Scalable Verification of MPI Programs
Anh Vo – Advisor: Ganesh Gopalakrishnan
School of Computing, University of Utah, SLC UT 84112

Problem / Motivation

- Large scale MPI applications are hard to verify:
  - Non-determinism is a big issue
  - Bugs appear intermittent, not in all traces
  - Bugs appear only at large scale
- Tradition testing: scalable, but no coverage guarantee over the space of non-determinism
- Model checking: guarantee coverage, but not scalable

Need scalable verification tool for non-deterministic MPI programs

Background/Related Work

- The current state-of-the-art dynamic verifier, ISP can:
  - Verify MPI programs: deadlocks, resource leaks
  - Guarantee coverage over non-determinism
  - Detect and enforce different execution schedules
- Example (above):
  - Both P0 & P2 sends can match P1 wildcard receive
  - One match causes error
  - Detect both choices and enforce them:
    - Intercept the wildcard Recv, force receive from P0
    - Restart the program, now force receive from P2

LIMITATIONS:
Verification does not scale beyond dozens of processes

Approach/Solution

Algorithm:
- Each process keeps a Lamport Clock (currLC)
- Increase currLC when observe an event
- Each LC is associated with an epoch
- Observable events are non-deterministic events
- Use piggyback (extra msg) to send LC
- Compare incoming m.LC with currLC:
  - If (m.LC < currLC) Then
    - m is “late”; /* i.e., arriving late */
    - check if m can be alternate matches to past events
    else (m.LC >= currLC)
    - currLC = max (currLC, m.LC)
- Upon program completion, output alternate matches
- Schedule Generator generates decisions for rerun (DFS)
- Proc’s are restarted & follow the decisions in each epoch by replacing Recv(*) w. specific Recv. (GUIDED_RUN)
- If new schedules are discovered, explore them as well
- Repeat until no more schedules

Experimental Results

Example showing how DMA discover matches

Experimental Results

- Time taken to verify MPI programs
- Comparison between ISP and DMA

Acknowledgements

We are thankful to Bronis R. de Supinski, Martin Schulz, and Greg Bronevetsky of Lawrence Livermore National Lab for their support, mentoring and encouragement.

Supported in part by Microsoft, NSF CNS-0509379, CCF-0811429, CCF-0903408