Understanding applications with Paraver

Judit Gimenez
judit@bsc.es

Sept 16-20, 2019

ANF Evaluation workshop
Humans are visual creatures

• Films or books?  
  • Two hours vs. days (months)  

• Memorizing a deck of playing cards  
  • Each card translated to an image (person, action, location)  

• Our brain loves pattern recognition  
  • What do you see on the pictures?
Our Tools

- Since 1991
- Based on traces
- Open Source
- [http://tools.bsc.es](http://tools.bsc.es)

- Core tools:
  - Paraver (paramedir) – offline trace analysis
  - Dimemas – message passing simulator
  - Extrae – instrumentation

- Focus
  - Detail, variability, flexibility
  - Behavioral structure vs. syntactic structure

- Intelligence: Performance Analytics

---

Barcelona Supercomputing Center
Centro Nacional de Supercomputación
Paraver
Paraver – Performance data browser

Trace visualization/analysis
+ trace manipulation

Timelines

2/3D tables (Statistics)

Comparative analyses
Multiple traces
Synchronize scales

Goal = Flexibility
No semantics
Programmable

Raw data

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

BSC
From timelines to tables

MPI calls profile

Useful Duration
Analyzing variability
Analyzing variability

- By the way: six months later ....
From tables to timelines

CESM: 16 processes, 2 simulated days

- Histogram useful computation duration shows high variability
- How is it distributed?

- Dynamic imbalance
  - In space and time
  - Day and night.
  - Season? 😊
Trace manipulation

- Data handling/summarization capability
  - Filtering
    - Subset of records in original trace
    - By duration, type, value,...
    - Filtered trace is a paraver trace and can be analysed with the same cfgs (as long as needed data kept)
  - Cutting
    - All records in a given time interval
    - Only some processes
  - Software counters
    - Summarized values computed from those in the original trace emitted as new even types
    - #MPI calls, total hardware count,...
Dimemas – Coarse grain, Trace driven simulation

- Simulation: Highly non linear model
  - MPI protocols, resource contention...

- Parametric sweeps
  - On abstract architectures
  - On application computational regions

- What if analysis
  - Ideal machine (instantaneous network)
  - Estimating impact of ports to MPI+OpenMP/CUDA/...
  - Should I use asynchronous communications?
  - Are all parts equally sensitive to network?

- MPI sanity check
  - Modeling nominal

- Paraver – Dimemas tandem
  - Analysis and prediction
  - What-if from selected time window

Detailed feedback on simulation (trace)
Network sensitivity

- MPIRE 32 tasks, no network contention

L = 5µs – BW = 1 GB/s

L = 1000µs – BW = 1 GB/s

L = 5µs – BW = 100MB/s

All windows same scale
Network sensitivity

- **WRF, Iberia 4Km, 4 procs/node**
  - Not sensitive to latency
- **NMM**
  - BW – 256MB/s
  - 512 – sensitive to contention
- **ARW**
  - BW - 1GB/s
  - Sensitive to contention
Would I benefit from asynchronous communications?

SPECFEM3D

Courtesy Dimitri Komatitsch
Ideal machine

The impossible machine: \( BW = \infty, \quad L = 0 \)

- Actually describes/characterizes Intrinsic application behavior
  - Load balance problems?
  - Dependence problems?

GADGET @ Nehalem cluster
256 processes

Real run

Ideal network

Impact on practical machines?
Impact of architectural parameters

- **Ideal speeding up ALL** the computation bursts by the CPU ratio factor
  - The more processes the less speedup (higher impact of bandwidth limitations) !!
Hybrid parallelization

- Hybrid/accelerator parallelization
  - Speed-up SELECTED regions by the CPU ratio factor

(Previous slide: speedups up to 100x)
Efficiency Model
Parallel efficiency model

- Parallel efficiency = LB eff * Comm eff
Parallel efficiency refinement: $LB \times \mu LB \times Tr$

- Serializations / dependences ($\mu LB$)
- Dimemas ideal network $\rightarrow$ Transfer (efficiency) $= 1$

$LB = 1$
Why scaling?

\[ \eta \parallel = LB \times Ser \times Trf \]

CG-POP mpi2s1D - 180x120

Good scalability!!

Should we be happy?

(speed up)

Efficiency

Parallel eff

Instr. eff

IPC eff

Parallel eff

LB

uLB

transfer

Efficiency

Parallel eff

Instr. eff

IPC eff
Why efficient?

Parallel efficiency = 93.28
Communication = 93.84

Parallel efficiency = 77.93
Communication = 79.79

Parallel efficiency = 28.84
Communication eff = 30.42
Using Clustering to identify structure

Automatic Detection of Parallel Applications Computation Phases (IPDPS 2009)
What should I improve?

What if ....

... we increase the IPC of Cluster1?

... we balance Clusters 1 & 2?

PEPC

13% gain

19% gain
Tracking scalability through clustering

- OpenMX (strong scale from 64 to 512 tasks)
Folding

- Instantaneous metrics with minimum overhead
  - Combine instrumentation and sampling
    - Instrumentation delimits regions (routines, loops, ...)
    - Sampling exposes progression within a region
  - Captures performance counters and call-stack references
“Blind” optimization

- From folded samples of a few levels to timeline structure of “relevant” routines

Recommendation without access to source code
CG-POP multicore MN3 study

- Unbalanced MPI application
- Same code
- Different duration
- Different performance
Methodology
Performance analysis tools objective

Help generate hypotheses

Help validate hypotheses

Qualitatively

Quantitatively
First steps

• Parallel efficiency – percentage of time invested on computation
  • Identify sources for “inefficiency”:
    • load balance
    • Communication /synchronization

• Serial efficiency – how far from peak performance?
  • IPC, correlate with other counters

• Scalability – code replication?
  • Total #instructions

• Behavioral structure? Variability?

Paraver Tutorial:
Introduction to Paraver and Dimemas methodology
BSC Tools web site

- tools.bsc.es
  - downloads
    - Sources / Binaries
    - Linux / windows / MAC
  - documentation
    - Training guides
    - Tutorial slides

- Getting started
  - Start wxparaver
  - Help → tutorials and follow instructions
  - Follow training guides
    - Paraver introduction (MPI): Navigation and basic understanding of Paraver operation
Same code, different behaviour

<table>
<thead>
<tr>
<th>Code</th>
<th>Parallel efficiency</th>
<th>Communication eff.</th>
<th>Load Balance eff.</th>
</tr>
</thead>
<tbody>
<tr>
<td>lulesh@mn3</td>
<td>90.55</td>
<td>99.22</td>
<td>91.26</td>
</tr>
<tr>
<td>lulesh@leftraru</td>
<td>69.15</td>
<td>99.12</td>
<td>69.76</td>
</tr>
<tr>
<td>lulesh@uv2 (mpt)</td>
<td>70.55</td>
<td>96.56</td>
<td>73.06</td>
</tr>
<tr>
<td>lulesh@uv2 (impi)</td>
<td>85.65</td>
<td>95.09</td>
<td>90.07</td>
</tr>
<tr>
<td>lulesh@mt</td>
<td>83.68</td>
<td>95.48</td>
<td>87.64</td>
</tr>
<tr>
<td>lulesh@cori</td>
<td>90.92</td>
<td>98.59</td>
<td>92.20</td>
</tr>
<tr>
<td>lulesh@thunderX</td>
<td>73.96</td>
<td>97.56</td>
<td>75.81</td>
</tr>
<tr>
<td>lulesh@jetson</td>
<td>75.48</td>
<td>88.84</td>
<td>84.06</td>
</tr>
<tr>
<td>lulesh@claix</td>
<td>77.28</td>
<td>92.33</td>
<td>83.70</td>
</tr>
<tr>
<td>lulesh@jureca</td>
<td>88.20</td>
<td>98.45</td>
<td>89.57</td>
</tr>
<tr>
<td>lulesh@mn4</td>
<td>86.59</td>
<td>98.77</td>
<td>87.67</td>
</tr>
<tr>
<td>lulesh@inti</td>
<td>88.16</td>
<td>98.65</td>
<td>89.36</td>
</tr>
<tr>
<td>lulesh@archer</td>
<td>88.01</td>
<td>97.95</td>
<td>89.86</td>
</tr>
<tr>
<td>lulesh@romeo</td>
<td>89.56</td>
<td>99.01</td>
<td>90.45</td>
</tr>
<tr>
<td>lulesh@mn4</td>
<td>91.02</td>
<td>98.38</td>
<td><strong>92.52</strong></td>
</tr>
<tr>
<td>lulesh@ stampede2 (skl)</td>
<td>85.76</td>
<td>97.63</td>
<td>87.84</td>
</tr>
<tr>
<td>lulesh@ stampede2 (knl)</td>
<td>89.21</td>
<td>98.42</td>
<td>90.64</td>
</tr>
<tr>
<td>lulesh@isambard</td>
<td>90.32</td>
<td>97.16</td>
<td>92.96</td>
</tr>
</tbody>
</table>

Warning::: Higher parallel efficiency does not mean faster!