OpenMP Programming Workshop @LRZ

The OpenMP Common Core and Beyond

OpenMP Enabling HPC since 1997

Manuel Arenaz | February 11-13, 2020

lrz

appentra
make code parallel

Agenda

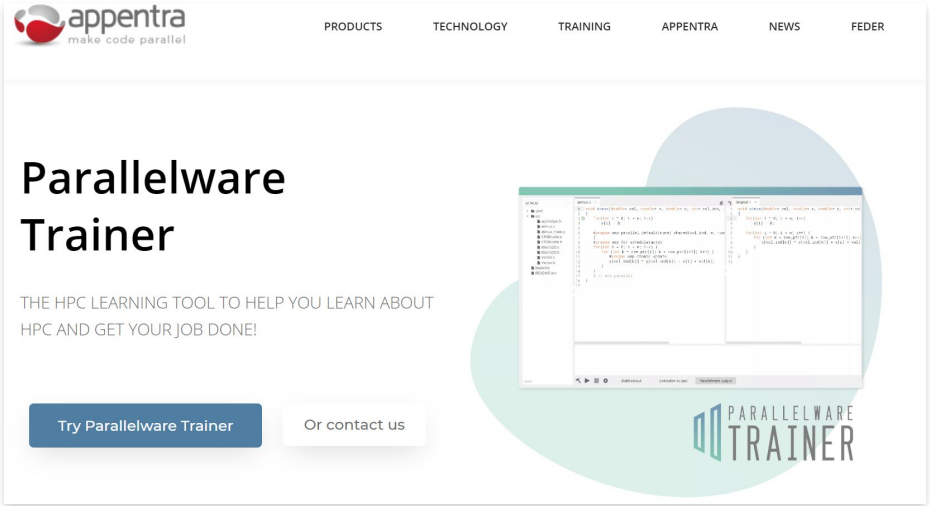
8:30 - 9:00	*Setup and welcome participants*
9:00 - 9:15	Overview
9:15 - 10:30	The OpenMP Common Core Decomposing code into patterns for parallelization Using Parallelware Trainer: A walk-through with PI example
10:30 - 11:00	Coffee
11:00 - 12:40	**Practicals: Examples codes PI, MANDELBROT, HEAT and LULESHmk** **Worksheet: Parallelizing PI and LULESHmk with OpenMP**
12:40 - 13:00	*Close*

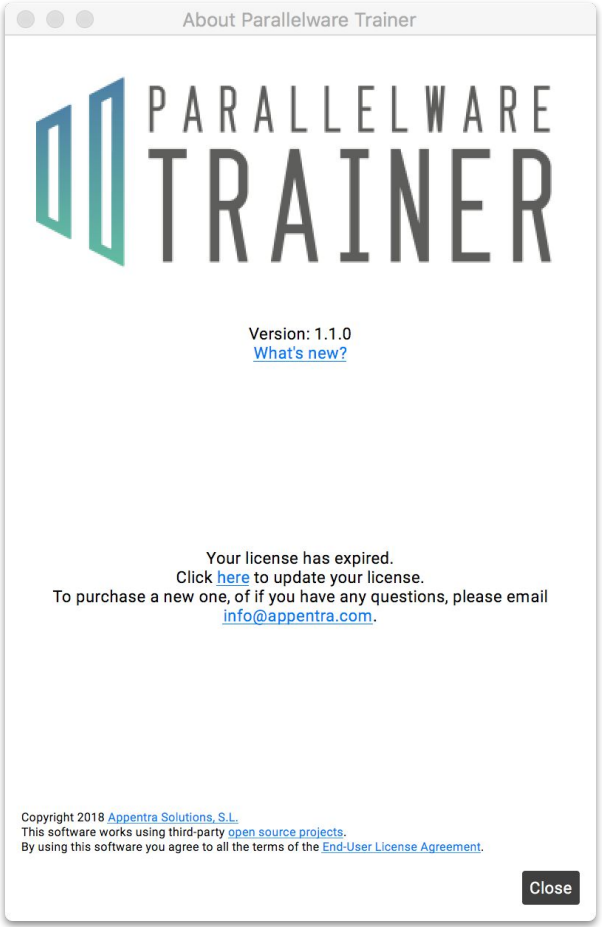
Hands-on Lab

- **Install and launch Parallelware Trainer:**
  https://www.appentra.com/
- Practicals:
  - Parallelizing the calculation of π
  - Parallelizing a micro-kernel of the CORAL-lulesh benchmark
  - Parallelizing HEAT and Mandelbrot.

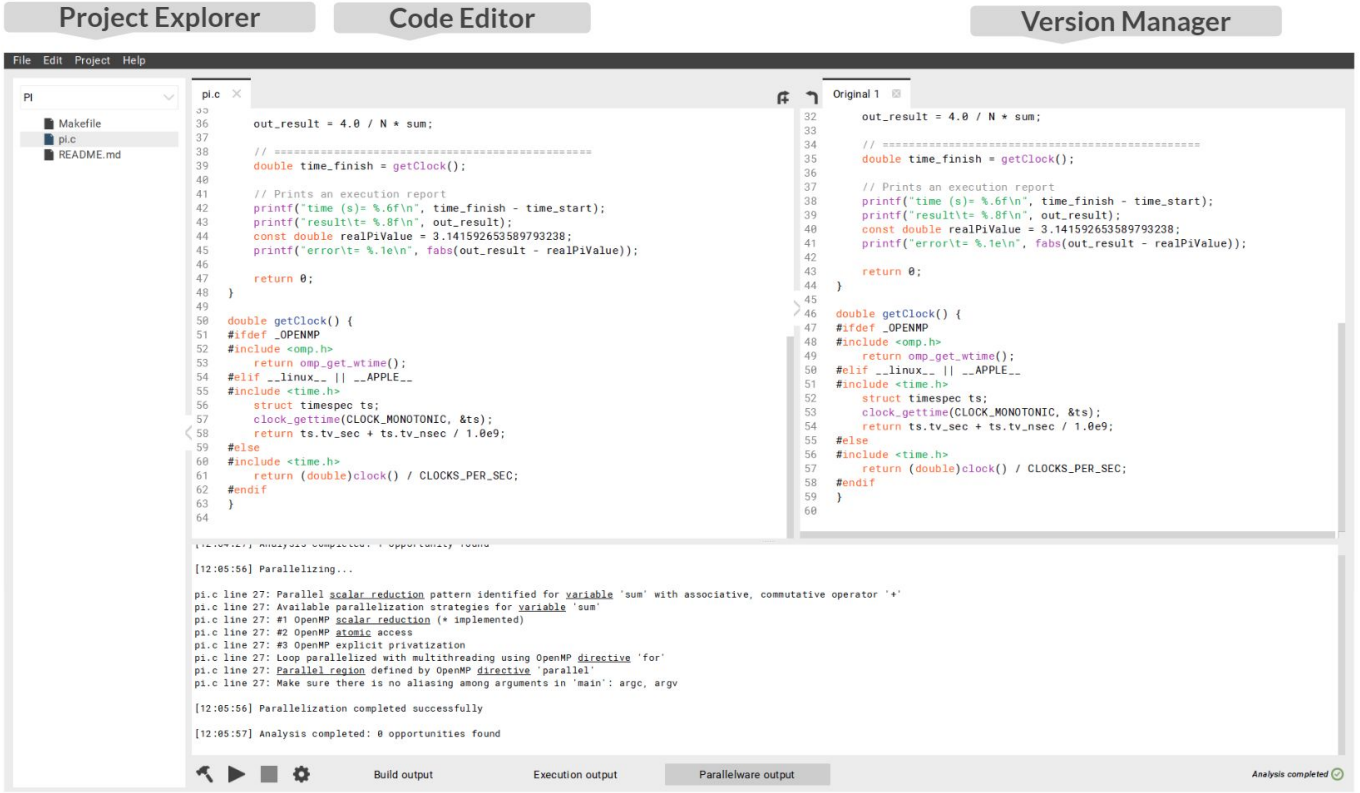
★ **Follow the instructions on the worksheet**

PARALLELWARE TRAINER




About Parallelware Trainer
PARALLELWARE
TRAINER
Version: 1.1.0
What's new?
Your license has expired.
Click here to update your license.
To purchase a new one, of if you have any questions, please email
info@appentra.com.
Copyright 2018 Appentra Solutions, S.L.
This software works using third-party open source projects.
By using this software you agree to all the terms of the End-User License Agreement.
Close

Parallelware Trainer



Parallelizing PI & LULESHmk

Goals:

- Follow the instructions in the worksheet for PI and LULESHmk.
  - Additional case studies: HEAT and MANDELBROT.
- Understand parallelization guided by code patterns.
- Generate, build and run multiple versions using Parallelware Trainer.

PI: Parallel scalar reduction

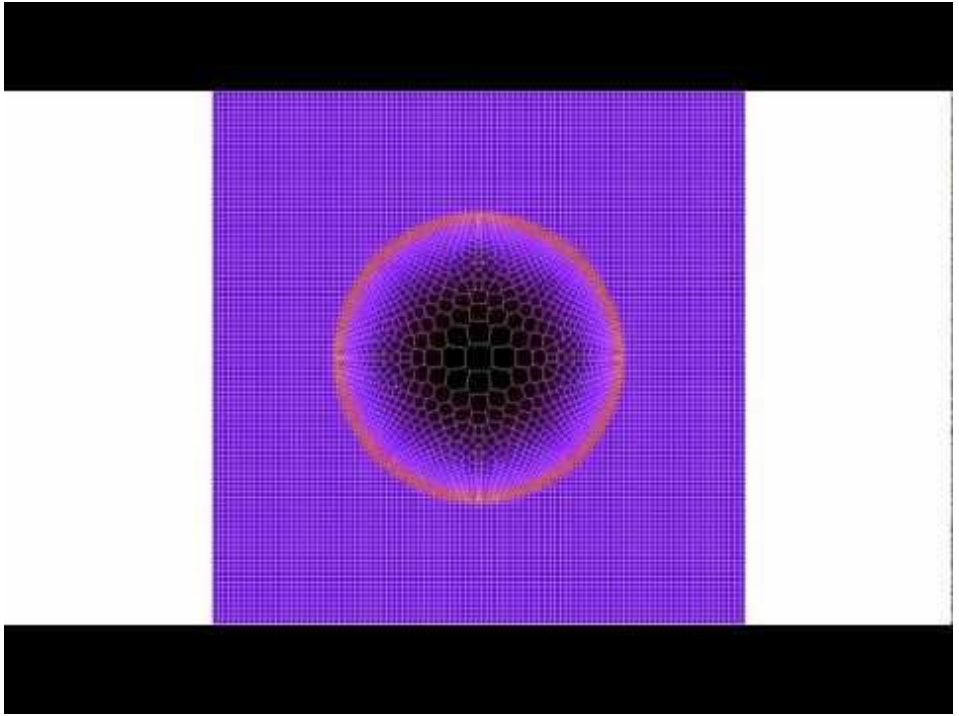
Code file "pi.c"		Pattern			
Function	Line	Forall	Scalar reduction	Sparse reduction	Convergence loop
main()	29		sum		

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



How to verify correctness of LULESHmk

```
$ ./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)
```

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
```

Decomposition of LULESHmk into patterns

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**	
ApplyMaterialPropertiesForElems_workload()	187					
ApplyMaterialPropertiesForElems()	210 217					
CalcElemVelocityGradient_workload()	231					
CalcKinematicsForElems()	258					
luleshmk()	292					iter (loop index)
main()	349					
	358					
	365					
VerifyAndWriteFinalOutput()	485					