SunTune Course Overview

Introduction − The SunTune Optimization Training has been developed by the SUN EMEA HPC Team in order to train end−users of SUN systems how to obtain good performance out of their application. This includes both manual optimization techniques, as well as using the SUN development environment in an optimal way.

The general philosophy of the training is to build up understanding of key concepts that are relevant to obtain good application performance. Once this is achieved, it is much easier to use the development environment in the best possible way.

The training covers both sequential optimization, as well as parallelization. Other than some programming experiences (preferably in Fortran or C), no specific background is assumed.

Table of Contents

1. Objectives

After this 3 day training, attendees should be much more knowledgeable on UltraSPARC−II performance, know how to tune and parallelize technical applications, be familiar with our software environment for HPC and be able to give practical advise to others what to do and what not to do when it comes to making their program(s) run more efficiently on SUN systems.

2. Target audience

Primarily we're looking for people that are interested to learn more about the technical details of High−Performance Computing. We do assume you have some practical programming experiences. Preferably this is in Fortran and/or C, but it is not a requirement.

3. The Training Modules

In this section we list the individual modules covered in the training. Per module, a short summary description is included.

3.1. The Memory Hierarchy

Today's microprocessors make heavy use of caches to bridge the gap between the speed of the processor and main memory. The resulting memory hierarchy can be rather complex. Often, tuning an application is about using the entire memory hierarchy in the best possible way.
In this module we explore the memory hierarchy to a reasonable level of detail. The concepts and characteristics presented here will be needed in the remainder of the training.

3.2. Architecture Overview

This relatively short module covers the UltraSPARC−II microprocessor and the server architectures. Only those characteristics relevant when tuning an application will be covered. This includes details on the memory hierarchy and a description of the most important instructions.

3.3. System Tools

The SUN Solaris Operating System has many powerful and useful commands that can assist the application developer. A subset of these is presented in this module.

3.4. Sequential Optimization Techniques

Sequential (also often called serial) performance is important for several reasons:

- Existing resources are used more optimally, typically without additional cost
- The performance increase can be noticable
- When parallelized, a program tuned for sequential performance will usually also give better performance with many processors

One may be tempted to think that sequential optimization is not worth considering because that is an area well under control. Even though a lot of progress has been made, there is often room for significant improvement in many applications.

A variety of techniques to optimize specific programming constructs have been developed over the years. Nowadays, the most important ones have been integrated into compilers and are part of standard optimizations. However, even though compilers get more sophisticated over time, they sometimes lack information to make the optimal decision.

Therefore we will cover all these key optimization techniques so that users can decide for themselves what should be done to assure good performance, with or without the help of the compiler.

In addition to these techniques, we will also present some example optimizations that go beyond what can be expected from a compiler.

3.5. The WorkShop Debugger and Performance Analyzer

The SUN WorkShop Environment offers a comprehensive program development suite. In this module we only cover two components: the symbolic debugger and the performance analyzer.

The debugger supports both command line controls and a GUI. Aside from the traditional functionality (breakpoint controls, printing of variables, call stack tracebacks, etc.), also source code browsing , data visualization and other features are available.

The performance analyzer is a powerful and useful tool to analyze the performance of an application. It can be used on existing, non−instrumented, executables and provides important information on the behavior of the application. Extensive post−processing features on the statistics gathered allow the user to obtain insight into the performance of the program.

3.6. The SUN Compilers − General

The SUN compilers provide a comprehensive set of options. In this module we will focus on those options that can be helpful when porting and/or debugging an application. These options can be useful when one tries to zoom in on a particular problem.

3.7. The SUN Compilers − Performance

This module is about performance relation options only. As will be explained and discussed, several categories of optimization switches are available.

3.8. The SUN Performance Libraries

In addition to compiler options, one can also use the highly tuned mathematical libraries that SUN provides as part of the WorkShop environment. We will present an overview of the available functionality.
Frequently used intrinsic operations have been optimized extensively. In addition to this, vectorized versions of important intrinsic functions are available.
The SUN Performance Library not only provides tuned (and shared memory parallelized) versions of the BLAS1, BLAS2, BLAS3, LINPACK, LAPACK, FFTPACK and VFFTPACK public domain packages, but also contains additional functionality for sparse matrices, convolutions, correlations, etc.

3.9. The SUN Compilers − Modulo Scheduling

Modulo Scheduling is used by the compiler as a technique to exploit the superscalar nature of the microprocessor and to hide instruction latencies. This module covers the concepts of modulo scheduling and the diagnostics given by the compiler. Several examples how to use this information to tune an application are presented.

3.10. Introduction Into Parallelization

The basics of parallel processing are introduced and explained. This module serves as a basis for the remainder of the training. Several important concepts (threads, speed−up, Amdahl's law, load balancing, parallel architectures, programming models, etc.) are introduced and discussed.

3.11. Data Dependency Analysis

In order to safely parallelize a specific program construct, one has to know whether there is a data dependency or nor. With automatic parallelization, the compiler has to answer this question. When explicitly parallelizing a program, the user has to be able to determine whether there is a data dependency or not.

3.12. Shared Memory Parallelization(incl. OpenMP)

Through a specific option, SUN compilers can auto−parallelize an application. The user can also obtain parallelization diagnostic messages out of the compiler. These will indicate whether the compiler has been successful or not in parallelizing a program.
It can happen that the compiler lacks information to resolve a data dependency. In such a case the user will have to provide additional information to the compiler through so−called directives (pragma's in C). We will present the directives/pragma's that are available on SUN systems.
In view of future developments, we will present an overview of the de−facto OpenMP standard for shared memory parallelization.

3.13. MPI and PRISM on SUN

SUN provides a highly tuned implementation of the Message Passing Interface (MPI) de−facto standard. We will present an overview of specific implementation features and give tips how to maximize the performance.
An overview of the powerful PRISM debugger and performance analyzer for testing, developing and tuning MPI applications will be given as well.

4. Hands−on Lab Sessions

The SunTune training is practically oriented. In order to benefit most from the material presented, several hands−on sessions are scheduled as part of the course. Examples will be made available. These have been chosen to highlight a specific feature discussed in the training. Attendees are encouraged to work on these examples and possibly extend them to reflect the typical characteristics of their own application.

5. Dress code

The course will be rather intense, so comfortable cloths may be most appropriate. However, we prefer not to impose any preferences regarding this. Please dress in the way you feel most comfortable with.