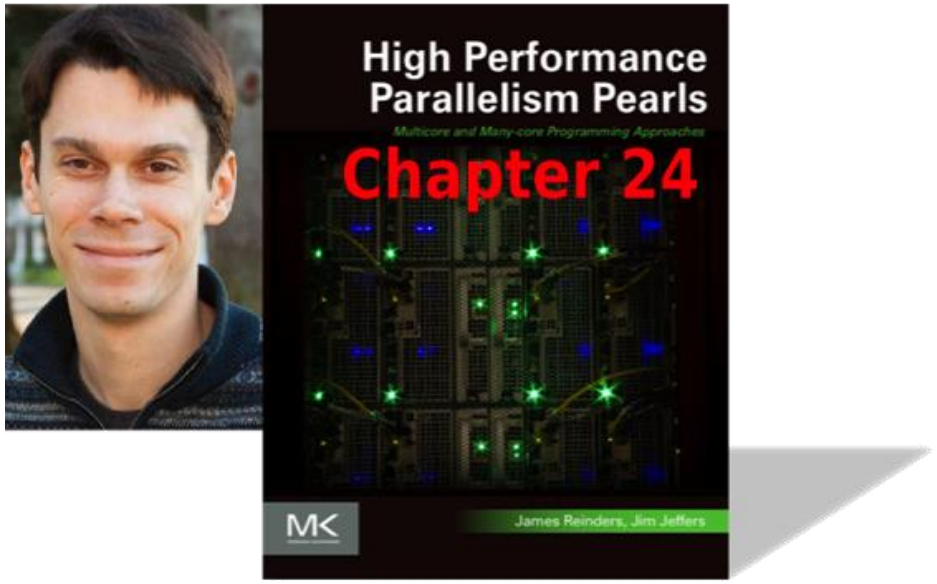
Hands-on “modernization of an application using Intel Advisor”

Matrix Transposition

- This Hands-on is based on chapter 24 (“Profiling-Guided Optimization”) - “High Performance Parallelism Pearls” book;

- This chapter exploits the use of Intel® VTune™ Amplifier XE reports to understand where to apply optimization on matrix transposition.



Agenda

- Speedup Estimation Analysis
  - Create Advisor Project
  - Collect Survey Data
  - Include Annotations
  - Collect Suitability Data

Agenda

- Speedup Estimation Analysis
  - Create Advisor Project
  - Collect Survey Data
  - Include Annotations
  - Collect Suitability Data
- ;

Intel Advisor

- Intel® Advisor XE is a shared memory threading designing and prototyping tool for C, C++, C# and Fortran.

- This tool provides a mechanism called annotation which is used to inform the regions of the code which can be parallelized. Based on such annotations a model is built in order to compare the performance scaling of different threading designs without the cost and disruption of implementation.

- In this part of Hands-on you will execute the following steps in order to perform threading prototyping:

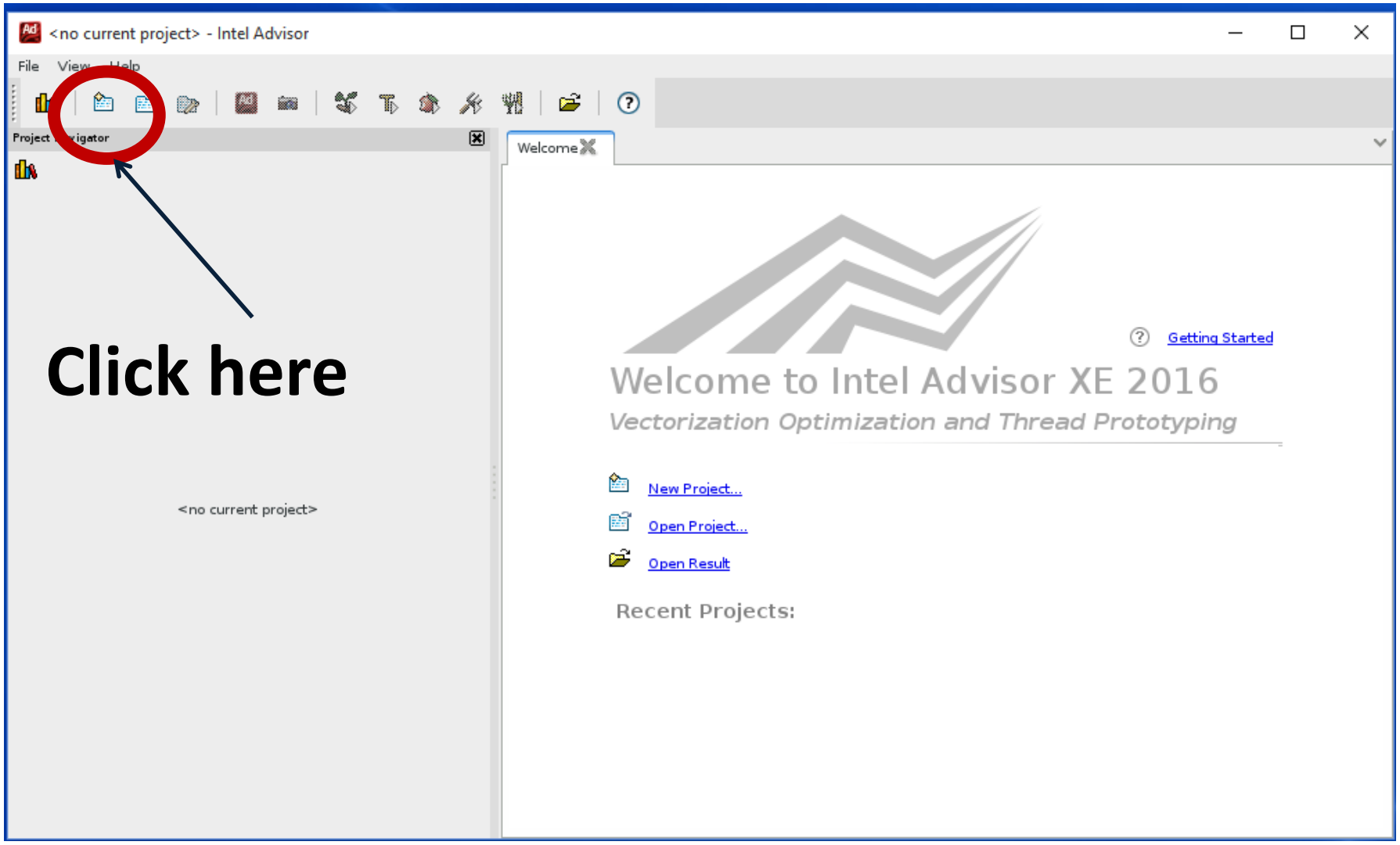
✓ Create project;
✓ Survey data: Discover opportunities for parallelization;
✓ Annotate sources: include annotations on source code to check scalability;
✓ Check suitability: evaluate the performance of annotated loops in different architectures and frameworks.

Agenda

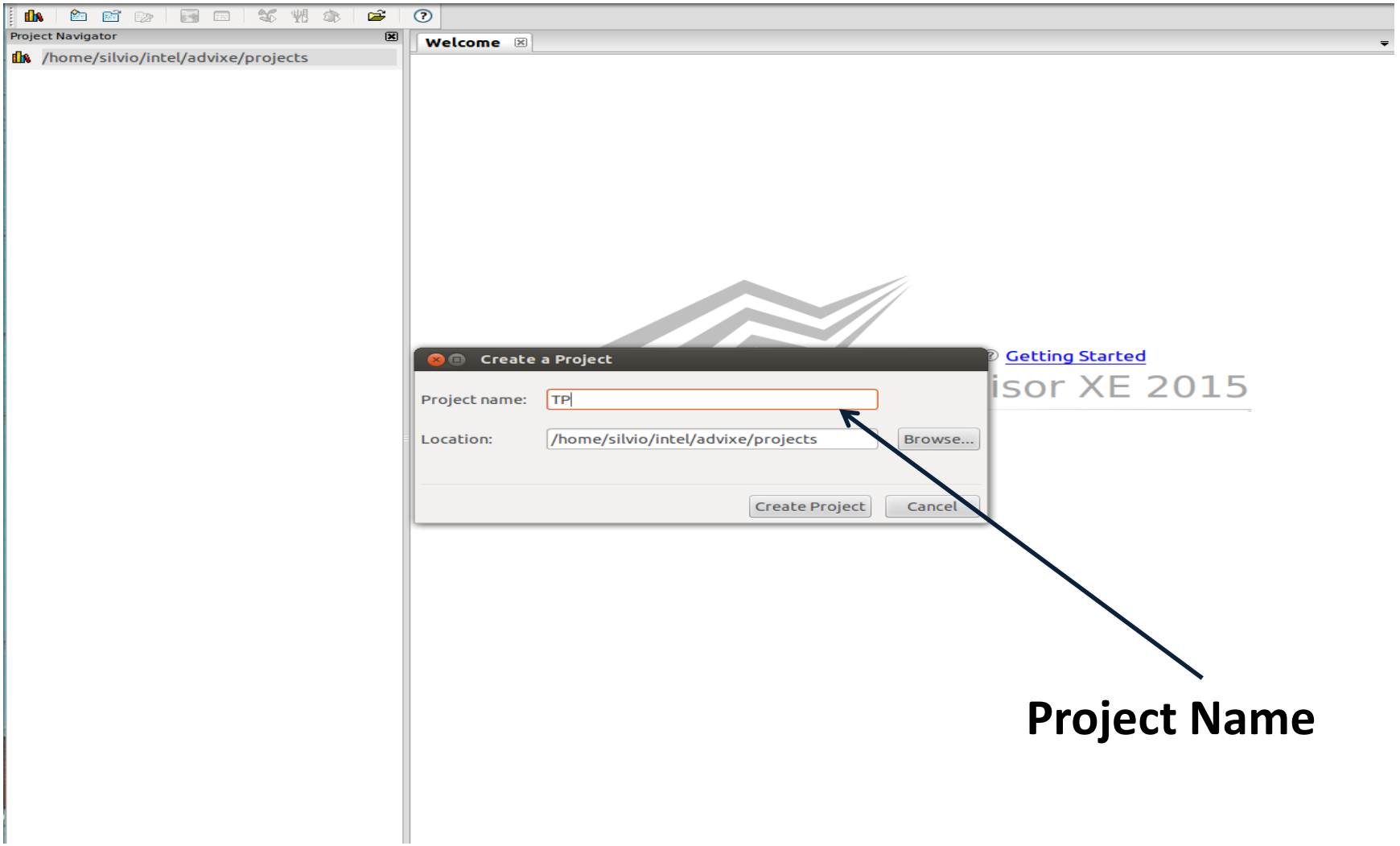
- Speedup Estimation Analysis
  - Create Advisor Project
  - Collect Survey Data
  - Include Annotations
  - Collect Suitability Data

Intel Advisor – Create New Project

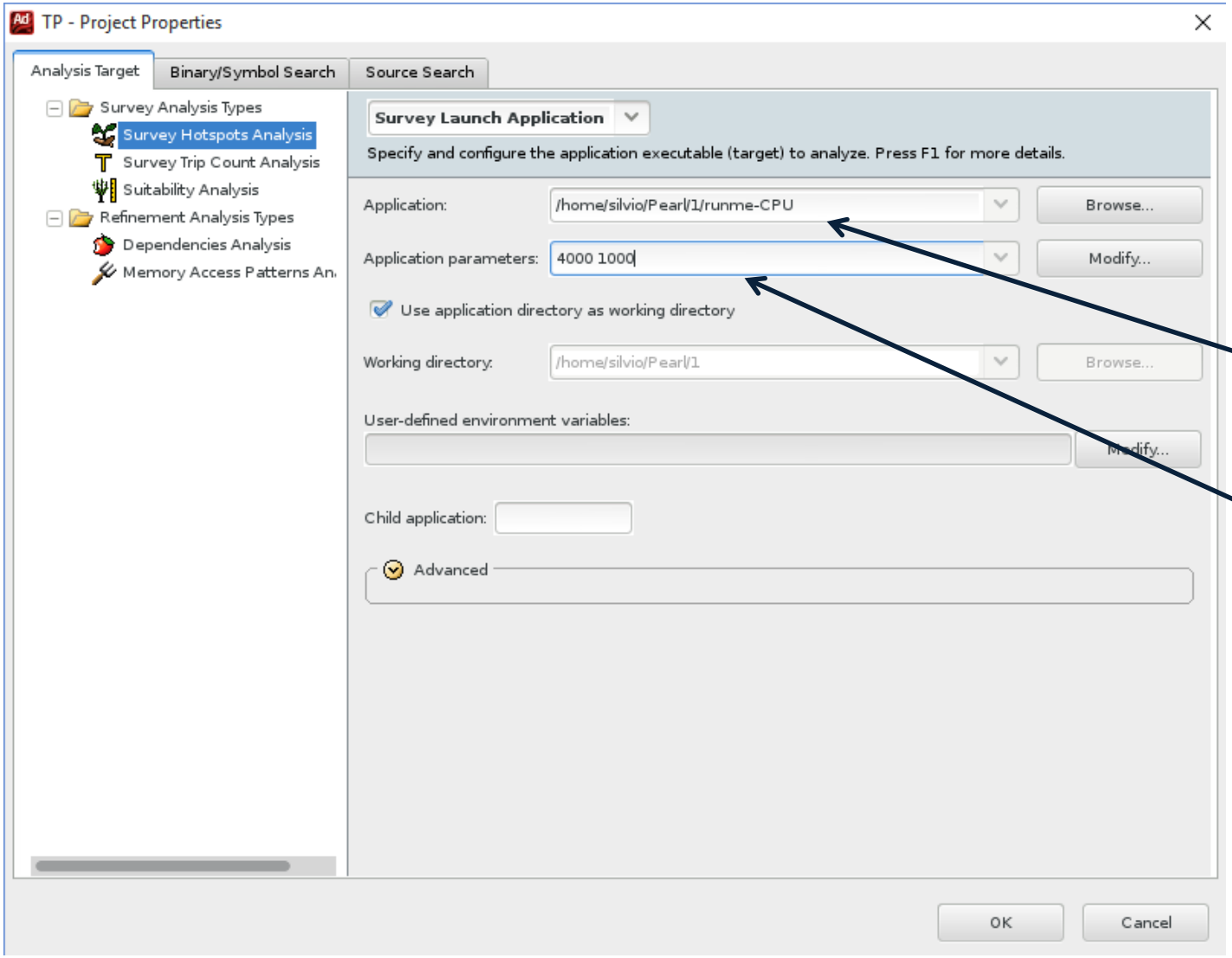
- Execute Intel Advisor on terminal: advixe-gui
- create new Advisor project:
  - name: TP
  - application: ~/hands-on/Pearl/1/runme-CPU
  - application parameters: 3000 100
  - Source Folder: ~/hands-on/Pearl/1/


<no current project> - Intel Advisor
File View Help
Project Navigator
Welcome
Click here
<no current project>
Getting Started
Welcome to Intel Advisor XE 2016
Vectorization Optimization and Thread Prototyping
New Project...
Open Project...
Open Result
Recent Projects:

Intel Advisor – Create New Project



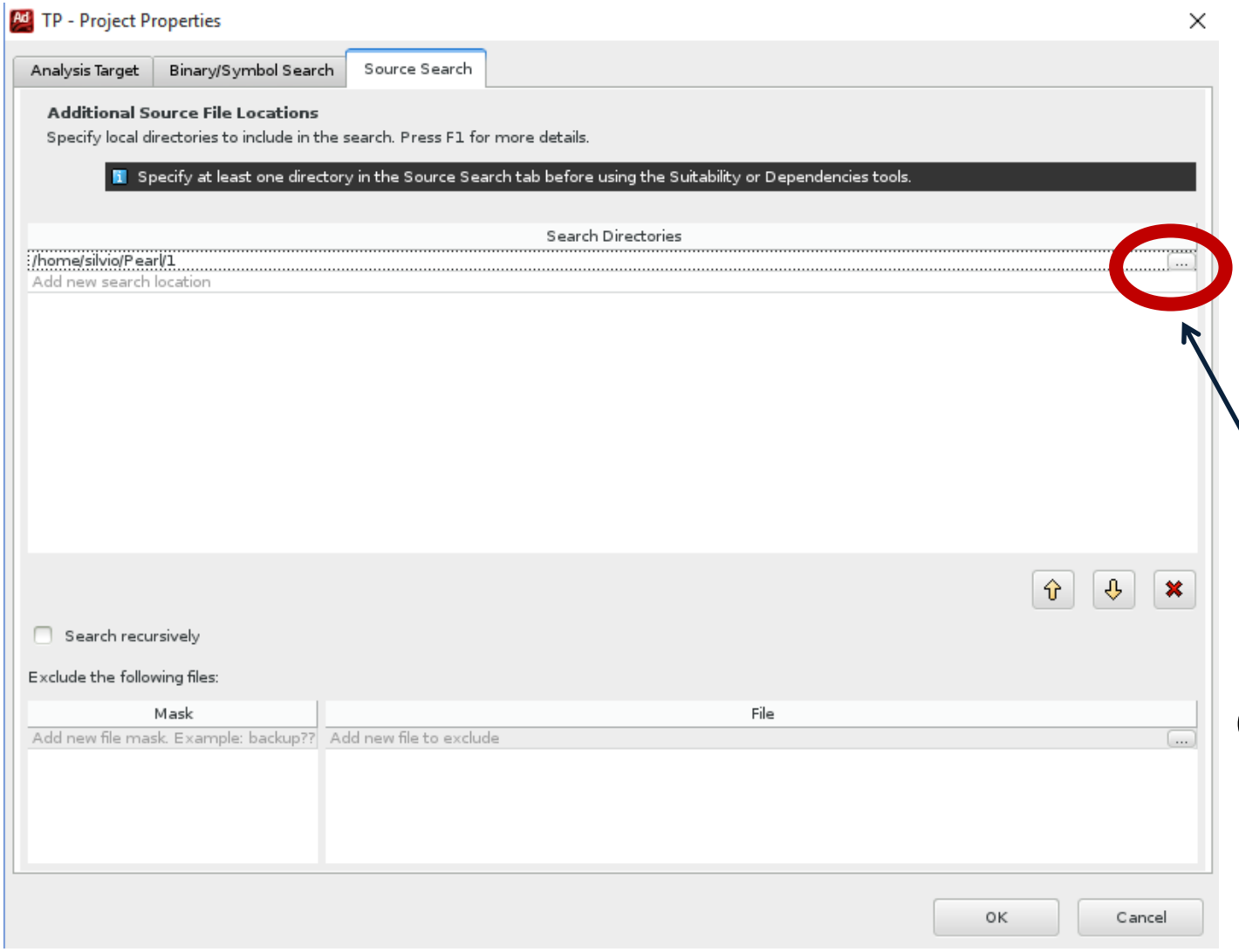
Intel Advisor – Create New Project



Application

Application Parameters

Intel Advisor – Source Search

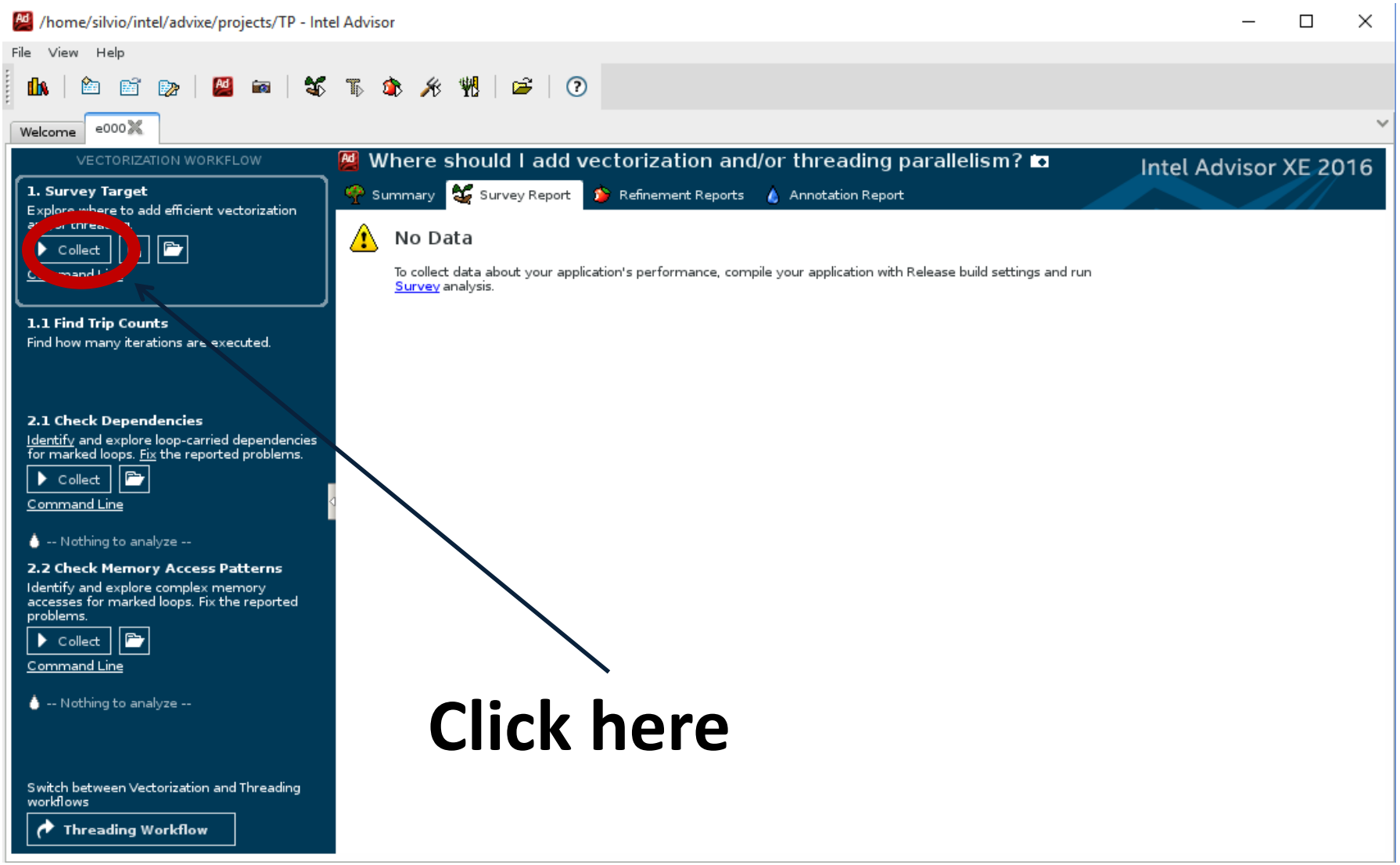


Click here

Agenda

- Speedup Estimation Analysis
  - Create Advisor Project
  - Collect Survey Data
  - Include Annotations
  - Collect Suitability Data

Intel Advisor - collect survey data



Intel Advisor – Compile Application

- After create advisor project recompile application:
  - Cd ~/hands-on/Pearl/1/
  - rm –rf runme-CPU
  - Make

Intel Advisor - survey data results

Higher instruction set architecture (ISA) available

Consider recompiling your application using a higher ISA.

Loops	Vector Issues	Self Time	Total Time	Loop Type	Why No Vectorization?	Vectorized Loops				Instruction Set Analysis		Advanced	Location
						Vector ISA	Gain Estimate	VL (Vector Length)	Compiler Estimated Gain	Traits	Data Types		
[loop in Transpose at Transpose.cc:20]		40.470s	40.470s	Vectorized (B...		SSE	3.43x	2; 4	3.43x		Float32; Float64	Unrolled by 4	Transpose.cc:
[loop in Transpose at Transpose.cc:20]		40.470s	40.470s	Vectorized (B...		SSE		2; 4	3.43x		Float32; Float64	Unrolled by 4	Transpose.cc:
[loop in __kmp_launch_thread at kmp_runtime.c:5900]		24.499s	24.820s	Scalar									kmp_runtime
[loop in VerifyTransposed at Main.cc:24]	1 Data type conversions present	0.171s	0.171s	Vectorized (B...		SSE2	2.73x	2; 4	2.73x	Type Conv...	Float32; Float64; Int32	Unrolled by 4	Main.cc:24
[loop in VerifyTransposed at Main.cc:24]	1 Data type conversions present	0.171s	0.171s	Vectorized (B...		SSE2		2; 4	2.73x	Type Conv...	Float32; Float64; Int32	Unrolled by 4	Main.cc:24
[loop in Transpose at Transpose.cc:20]	**1 Ineffective peeled/remainder loop ...**	**0.080s**	**0.080s**	**Vectorized ...**		**SSE**	**1.86x**	**2; 4**	**1.86x**		**Float32; Float64**		**Transpose.**
[loop in Transpose at Transpose.cc:20]		0.050s	0.050s	Vectorized (R...		SSE		2; 4	1.86x		Float32; Float64		Transpose.cc:
[loop in Transpose at Transpose.cc:20]		0.030s	0.030s	Remainder									Transpose.cc:
[loop in Transpose at Transpose.cc:17]		0.030s	40.500s	Scalar	inner loop was al...								Transpose.cc:
[loop in start_thread]		0.000s	24.820s	Scalar									
[loop in __libc_start_main]		0.000s	40.670s	Scalar									
[loop in main at Main.cc:74]		0.000s	40.600s	Scalar	loop with multipl...						Float32; Float64		Main.cc:74
[loop in VerifyTransposed at Main.cc:23]		0.000s	0.171s	Scalar						Unpacks	Float32; Float64; Int32; Int64		Main.cc:23
[loop in [OpenMP worker] at z_Linux_util.c:786]		0.000s	24.820s	Scalar									z_Linux_util.c

Source | Top Down | Loop Assembly | Recommendations | Compiler Diagnostic Details

File: Transpose.cc:20 Transpose

Line	Source	Total Time	%	Loop Time	%	Traits
7	// You are free to use, modify and distribute this code as long as you acknowledge					
8	// the above mentioned publication.					
9	// (c) Colfax International, 2013					
10						
11	#include "Transpose.h"					
12	#include <cstdlib>					
13						
14						
15	void Transpose(FTYPE* const A, const int n) {					
16						
17	⊞ for (int i = 0; i < n; i++) {			40.499s		
18						
19						
20	⊞ for (int i = 0; i < i; i++) {	0.280s		40.550s		
21						
22	const FTYPE c = A[i*n + j];	29.950s				
23	A[i*n + j] = A[j*n + i];	10.180s				
24	A[j*n + i] = c;	0.190s				
25	}					
26						
27	}					
28						
29	}					
30						

Intel Advisor - survey data results

Loops	Vector Issues	Self Time▾	Total Time	Loop Type	Why No Vectorization?	Vectorized Loops				Instruction Set Analysis		Advanced	Location
						Vector ISA	Gain Estimate	VL (Vector Length)	Compiler Estimated Gain	Traits	Data Types		
[loop in Transpose at Transpose.cc:20]		40.470s	40.470s	Vectorized (B...		SSE	3.43x	2; 4	3.43x		Float32; Float64	Unrolled by 4	Transpose.cc:
[loop in Transpose at Transpose.cc:20]		40.470s	40.470s	Vectorized (B...		SSE		2; 4	3.43x		Float32; Float64	Unrolled by 4	Transpose.cc:
[loop in __kmp_launch_thread at kmp_runtime.c:5900]		24.499s	24.820s	Scalar									kmp_runtime
[loop in VerifyTransposed at Main.cc:24]	1 Data type conversions present	0.171s	0.171s	Vectorized (B...		SSE2	2.73x	2; 4	2.73x	Type Conv...	Float32; Float64; Int32	Unrolled by 4	Main.cc:24
[loop in VerifyTransposed at Main.cc:24]	1 Data type conversions present	0.171s	0.171s	Vectorized (B...		SSE2		2; 4	2.73x	Type Conv...	Float32; Float64; Int32	Unrolled by 4	Main.cc:24
[loop in Transpose at Transpose.cc:20]	**1 Ineffective peeled/remainder loop ...**	**0.080s**	**0.080s**	**Vectorized ...**		**SSE**	**1.86x**	**2; 4**	**1.86x**		**Float32; Float64**		**Transpose.**
[loop in Transpose at Transpose.cc:20]		0.050s	0.050s	Vectorized (R...		SSE		2; 4	1.86x		Float32; Float64		Transpose.cc:
[loop in Transpose at Transpose.cc:20]		0.030s	0.030s	Remainder									Transpose.cc:
[loop in Transpose at Transpose.cc:17]		0.030s	40.500s	Scalar	inner loop was al...								Transpose.cc:
[loop in start_thread]		0.000s	24.820s	Scalar									
[loop in __libc_start_main]		0.000s	40.670s	Scalar									
[loop in main at Main.cc:74]		0.000s	40.600s	Scalar	loop with multipl...						Float32; Float64		Main.cc:74
[loop in VerifyTransposed at Main.cc:23]		0.000s	0.171s	Scalar						Unpacks	Float32; Float64; Int32; Int64		Main.cc:23
[loop in [OpenMP worker] at z_Linux_util.c:786]		0.000s	24.820s	Scalar									z_Linux_util.c

Source | Top Down | Loop Assembly | Recommendations | Compiler Diagnostic Details

Function Call Sites and Loops	Total Time %	Total Time	Self Time	Loop Type	Why No Vectorization?	Vectoriz...		Instruction Set Analysis		Advanced	Location
						Vec...	VL ...	Traits	Data Types		
Total	100.0%	65.490s	0s								
__libc_start_main	62.1%	40.670s	0s								
[loop in __libc_start_main]	62.1%	40.670s	0s	Scalar							
[Unknown stack frame(s)]	62.0%	40.600s	0s								
main	62.0%	40.600s	0s								Main.cc:38
[loop in main at Main.cc:74]	62.0%	40.600s	0s	Scalar	loop with multiple exits cannot be vectorized unless it meets search loop idiom criteria				Float32; Float64		Main.cc:74
Transpose	62.0%	40.600s	0.0200s								Transpose.cc:15
[loop in Transpose at Transpose.cc:17]	61.8%	40.500s	0.0300s	Scalar	inner loop was already vectorized						Transpose.cc:17
[loop in Transpose at Transpose.cc:20]	61.8%	40.470s	0s								Transpose.cc:20
[loop in Transpose at Transpose.cc:20]	61.8%	40.470s	40.4696s	Vectorized (Body)		SSE	2; 4		Float32; Float64	Unrolled by 4	Transpose.cc:20
[loop in Transpose at Transpose.cc:20]	0.1%	0.080s	0s								Transpose.cc:20
[loop in Transpose at Transpose.cc:20]	0.1%	0.050s	0.0500s	Vectorized (Remainder)		SSE	2; 4		Float32; Float64		Transpose.cc:20
[loop in Transpose at Transpose.cc:20]	0.0%	0.030s	0.0300s	Remainder							Transpose.cc:20
main	0.1%	0.070s	0s								Main.cc:38
__clone	37.9%	24.820s	0s								

Intel Advisor - survey data results

- What lines are the hot spots in this code?

Agenda

- Speedup Estimation Analysis
  - Create Advisor Project
  - Collect Survey Data
  - Include Annotations
  - Collect Suitability Data

Intel Advisor – Include Annotations

- Include the annotations on “~/hands-on/Pearl/1/transpose.cc” in the following way:
  - **#include "advisor-annotate.h“**: include header file
  - **ANNOTATE_SITE_BEGIN(id)**: before beginning of loop;
  - **ANNOTATE_ITERATION_TASK(id)**: first line inside the loop;
  - **ANNOTATE_SITE_END()**: after end of loop;

Intel Advisor – Include Annotations

```
#include "Transpose.h"
#include <cstdlib>


void Transpose(FTYPE* const A, const int n) {

 for (int j = 0; j < n; j++) {


  for (int i = 0; i < j; i++) {

     const FTYPE c = A[i*n + j];
     A[i*n + j] = A[j*n + i];
     A[j*n + i] = c;
    }

  }

}
```

```
#include "Transpose.h"
#include <cstdlib>
#include "advisor-annotate.h"

void Transpose(FTYPE* const A, const int n) {
  ANNOTATE_SITE_BEGIN( MySite1 );
  for (int j = 0; j < n; j++) {
    ANNOTATE_ITERATION_TASK( MyTask1 );
    ANNOTATE_SITE_BEGIN( MySite2 );
    for (int i = 0; i < j; i++) {
      ANNOTATE_ITERATION_TASK( MyTask2 );
      const FTYPE c = A[i*n + j];
      A[i*n + j] = A[j*n + i];
      A[j*n + i] = c;
    }
    ANNOTATE_SITE_END();
  }
  ANNOTATE_SITE_END();
}
```

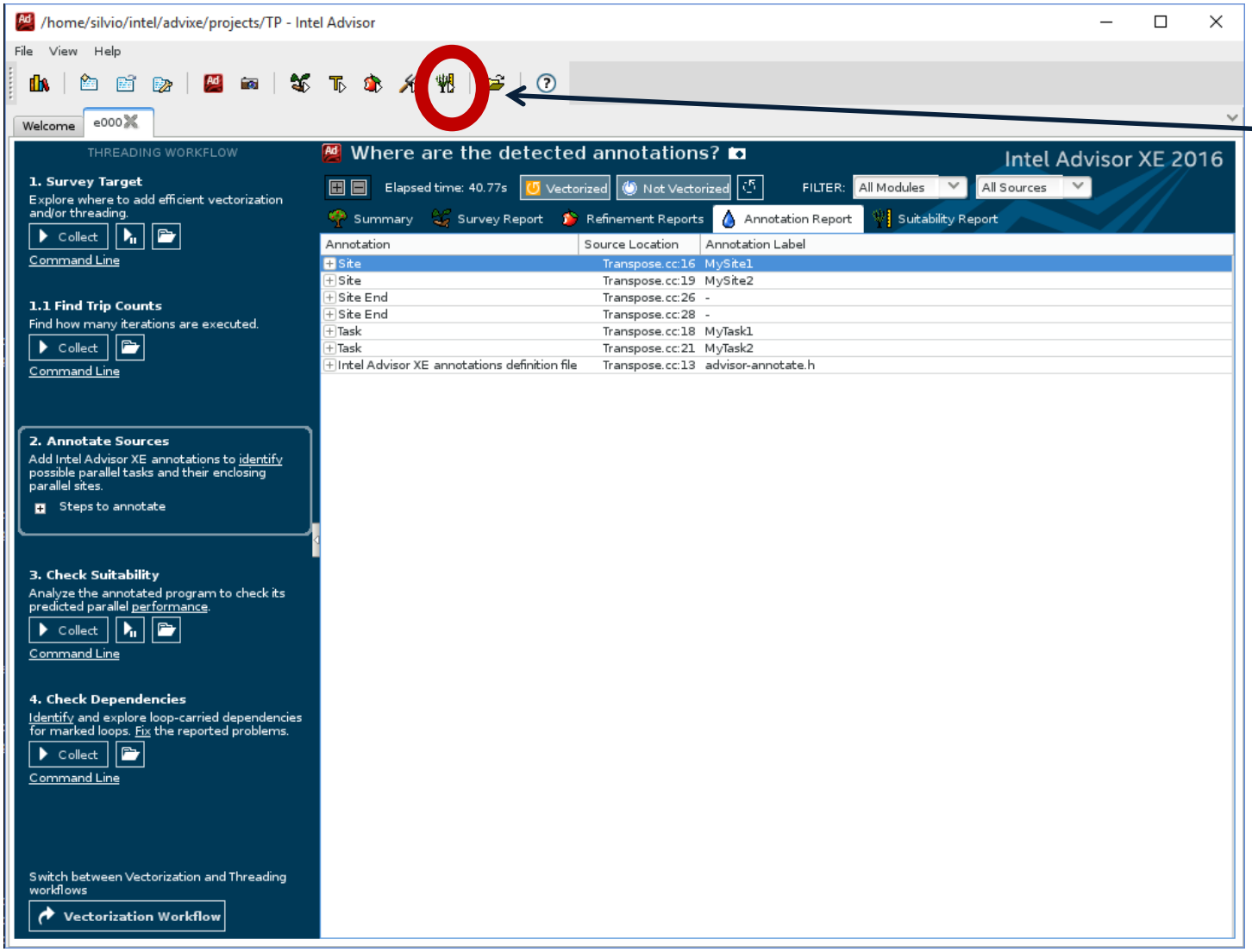
Intel Advisor – Include Annotations

- Red Lines shows the lines that has to be included in original source code;
- After include these lines recompile application:
  - Cd ~/hands-on/Pearl/1/
  - rm –rf runme-CPU
  - Make
- Check your annotations on “view annotations” options;

Agenda

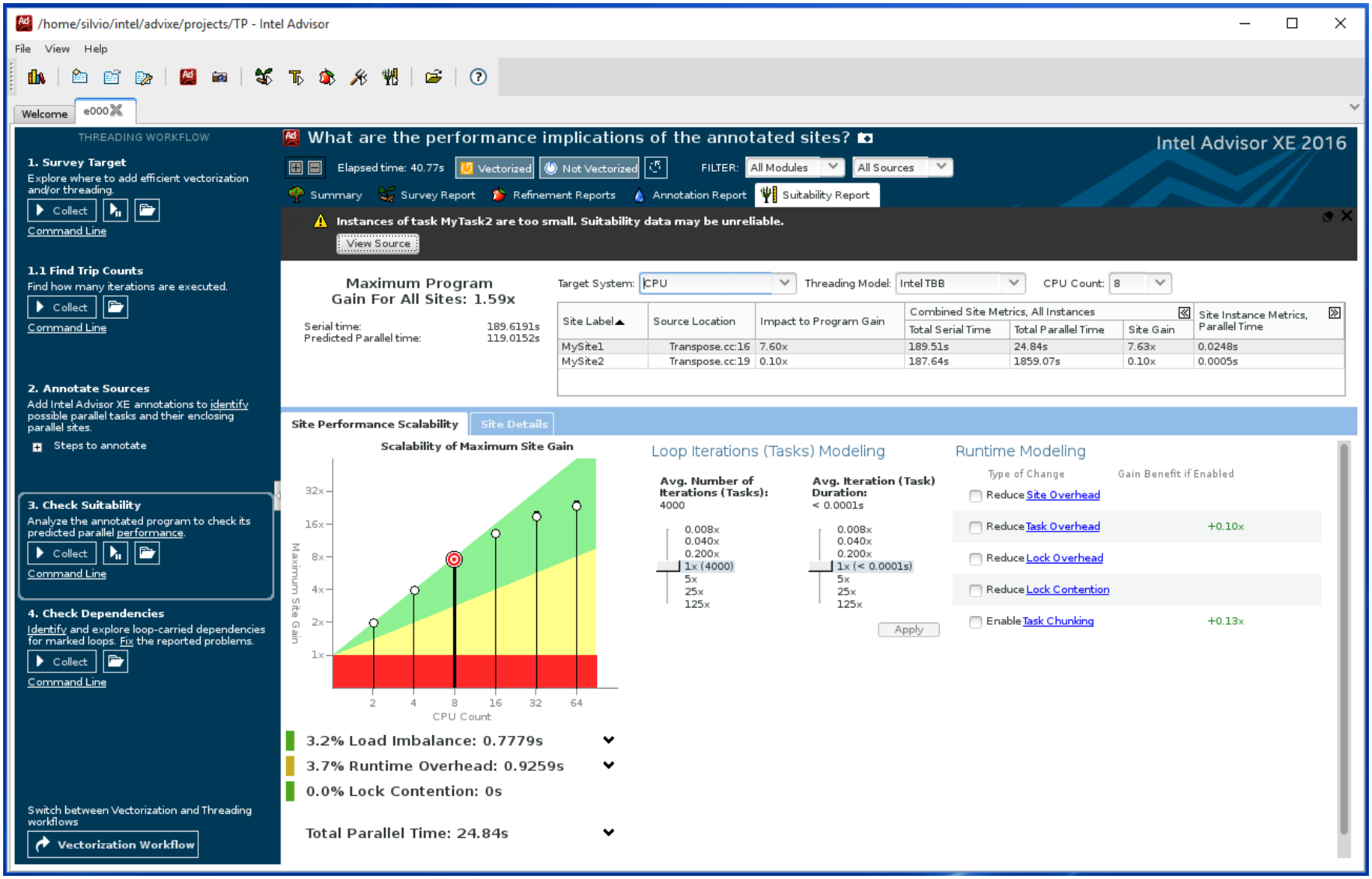
- Speedup Estimation Analysis
  - Create Advisor Project
  - Collect Survey Data
  - Include Annotations
  - Collect Suitability Data

Intel Advisor - collect suitability data

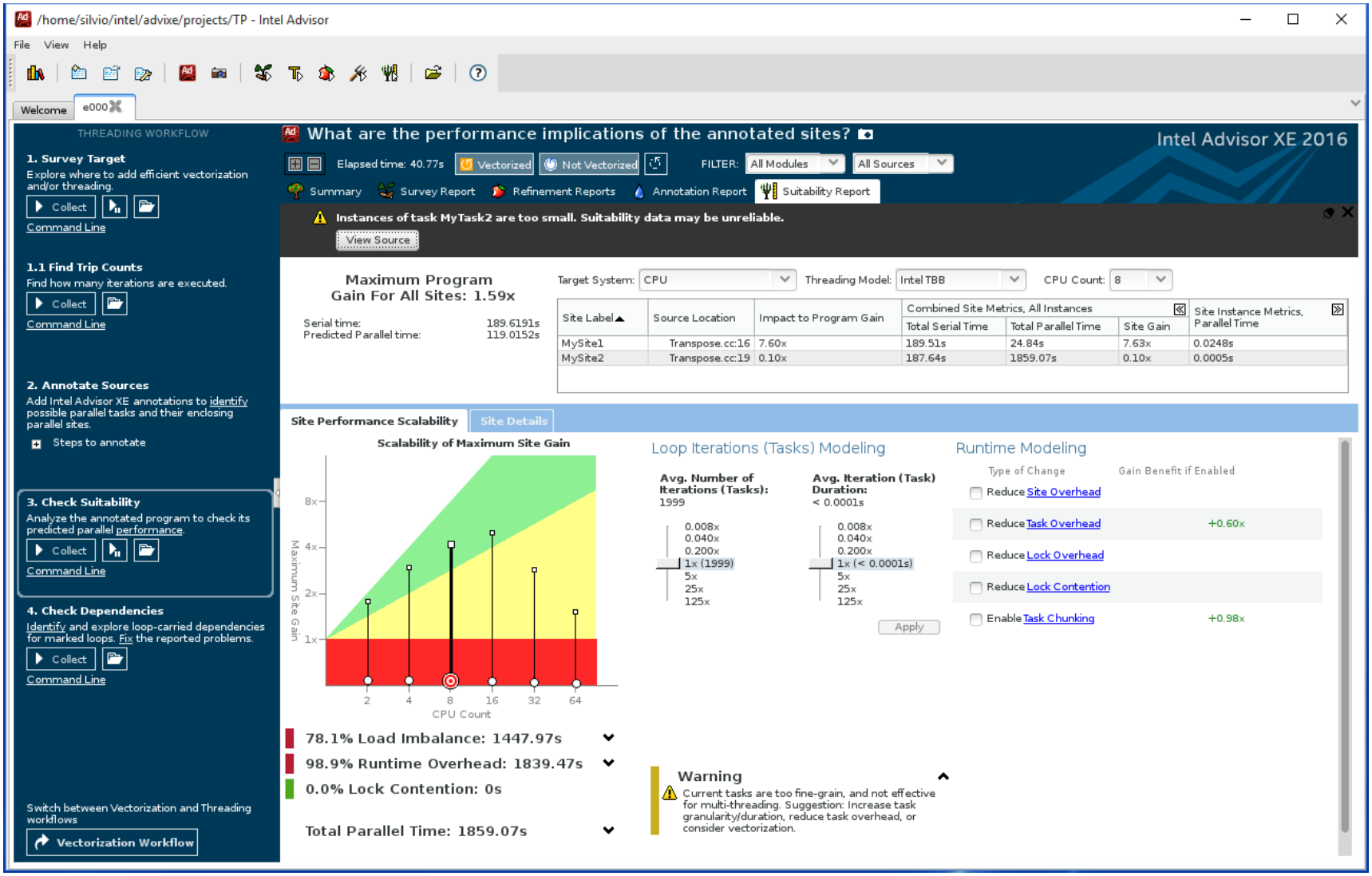


Click here

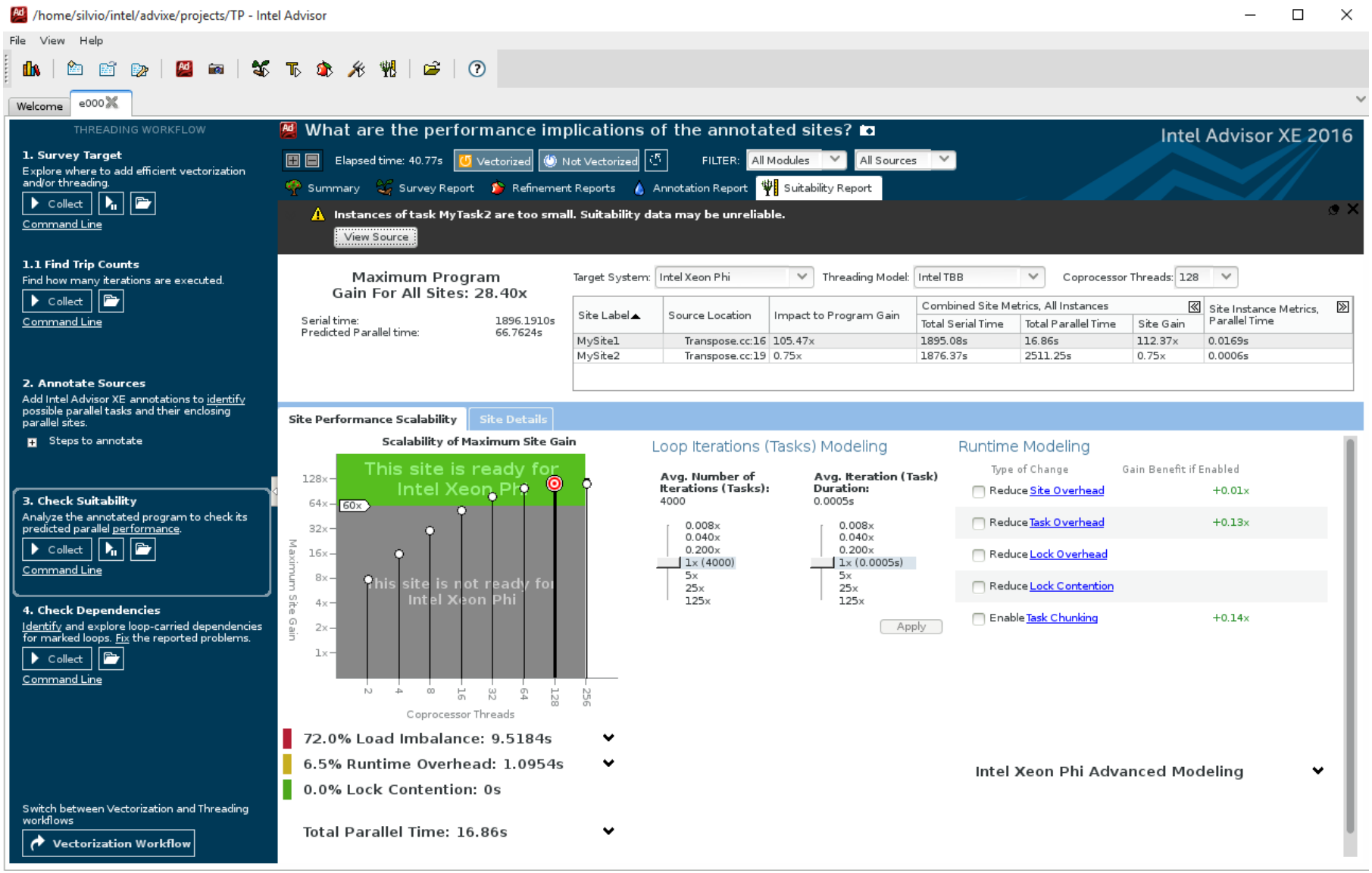
Check Suitability Expected Results (Intel Xeon) Site 1



Check Suitability Expected Results (Intel Xeon) Site 2



Check Suitability Expected Results (Intel Xeon Phi) Site 1



Check Suitability Expected Results (Intel Xeon Phi) Site 2