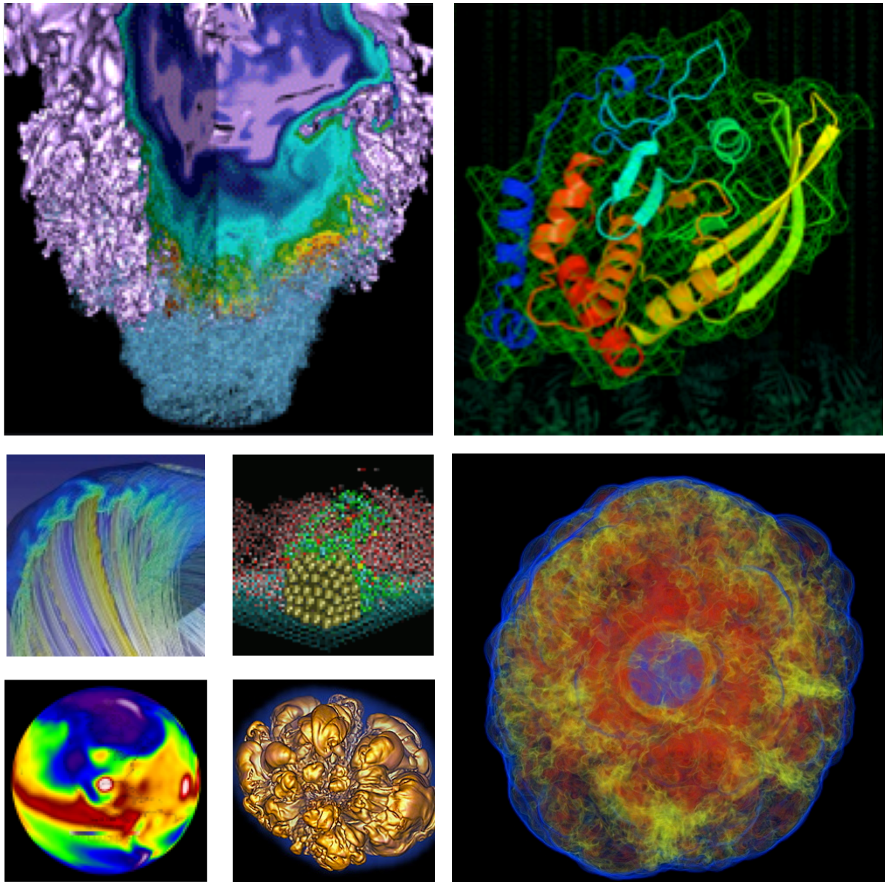
Appentra's
Parallelware Trainer

ECP/NERSC OpenMP Hackathon,
August 2019



Woo-Sun Yang
NERSC User Engagement Group

August 27, 2019

Intro to Parallelware Trainer

- **GUI tool developed by Appentra (https://www.appentra.com)**
- **It detects code segments that can be parallelized with OpenMP and OpenACC, and provides necessary directives**
  - Implementation of a reduction is provided in 3 flavors (built-in reduction, atomic operation & explicit privatization for the reduction var)
  - Offloading supported
  - Tasking directives provided but for loop-level parallelism
  - Deferred support for some parallel (sparse) scatter operations
  - Some notable features not being explored (e.g., loop collapsing)
  - Current support limited to C codes – C++ and Fortran support to come

Intro to Parallelware Trainer (cont'd)

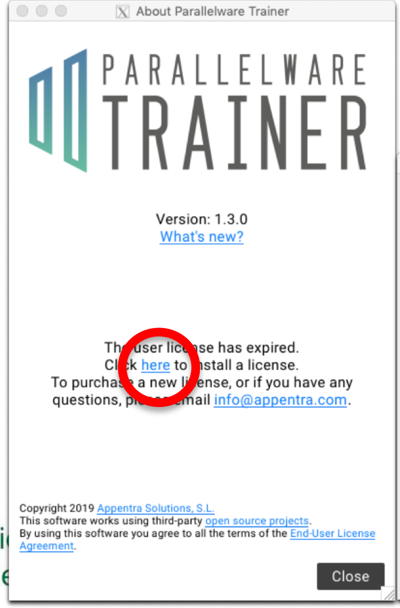
- **Based on static code pattern analysis**
  - Profiling not part of the tool workflow
  - Users need to profile with a profiler of own choice, to identify hot spots or to evaluate the resulting performance
  - Users are expected to work further for optimizations (memory use optimization, chunk scheduling, loop collapsing, etc.)
- **"Trainer"**
  - OpenMP and OpenACC beginners can learn about quick coding changes they can make in their codes

How to use Parallelware Trainer

- **Start an interactive job and start pwtrainer:**

```
$ salloc -N 1 -t 30 -C haswell -q debug
...
$ module load pwtrainer
$ pwtrainer
```

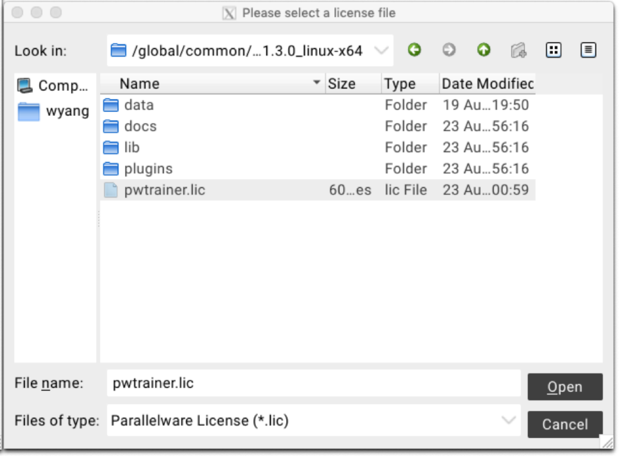
- **It will ask for a license file. Click the ‘here’ link in the pwtrainer window and select pwtrainer.lic in /global/common/cori_cle7/software/pwtrainer/pwtrainer-1.3.0_linux-x64 (this obviously tedious process may change in the future)**



Version: 1.3.0
What's new?

The user license has expired.
Click here to install a license.
To purchase a new license, or if you have any questions, please email info@appentra.com.

Copyright 2019 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.



Look in: /global/common/…1.3.0_linux-x64

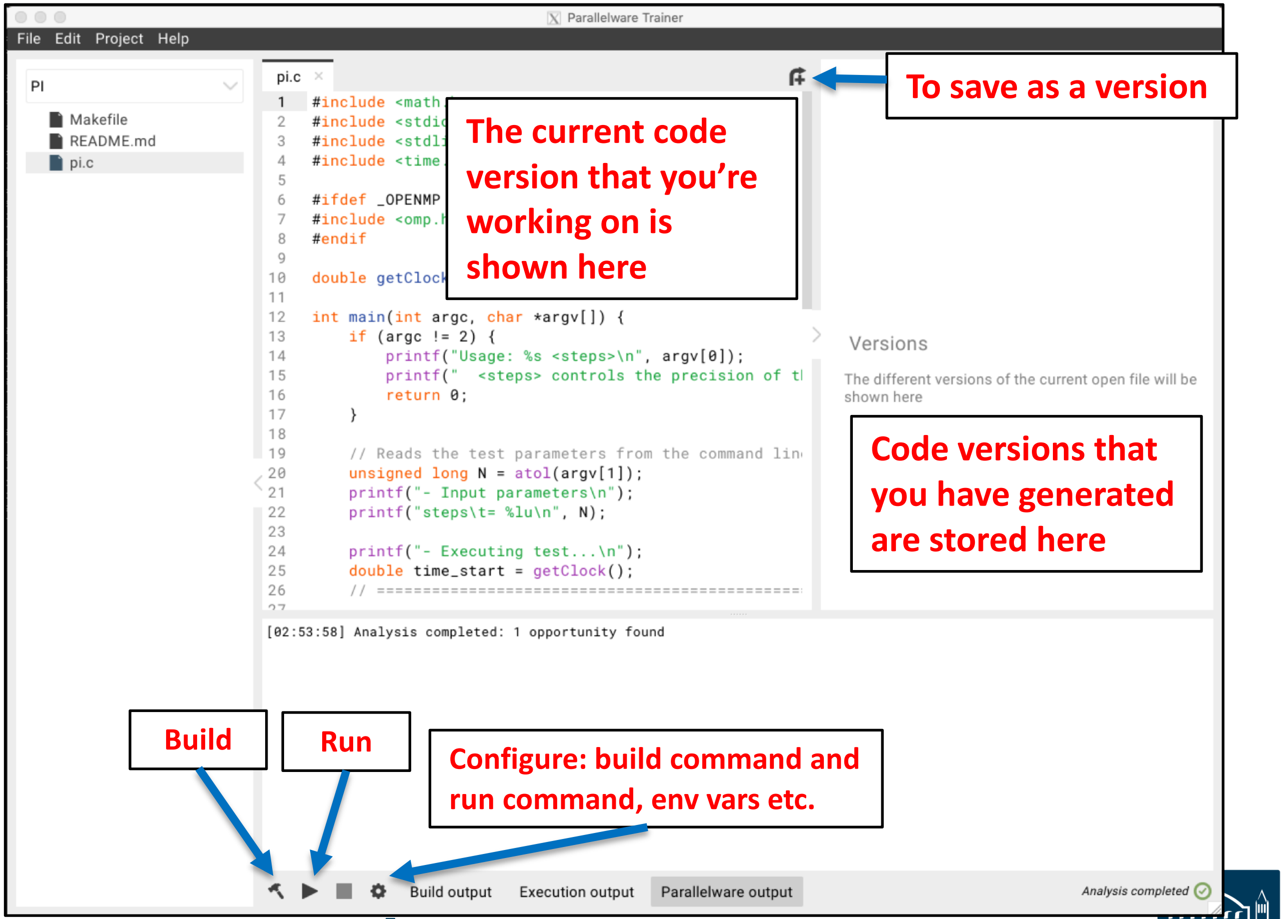
Name	Size	Type	Date Modifiec
data		Folder	19 Au…19:50
docs		Folder	23 Au…56:16
lib		Folder	23 Au…56:16
plugins		Folder	23 Au…56:16
pwtrainer.lic	60…es	lic File	23 Au…00:59

File name: pwtrainer.lic

Files of type: Parallelware License (*.lic)

How ot use Parallelware Trainer (cont'd)

- Open a new project (File → Open Project) and select a directory where you want to run pwtrainer for the codes there
- You will see



```
#include <math.
#include <stdic
#include <stdli
#include <time.

#ifdef _OPENMP
#include <omp.h
#endif

double getClock

int main(int argc, char *argv[]) {
    if (argc != 2) {
        printf("Usage: %s <steps>\n", argv[0]);
        printf("  <steps> controls the precision of tl
        return 0;
    }

    // Reads the test parameters from the command lin
    unsigned long N = atol(argv[1]);
    printf("- Input parameters\n");
    printf("steps\t= %lu\n", N);

    printf("- Executing test...\n");
    double time_start = getClock();
    // ==========================================
```

```
[02:53:58] Analysis completed: 1 opportunity found
```

The different versions of the current open file will be shown here

To save as a version

The current code version that you're working on is shown here

Code versions that you have generated are stored here

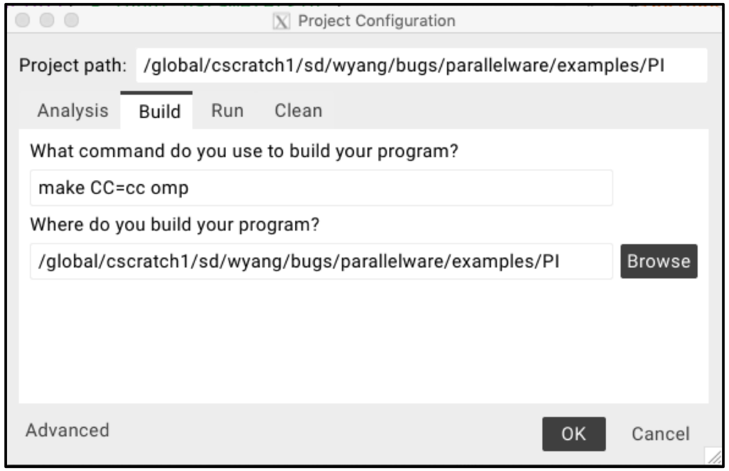
Build

Run

Configure: build command and run command, env vars etc.

How to use Parallelware Trainer (cont'd)

- **Set configuration for build, run and clean, etc.**



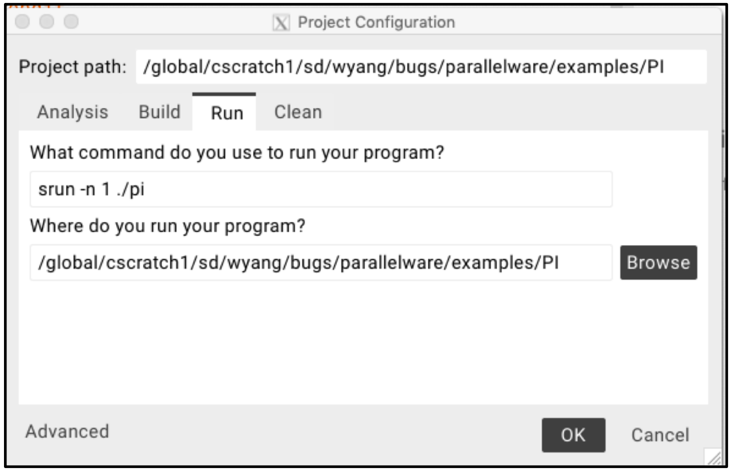
Project path: /global/cscratch1/sd/wyang/bugs/parallelware/examples/PI

What command do you use to build your program?

make CC=cc omp

Where do you build your program?

/global/cscratch1/sd/wyang/bugs/parallelware/examples/PI



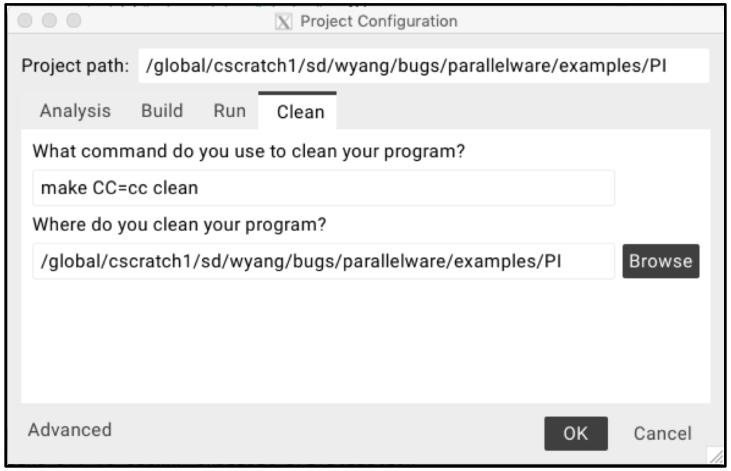
Project path: /global/cscratch1/sd/wyang/bugs/parallelware/examples/PI

What command do you use to run your program?

srun -n 1 ./pi

Where do you run your program?

/global/cscratch1/sd/wyang/bugs/parallelware/examples/PI



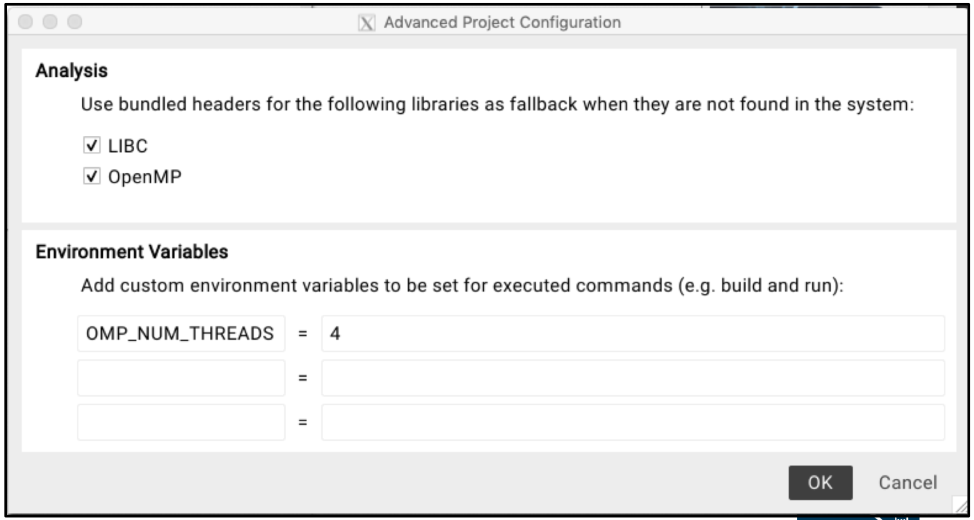
Project path: /global/cscratch1/sd/wyang/bugs/parallelware/examples/PI

What command do you use to clean your program?

make CC=cc clean

Where do you clean your program?

/global/cscratch1/sd/wyang/bugs/parallelware/examples/PI



Analysis

Use bundled headers for the following libraries as fallback when they are not found in the system:

☑ LIBC

☑ OpenMP

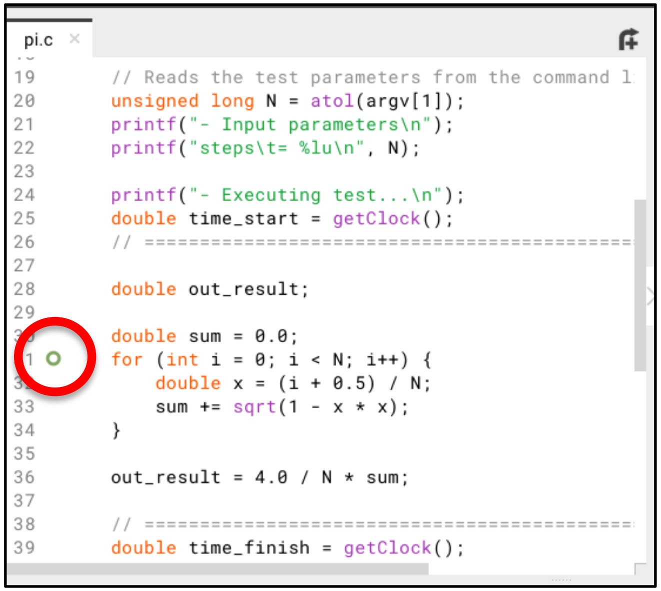
Environment Variables

Add custom environment variables to be set for executed commands (e.g. build and run):

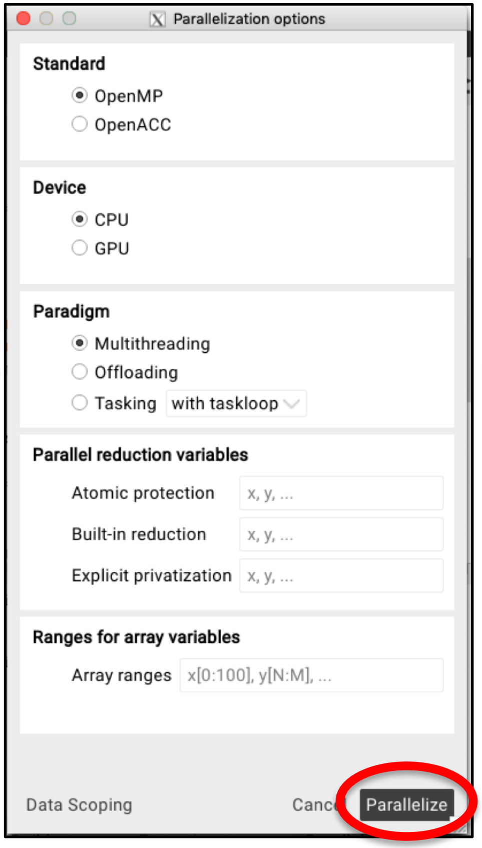
OMP_NUM_THREADS = 4

How to use Parallelware Trainer (cont'd)

- **Parallelize a code section that the tool has identified (green circle icon in the source pane)**
  - Click the icon and set parallelization option details



```
    // Reads the test parameters from the command l
    unsigned long N = atol(argv[1]);
    printf("- Input parameters\n");
    printf("steps\t= %lu\n", N);

    printf("- Executing test...\n");
    double time_start = getClock();
    // ============================================

    double out_result;

    double sum = 0.0;
    for (int i = 0; i < N; i++) {
        double x = (i + 0.5) / N;
        sum += sqrt(1 - x * x);
    }

    out_result = 4.0 / N * sum;

    // ============================================
    double time_finish = getClock();
```



Standard

● OpenMP
○ OpenACC

Device

● CPU
○ GPU

Paradigm

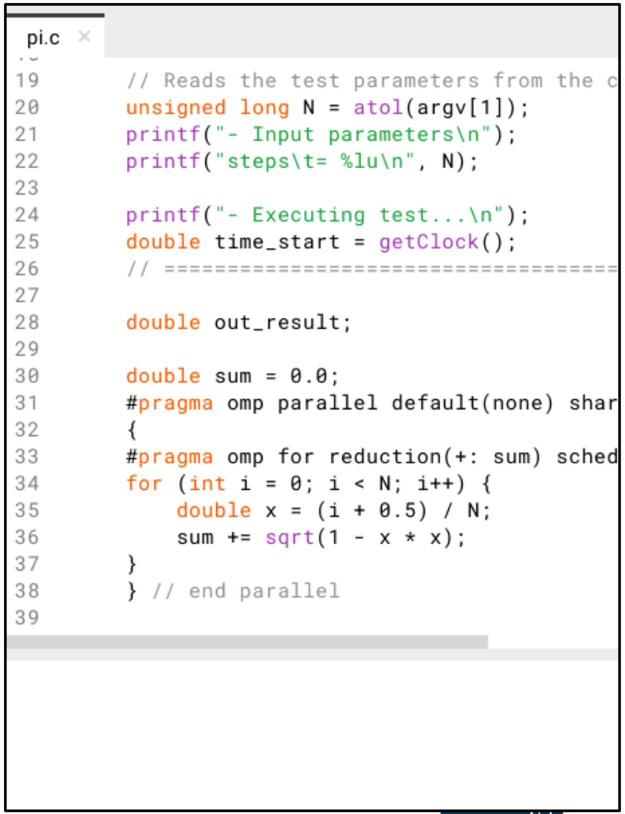
● Multithreading
○ Offloading
○ Tasking with taskloop

Parallel reduction variables

Atomic protection: x, y, ...
Built-in reduction: x, y, ...
Explicit privatization: x, y, ...

Ranges for array variables

Array ranges: x[0:100], y[N:M], ...



```
    // Reads the test parameters from the c
    unsigned long N = atol(argv[1]);
    printf("- Input parameters\n");
    printf("steps\t= %lu\n", N);

    printf("- Executing test...\n");
    double time_start = getClock();
    // ==================================

    double out_result;

    double sum = 0.0;
    #pragma omp parallel default(none) shar
    {
    #pragma omp for reduction(+: sum) sched
    for (int i = 0; i < N; i++) {
        double x = (i + 0.5) / N;
        sum += sqrt(1 - x * x);
    }
    } // end parallel
```

How to use Parallelware Trainer (cont'd)

- **You can edit in the source pane what the tool suggested**
- **If you are satisfied, click the build button at the bottom to build**
- **To run, click the run button**
- **You can save this version by clicking the up-right arrow at the top right corner of the current code pane**
  - This will create a version in the version pane
  - To bring a previous version to the current version, choose the tab for the corresponding version and click the up-left arrow
- **You repeat the above processes with different parallelization options (and offload option, too) and implementation options for your needs**

NERSC

National Energy Research Scientific Computing Center