TEXAS ADVANCED COMPUTING CENTER
WWW.TACC.UTEXAS.EDU

The University of Texas at Austin

# Lab: Tools for Parallelism Performance

Todd Evans

Feb 3rd, 2017

# Login

## Setup

1. Login to Stampede KNL

   ```
   $ ssh <username>@login-knl1.stampede.tacc.utexas.edu
   ```

2. Enter your password and token
3. Extract the lab if you don't have it

   ```
   $ tar -xf ~train00/knl_tools_training.tgz
   ```

4. start an idev session

   ```
   $ idev -m 60
   ```

5. Move into the directory

   ```
   $ cd knl_tools
   ```

# Ex 1: Load Imbalance Identification

## We will use VTune to identify a load imbalance we are familiar with

1. `module load vtune`
2. compile `omp_prime` with static scheduling
   - 2.1 run with `amplxe-cl` using 68 threads

     ```
     $ amplxe-cl --collect hotspots -r static ./omp_prime
     ```
3. open results in the `amplxe-gui` and inspect summary
   - 3.1 `open result` → `static` → `static.amplxe`
   - 3.2 is anything flagged?
4. compile `omp_prime` with dynamic scheduling
   - 4.1 run with `amplxe-cl`

     ```
     $ amplxe-cl --collect hotspots -r dynamic ./omp_prime
     ```
5. Open new result. What is flagged now?
   - 5.1 Does this make sense?
   - 5.2 What advice is given?
6. Modify the code to reflect the advice and rerun (hint try chunksize=100)

# Ex 2: Parallelize With Advisor

### We will parallelize a matrix multiplication program

1. `matrix.c` multiplies two square matrices $C = AB$
2. compile `matrix.c`

   ```
   $ icc matrix.c -o matrix -g
   ```

3. run it with advisor

   ```
   $ module load advisor
   $ advixe-cl -c survey --search-dir src:=./ ./matrix
   ```

4. run advisor gui and open and inspect results

   ```
   $ advixe-gui
   ```

   4.1 Where's the hotspot? (look under the survey button)

   4.2 What's the first recommendation?

# Ex 2: Parallelize With Advisor

## Vectorize and annotate

1. compile with vectorization `matrix.c`
   ```
   $ icc matrix.c -o matrix -xhost -g
   ```
2. rerun and observer performance
3. Now let’s predict OpenMP parallelization effects
4. We have to annotate the code for this
5. Where should annotation go based on granularity considerations?

# Ex 2: Parallelize With Advisor

## Annotate

```
#include ”advisor−annotate.h”
.
.
.
ANNOTATE_SITE_BEGIN(matrix);
for(int i=0;i<N;i++) {
  ANNOTATE_ITERATION_TASK(matrix_task);
  for(int j=0;j<N;j++) {
   double r = 0;
   for(int k=0;k<N;k++) {
    r += A[i][k]*B[k][j];
   }
   C[i][j] = r;
  }
}
ANNOTATE_SITE_END();
```

# Ex 2: Parallelize With Advisor

## Run suitability analysis

- compile
  ```
  $ icc matrix.c -o matrix -xhost -g
  -I$ADVISOR_2017_DIR/include
  ```
- run the suitability analysis
  ```
  $ advixe-cl -c suitability --search-dir src:=./
  ./matrix
  ```
- inspect results
- What is the predicted speed-up? It's shown under suitability report

# Ex 2: Parallelize With Advisor

## Parallelize code

1. add OpenMP pragma to `matrix.c`

   ```
   $ icc matrix.c -o debug_matrix -xhost -g
   ```

2. examine speedup at various thread counts