# Performance Analysis Tools for MPI

The areas that are performing badly can be identified by the performance analysis tools. Those tools can locate the areas and hence it can be able to improve program efficiency. The analysis tools should be robust and portable.

There are a lot of tools available and here I choose,

1.DEEP/MPI

2.VAMPIR

## DEEP/MPI:

- Development Environment for Parallel Programs called as DEEP was developed by Veridian-Pacific Sierra Research.
- The program has to be compiled with mpiprof DEEP profiling driver.
- It has an integrated graphics interface through with the shared and distributed memory programs can be monitored.
- It stores the information in the subdirectory and displays the performance data after executing the code and displays the wall clock time.
- The loop performance table which is one of the additional information that user brings up.
- CPU and message balance displays the work distribution and number of messages among the processes.
- PAPI Interface to hardware was supported and it supports the profiling.
- Supports analysis of shared memory parallelism and can be used to analyzed mixed MPI and shared memory parallel programs.

## VAMPIR:

- The tool was developed by Pallas GmbH.
- Vampirtrace is a profiling library, and it produces trace files, and the analysis is handled using trace file.
- It has an API for starting and stopping tracing and also inserts the trace file with user defined events.

- It includes MPI I/O calls and a runtime filtering mechanism which is used to limit the amount of trace data.
- Vampitrace automatically corrects clock offset and skew, for systems without globally consistent clock.
- The current version can display up to 512 processes and the display is clustered with cluster, node and process with maximum of 200 objects.
- The displays include analysis of program execution, statistical analysis and dynamic calling tree.

**Differences:**

- The profiling library used for Vampir is Vampitrace and for DEEP/MPI, mpiprof is used.
- DEEP has an integrated graphics interface which fetches data from shared memory  and displays data for only the small number of processes while Vampir uses cluster mechanism and displays status of lots of processes.
- Vampir traces user defined events while DEEP does not support this functionality.
- Vampitrace automatically corrects the clock and make it consistent while DEEP doesn't.
- DEEP includes a call tree viewer for program structure browsing while Vampitrace uses dynamic calling tree.