DESRES/TR--2014-01

Desmond/GPU Performance as of November 2014

Michael Bergdorf,[1] Eric T. Kim,[1] Charles A. Rendleman, [1] and David E. Shaw[1,2,*]

Desmond/GPU Software Version 1.2.2, November 12, 2014

Summary

Desmond/GPU is a code, written in CUDA C++, that is designed for the execution of molecular dynamics (MD) simulations of biological systems on NVIDIA Graphics Processing Units (GPUs). This paper reports the performance that Desmond/GPU, running on a range of different GPUs and GPU cluster configurations, achieves on three biological system benchmarks as of November 2014. Our benchmark results show that on a single GPU, Desmond can deliver the same simulation throughput that it delivers on a dozen CPUs.

Benchmark Chemical Systems and Simulation Parameters

Performance results were measured for two chemical systems that are commonly used for MD code benchmarking, and one additional system that is characteristic for free energy perturbation (FEP) simulations.

[1] D. E. Shaw Research, New York, NY 10036, USA.
[2] Department of Biochemistry and Molecular Biophysics, Columbia University, New York, NY 10032, USA.
[*] To whom correspondence should be addressed: David.Shaw@DEShawResearch.com.

The characteristics of the benchmark systems are summarized in Table 1. The parameters were chosen to optimize Desmond/GPU performance without compromising accuracy [1]. DHFR [2] and ApoA1 [3] were run with a non-bonded interaction cutoff of 9 Å in the NVE ensemble, that is, without temperature or pressure control. The P38S and P38C cases [4] were run with a non-bonded interaction cutoff of 9 Å in the NVT ensemble using a Nosé-Hoover thermostat.

Free energy perturbation/replica exchange with solute tempering (FEP/REST) [5] combines enhanced sampling through replica exchange with FEP to accelerate structural reorganization and thus convergence of relative protein-ligand binding affinities. Our two FEP/REST benchmarks, P38C and P38S, consist of 12 replicas (or "windows") each. Every 1.2 simulated picoseconds, replicas are exchanged and energy differences (dE) are computed and written out.

Both the p38 complex (P38C) and the p38 inhibitor (P38S) system were run on different numbers of GPUs.

Table 1. Production parameters. The PME frequency is the interval at which the "far" electrostatics forces are evaluated.

Label	Name	Time step	PME frequency	PME grid size	# of atoms	System size (Å^3)
DHFR	Dihydrofolate reductase	2.5 fs	2 steps	64 × 64 × 64	23,558	62 × 62 × 62
ApoA1	Apolipoprotein A1	2.5 fs	2 steps	128 × 128 × 128	92,224	109 × 109 × 78
P38S	p38 inhibitor	2.0 fs	3 steps	32 × 32 × 32	2,853	29 × 29 × 36
P38C	p38 MAP kinase inhibitor complex	2.0 fs	3 steps	64 × 64 × 64	25,550	80 × 61 × 56

Desmond/GPU has two communication-layer implementations. The first, the *MT collective*, is the default choice for single-GPU simulations. The MT collective also allows parallelization of a system across multiple GPUs through NVIDIA GPUDirect™ 2.0. The second, the *MPI collective*, is used for simulations with multiple interacting replicas (e.g., replica-exchange molecular dynamics, REMD). The DHFR and ApoA1 benchmarks were run with the MT collective, while both P38S and P38C were run using the MPI collective.

Hardware and Operating Environment

We ran the benchmarks on GPUs representing three different architectures: "Fermi" GF110, "Kepler" GK104, and "Kepler II" GK110/GK110B. For each GPU architecture, we sampled both consumer-grade "GeForce" GPUs and "Tesla" high-performance computing parts.

The DHFR and ApoA1 benchmarks were run on a variety of host systems: single-processor Dell workstations T3500 (GeForce GTX 480) and T3600 (TITAN and Tesla K20c); a Colfax CXT8000 8-GPU server (GeForce GTX 580); a Super Micro 4027GR 8-GPU server (GeForce GTX 780 Ti); a dual-socket Cirrascale BladeRack-XL 5GVU 8-GPU server with PCIe-Gen2 (GeForce GTX680 and GTX 780); and a range of servers of the NVIDIA GPU Test Drive PSG cluster (Tesla models M2900, K10, K20m, and K20X).

The FEP/REMD benchmarks were run on: a dual-socket Colfax CXT8000 8 × GTX 580 server; a dual-socket Cirrascale BladeRack-XL 5GVU 8-GPU server with PCIe-Gen2 (GeForce GTX 680 and GTX 780); 2 × Tesla M2090, 2 × Tesla K10, 2 × Tesla K20m, and 4 × Tesla K20X nodes of the NVIDIA GPU Test Drive PSG cluster.

CPU architectures ranged from Westmere to Ivy Bridge and affected simulation performance very weakly, since in Desmond/GPU the CPU is only used to dispatch work to the GPUs. All host systems ran under either CentOS 5 or 6. Desmond/GPU was compiled using CUDA 5.0.35

and GCC 4.5.3. The Tesla simulations used NVIDIA driver version 319.32; the GeForce simulations were done with driver version 319.76, except for the GTX 680 simulation, which used driver version 319.60. Note that performance may vary depending on the driver version. In general, we chose the oldest driver that had all necessary bug fixes and supported our CUDA version. On the NVIDIA GPU Test Drive cluster we used the drivers that came installed on the system. Note also that in our experience GeForce cards require an extensive testing, selection, and burn-in period before a stable and reliable set is found suitable for use in a production setting.

Results

In this section, Desmond/GPU simulation rates are reported in units of simulated nanoseconds per wall-clock day.

In order to provide a comparison point for CPUs versus GPUs, we also ran the DHFR and ApoA1 benchmarks using Desmond 3.6 on recent CPU compute nodes. The single-socket nodes consist of 4-core Intel Xeon E3-1270 V2 Ivy Bridge CPUs running at 3.50 GHz and are connected using Mellanox FDR InfiniBand OFED 3.5. The performance numbers for DHFR and ApoA1 are reported in Table 2 and Figure 1, left panel, in units of simulated nanoseconds per wall-clock day for different numbers of MPI processes, with and without hyper-threading. Each MPI process was bound to an individual core. The ApoA1 rate on a single CPU core with hyper-threading enabled is 0.7 ns/day. If we compare this with the rate of 32.6 ns/day on a GeForce GTX 780 GPU, we note that the compute throughput of a single GPU is comparable to that of a dozen 4-core CPUs.

Table 2. Desmond 3.6 DHFR/ApoA1 benchmarks. Since hyper-threading was enabled on the nodes, we ran both with one thread and two threads per process. Running with more than two threads and fewer processes did not show any improvement in performance.

Threads per process (TPP)	# of nodes	# of processes	DHFR rate	ApoA1 rate
1	1	1	4.5	0.6
	1	2	8.5	1.1
	1	4	15.1	1.9
	2	8	27.9	3.6
	4	16	49.4	7.5
	8	32	90.5	14.0
2	1	1	5.3	0.7
	1	2	10.0	1.3
	1	4	17.5	2.3
	2	8	33.2	4.6
	4	16	58.4	8.9
	8	32	101.9	16.6

Table 3 and Figure 1, right panel, report the performance of Desmond/GPU in units of simulated nanoseconds per wall-clock day for DHFR and ApoA1 for different GPU/host configurations. Table 4 and Figure 2 report the performance for the two representative FEP/REST systems described above.

Table 3. DHFR/ApoA1 benchmarks. Both systems were run on 1, 2, and 4 GPUs where available. All GPUs were required to reside under the same PCI root complex in order to enable GPU Direct 2.0 peer-to-peer communication. Simulation rates are given in simulated nanoseconds per wall-clock day.

GPU	DHFR rate			ApoA1 rate		
	1 GPU	2 GPUs	4 GPUs	1 GPU	2 GPUs	4 GPUs
GeForce GTX 780 Ti	147.4	198.0		36.2	57.9	
GeForce GTX TITAN	148.4			35.5		
GeForce GTX 780	131.4	181.2	224.4	32.6	51.9	70.1
Tesla K20Xm	109.8	153.5		27.3	43.6	
Tesla K20Xm ECC	106.2	153.2		25.6	41.0	
Tesla K20m	97.6	139.9		23.9	38.5	
Tesla K20m ECC	94.1	135.2		22.1	35.5	
GeForce GTX 680	98.1	142.8	192.5	18.9	31.5	44.1
GeForce GTX 580	84.7	131.8	185.2	18.6	31.6	44.9
Tesla M2090	72.5	113.7		15.9	27.0	
Tesla K10	70.9	105.2	144.4	15.6	26.3	37.6

Figure 1. Desmond 3.6 and Desmond/GPU DHFR/ApoA1 benchmarks. Left: the performance of Desmond 3.6 running the ApoA1 (dark green) and the DHFR (light green) benchmark at different degrees of parallelism (see Table 2). The darkened extensions indicate improvement in performance that can be obtained by running the code with two threads per process (2 TPP). Right: Desmond/GPU performance for the same benchmarks for different GPU models in order of increasing ApoA1 performance (see Table 3 for details).

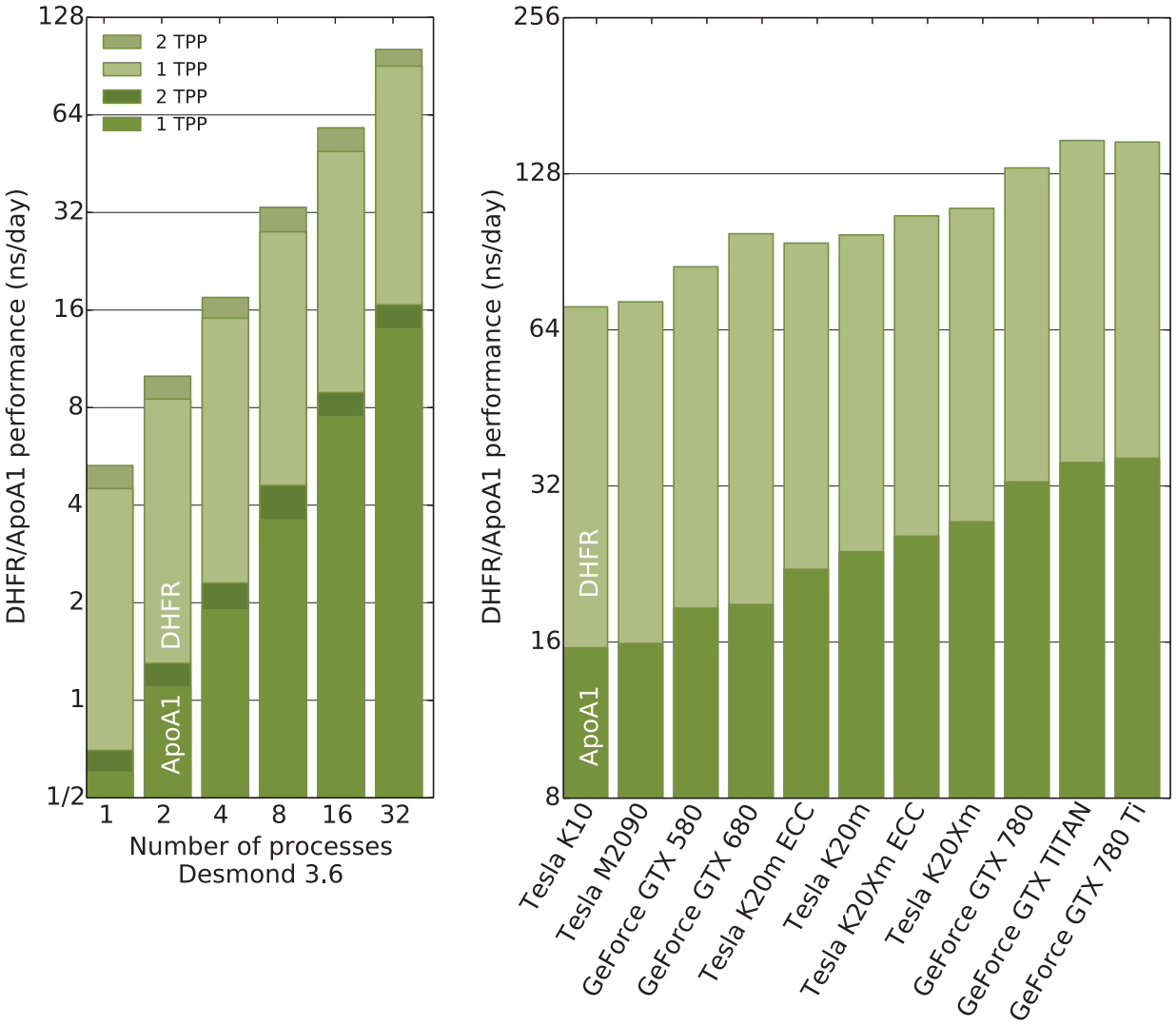
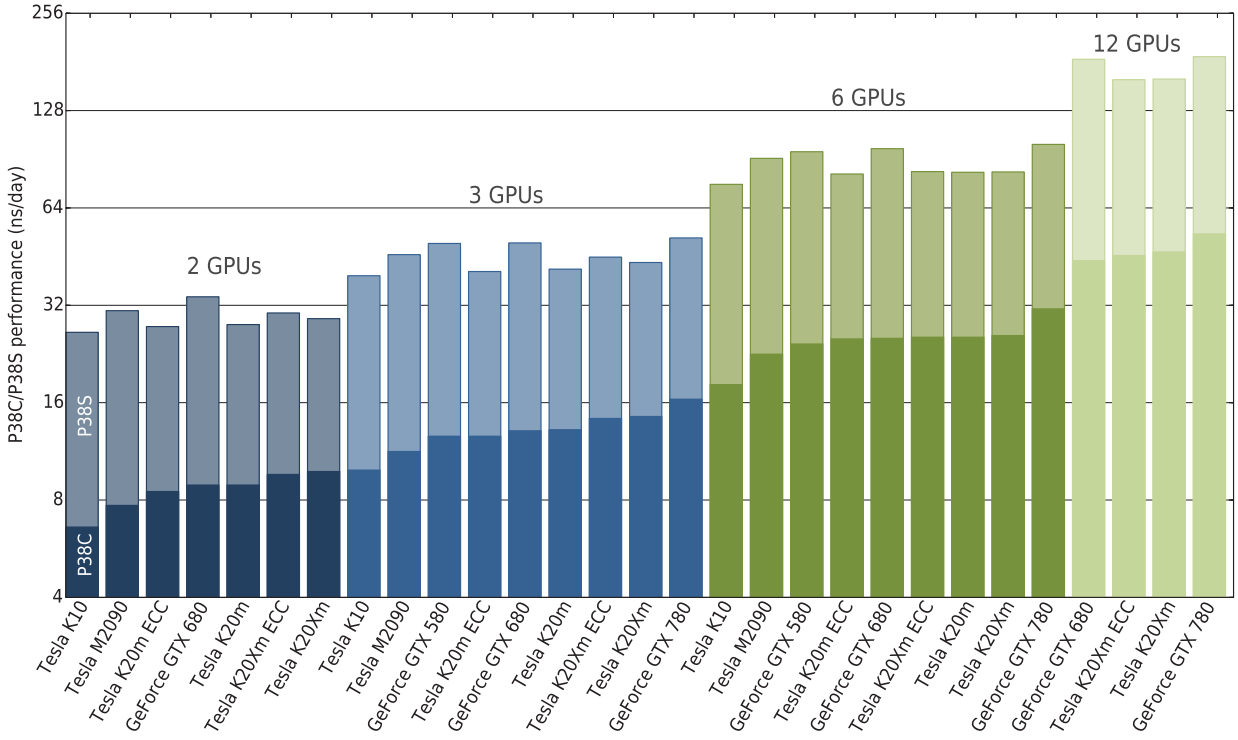


Table 4. FEP/REST benchmark systems. The "# of GPUs" column gives the number of GPUs used to simulate the 12 replicas. Simulation rates are given in simulated nanoseconds per wall-clock day.

GPU	# of nodes	# of GPUs	P38C rate	P38S rate
GeForce GTX 580	1	3	12.6	49.7
	1	6	24.3	95.5
GeForce GTX 680	1	2	8.9	34.0
	1	3	13.1	49.9
	1	6	25.3	97.6
	2	12	44.0	184.4
GeForce GTX 780	1	3	16.4	51.7
	1	6	31.2	100.6
	2	12	53.2	187.9
Tesla K10	1	2	6.6	26.4
	1	3	9.9	39.5
	2	6	18.2	75.7
Tesla K20Xm	1	2	9.8	29.1
	1	3	14.5	43.4
	2	6	25.8	82.7
	3	12	46.8	160.2
Tesla K20Xm ECC	1	2	9.6	30.3
	1	3	14.3	45.1
	2	6	25.5	82.9
	3	12	45.6	159.5
Tesla K20m	1	2	8.9	27.9
	2	3	13.2	41.4
	3	6	25.5	82.5
Tesla K20m ECC	1	2	8.5	27.5
	2	3	12.6	40.7
	3	6	25.2	81.5
Tesla M2090	1	2	7.7	30.8
	2	3	11.3	45.9
	3	6	22.6	91.1

Figure 2. FEP/REST benchmark systems. The dark bars give the performance for the complex, P38C, while the light bars show the performance for the smaller p38 inhibitor FEP/REST system, P38S (see Table 4 for details). The results are presented in order of increasing P38C performance.



Desmond/GPU Availability

Desmond/GPU is available without cost from D. E. Shaw Research[3] for non-commercial research use by non-profit institutions, and under commercial license from Schrödinger, LLC[4] for other purposes or parties.

The current release is based on CUDA 5.0 and supports NVIDIA GPUs with compute capabilities 2.0, 3.0, and 3.5.

[3] http://www.deshawresearch.com/resources_desmond.html
[4] http://www.schrodinger.com/Desmond

References

1. Edmond Chow, Charles A. Rendleman, Kevin J. Bowers, Ron O. Dror, Douglas H. Hughes, Justin Gullingsrud, Federico D. Sacerdoti, and David E. Shaw, "Desmond Performance on a Cluster of Multicore Processors," D. E. Shaw Research Technical Report DESRES/TR--2008-01, July 2008.

2. MD Benchmarks for Amber, CHARMM and NAMD. http://ambermd.org/amber8.bench2.html

3. James C. Phillips, Gengbin Zheng, Sameer Kumar, and Laxmikant V. Kalé, "NAMD: Biomolecular Simulation on Thousands of Processors," *Proceedings of the 2002 ACM/IEEE Conference on Supercomputing (SC02)*, Baltimore, Maryland, November 16–22, 2002.

4. David M. Goldstein, Michael Soth, Tobias Gabriel, Nolan Dewdney, Andreas Kuglstatter, Humberto Arzeno, Jeffrey Chen, William Bingenheimer, Stacie A. Dalrymple, James Dunn, Robert Farrell, Sandra Frauchiger, JoAnn La Fargue, Manjiri Ghate, Bradford Graves, Ronald J. Hill, Fujun Li, Renee Litman, Brad Loe, Joel McIntosh, Daniel McWeeney, Eva Papp, Jaehyeon Park, Harlan F. Reese, Richard T. Roberts, David Rotstein, Bong San Pablo, Keshab Sarma, Martin Stahl, Man-Ling Sung, Rebecca T. Suttman, Eric B. Sjogren, Yunchou Tan, Alejandra Trejo, Mary Welch, Paul Weller, Brian R. Wong, and Hasim Zecic, "Discovery of 6-(2,4-Difluorophenoxy)-2-[3-hydroxy-1-(2-hydroxyethyl)propylamino]-8-methyl-8H-pyrido[2,3-d]pyrimidin-7-one (Pamapimod) and 6-(2,4-Difluorophenoxy)-8-methyl-2-(tetrahydro-2H-pyran-4-ylamino)pyrido[2,3-d]pyrimidin-7(8H)-one (R1487) as Orally Bioavailable and Highly Selective Inhibitors of p38α Mitogen-Activated Protein Kinase," *Journal of Medicinal Chemistry*, vol. 54, no. 7, 2011, pp. 2255–2265.

5. Lingle Wang, B. J. Berne, and Richard A. Friesner, "On Achieving High accuracy and Reliability in the Calculation of Relative Protein–Ligand Binding Affinities," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 109, no. 6, 2012, pp. 1937–1942.

Acknowledgements

We thank Goran Pocina, Michael Fenn, Chris R. Harwell, Clayton Falzone, and Douglas H. Hughes for setting up our GPU infrastructure, Mark Berger (NVIDIA) and Adam DeConinck (NVIDIA) for generously providing us with the opportunity to run some of our benchmarks on the NVIDIA PSG cluster, and Mollie Kirk and Rebecca Bish-Cornelissen for their help with the preparation of this report.

Appendix

The configuration and structure files used in the simulations reported in this document are available on our website.

Running a simulation across multiple GPUs

For our parallel runs across multiple GPUs we used the following tuned command-line options:

```
numactl --cpubind=0 gdesmond-actual --collective MT --tpp N --cfg force.overlap_kernels=true \
  --cfg aspect_ratio=4 --cfg expected_np=20
```

The option `force.overlap_kernels=true` allows Desmond/GPU to overlap some of the force calculations.

Running a Desmond 3.6 simulation across multiple CPUs

For our parallel runs we used the following command line:

```
mpirun –n #NPROCS -bycore --bind-to-core  --report-bindings -- desmond --destrier mpi
```

In order to use two threads per process we used:

```
mpirun –n #NPROCS -bycore --bind-to-core  --report-bindings -- desmond --destrier mpi –tpp 2
```

Desmond/GPU memory requirements

The memory requirements of Desmond/GPU for intermediately sized systems can be estimated as

750 MB + 6 × number of particles in thousands,

thus, a system with 90,000 atoms will roughly require a GPU with at least 1,290 MB of RAM.