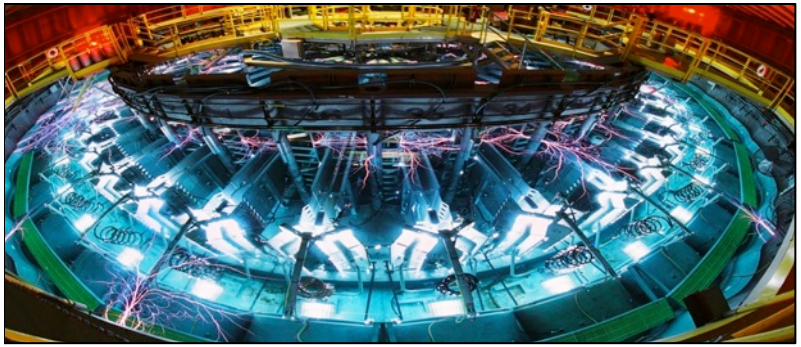
Exceptional service in the national interest

Sandia National Laboratories



KokkosP: Runtime Hooks for Portable Performance Analysis

S.D. Hammond, C.R. Trott, H.C. Edwards, N. Ellingwood

Center for Computing Research,

Scalable Computer Architectures

Sandia National Laboratories, NM

sdhammo@sandia.gov

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Our Research Vision

"..we should have a **single application code base** which runs across multiple platforms and architectures *and* achieves as **close to 90% of the native/best programming model/solution** as possible."

- This is *really* hard
- But .. its what our code teams expect (and deserve)

- Requires us not just to be content to write Kokkos but also to write solutions in other models to assess how well we are doing
- We have learned **a huge** amount about Kokkos (and other models) this way

Performance Tools

- One of the problems with this approach has been that we find performance differences
- Then .. we need code ninjas to work out where some of the time is going
- Existing performance tools have some real weaknesses
  - They don't understand our abstractions
  - Some don't have a one-to-one mapping
  - Most are single-type of device
  - Non-uniformity of tools across architectures

KokkosP Hooks

- Integrated hooks directly into the Kokkos runtime
  - Parallel kernel dispatch
  - Memory allocation using Kokkos views
  - User-defined section naming

- Dynamically loaded at runtime (no LD_PRELOAD)

- Stackable (one tool calls another, calls another)
  - Record multiple items of interest in a single run
  - Easily filter

- Working on dynamic tool feedback API
  - Control threads, data placement etc

Simple Tools

```
KOKKOS_PROFILE_LIBRARY=/home/sdhammo/git/kokkos-profiling-github-repo/
src/tools/simple-kernel-timer/kp_kernel_timer.so ./lulesh.host -s 40
```

1. sdhammo@shepard-lsm1:~/git/asc-codesign-repo-3/code/lulesh/cleaned/kokkos-opt-3 (ssh)

bash | nano | bash | bash | bash | sdhammo@shepard-lsm1:~... | sdhammo@shepard-lsm1:~...

```
compile-time-node60.out      Kokkos_Profiling_Interface.o  .project
compile-time-node61.out      Kokkos_Serial.o               shepard-lsm1-104642.dat
compile-time-node63.out      Kokkos_Serial_TaskPolicy.o
[sdhammo@shepard-lsm1 kokkos-opt-3]$ /home/sdhammo/git/kokkos-profiling-github-repo/src/tools/simple-kernel-timer/kp_reader shepard-lsm1-104642.dat
                                                    EvalEOSForElems A        1.25549        45290        0.00003  16.900  12.539
                                                CalcFBHourglassForceForElems A        0.80554         1294        0.00062  10.844   8.045
ZN30_INTERNAL_9_lulesh_cc_c58402ba28CalcHourglassControlForElemsER6DomainPddEUliRiE_        0.69605         1294        0.00054   9.370   6.951
                                                  CalcMonotonicQRegionForElems        0.51277        14234        0.00004   6.902   5.121
                                                     IntegrateStressForElems B        0.45641         1294        0.00035   6.144   4.558
                                                        CalcKinematicsForElems        0.44597         1294        0.00034   6.003   4.454
                                                CalcFBHourglassForceForElems B        0.43574         1294        0.00034   5.866   4.352
                                                       InitStressTermsForElems        0.41886         1294        0.00032   5.638   4.183
                                                    IntegrateStressForElems A        0.36334         1294        0.00028   4.891   3.629
                                                        CalcEnergyForElems        0.27219        14234        0.00002   3.664   2.718
ZN30_INTERNAL_9_lulesh_cc_c58402ba29CalcCourantConstraintForElemsER6DomainiPidRdEUliR9MinFinderE_        0.25953        14234        0.00002   3.494   2.592
                                                      EvalEOSForElems F        0.24330        14234        0.00002   3.275   2.430
                                                CalcMonotonicQGradientsForElems        0.22903         1294        0.00018   3.083   2.287

ZN30_INTERNAL_
                                                    EvalEOSForElems A        1.25549
                                                CalcFBHourglassForceForElems A        0.80554
ZN30_INTERNAL_9_lulesh_cc_c58402ba28CalcHourglassControlForElemsER6DomainPddEUliRiE_        0.69605
                                                  CalcMonotonicQRegionForElems        0.51277
                                                     IntegrateStressForElems B        0.45641
ZN
                                                        CalcKinematicsForElems        0.44597
                                                CalcFBHourglassForceForElems B        0.43574
                                                       InitStressTermsForElems        0.41886

                                                ApplyMaterialPropertiesForElems B        0.00781         1294        0.00001   0.105   0.078
                                                ApplyMaterialPropertiesForElems C        0.00693         1294        0.00001   0.093   0.069
                                    N6Kokkos4Impl20ViewDefaultConstructINS_6OpenMPEdLb1EEE        0.00502           35        0.00014   0.068   0.050
                                    N6Kokkos4Impl20ViewDefaultConstructINS_6OpenMPEiLb1EEE        0.00136           11        0.00012   0.018   0.014
                                    N6Kokkos4Impl20ViewDefaultConstructINS_6OpenMPEjLb1EEE        0.00056           92        0.00001   0.008   0.006
N6Kokkos4Impl9ViewRemapINS_4ViewIPdNS_10LayoutLeftENS_6DeviceINS_6OpenMPENS_9HostSpaceEEEvNS0_11ViewDefaultEEESA_Lj1EEE        0.00021           35        0.00001   0.003   0.002
N6Kokkos4Impl9ViewRemapINS_4ViewIPiNS_10LayoutLeftENS_6DeviceINS_6OpenMPENS_9HostSpaceEEEvNS0_11ViewDefaultEEESA_Lj1EEE        0.00006           11        0.00001   0.001   0.001

----------------------------------------------------------------------
Summary:
```

Profiling Kokkos with VTune

- Programming abstractions obscure the call stacks
- Confusing identification of Parallel Regions
  - OpenMP parallel for is in a single file: Kokkos_OpenMP_Parallel.hpp
- Very long function names which users absolutely hate

Grouping: Class / Function / Call Stack

Class / Function / Call Stack	Effective Time by Utilization (Idle, Poor, Ok, Ideal, Over)	Imb
[Not part of any known object class]	53.338s	
LAMMPS_NS::NeighborKokkosExecute<Kokkos::OpenMP>	25.571s	
Kokkos::Impl	21.539s	
LAMMPS_NS::PairComputeFunctor<LAMMPS_NS::PairLJCutKokkos<Kokkos::OpenMP>, (int)1, (bool)	0.794s	
LAMMPS_NS::FixNVEKokkos<Kokkos::OpenMP>	0.634s	
LAMMPS_NS::AtomNeighbors	0.431s	
Kokkos::Impl::OpenMPexecTeamMember	0.344s	
Kokkos::Impl::OpenMPexec	0.235s	

Profiling Kokkos: KokkosP+VTune

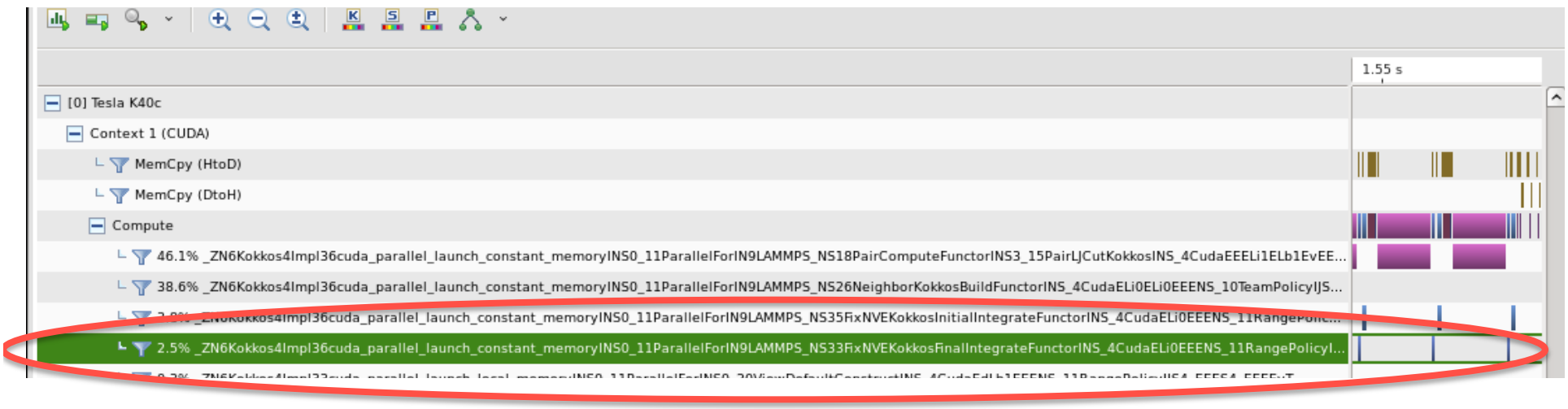
- KokkosP tools add domain marking for kernels
- VTune allows filtering, zoom in, etc. based on Domain and Frames
- Track memory allocations in timeline viewer
- Domain markings can also make CUDA kernel execution visible

Grouping: Frame Domain / Frame / Function / Call Stack

Frame Domain / Frame / Function / Call Stack	CPU Time								
	Effective Time by Utilization				Spin Time			Overhead Time			
	Idle	Poor	Ok	Ideal	Over	Imbalanc...	Lock ...	Other	Creation (OpenMP)	Sch...	Red...	Ato...
ParallelReduce.PairForce	42.993s					3.226s	0s	0s	0.011s	0s	0s	0s
ParallelFor.NEigh::Build	35.089s					0.798s	0s	0.000s	0.004s	0s	0s	0s
ParallelFor.22BuildBorderList	22.546s					0.010s	0.001s	-0.00 ...	0.000s	0s	0s	0s
[No frame domain - Outside a	0.204s					2.250s	0.011s	0.005s	0.022s	0s	0s	0s
ParallelFor.N6Kokkos4Impl20	0.636s					0.271s	0s	0.000s	0.027s	0s	0s	0s
ParallelFor.N6Kokkos4Impl20	0.016s					0.260s	0s	0s	0.602s	0s	0s	0s
ParallelFor.NVE::final	0.509s					0.107s	0s	0s	0.002s	0s	0s	0s

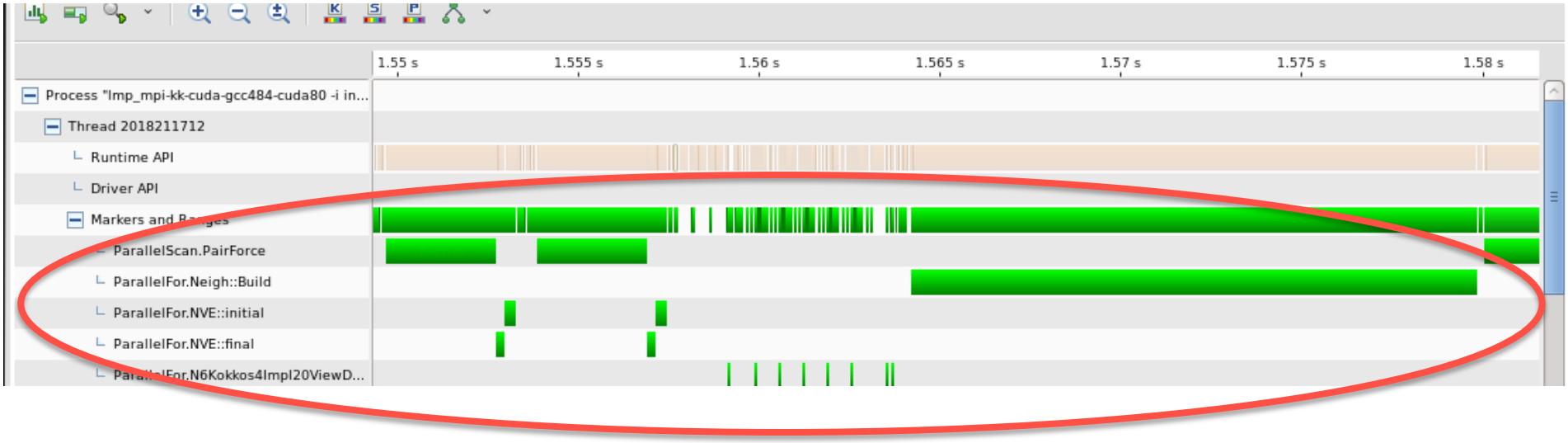
Profiling Kokkos: NSight

- NSight critical for performance optimization on GPUs
  - Bandwidth analysis
  - Memory access patterns
  - Stall reasons
- **Problem:** again template based abstraction layers make awful function names, even worse than in VTune for some problems



Profiling Kokkos: KokkosP+ NSight

- KokkosP uses the CUDA8-NVTX Interface
  - Named Domains in addition to named Ranges
  - Using KokkosP NVProf-Connector to pass user-provided names through
  - Shows Host Regions + GPU Regions



Some Late Breaking News…

- Agreement to work on one set of runtime hooks to rule them all (with RAJA and Kokkos)
- Means a single set of connectors/hooks will work across C++ abstraction runtimes
- Collaboration with Dave Beckingsale, Jeff Keasler and Rich Hornung (LLNL)
- Prototype(y-ish) hooks already working
- Will mean a new name (Dave is in charge of suggestions .. unfortunately)

Summary

- Just a snapshot of the tools interfaces and *connectors* we have been working on
  - Connections for parallel execution patterns
  - Memory allocations
  - User-defined named regions

- Simple tools require no vendor products (all in-house)
- But .. users can also use their favorite vendor tools (VTune, Nsight, etc)

- Research prototypes:
  - Feedback API (tools dynamically feed info back to application/Kokkos)
  - Debugging connectors
  - Vectorization and Instruction Analysis (APEX LDRD at Sandia)

- http://www.github.com/kokkos

Sandia National Laboratories

Exceptional service in the national interest

http://www.github.com/kokkos