Case Studies with Projections

Ronak Buch & Laxmikant (Sanjay) Kale
http://charm.cs.illinois.edu
Parallel Programming Laboratory
Department of Computer Science
University of Illinois at Urbana-Champaign

11th Workshop of the INRIA-Illinois-ANL JLPC, Sophia Antipolis, France
June 12, 2014
Basic Problem

- We have some Charm++ program
- Performance is worse than expected

How can we:
  - Identify the problem?
  - Measure the impact of the problem?
  - Fix the problem?
  - Demonstrate that the fix was effective?
Key Ideas

● Start with high level overview and repeatedly specialize until problem is isolated
● Select metric to measure problem
● Iteratively attempt solutions, guided by the performance data
Key Ideas

- Start with high level overview and repeatedly specialize until problem is isolated
- Select metric to measure problem
- Iteratively attempt solutions, guided by the performance data
Stencil3d Performance
Stencil3d

- Basic 7 point stencil in 3d
- 3d domain decomposed into blocks
- Exchange faces to neighbors

- Synthetic load balancing experiment
- Calculation repeated based on position in domain
No Load Balancing
No Load Balancing

Clear load imbalance, but hard to quantify in this view
No Load Balancing

Clear that load varies from 90% to 60%
Next Steps

- Poor load balance identified as performance culprit
- Use Charm++’s load balancing support to evaluate the performance of different balancers
- Trivial to add load balancing
  - Relink using -module CommonLBs
  - Run using +balancer <loadBalancer>
GreedyLB

Much improved balance, 75% average load
RefineLB

Much improved balance, 80% average load
Multirun Comparison

Greedy on left, Refine on right.
ChaNGa Performance
ChaNGa

- Charm N-body GrAVity solver
- Used for cosmological simulations
- Barnes-Hut force calculation

- Following data uses *dwarf* dataset on 8K cores of Blue Waters
- *dwarf* dataset has high concentration of particles at center
Original Time Profile
Original Time Profile

Time Profile

Why is utilization so low here?
Some PEs are doing work.
Next Steps

● Are all PEs doing a small amount of work, or are most idle while some do a lot?
● Outlier analysis can tell us
  ○ If no outliers, then all are doing little work
  ○ If outliers, then some are overburdened while most are waiting
Outlier Analysis

Extrema: Least Idle Time (20 Extrema PEs)
Outlier Analysis

Large gulf between average and extrema => Load imbalance
Next Steps

- Why does this load imbalance exist? What are the busy PEs doing and why are other waiting?

- Outlier analysis tells us which PEs are overburdened

- Timeline will show what methods those PEs are actually executing
Wrote new tool to parse Projections logs. Large disparity of messages across processors.
Next Steps

- Can we distribute the work?

- After identifying the problem, the code revealed that this was caused by tree node contention.

- To solve this, we tried randomly distributing copies of tree nodes to other PEs to distribute load.
Final Time Profile

Time Profile

Percentage Utilization

Time (1.851ms resolution)
Used to have 30000+ messages on some PEs, now all process <5000. Much better balance.