INTERNATIONAL JOURNAL OF LATEST TECHNOLOGY IN ENGINEERING, MANAGEMENT & APPLIED SCIENCE (IJLTEMAS)

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XII, Issue II, February 2023

Survey of Modern Tools and Techniques for High Performance Computing

Himanshu N. Patel

Dr. Babasaheb Ambedkar Open University, Ahmedabad, Gujarat, India- 388060

Abstract— High performance computing (HPC) is one of the most essential tools fueling the advancement of computational science. And the universe of scientific computing has expanded in all directions [11]. From weather forecasting and energy exploration, to computational fluid dynamics and life sciences, researchers are fusing traditional simulations with artificial intelligence, machine learning, deep learning, big data analytics, and edge-computing to solve the mysteries of the world around us. In this paper I have discuss some of the modern tools and techniques for HPC and how it works with advantages it offers. This paper also provides useful information regarding types of tools for High Performance Computing.

Keywords— High Performance Computing (HPC), Development tools for HPC, IDE for HPC, Debugging tools for HPC, Analysis tools for HPC.

I. Introduction

High-performance computing (HPC) is the use of distributed computing facilities for solving problems that need large computing power. The general profile of HPC applications is constituted by a large collection of compute-intensive tasks that need to be processed in a short period of time. It is common to have parallel and tightly coupled tasks, which require low-latency interconnection networks to minimize the data exchange time. The metrics to evaluate HPC systems are floating-point operations per second (FLOPS), now tera-FLOPS or even peta-FLOPS, which identify the number of floating-point operations per second that a computing system can perform [12].

Typical use of HPC is in the field of Climate modeling, Geographical data, Electronic design automation, Oil and gas industry modeling, Biosciences and Media & entertainment. Major area of applications [11] of high performance computing is

Super Computing: Exploring supernova explosions. Mapping the Earth's interior. Predicting hurricanes.

Healthcare & Life Sciences: Discovering drugs. Uncovering genetic mutations. Analyzing images.

Energy: Producing energy. Refining and distributing oil. Reducing environmental impact.

Public Sector: Cyber security. Disaster response. Humanitarian assistance.

II. How does HPC work?

A standard computing system solves problems primarily using serial computing—it divides the workload into a sequence of tasks, and then executes the tasks one after the other on the same processor

In contrast, HPC leverages

- Massively parallel computing. Parallel computing runs multiple tasks simultaneously on multiple computer servers or processors. Massively parallel computing is parallel computing using tens of thousands to millions of processors or processor cores.
- Computer clusters (also called HPC clusters). An HPC cluster consists of multiple high-speed computer servers networked together, with a centralized scheduler that manages the parallel computing workload. The computers, called nodes, use either high-performance multi-core CPUs or, more likely today, GPUs (graphical processing units), which are well suited for rigorous mathematical calculations; machine learning models and graphics-intensive tasks. A single HPC cluster can include 100,000 or more nodes.
- High-performance components: All the other computing resources in an HPC cluster networking, memory, storage and file systems are high-speed, high-throughput and low-latency components that can keep pace with the nodes and optimize the computing power and performance of the cluster.

III. Types of Tools for High Performance Computing

We can divide tools for high performance computing into three categories:

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XII, Issue II, February 2023

- Integrated Development Environments
- Parallel Communication and Debugging
- Performance Analysis Tools

In following section we discuss some of the tools available in each of the above categories.

A. **Integrated Development Environments (IDE) for HPC**

It provides facility for performance related tools and libraries to helps high performance computing applications. Following section discuss important IDEs

Google Cloud HPC Toolkit [10]

Cloud HPC Toolkit is open-source software offered by Google Cloud which makes it easy for you to deploy HPC. It is designed to be highly customizable and extensible, and intends to address the HPC deployment needs of a broad range of use cases. Cloud HPC Toolkit provides you with the following benefits:

- Fast creation and deployment of turnkey HPC environments that follow Google Cloud best practices
- An open source solution that is configurable and extensible
- Seamless integration with various partners such as Intel DAOS, DDN EXAscaler, and Slurm
- Monitoring and performance visibility through integration with Cloud Monitoring

Cloud HPC Toolkit only supports creating and deleting a cluster.

IBM High Performance Computing Toolkit [1]

The HPC Toolkit is an integrated set of performance analysis tools that help the application developer tune a serial or parallel application. The HPC Toolkit is a component of IBM Parallel Environment Developer Edition. IBM Parallel Environment (PE) Developer Edition Version 1 Release 3 provides an Eclipse-based integrated development environment (IDE) that improves parallel application developer productivity throughout the edit, compile, debug, and run development cycle for C, C++ and FORTRAN applications. The IBM HPC Toolkit provides three primary interfaces for you to use [13]:

- Eclipse IBM HPC Toolkit plug-in.
- peekperf
- Xprof

Using above interface, you can:

- Collect performance data from an instrumented application by setting environment variables for the specific analysis tool and then invoking the application from the command line.
- Modify your application to add calls to the hardware performance counter tool, rebuild your application linking with the hardware performance counter library, and then run your application to get hardware performance counter data.
- Link your application directly with the MPI profiling and trace library then run your application to get MPI trace and profiling data.
- Link your application with the I/O profiling and trace library to get I/O profiling and trace information.

Intel Parallel Studio XE [2]

It is a set of tools that helps software architects and lead designers add effective parallelism to an application to improve performance on multi-core systems. It allows to quickly prototype a parallel design, project its scalability and identify synchronization issues. It Deliver top application performance while minimizing development, tuning and testing time, and effort.

It is a threading prototyping tool for C, C++, C# and FORTRAN software architects. Quickly model and compare the performance scaling of different threading designs without the cost and disruption of implementation. Find and eliminate data sharing issues during design when they are less expensive to fix. Model the performance impact of any added synchronization and project the scaling on systems with larger core counts.

Sun HPC ClusterTools [3]

The Sun HPC ClusterTools 7 release was Sun's first binary distribution of the Open MPI software. This release marked a change in source-code base for Sun from a proprietary code base derived from the Thinking Machines Corporation Globalworks™ software to the open-source Open MPI software. Sun HPC ClusterTools includes packages of binaries built from the Open MPI source code

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XII, Issue II, February 2023

by the Sun™ Studio compilers and install scripts for installing those packages across a cluster of nodes. The Sun HPC ClusterTools team contributed a Sun Grid Engine plug-in and developed the uDAPL Byte Transfer Layer module as its Infiniband solution on Solaris™ operating system. Additionally, Sun HPC ClusterTools includes examples of using DTrace to analyze performance and debug MPI applications. Other product-focused activity included significant contribution to the development of the MPI Test Tool (MTT) and development of a set of user documentation. This paper describes the new Sun HPC ClusterTools based on Open MPI, focusing on areas where Sun has contributed to Open MPI.

B. Parallel Debugging Tools For HPC[14]

This section discusses tools available for parallel communication and debugging.

Valgrind [4]

Valgrind is a system for debugging and profiling Linux programs. With Valgrind's tool suite you can automatically detect many memory management and threading bugs, avoiding hours of frustrating bug-hunting, making your programs more stable. You can also perform detailed profiling to help speed up your programs. The Valgrind distribution includes the following debugging and profiling tools:

TABLE I

Tool	Use
Memcheck	Memcheck detects memory-management problems, and is aimed primarily at C and C++ programs. When a program is run under Memcheck's supervision, all reads and writes of memory are checked, and calls to malloc/new/free/delete are intercepted.
Cachegrind	Cachegrind is a cache profiler. It performs detailed simulation of the I1, D1 and L2 caches in your CPU and so can accurately pinpoint the sources of cache misses in your code. It identifies the number of cache misses, memory references and instructions executed for each line of source code, with per-function, per-module and whole-program summaries. It is useful with programs written in any language. Cachegrind runs programs about 20--100x slower than normal.
Callgrind	It provides all the information that Cachegrind does, plus extra information about callgraphs. It is also available separately is an amazing visualisation tool, KCachegrind, which gives a much better overview of the data that Callgrind collects; it can also be used to visualise Cachegrind's output.
Massif	Massif is a heap profiler. It performs detailed heap profiling by taking regular snapshots of a program's heap. It produces a graph showing heap usage over time, including information about which parts of the program are responsible for the most memory allocations. The graph is supplemented by a text or HTML file that includes more information for determining where the most memory is being allocated. Massif runs programs about 20x slower than normal.
Helgrind	Helgrind is a thread debugger which finds data races in multithreaded programs. It looks for memory locations which are accessed by more than one (POSIX p-) thread, but for which no consistently used (pthread_mutex_) lock can be found. Such locations are indicative of missing synchronization between threads, and could cause hard-to-find timing-dependent problems. It is useful for any program that uses pthreads. It is a somewhat experimental tool, so your feedback is especially welcome here
DRD	DRD is a tool for detecting errors in multithreaded C and C++ programs. The tool works for any program that uses the POSIX threading primitives or that uses threading concepts built on top of the POSIX threading primitives. While Helgrind can detect locking order violations, for most programs DRD needs less memory to perform its analysis.

Marmot [5]

Debugging MPI-programs normally is very frustrating. MARMOT surveys the MPI-calls made and automatically checks the correct usage of these calls and their arguments during runtime. It does not replace classical debuggers, but can be used in addition to them.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XII, Issue II, February 2023

MemoryScape [6]

It is advanced memory debugging and analysis capability that helps in identifying and resolve difficult memory problems in C, C++ and FORTRAN. It is graphical, real-time view into heap memory, memory usage, memory allocation bounds violations and memory leaks, occurs without instrumentation. Its built-in scripting language makes batch mode testing easy and efficient, incorporating scripts into nightly processing to verify that new development has introduced no new memory errors.

C. Performance Analysis Tools For HPC

This section discusses tools for performance analysis:

Vampir Performance Analysis Tool-Set [7]

Vampir is available as a commercial product since 1996 and has been enhanced in the scope of many research and development projects. The tool is well-proven and widely used in the high performance computing community for many years. A growing number of performance monitoring environments like TAU, KOJAK or VampirTrace can produce tracefiles that are readable by Vampir. Unfortunately, Vampir no longer supports Intel's structured trace file (STF) format because of licensing reasons. Since version 5.0, Vampir supports the new Open Trace Format (OTF). This trace format is especially designed for massively parallel programs.

During a program run of an application, VampirTrace generates an OTF trace file, which can be analyzed and visualized by the visualization tool Vampir. The VampirTrace library allows MPI communication events of a parallel program to be recorded. Additionally, certain program-specific events can also be included.

Vampir is very portable due to its X-based graphical user interface and available for many computing platforms.

SCALASCA - Scalable performance analysis of large-scale parallel applications [8]

SCALASCA is a software tool that supports the performance optimization of parallel programs by measuring and analyzing their runtime behavior. The analysis identifies potential performance bottlenecks in particular those concerning communication and synchronization and offers guidance in exploring their causes.

SCALASCA supports the performance optimization of simulation codes on a wide range of current HPC platforms. Its powerful analysis and intuitive result presentation guides the developer through the tuning process.

SCALASCA targets mainly scientific and engineering applications based on the programming interfaces MPI and OpenMP, including hybrid applications based on a combination of the two. The tool has been specifically designed for use on large-scale systems including IBM Blue Gene and Cray XT, but is also well suited for small and medium scale HPC platforms. The software is available for free download under the New BSD open-source license.

TAU Performance System [9]

TAU Performance System® is a portable profiling and tracing toolkit for performance analysis of parallel programs written in FORTRAN, C, C++, Unified Parallel C (UPC), Java, and Python.

TAU (Tuning and Analysis Utilities) is capable of gathering performance information through instrumentation of functions, methods, basic blocks, and statements as well as event based sampling. All C++ language features are supported including templates and namespaces. The API also provides selection of profiling groups for organizing and controlling instrumentation. The instrumentation can be inserted in the source code using an automatic instrumentor tool based on the Program Database Toolkit (PDT), dynamically using DyninstAPI, at runtime in the Java Virtual Machine, or manually using the instrumentation API.

TAU's profile visualization tool, paraprof, provides graphical displays of all the performance analysis results, in aggregate and single node/context/thread forms. The user can quickly identify sources of performance bottlenecks in the application using the graphical interface. In addition, TAU can generate event traces that can be displayed with the Vampir, Paraver or JumpShot trace visualization tools.

IV. Conclusion

With the introduction of multi-core processors, applying parallel programming methods in application development is important to achieve efficient performance. 8 core and possibly 16 core processors will be available soon for desktop computers. Some diverse tools requires for supporting application developers. This paper gives the reader an overview of the available tools in the area of integrated development environments, parallel debuggers, and recent performance analysis tools. The paper gives reader, a technical overview to assist them to decide tool suitable for the development task at hand.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XII, Issue II, February 2023

References

1. "http://researcher.watson.ibm.com/researcher/view_group.php?id=2754"
2. "https://software.intel.com/en-us/intel-parallel-studio-xe"
3. "https://link.springer.com/chapter/10.1007/978-3-540-68564-7_1"
4. "http://valgrind.org/info/about.html"
5. "http://www.hlrs.de/organization/av/amt/projects/marmot"
6. "http://www.roguewave.com/products/memoryscape.aspx"
7. "http://www.vampir.eu/history"
8. "http://www.scalasca.org/about/about.html"
9. "http://www.cs.uoregon.edu/research/tau/home.php"
10. "https://cloud.google.com/hpc-toolkit/docs/overview"
11. "https://www.nvidia.com/en-in/high-performance-computing/"
12. "https://www.sciencedirect.com/topics/computer-science/high-performance-computing"
13. "https://www.ibm.com/docs/en/SSFK5S_EOS/eos/install_130.pdf"
14. H. Patel and P. Virparia, "Tools for efficient parallelism in high performance computing", in National Journal of System and Information Technology, 2014, pp 21-28, ISSN 0974-3308.