

#### USING NSIGHT TOOLS TO OPTIMIZE THE NAMD MOLECULAR DYNAMICS SIMULATION PROGRAM

Robert (Bob) Knight - Software Engineer, NVIDIA

David Hardy - Senior Research Programmer, U. Illinois at Urbana Champaign March 19, 2019



Home

Financial Advisors - Why Register? -

Investing

FIDELITY INSTITUTIO



D Ameritrade

#### The TICKER TAPE

Market News

O Search Ticker Tape

Markets & Economy > Viewpoints & White Papers

Funds & Products

#### Rebalancing Your Investment Mix

Posted: 10/24/2018 by Fidelity Viewpoints

#### Even during a strong stock market, you should evaluate risks and returns.

Defined Contribution





By TOM PETRUNO, Contributing Writer October 3, 2018

#### How to Rebalance Your Portfolio

A risk of this long-running bull market: You may have more money in stocks than you should.



Home > Investing > Investment Strategies > Portfolio Strategy

#### Mid-Year Tune-Up: Reassess, Realign, **Rebalance?**

June 15, 2017 06 min read



Morgan Stanley

Ideas People What We Do About U:

Kiplinger's

FINANCIAL PLANNING



#### It's Time to Rebalance

#### WHY REBALANCE?

🗼 NVIDIA.

#### **GPU Motivation:** Performance Trends



Peak Double Precision FLOPS

1400 GB/s K40 K20 M2090 M1060 M2050 

Peak Memory Bandwidth

#### **BECAUSE PERFORMANCE MATTERS**

#### DELIVERING MOST OF THE NEW COMPUTING PERFORMANCE



# HOW? NSIGHT SYSTEMS & NSIGHT COMPUTE



- Nsight Systems
  - Focus on the application's algorithm a unique perspective
  - Rebalance your application's compute cycles across the system's CPUs & GPUs



#### Nsight Compute

CUDA kernel profiling



### NAMD - NANOSCALE MOLECULAR DYNAMICS

- 25 years of NAMD
- ► 50,000+ Users



- Awards: 2002 Gordon Bell, 2012 Sidney
   Fernbach
- Solving Important Biophysics/Chemistry Problems
- Focused on scaling across GPUs Biggest Bang for Their Compute \$

#### NAMD & VISUAL MOLECULAR DYNAMICS COMPUTATIONAL MICROSCOPE

Enable researchers to investigate systems at the atomic scale

NAMD - molecular dynamics simulation VMD - visualization, system preparation and analysis



### NAMD OVERVIEW

- Simulate the physical movement of atoms within a molecular system
- Atoms are organized in fixed volume patches within the system
- Forces that move atoms are calculated at each timestep
- After a cycle (e.g. 20 timesteps), atoms may migrate to an adjacent patch
- Performance measured as ns/day the number of nanoseconds of simulation that could be calculated in one day of running the workload (higher is better)

#### Molecule with 4x4x4 Patches



#### PARALLELISM IN MOLECULAR DYNAMICS LIMITED TO EACH TIMESTEP



### TIMESTEP COMPUTATIONAL FLOP COST

90% — short-range non-bonded forces 5% – long-range PME electrostatics force calculation 2% – bonded forces 2% – corrections for excluded interactions update \_\_\_\_\_ - 1% – numerical integration

#### Start applying GPU acceleration to most expensive parts

#### **BREAKDOWN A WORKLOAD**



# NVIDIA TOOLS EXTENSION (NTVX) API



- Instrument application behavior
  - Supported by all NVIDIA tools
- Insert markers, ranges
- Name resources
  - OS thread, CUDA runtime
- Define scope using *domains*

```
void Wait(int waitMilliseconds)
nvtxNameOsThread("MAIN");
nvtxRangePush( FUNCTION );
nvtxMark(>"Waiting...");
Sleep(waitMilliseconds);
nvtxRangePop();
int main(void)
nvtxNameOsThread("MAIN");
nvtxRangePush(__FUNCTION__);
Wait();
nvtxRangePop();
```

### NAMD ALGORITHM SHOWN WITH NVTX



# NAMD CYCLE



# 20 Timesteps followed by Atom Migration



### **ONE NAMD TIMESTEP**



### **TIMESTEP - SINGLE PATCH**

| Timeline View              |                     |                            |                    |                       | 2 1x -   | 🗴 <u>1 warning, 17 message</u> |
|----------------------------|---------------------|----------------------------|--------------------|-----------------------|----------|--------------------------------|
|                            | <b>19s</b> +376.24n | ns +376.25ms +             | 376.26ms +376.27ms | +376.28ms +376.29ms   | +376.3ms | +376.31ms +376.32ms            |
| ► CPU (12)                 |                     |                            |                    |                       |          |                                |
| - Threads (13)             |                     |                            |                    |                       |          |                                |
| ▼ √ [1816] NAMD masterPe - |                     |                            |                    |                       |          |                                |
| OS runtimo librarios       |                     |                            |                    |                       |          | 1                              |
| <ul> <li>NVTX</li> </ul>   | <b>*</b>            |                            | ~88us              |                       |          |                                |
| [Default]                  | i integ integrate   | positionsReady: 1873 [13.1 |                    | integrate 3 [59.488 µ | s]       | int                            |
| Charm++                    |                     |                            |                    |                       |          |                                |
| CUDA API                   |                     |                            |                    |                       |          |                                |
| Profiler overhead          |                     |                            |                    |                       |          | -                              |
|                            | 4                   |                            |                    |                       |          | •                              |

- Zoom In
- Patches are implemented as user-level threads

### **NSIGHT SYSTEMS**

- User
   Instrumentation
- API Tracing
- Backtrace
   Collection
- Custom Data Mining
- Nsight Compute Integration



### **API TRACING**

#### Process stalls on file I/O while waiting for 160ms mmap64 operation +550ms +600ms +650ms +700ms +750ms +800ms 72s 725 CPU (80) Processes (41) (63) python Threads (12) ▼ ▼ [63] python ▼ mmap64 OS runtime libraries CUDA API elementwise kernel CuDNN infiduation and the second distribution of the CuBLAS Drofiler overhead

#### Thread communicates over socket



APIs: CUDA, cuDNN, cuBLAS, OSRT (OS RunTime), OpenGL, OpenACC, DirectX 12, Vulkan\*

\* Available in next release

# SAMPLING DATA UNCOVERS CPU ACTIVITY

*Filter By Selection* shows a specific thread's activity

Blocked State Backtrace shows the path leading to an OS runtime library call

| Timeline View •   |             |   |   |   |  |   |
|---|-------------|---|---|---|--|---|
|   | 18s +       | 470ms   | +480ms  | +490ms  | +500ms                                 | 18s 508.33ms +520ms +   |
| INVIA   |             |   |   |   |  | 1   |
| CUDA API  |             |   | Filte   | r By  |  | Disclored State   |
| Profiler overhead   |             |   | Selec   | tion  |  | DIOCKED SLALE   |
| ▼ ✔ [22262] NAMD masterPe -   | *           |   |   |   |  | Backtrace   |
| OS runtime libraries  | *           |   | pt pthre ioctl                                  | D pthread_nutex                                 | I pt/read_                             | mute lock Waiting (with callchain)  |
| NVTX  | 💉 🖬         |   |   | pute  | cor putePMEC                           | udə [12] 98 m Time: 18.5083s  |
|   | *           |   | cuda  |   |  | Call stack at 18.497s:  |
| Profiler overhead   | -           |   |   |   |  | libpthread-2.23.so!_pthread_mutex_lock<br>namd2!PmeAtomStorage::addAtoms ()                           |
| Profiler overhead   |             |   |   |   |  | namd2!ComputePmeCUDADevice::recvAtoms()   |
| [22270] namd2 •   |             |   |   |   |  | namd2!ComputePmeCUDA::sendAtoms()<br>namd2!ComputePmeCUDA::doWork()                                   |
| Tilter 0.00% (5 samples) of da  | ata is show | n due to app                                    | olied filters. Time fil                         | ter: 18.50 to 18.50 (0.                         | 00 seconds or <b>0.0</b> 4             | %). namd2!ScdSchedulerorever<br>namd2!ScdScheduler<br>mamd2!ScriptTcl::run()<br>namd2!sriptTcl::run() |
| Symbol Name   |             | Se  | elf, % 🔻 Module Na                              | ame   |  | namd2!main<br>libc.2.23 sol_libc.start.main   |
| ComputePmeCUDA::sendAtoms()   |             |   | 80.00 /home/rkn                                 | 80.00 /home/rknight/namd/Linux-x86_64-g++/namd2 |  | namd2!_start  |
| <ul> <li>ComputePmeCUDA::doWork()</li> </ul>  |             | 1923  | 80.00 /home/rkn                                 | hight/namd/Linux-x86                            | 5_64-g++/namd2                         |   |
| <ul> <li>WorkDistrib::enqueueWorkC(Loc</li> </ul>                                   | alWorkMs    | g*)   | 80.00 /home/rknight/namd/Linux-x86_64-g++/namd2 |   |  |   |
| ✓ CkDeliverMessageFree  |             | 80.00 /home/rknight/namd/Linux-x86_64-g++/namd2 |   |   |  |   |
| _processHandler(Vold^, CkCoreState^)  |             | 80.00 /home/rknight/namd/Linux-x86_64-g++/namd2 |   |   |  |   |
| <ul> <li>CsdScheduleForever</li> <li>[Max dopth]</li> </ul>                         |             |   | 80.00 /nome/rkn                                 | hight/hamd/Linux-x80                            | 5_64-g++/namoz                         |   |
| [wax depth]   |             |   |   | ni<br>hight/namd/Linuv-v96                      | $5.64 \cdot a + \pm / n \cdot a = d^2$ |   |
| CdcEifo Pon   |             |   |   |   |  |   |
| CdsFifo_Pop   |             |   | 20.00 /home/rkn                                 | ight/namd/Linux-x86                             | 5.64 - q + + /namd2                    |   |
| <ul> <li>CdsFifo_Pop</li> <li>CsdNextMessage</li> <li>CsdScheduleForever</li> </ul> |             |   | 20.00 /home/rkn<br>20.00 /home/rkn              | hight/namd/Linux-x86                            | $5_{64-g++/namd2}$                     |   |

#### **REPORT NAVIGATION DEMO**

# **CUSTOM DATA MINING**

5076271

\* Available in next release

#### Find Needles in Haystacks of Data

8

0



21

NVIDIA.

| nsys-exporter*                               | Kernel Stat  | tistics – all  | times in na   | noseconds   |   |
|--|--|--|---|---|---|
| QDREP->SQLite                                | minimum<br>  | maximum  | average   | kernel  |   |
| Use Cases<br>• Outlier<br>Discovery          | 1729557<br>561821<br>474173<br>454621<br>393470<br>52288 | <b>5347138</b><br>631674<br>574618<br>593402<br>676060<br>183455 | 2403882.7<br>581409.6<br>489148.1<br>465637.6<br>420914.9<br>116258.2 | <b>nonbondedFo</b><br>batchTransp<br>batchTransp<br>spread_char<br>gather_forc<br>bondedForce | orceKernel<br>oose_xyz_yzx_kernel<br>oose_xyz_zxy_kernel<br>oge_kernel<br>ee<br>sKernel |
| <ul> <li>Regression<br/>Analysis</li> </ul>  | <br>nonbondedFo  | orceKernel   | The   | e longest nonBondedForceKernel is at 35.453s on GPU1, stream 133                              |   |
| <ul> <li>Scripted</li> <li>Custom</li> </ul> | duration   | start<br>  | stream  | context   | GPU   |
| Report                                       | 5347138  | 35453528745  | 133   | 7   | 1   |
| Generation                                   | 5245934  | 39527523457  | 132   | 8   | 0   |

132

41048810842

### **KERNELSTATS SCRIPT**



#!/bin/bash

sqlite3 \$DB "ALTER TABLE CUPTI\_ACTIVITY\_KIND\_KERNEL ADD COLUMN duration INT;"

sqlite3 \$DB "UPDATE CUPTI\_ACTIVITY\_KIND\_KERNEL SET duration = end-start;"

// add duration column, set duration column's value

sqlite3 \$DB "CREATE TABLE kernelStats (shortName INTEGER, min INTEGER, max INTEGER, avg INTEGER);"

sqlite3 \$DB "INSERT INTO kernelStats SELECT shortName, min(duration), max(duration), avg(duration) FROM CUPTI\_ACTIVITY\_KIND\_KERNEL GROUP BY shortName;"

// create new table, insert kernel name IDs, min, max, avg into it

sqlite3 -column -header \$DB SELECT min as minimum, max as maximum, round(avg,1) as average, value as kernel FROM kernelStats INNER JOIN StringIds ON StringIds.id = kernelStats.shortName ORDER BY avg DESC;"

// print formatted min, max, avg, and kernel name values, order by descending avg

### **NSIGHT COMPUTE INTEGRATION**





# DATA COLLECTION



rknight@RKNIGHT-WS: ~

rknight@RKNIGHT-WS:~\$ nsys launch -t cuda,nvtx,osrt --osrt-threshold 100000 -w true /usr/local/namd/bin/namd2 --beginEventStep 1010 --endEventStep 1070 +p40 +setcpuaffi nity +idlepoll +devices 0,1,2,3 ./stmv\_nve\_cuda.namd &

#### **Command Line Interface**

- No root access required
- Works in Docker containers
- Interactive Mode
- Supports cudaProfilerStart/Stop
   APIs

# What about NAMD...

Profiling Simulations of Satellite Tobacco Mosaic Virus (STMV) ~ 1 Million Atoms



### V2.12 TIMESTEP COMPUTATIONAL FLOP COST

90% — short-range non-bonded forces 5% – long-range PME electrostatics GPU force calculation 2% – bonded forces CPU 2% – corrections for excluded interactions update \_\_\_\_ 1% — numerical integration

#### **NAMD V2.12** Profiling STMV with nvprof - Maxwell GPU Fully Loaded





### NAMD PERFORMANCE

| GPUs | Architecture | V2.12<br>Nanoseconds/Day |
|------|--------------|--------------------------|
| 1    | Maxwell      | 0.65                     |
| 1    | Volta        | 5.34619                  |
| 2    | Volta        | 5.45701                  |
| 4    | Volta        | 5.35999                  |
| 8    | Volta        | 5.31339                  |

Volta (2018) delivers ~10x performance boost relative to Maxwell (2014)

Failure to scale is caused by unbalanced resource utilization

### V2.13 TIMESTEP COMPUTATIONAL FLOP COST



# NAMD V2.13

#### Moving all force calculations to GPU shrinks timeline gap



#### NAMD PERFORMANCE

| GPUs<br>(Volta) | V2.12<br>Nanoseconds/Day | V2.13<br>Nanoseconds/Day<br>(% Gain vs 2.12) |
|-----------------|--------------------------|--|
| 1               | 5.34619                  | 5.4454 (1.2%)                                |
| 2               | 5.45701                  | 5.97838 (9.5%)                               |
| 4               | 5.35999                  | 7.49265 (39.8%)                              |
| 8               | 5.31339                  | 7.55954 (42.3%)                              |

### NEXT TIMESTEP COMPUTATIONAL FLOP COST

90% — short-range non-bonded forces 5% – long-range PME electrostatics force calculation 2% – bonded forces 2% – corrections for excluded interactions update coordinates 1% — numerical integration GPU

#### **Bonded Kernel Optimization**

| Symbol Name  | Self, %             | Module Name  |
|--|---------------------|--|
| Sequencer::submitReductions_SOA()                                      | 19.0 <mark>1</mark> | /Projects/dhardy/namd/Linux-x86_64-icc-smp-CUDA_devel/namd2_ |
| Sequencer::submitHalfstep_SOA()  | 16.59               | /Projects/dhardy/namd/Linux-x86_64-icc-smp-CUDA_devel/namd2_ |
| Sequencer::addForceToMomentum_SOA(double, double, double, double, int) | 6.40                | /Projects/dhardy/namd/Linux-x86_64-icc-smp-CUDA_devel/namd2  |
| CmiWallTimer   | 5.55                | /Projects/dhardy/namd/Linux-x86_64-icc-smp-CUDA_devel/namd2  |
| HomePatch::doMarginCheck_SOA()   | 4.18                | /Projects/dhardy/namd/Linux-x86_64-icc-smp-CUDA_devel/namd2  |
| 0x7ffe51ffee2f   | 3.86                | [vdso]   |
| ComputeBondedCUDA::finishPatchesOnPe()                                 | 3.53                | /Projects/dhardy/namd/Linux-x86_64-icc-smp-CUDA_devel/namd2  |
| CmiGetNonLocal   | 3.52                | /Projects/dhardy/namd/Linux-x86_64-icc-smp-CUDA_devel/namd2  |
| ComputePmeCUDA::sendAtoms()  | 2.93                | /Projects/dhardy/namd/Linux-x86_64-icc-smp-CUDA_devel/namd2  |
| CcdCallBacks   | 2.42                | /Projects/dhardy/namd/Linux-x86_64-icc-smp-CUDA_devel/namd2  |

Host-side postprocessing of bonded forces still a significant bottleneck



#### CPU integrator causing bottleneck



**Integrator Development Phases** 



Integrator - Phase 1

#### CPU vectorization - arrange data into SOA (structure of arrays) form



Speedups: 26.5% for integrate\_SOA\_2, 52% for integrate\_SOA\_3



#### integrator - Phase Z

#### Per Patch Integrator Offload - Zoom In

₽ 1x Timeline View A 1 warning, 18 messag +30ms +35ms +40ms +45ms +50ms +55ms +60ms +65ms 25s +70ms OS runtime libraries pthr... pthre... pthread\_mutex... pthrea... pthread\_mutex\_lock pthread mutex lock pthread\_mutex\_lock pthread\_mutex\_lock pthread\_mutex\_lock NVTX integrate SOA 1: 2161 [17,433 ms] ntegrate SOA 1: 1159 [12.649 CUDA API aMemcovAs... | cudaM... | cuda. addForceToMoment.. cuda | Profiler overhead Memory Transfer Hell ✓ √ [14613] namd2 devel **GPU Underutilized** OS runtime libraries pthread\_mutex\_lock pthr... pthre... pthread\_m... pthread\_mutex\_lock pthread\_mutex\_lock pthread\_mutex\_lock pthre... pthread\_mutex\_lock pthread\_mut. pthre... pthre.. NVTX ntegrate SOA 1: 2130 [10.801 ms integrate SOA 1: 375 [11.769 Each Kernel Handles CUDA API cud... | cudaMemo Profiler overhead ~500 atoms 23 threads hidden. CUDA (TITAN V. 0000:04:00.0) STMV includes 1M atoms Default stream (7) Kernels Memory Memset DtoH memcpy DtoD memcpy NVTX 2200 more streams 2200 streams hidder . Þ

#### Integrator - Phase 3

#### Per <u>CPU Core</u> Integrator Offload - Zoom In



Integrator - Phase 3

#### Small Memory Copy Penalties

Small memory copy operations should be avoided by grouping data together

Improve memory access performance by using Pinned memory



### NAMD PERFORMANCE

| GPUs | V2.12<br>Nanoseconds/Day | V2.13<br>Nanoseconds/Day<br>(% Gain vs 2.12) | NEXT<br>Nanoseconds/Day<br>SOA + Incomplete<br>Integrator Phase 3<br>(% Gain vs 2.12) |
|------|--------------------------|--|---|
| 1    | 5.34619                  | 5.4454 (1.2%)                                | 4.1451 (-22.5%)   |
| 2    | 5.45701                  | 5.97838 (9.5%)                               | 5.76149 (5.6%)  |
| 4    | 5.35999                  | 7.49265 (39.8%)                              | 7.11889 (32.8%)   |
| 8    | 5.31339                  | 7.55954 (42.3%)                              | 8.10406 (52.5%)   |

Integrator Phase 3 Optimization Development In Progress

# What? Charm++ Runtime using 21.3% on gettimeofday()!

#### Replace with x86\_64 RDTSC instruction, avoid unnecessary calculations.



#### Charm++ Runtime Lock Optimization

| NVIDIA Nsight Systems 2019.1.0        |  | - 🗆 ×                     |
|---------------------------------------|--|---------------------------|
| ile <u>V</u> iew <u>H</u> elp         |  |                           |
| Project 1 × 1GPU-cno-100us.qdrep × te | st-cno-100us-121.qdrep 🗙   |                           |
| Timeline View +                       |  | pr, 1 warning, 17 message |
| 0s                                    | +405ms +410ms +415ms +420ms +425ms 0s 429.41ms +435ms +440ms   | +445ms                    |
| <ul> <li>CPU (96)</li> </ul>          |  |                           |
| <ul> <li>Threads (43)</li> </ul>      |  |                           |
| ▼ [1639] namd2 ▪                      |  |                           |
| OS runtime libraries                  |  |                           |
| ▼ NVTX                                |  |                           |
| [Default]                             |  |                           |
| Charm++                               | Replace pthread_mutex_lock with  | idle [4.115 ms]           |
| CUDA API                              | pthread spin lock.   |                           |
| ▼ [1627] namd2 ▼                      |  |                           |
| OS runtime libraries                  | pth p pt pt pt pt pthr   |                           |
| ▼ NVTX                                | pthread_mutex_lock   |                           |
| [Default]                             | comput   com. c com. [c] c c c [c] c com] [] comp c] | in in <mark>Cont</mark> i |
| Charmate                              |  | idle [3.814 ms]           |
|                                       |  | - Interprotecting         |
| ▼ [1629] namd2 •                      |  |                           |
| OS runtime libraries                  | m p., pt., pt., p., pt., pt., p., p., p., p., p., p., p., p., p  |                           |
| ▼ NVTX                                |  |                           |
| [Default]                             |  | <b>.</b>                  |
| Charm++                               | idle [10.036 ms]   | idle [3.652 ms]           |
| CUDA (Tesla V100-SXM3-32GB, 📌 🚃       |  |                           |
| •                                     |  | ۱.                        |

43

### NAMD PERFORMANCE

| GPUs | V2.12<br>Nanoseconds/Day | V2.13<br>Nanoseconds/Day<br>(% Gain vs 2.12) | NEXT<br>Nanoseconds/Day<br>SOA+Runtime<br>(% Gain vs 2.12) |
|------|--------------------------|--|--|
| 1    | 5.34619                  | 5.4454 (1.2%)                                | 6.7032 (25.4%)   |
| 2    | 5.45701                  | 5.97838 (9.5%)                               | 7.24473 (32.8%)  |
| 4    | 5.35999                  | 7.49265 (39.8%)                              | 8.95371 (67.0%)  |
| 8    | 5.31339                  | 7.55954 (42.3%)                              | 9.26881 (74.4%)  |

Integrator Phase 3 Optimization Development In Progress

### NAMD SUMMARY

- Nsight Systems guides development process
  - Estimated best case performance: 10 to 12 nanoseconds/day (single GPU)
  - Data transfer activity constraining performance



# **COMMON OPTIMIZATION OPPORTUNITIES**

#### CPU

- Thread Synchronization
- Algorithm bottlenecks starve the GPU(s)
- Multi GPU
  - Communication between GPUs
  - Lack of Stream Overlap in memory management, kernel execution

#### Single GPU

- Memory operations blocking, serial, unnecessary
- Too much synchronization device, context, stream, default stream, implicit
- CPU GPU Overlap avoid excessive communication

#### **NSIGHT PRODUCT FAMILY**

- Nsight Systems Analyze application algorithm system-wide
- Nsight Compute Debug/optimize CUDA kernel
- Nsight Graphics Debug/optimize graphics workloads



#### ACKNOWLEDGMENTS

- U. of Illinois
  - Julio Maia, Ronak Buch, John Stone, Jim Phillips
- NVIDIA
  - Daniel Horowitz, Antoine Froger, Sneha Kottapalli, Peng Wang

#### **THANK YOU!**

Visit us at the NVIDIA booth for a live demo!

Download latest public version https://developer.nvidia.com/nsight-systems

Also available in CUDA Toolkit (v10.1 and later)

Forums: https://devtalk.nvidia.com

Email: nsight-systems@nvidia.com

# **DEVELOPER TOOLS AT GTC19**

#### Talks

- S9345: CUDA Kernel Profiling using NVIDIA Nsight Compute
- S9661: Nsight Graphics DXR/Vulkan Profiling/Vulkan Raytracing
- S9751: Accelerate Your CUDA Development with Latest Debugging and Code Analysis Developer Tools
- S9866: Optimizing Facebook AI Workloads for NVIDIA GPUs
- S9339: Profiling Deep Learning Networks

Demos of DevTools products on Linux, DRIVE AGX, & Jetson AGX at the showfloor

- Wednesday @12-7
- Thursday @11-1



#### **DEMO BACKUP**

#### **CORRELATE ACTIVITY**

| Timeline View +  |   | P 1x I warning, 18 message |
|--|---|----------------------------|
| CUDA API<br>Profiler overhead  | 18s + 18s 525 860ms -526ms +526.2ms +526.4ms +526.6ms +526.8ms +527ms +527.2ms ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓  | +527.4ms +527.6ms +527.8ms |
| OS runtime libraries<br>NVTX<br>CUDA API                                     | computePMECuda [14.0]     computeBonded-OpenBoxesOnPe [1.044 ms]       Sprea     Selecting one  |                            |
| Profiler overhead  • I [22262] NAMD masterPe • OS runtime libraries          | Call to spread_charge_kernel  Kernel launcher Begins: 18.5259s Ends: 18.5259s (+19.762 µs)  |                            |
| NVTX<br>CUDA API<br>Profiler overhead  | Kernel name: spread_charge_kernel       and effect, i.e.         Return value: 0       GPU: TITAN V, 0000:01:00.0         Stream: 14       analysis         Latency: 1.626 ms→       analysis |                            |
| <ul> <li>[22266] [NS] •</li> <li>[22267] CUPTI worker thread •</li> </ul>    | Correlation ID: 867   |                            |
| CUDA (TITAN V, 0000:01:00.0)<br>Stream 14<br>Kernels<br>spread_charge_kernel | Pinned Rows   | spread_charge_kernel       |
| NVTX   | computePMECuda [2.886 ms]   | spreadCharge [1.296 ms]    |

### **FINDING A CORRELATION**

