OpenMP Common Core

Learning parallelization of real applications from the ground-up

OpenMP Enabling HPC since 1997

Manuel Arenaz | June 25, 2021

appentra code performance

Brookhaven National Laboratory

Stony Brook University

Oak Ridge National Laboratory

Agenda

14:00 - 14:15	Setup and welcome participants
14:15 - 14:30	Overview
14:35 - 15:45	The OpenMP Common Core The Parallelware Analyzer performance optimization report Key technology differentiation: Code into patterns for parallelization Using Parallelware Analyzer: A walk-through with MATMUL example
15:45 - 16:15	Coffee
16:15 - 17:50	**Practicals: Use Parallelware Analyzer to parallelize PI and LULESHmk with OpenMP**
17:50 - 18:00	Close

Hands-on Lab using Parallelware Analyzer

- **First, install and launch Parallelware Analyzer:**
  https://www.appentra.com/products/parallelware-analyzer/trial/

- **Second, follow the step-by-step instructions to parallelize PI and LULESHmk**
  - Decomposition of the codes into patterns
  - Generate, build, run and benchmark parallel versions of the codes using Parallelware Analyzer
  - https://www.appentra.com/parallelware-analyzer-openmp-common-core-exercise-pi/
  - https://www.appentra.com/parallelware-analyzer-openmp-common-core-exercise-luleshmk/

PI: Decomposition of into patterns

Code file "pi.c"		Pattern			
Function	Line	Forall	Scalar reduction	Sparse reduction	Convergence loop
main	27		sum		

Parallelization strategy:
Parallel Loop w/ Built-in reduction

PI: Performance on Appentra CI

Version	Problem size	#threads	Time (secs) [Speedup]	Result	Error
pi	N=900000000	n/a	15.000000	3.14159265	2.7e-15
pi_v1_omp_reduction	N=900000000	1	14.331680	3.14159265	2.7e-15
		2	[1.99x] 7.175246	3.14159265	8.4e-14
		4	[3.79x] 3.785248	3.14159265	8.5e-14

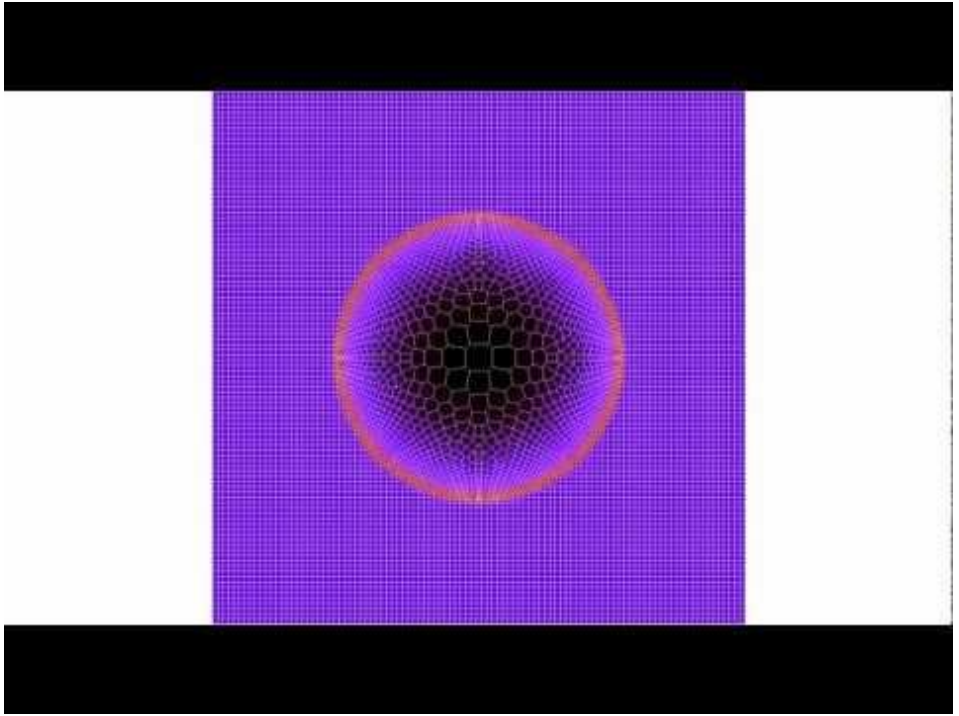
CPU: Intel(R) Core(TM) i7-3770 CPU
MEMORY: 4x DDR3 Kingston KHX1600C10D3/8GX @ 1333 MT/s (32 GiB)
GPU: NVIDIA GeForce GTX 670 (2 GiB GDDR5)
GPU BUS: PCIe 3.0 x16

LULESHmk: Parallel Sparse Reduction

Livermore **U**nstructured **L**agrange
Explicit **S**hock **H**ydrodynamics

Part of a Physics Simulation software (ALE3D)

Models the propagation of a Sedov blast wave using Lagrangian hydrodynamics



LULESHmk: How to verify correctness?

```
$ ./luleshmk
- Configuring the test...
- Executing the test...
gprof ./luleshmk
- Verifying the test...
Run completed:
   Problem size        =  30
   MPI tasks           =  1
   Iteration count     =  932
   Final Origin Energy = 1.000000e+00
   Number of nodes    = 27000
   Number of elements = 30000
   Number of regions  = 1
      Region 1 of size 30000
   Testing Plane 0 of Energy Array on rank 0:
        MaxAbsDiff   = 8.410000e+02
        TotalAbsDiff = 1.303550e+05
        MaxRelDiff   = 9.655568e-01

Elapsed time         =      71.00 (s)
Grind time (us/z/c)  =   2.821491 (per dom)  (  2.821491
overall)
FOM                  =  354.42254 (z/s)
```

Decomposition of LULESH into patterns

Code file "luleshmk.c"		**Pattern**				
Function	**Line**	**Forall**		**Scalar reduction**	**Sparse reduction**	**Convergence loop**
CalcElemFBHourglassForce_workload	61			sum		
CalcElemFBHourglassForce	74	hgfx hgfy hgfz				
CalcFBHourglassForceForElems	**132**				**domain_m_fx** **domain_m_fy** **domain_m_fz**	
ApplyMaterialPropertiesForElems_workload	188			sum		
ApplyMaterialPropertiesForElems	211	vnew				
	218	vnew				
CalcElemVelocityGradient_workload	232			sum		
CalcKinematicsForElems	259	domain_m_dxx domain_m_dzz	domain_m_dyy			
luleshmk	292					iter (loop index)
main	386	locDom_e locDom_m_dxx locDom_m_dyy	locDom_m_dzz vnew			
	395	locDom_m_nodelist				
	403	locDom_fx locDom_fy locDom_fz				
VerifyAndWriteFinalOutput	529			MaxAbsDiff TotalAbsDiff MaxRelDiff		

Version 1: “luleshmk_v1_omp_atomic”

Code file “luleshmk.c”		Pattern				
Function	Line	Forall		Scalar reduction	Sparse reduction	Convergence loop
CalcElemFBHourglassForce_workload()	60			sum		
CalcElemFBHourglassForce()	73	hgfx hgfy hgfz				
CalcFBHourglassForceForElems()	**131**				**domain_m_fx** **domain_m_fy** **domain_m_fz**	
ApplyMaterialPropertiesForElems_workload()	187			sum		
ApplyMaterialPropertiesForElems()	210	vnew				
	217	vnew				
CalcElemVelocityGradient_workload()	231			sum		
CalcKinematicsForElems()	258	domain_m_dxx domain_m_dzz	domain_m_dyy			
luleshmk()	292					op index)
main()	349	locDom_e locDom_m_dxx locDom_m_dyy	locDom_m_dzz vnew			
	358	locDom_m_nodelist				
	365	locDom_fx locDom_fy locDom_fz				
VerifyAndWriteFinalOutput()	485			MaxAbsDiff TotalAbsDiff MaxRelDiff		

Parallelization strategy:
Parallel Loop w/ Atomic

Version 2: “luleshmk_v2_omp_explicit”

Code file “luleshmk.c”		Pattern				
Function	Line	Forall		Scalar reduction	Sparse reduction	Convergence loop
CalcElemFBHourglassForce_workload()	60			sum		
CalcElemFBHourglassForce()	73	hgfx hgfy hgfz				
CalcFBHourglassForceForElems()	**131**				**domain_m_fx** **domain_m_fy** **domain_m_fz**	
ApplyMaterialPropertiesForElems_workload()	187			sum		
ApplyMaterialPropertiesForElems()	210	vnew				
	217	vnew				
CalcElemVelocityGradient_workload()	231			sum		
CalcKinematicsForElems()	258	domain_m_dxx domain_m_dzz	domain_m_dyy			
luleshmk()	292					index)
main()	349	locDom_e locDom_m_dxx locDom_m_dyy	locDom_m_dzz vnew			
	358	locDom_m_nodelist				
	365	locDom_fx locDom_fy locDom_fz				
VerifyAndWriteFinalOutput()	485			MaxAbsDiff TotalAbsDiff MaxRelDiff		

Parallelization strategy:

Parallel Loop w/ Explicit Privatization

Profiling of LULESHmk

```
$ gcc -pg -o luleshmk luleshmk.c -lm
$ ./luleshmk
$ gprof ./luleshmk
```

```
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls   s/call   s/call  name
 56.23     24.62    24.62 223680000     0.00     0.00  CalcElemFBHourglassForce_workload
 22.76     34.59     9.97      932     0.01     0.01  ApplyMaterialPropertiesForElems_workload
 15.26     41.27     6.68 27960000     0.00     0.00  CalcElemVelocityGradient_workload
  2.26     42.26     0.99      932     0.00     0.03  CalcFBHourglassForceForElems
  2.24     43.24     0.98 27960000     0.00     0.00  CalcElemFBHourglassForce
  0.75     43.57     0.33 27960000     0.00     0.00  CalcElemVelocityGradient
  0.34     43.72     0.15        1     0.15    43.76  luleshmk
  0.09     43.76     0.04      932     0.00     0.01  CalcKinematicsForElems
  0.09     43.80     0.04                             frame_dummy
  0.00     43.80     0.00      932     0.00     0.01  ApplyMaterialPropertiesForElems
  0.00     43.80     0.00        2     0.00     0.00  calculate_checksum
  0.00     43.80     0.00        2     0.00     0.00  getClock
  0.00     43.80     0.00        1     0.00     0.00  Parameters_create
  0.00     43.80     0.00        1     0.00     0.00  Parameters_free
  0.00     43.80     0.00        1     0.00     0.00  VerifyAndWriteFinalOutput
```

Version 3: “luleshmk_v3_omp”

Code file “luleshmk.c”		Pattern				
Function	Line	Forall		Scalar reduction	Sparse reduction	Convergence loop
CalcElemFBHourglassForce_workload()	60			sum		
CalcElemFBHourglassForce()	73	hgfx				
CalcFBHourglassForceForElems()	**131**				**domain_m_fx domain_m_fy domain_m_fz**	
ApplyMaterialPropertiesForElems_workload()	187			sum		
ApplyMaterialPropertiesForElems()	210 217					
CalcElemVelocityGradient_workload()	231			sum		
CalcKinematicsForElems()	258	domain_m_dxx domain_m_dzz	domain_m_dyy			
luleshmk()	292					er (loop index)
main()			m_m_dzz			
VerifyAndWriteFinalOutput()	485			MaxAbsDiff TotalAbsDiff MaxRelDiff		

Parallelization strategy:
Parallel Loop w/ Built-in reduction

Parallelization strategy:
Parallel Loop w/ Atomic

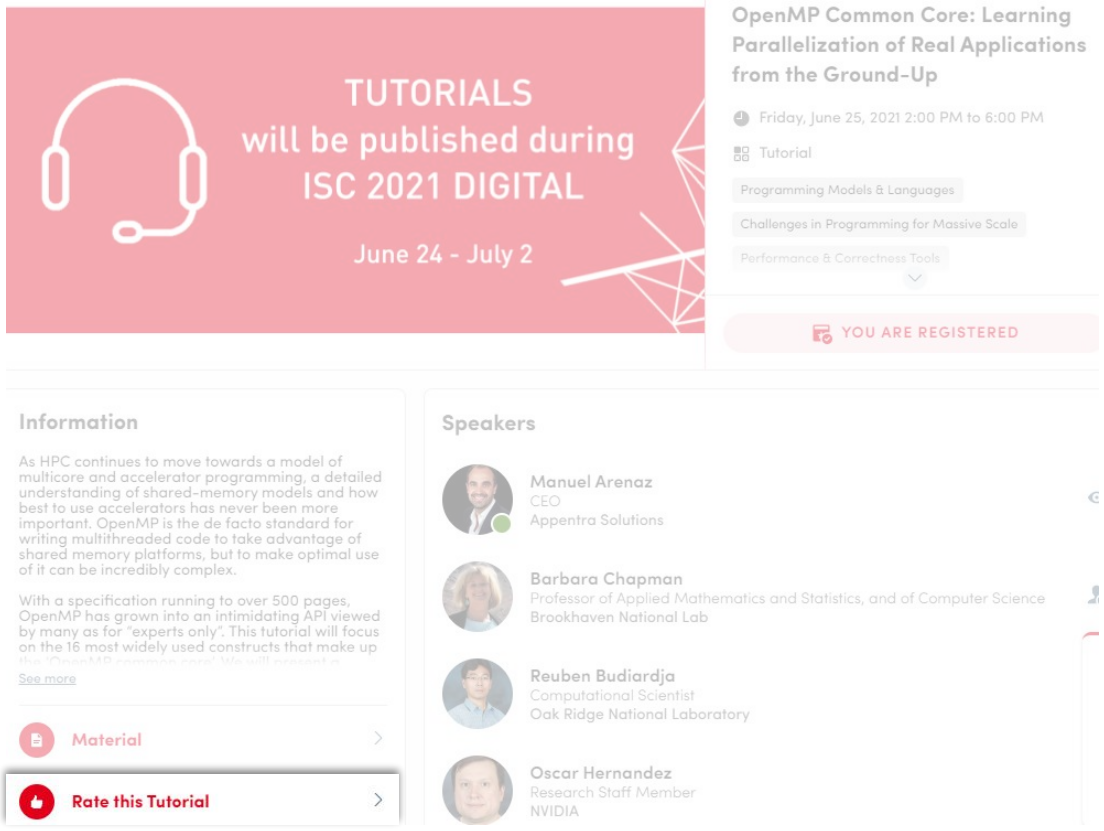
Parallelization strategy:
Parallel Loop

LULESHmk: Performance on Appentra CI

Version	Problem size	#threads	Time (sec) [Speedup]	checksum_f	checksum_e
luleshmk	NumNodes 27000 NumElems 30000	n/a	70.67	3.28901e+11	4.37594e+12
luleshmk_v1_omp_atomic		4	[1.82x] 38.80	3.28901e+11	4.37594e+12
luleshmk_v2_omp_explicit		4	[1.80x] 39.16	3.28901e+11	4.37594e+12
luleshmk_v3_omp		4	[3.63x] 19.46	3.28901e+11	4.37594e+12

CPU: Intel(R) Core(TM) i7-3770 CPU
MEMORY: 4x DDR3 Kingston KHX1600C10D3/8GX @ 1333 MT/s (32 GiB)
GPU: NVIDIA GeForce GTX 670 (2 GiB GDDR5)
GPU BUS: PCIe 3.0 x16

Give feedback!



OpenMP Common Core

Learning parallelization of real applications from the ground-up

OpenMP Enabling HPC since 1997

Manuel Arenaz | June 25, 2021

appentra code performance

Brookhaven National Laboratory

Stony Brook University

Oak Ridge National Laboratory