Performance comparison between MPI and AMPI

Rabin Ranabhat
Loras College
rabin@ucar.edu
July 27, 2011

Mentors: Jim Edwards
: John Dennis
Overview

• Motivation
• Objective
• Parallel Program
• MPI
• AMPI
  – Virtualization
  – Migration
Motivation
Objective

• Investigate performance of AMPI and MPI with a parallel program
Parallel Program

- Mandelbrot used as the parallel program
- 2-dimensional decomposition

![Parallel Program Diagram]
Message Passing Interface (MPI)

• Defacto standard for running parallel programs
• First implemented by William Gropp, Ewing Lusk
• Allows processes to communicate
Adaptive MPI (AMPI)

- MPI implementation based on Charm++
- Developed by University of Illinois – Urbana Champaign
- AMPI was introduced because
  - Processor set available may not be what the application needs.
AMPI

• has virtualizations
  – Number of tasks implemented as user level threads
VP ratio vs. time

- Time taken (in seconds)
- Ratio of Virtual Processor and Physical Processor

*400 x 400 grid size
*32 physical processors
Adaptive MPI: migration

- Over-decomposition contribute to migrating threads which lead to dynamic load balancing
- Tradeoff between load balancing overhead and performance degradation
- Subroutine call for migration – `MPI_Migrate()`
Migration frequency

Case 1

Case 2
MPI and AMPI task with migration

![Graph showing the time taken for MPI and AMPI tasks with migration. The x-axis represents the number of processes, while the y-axis represents the time taken in seconds. The graph compare MPI and AMPI tasks, with MPI showing a faster decline with increasing number of processes.]
Conclusion

• AMPI does a better job granted certain conditions
• On average, AMPI jobs with load balancing 5% faster than MPI
• Not successful to use MPI-IO with AMPI
Questions

Email: rabin@ucar.edu