.0, 28463.0, 28270.0, 27771.0	
FENE USER-INTEL (1)	26794.0, 27784.0, 27484.0, 27657.0	19579.0, 20292.0, 20049.0, 19759.0	
FENE USER-INTEL (2)	30911.0, 31209.0, 30491.0, 30922.0	21391.0, 21680.0, 21534.0, 21526.0	
FENE USER-INTEL (4)	N/A	24312.0, 24514.0, 24297.0, 23948.0	
GRANULAR NO-PACK (1)	24885.0, 25276.0, 24839.0, NaN	12117.0, 12547.0, 12417.0, 12182.0	
GRANULAR NO-PACK (2)	30307.0, 30728.0, 29387.0, NaN	15424.0, 15597.0, 15398.0, 15422.0	
GRANULAR NO-PACK (4)	N/A	20752.0, 20951.0, 20653.0, 20326.0	
GRANULAR USER-OMP (1)	26717.0, 27574.0, 26743.0, 26576.0	13037.0, 13521.0, 13296.0, 13121.0	
GRANULAR USER-OMP (2)	32843.0, 33177.0, 31969.0, 32900.0	14663.0, 14733.0, 14585.0, 14826.0	
GRANULAR USER-OMP (4)	N/A	17999.0, 18202.0, 18346.0, 17885.0	
EAM NO-PACK (1)	119499.0, 123892.0, 122717.0, 120149.0	111919.0, 116133.0, 114393.0, 113207.0	
EAM NO-PACK (2)	NaN	136520.0, 135315.0, 136914.0, 136706.0	
EAM NO-PACK (4)	N/A	158741.0, 160742.0, 160046.0, 156353.0	
EAM USER-OMP (1)	107590.0, 111889.0, 110716.0, 108232.0	98270.0, 101547.0, 100615.0, 98790.0	
EAM USER-OMP (2)	NaN	75665.0, 75953.0, 75939.0, 76141.0	
EAM USER-OMP (4)	N/A	72446.0, 73303.0, 73415.0, 71655.0	
EAM KOKKOS (1)	NOT COMPILED	154634.0, 159808.0, 157919.0, 155534.0	
EAM KOKKOS (2)		75665.0, 75953.0, 75939.0, 76141.0	
EAM KOKKOS (4)		102896.0, 104510.0, 104574.0, 103404.0	
EAM USER-INTEL (1)	118853.0, 123097.0, 122145.0, 119505.0	112321.0, 115566.0, 114313.0, 112192.0	
EAM USER-INTEL (2)	NaN	118980.0, 120370.0, 120794.0, 120047.0	
EAM USER-INTEL (4)	N/A	115424.0, 117438.0, 115936.0, 115247.0	
RHODOPSIN NO-PACK (1)	NaN	704583.0, 726291.0, 720281.0, 706816.0	
RHODOPSIN NO-PACK (2)	NaN	901530.0, 905965.0, 902622.0, 898051.0	
RHODOPSIN NO-PACK (4)	N/A	1050652.0, 1073758.0, 1060657.0, 1051333.0	
RHODOPSIN USER-OMP (1)	NaN	588449.0, 606358.0, 601745.0, 584760.0	
RHODOPSIN USER-OMP (2)	NaN	545571.0, 547128.0, 546088.0, 546976.0	
RHODOPSIN USER-OMP (4)	N/A	588680.0, 592412.0, 592071.0, 575649.0	
RHODOPSIN USER-INTEL (1)	NaN	221398.0, 227380.0, 224115.0, 222665.0	
RHODOPSIN USER-INTEL (2)	NaN	225953.0, 227774.0, 225349.0, 226312.0	
RHODOPSIN USER-INTEL (4)	N/A	230804.0, 233045.0, 233879.0, 226106.0	

Table 4: Energy use (in Joules) reported by Craypat on ARCHER (failures in instrumentation indicated by “NaN”) and RUR on ARCHER-KNL for four repeats. No energy figures are available for CIRRUS at present.