Intel® Advisor XE 2013

Threading Prototyping Tool

Code the Future

Additional Material

Intel® Advisor XE - Threading Prototyping Tool for Architects

Intel Advisor XE:

- Product page – overview, features, FAQs…
- Training materials – movies, tech briefs, documentation…
- Evaluation guides – step by step walk through
- Case studies
- Support – forums, secure support…

More Analysis Tools:

- Intel® Inspector XE - memory and thread checker / debugger
- Intel® VTune™ Amplifier XE – performance profiler

Intel Software Development Products

Data Driven Threading Design

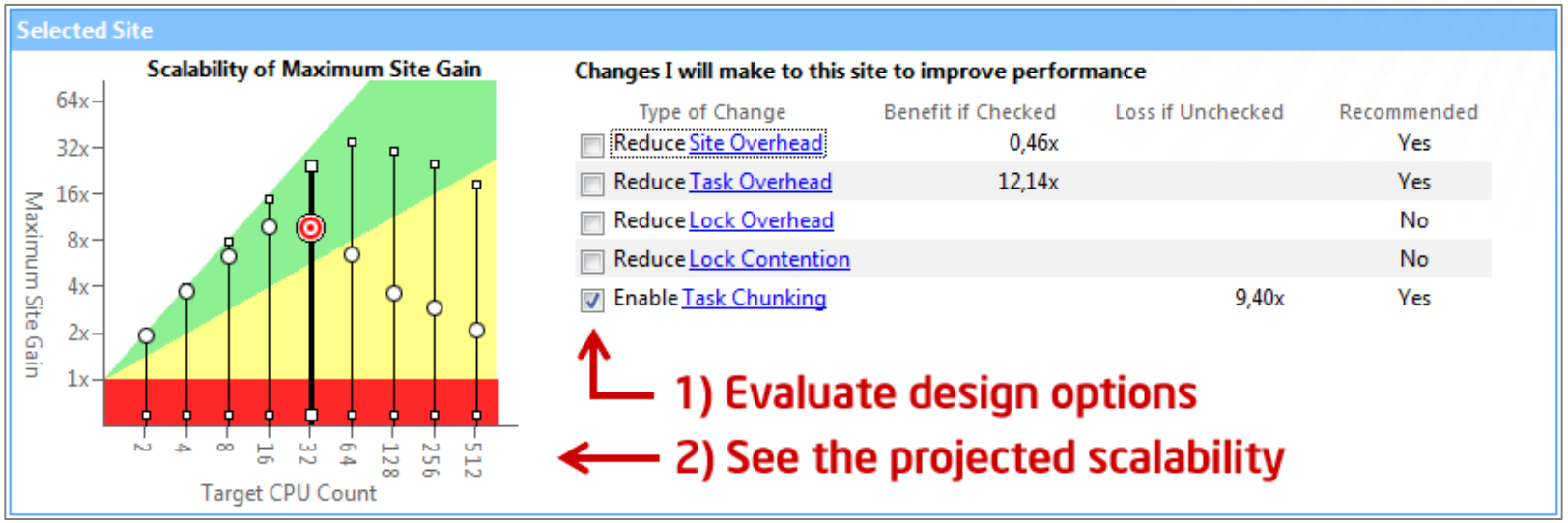
Intel® Advisor XE – Threading Prototyping Tool for Architects

Have you:

- Tried threading an app, but seen little performance benefit?
- Hit a "scalability barrier"? Performance gains level off as you add cores?
- Delayed a release that adds threading because of synchronization errors?

Breakthrough for threading design:

- Quickly prototype multiple options
- Project scaling on larger systems
- Find synchronization errors before implementing threading
- Separate design and implementation, design without disrupting development



Add Parallelism with Less Effort, Less Risk and More Impact

Design Then Implement

Intel® Advisor XE 2013 – Threading Assistant

Design Parallelism

- No disruption to regular development
- All test cases continue to work
- Tune and debug the design before you implement it

1) Analyze it.

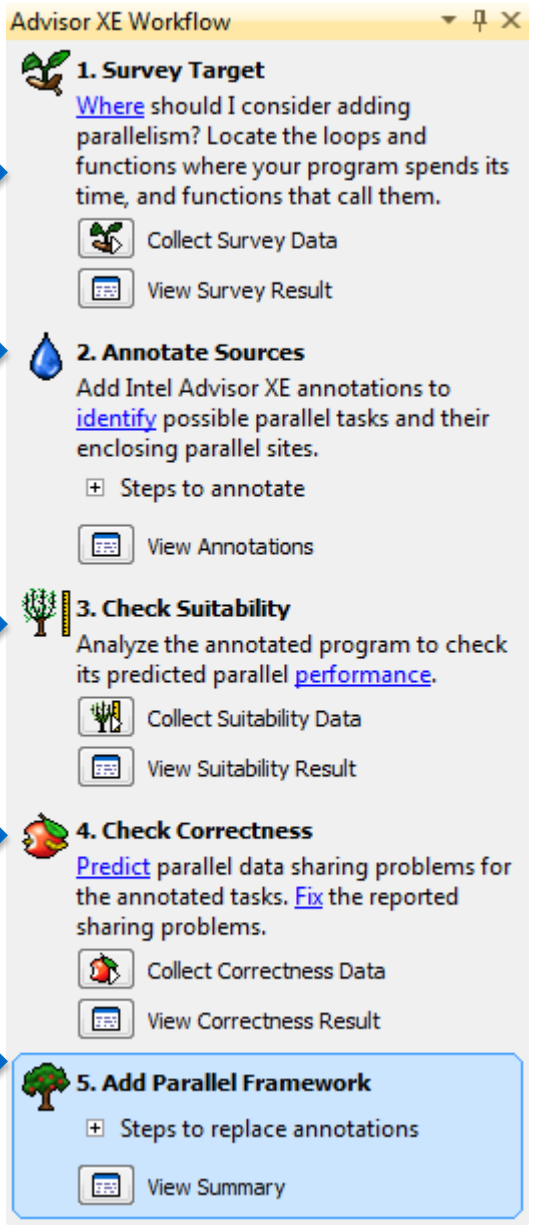
2) Design it.
(Compiler ignores these annotations.)

3) Tune it.

4) Check it.

Implement Parallelism

5) Do it!



Less Effort, Less Risk, More Impact

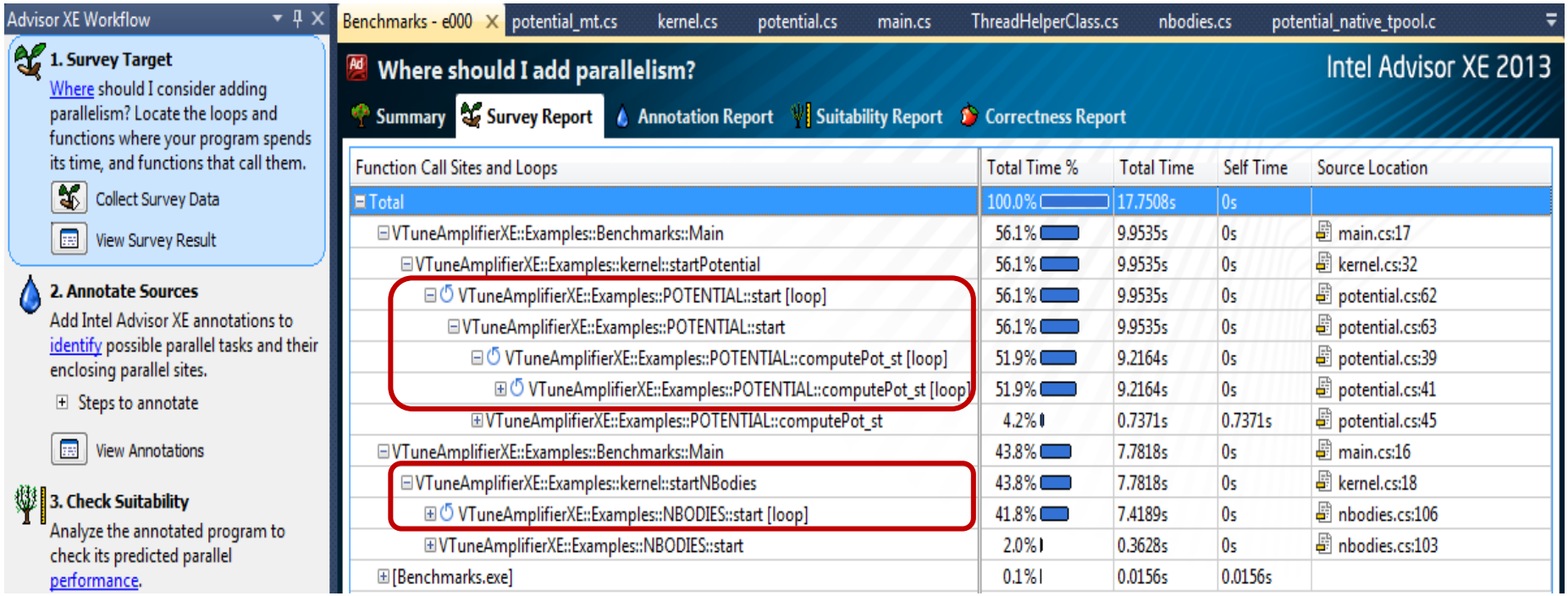
Amdahl's Law

(paraphrased) “The benefit from parallelism is limited by the computation which remains serial”

If you perfectly execute ½ of your application in parallel you will achieve < 2x speedup

The implication of this is that you must focus your attention where your application spends its time

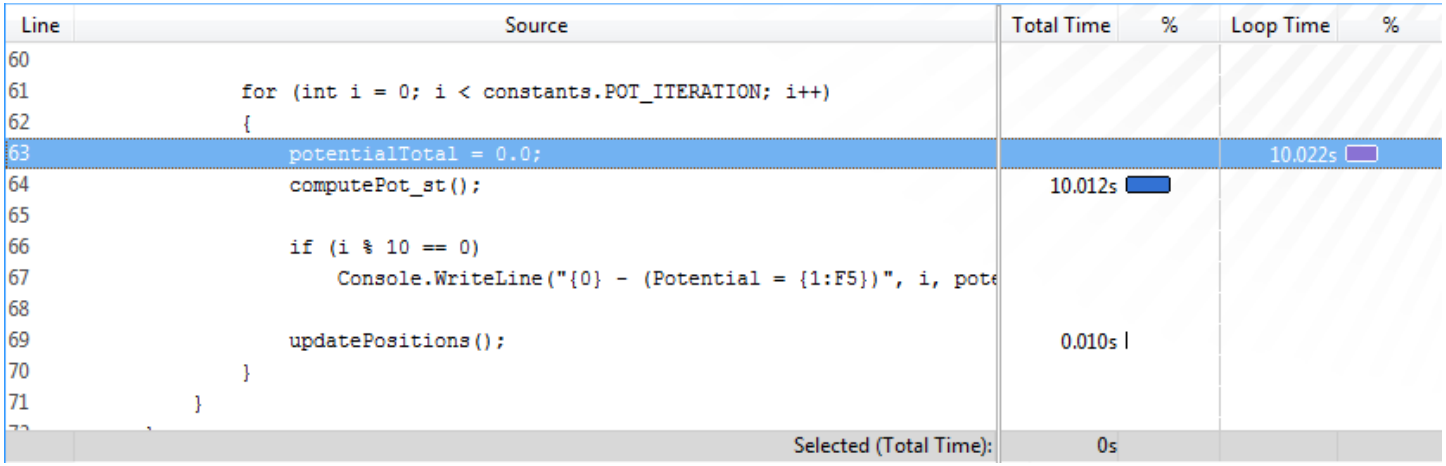
Survey



Find the places that are important to your application

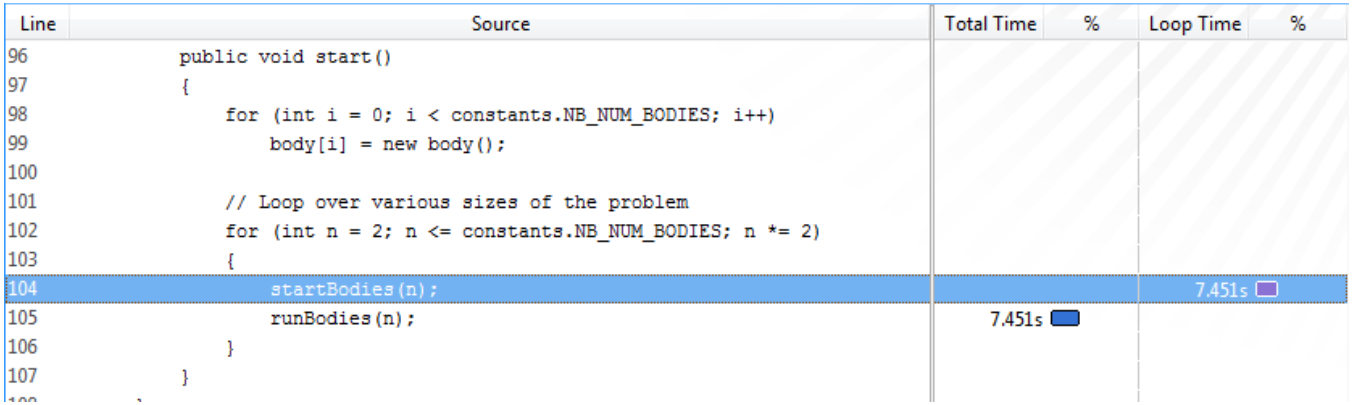
Two Candidate loops

- 56%: POTENTIAL::start (loop)



Line	Source	Total Time	%	Loop Time	%
60					
61	for (int i = 0; i < constants.POT_ITERATION; i++)				
62	{				
63	potentialTotal = 0.0;			10.022s	
64	computePot_st();	10.012s			
65					
66	if (i % 10 == 0)				
67	Console.WriteLine("{0} - (Potential = {1:F5})", i, pote				
68					
69	updatePositions();	0.010s			
70	}				
71	}				
	Selected (Total Time):	0s			

- 41.8%: NBODIES::start (loop)



Line	Source	Total Time	%	Loop Time	%
96	public void start()				
97	{				
98	for (int i = 0; i < constants.NB_NUM_BODIES; i++)				
99	body[i] = new body();				
100					
101	// Loop over various sizes of the problem				
102	for (int n = 2; n <= constants.NB_NUM_BODIES; n *= 2)				
103	{				
104	startBodies(n);			7.451s	
105	runBodies(n);	7.451s			
106	}				
107	}				

Advisor XE Annotation Concepts

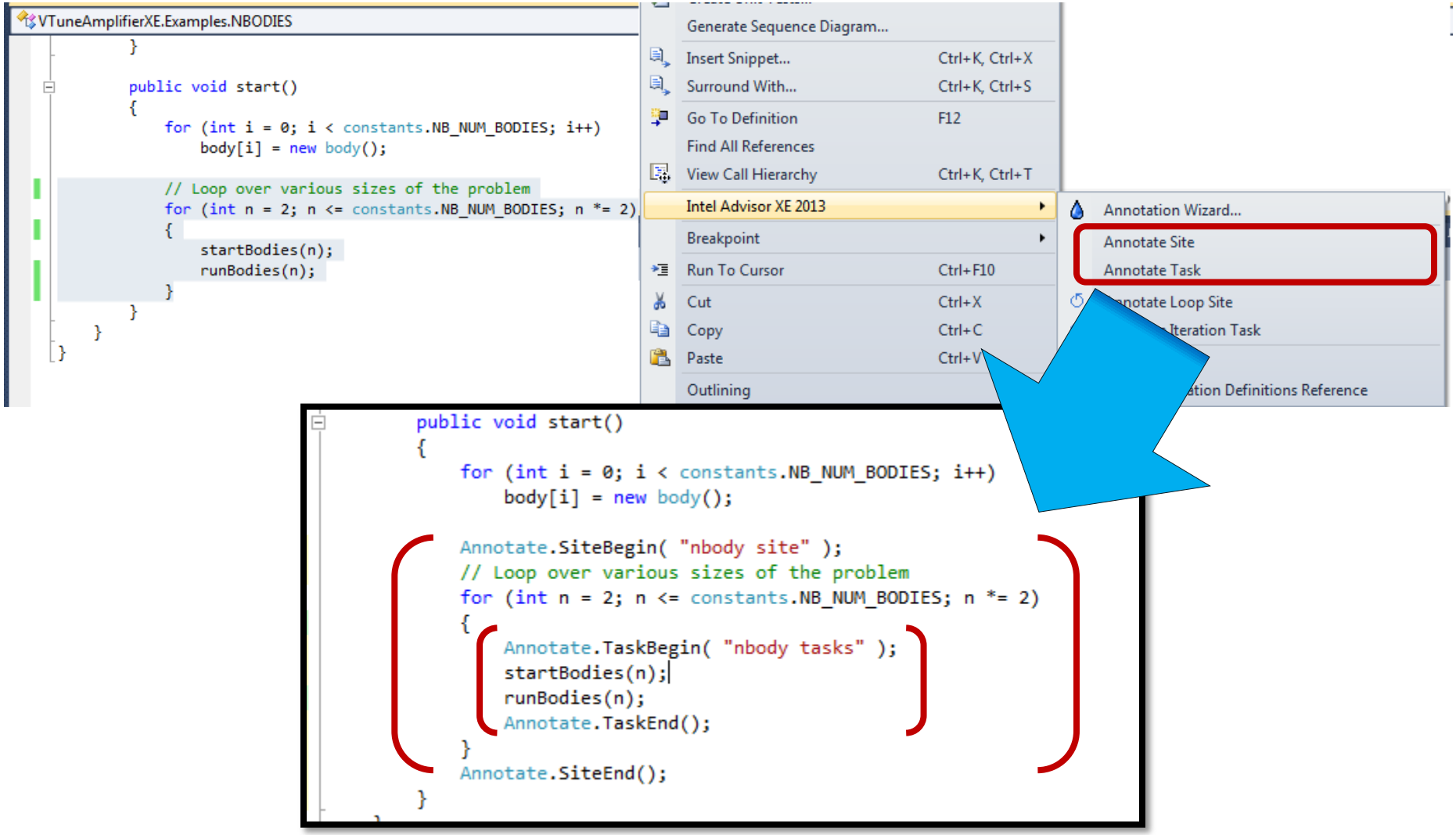
Advisor XE uses 3 primary concepts to create a model

- **SITE**
  - A region of code in your application you want to transform into parallel code
- **TASK**
  - The region of code in a SITE you want to execute in parallel with the rest of the code in the SITE
- **LOCK**
  - Mark regions of code in a TASK which must be serialized

NOTE

- All of these regions may be nested
- You may create more than one SITE
- Just macros, so work with any C/C++ compiler

Add Annotation
NBODIES::start (loop)

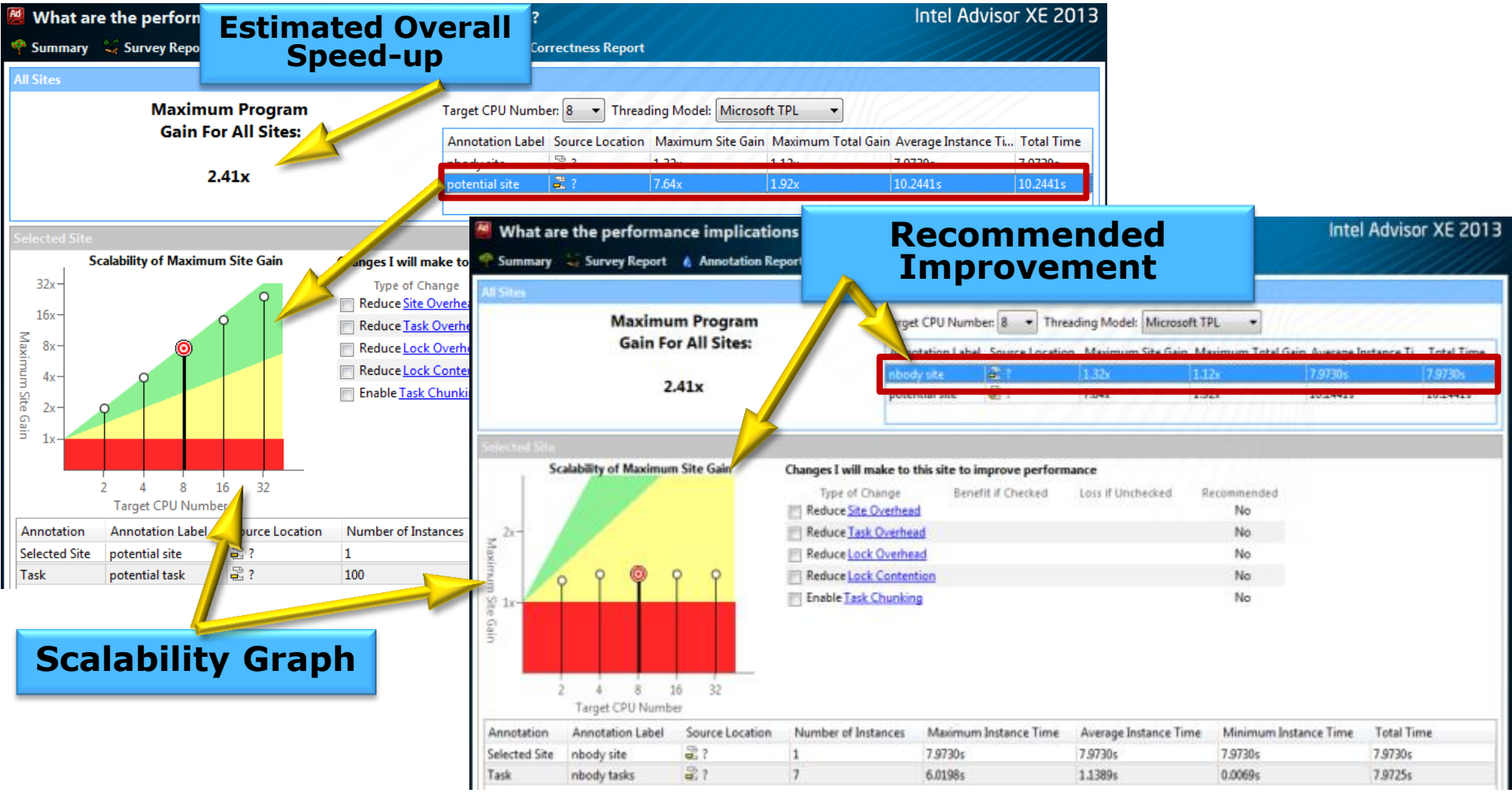


Add Annotation
POTENTIAL::start (loop)

VTuneAmplifierXE.Examples.POTENTIAL — start()

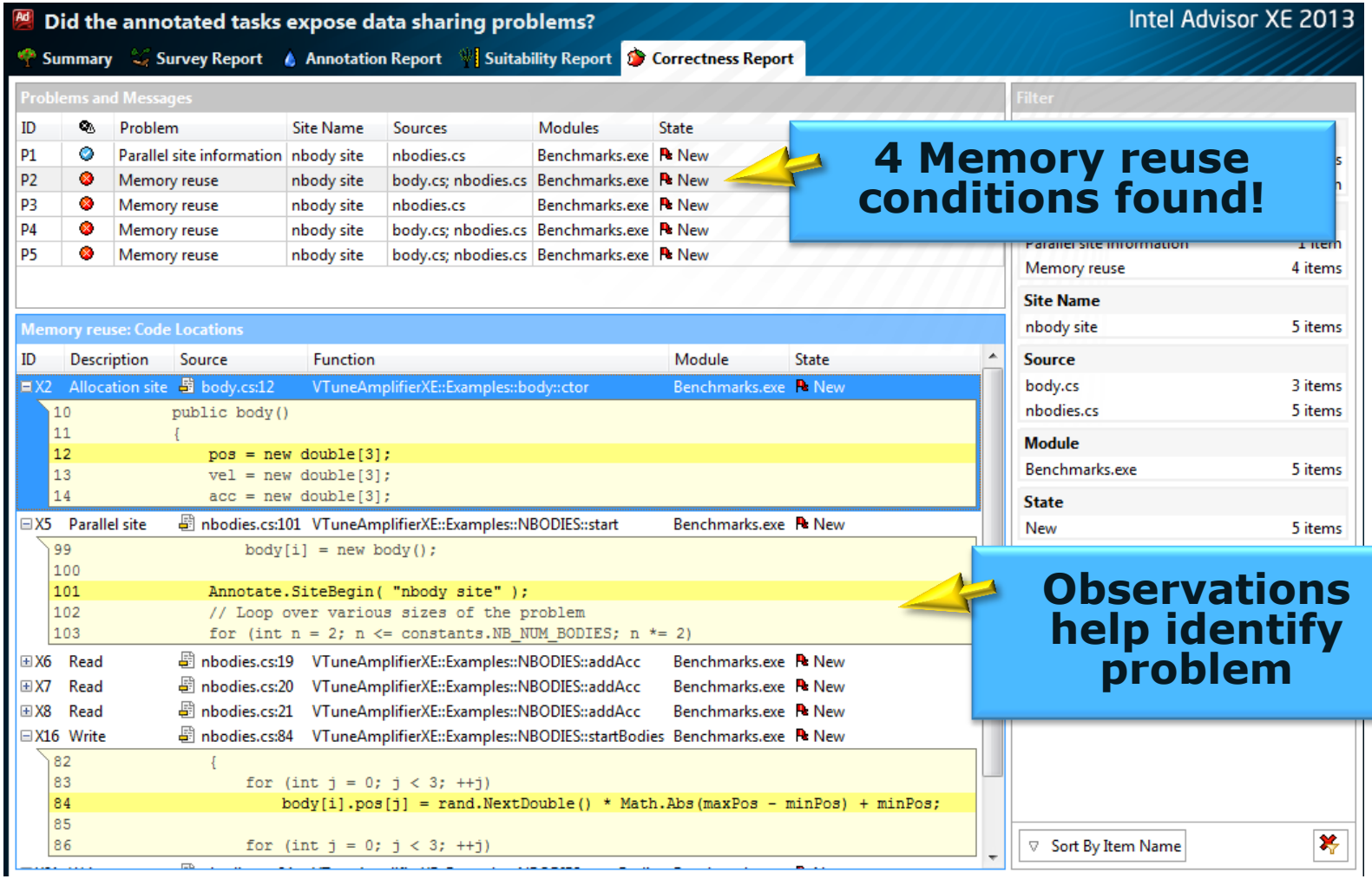
```
Annotate.SiteBegin( "potential site" );
for (int i = 0; i < constants.POT_ITERATION; i++)
{
    potentialTotal = 0.0;
    Annotate.TaskBegin( "potential task" );
    computePot_st();
    Annotate.TaskEnd();

    if (i % 10 == 0)
        Console.WriteLine("{0} - (Potential = {1:F5})", i, potentialTotal);

    updatePositions();
}
Annotate.SiteEnd();
```

Suitability – How Fast Will It Be?



Analyze the performance of your proposal

Correctness – Any Data Sharing Bugs?



Analyze your design for errors

and then Repeat…

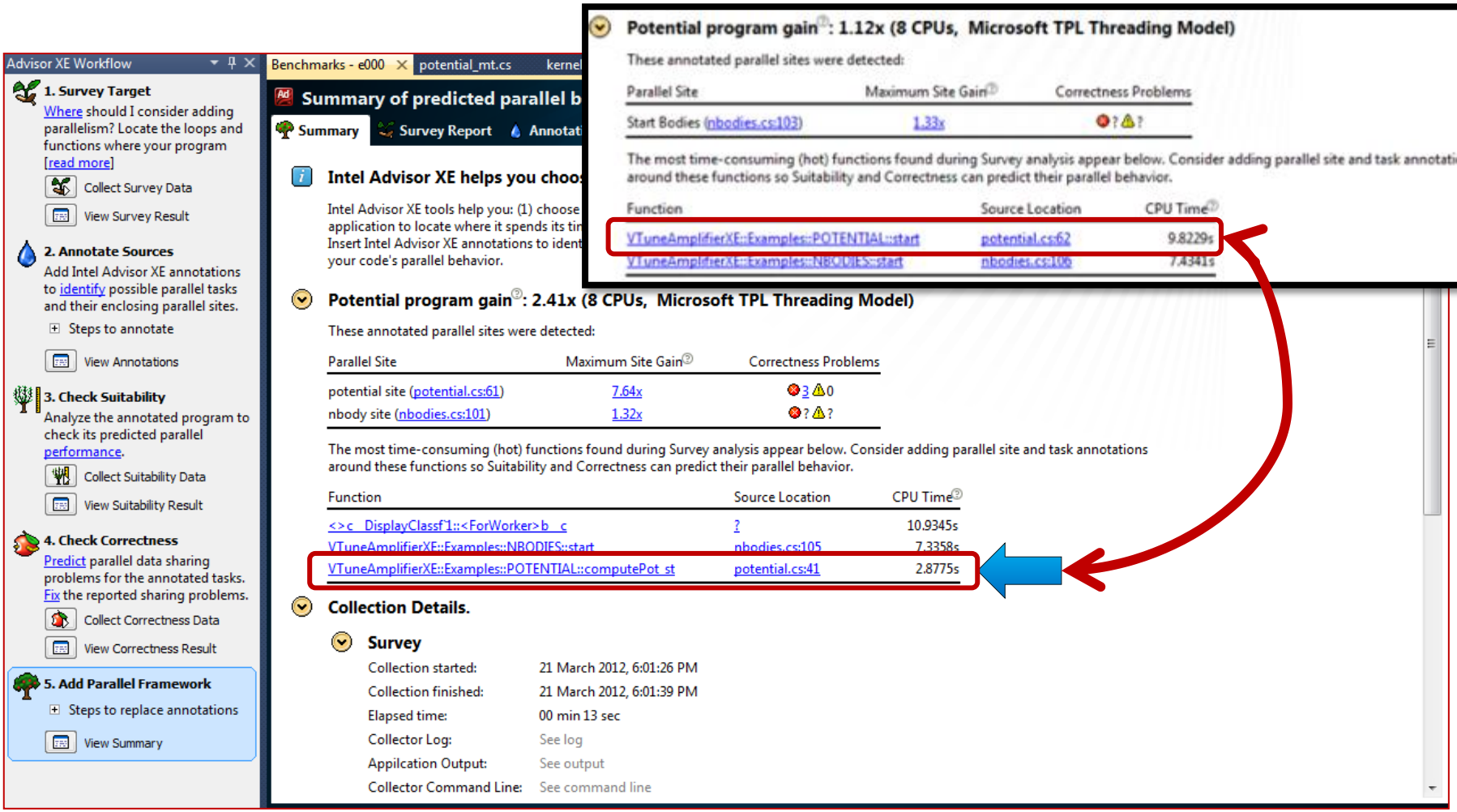
You do not have to choose the perfect answer the first time, so you can go back and modify your choices

Iterative refinement will either

- **Create a suitable and correct annotation proposal**
- **Conclude no viable sites are possible**

Efficiently arriving at either answer is valuable

Add Parallel Framework



Summary

The Intel Advisor XE is a unique tool

- assists you to work smarter though detailed modeling
- guides you through the necessary steps
- leaves you in control of code and architectural choices
- lets you transform serial algorithms into parallel form faster

The parallel modeling methodology

- maintains your original application's semantics and behavior
- helps find the natural opportunities to exploit parallel execution

Intel® Advisor XE

Available in these performance suites

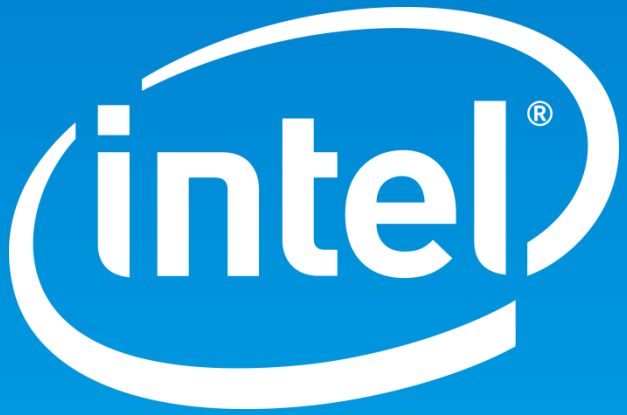
	Intel® Cluster Studio XE	Intel® Parallel Studio XE	
Analysis	●	●	**Intel® VTune™ Amplifier XE** - Performance Profiler
	●	●	**Intel® Inspector XE** - Memory & Thread Analyzer
	●	●	**Static Analysis & Pointer Checker** - Find Coding & Security Errors
	●	●	**Intel® Advisor XE** - Threading Prototyping Tool
	●		**Intel® Trace Analyzer & Collector** - MPI Optimizing Tool
Compilers & Libraries	●	●	**Intel® Compiler** - Optimizing Compiler for C, C++ and Fortran
	●	●	**Intel® Integrated Performance Primitives**† - Media and Data Optimizations
	●	●	**Intel® Threading Building Blocks**† **-** Parallelize Applications for Performance
	●	●	**Intel® Math Kernel Library -** High Performance Math
	●		**Intel® MPI Library** - Flexible, Efficient and Scalable Messaging

http://software.intel.com/en-us/intel-advisor-xe

† Available for C, C++ only

C, C++ only and Fortran only versions of Parallel Studio XE are also available.

Create fast, reliable code


intel

Legal Disclaimer & Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804