On-line detection of large-scale parallel application's structure

UPC

BSC Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Germán Llort
gllort@bsc.es

Tools scalability

- >10k processes + long runs = large traces
- Blind tracing is not an option
- Profilers also start presenting issues
- Can you even store the data?
- How patient are you?

Work with large traces

- Past methodology: Filters driven by the expert
  - Get the whole trace
  - Summarize for a global view
  - Focus on a representative region

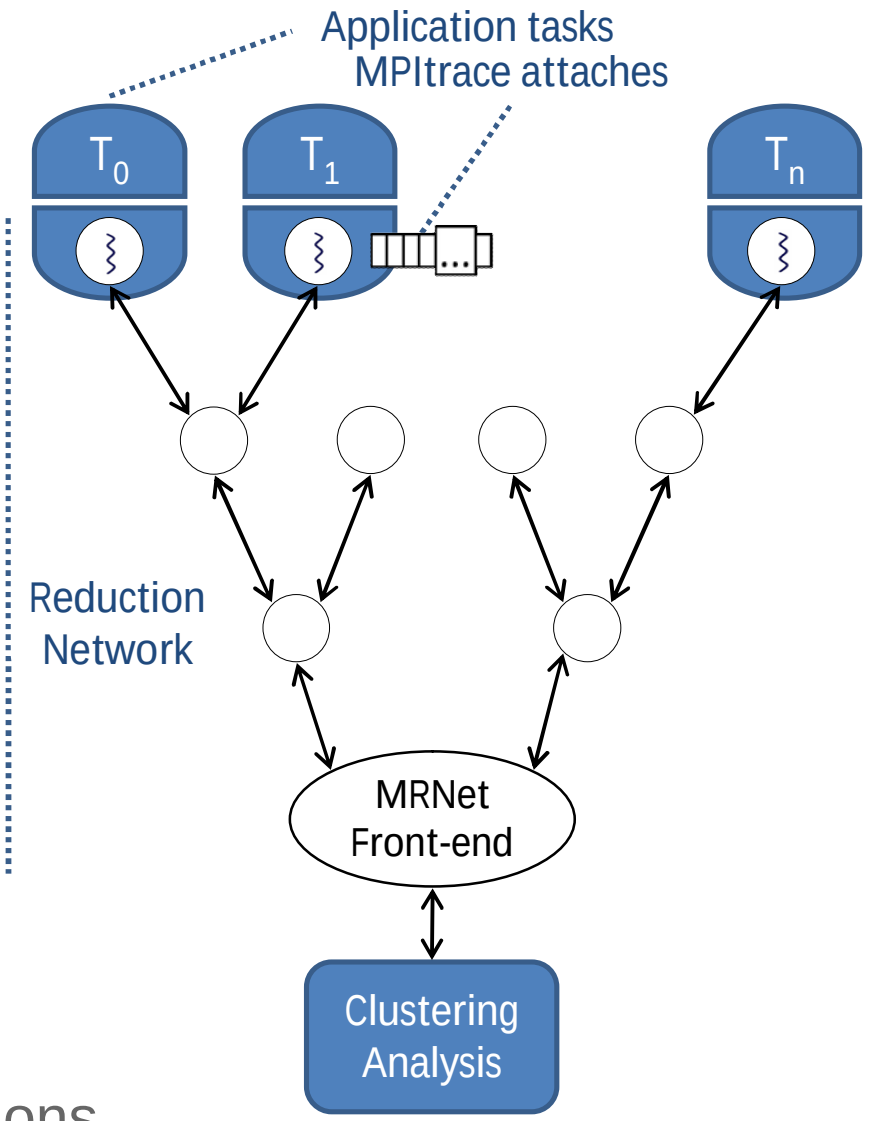
- Goal: Transfer the expertise to the run-time

Objectives

- Traces of “100 Mb”
  - Best describe the application behavior
  - Trade-off: Maximize information / data ratio
- The challenge?
  - Intelligent selection of the information
- How?
  - On-line analysis framework
    - Decide at run-time what is most relevant

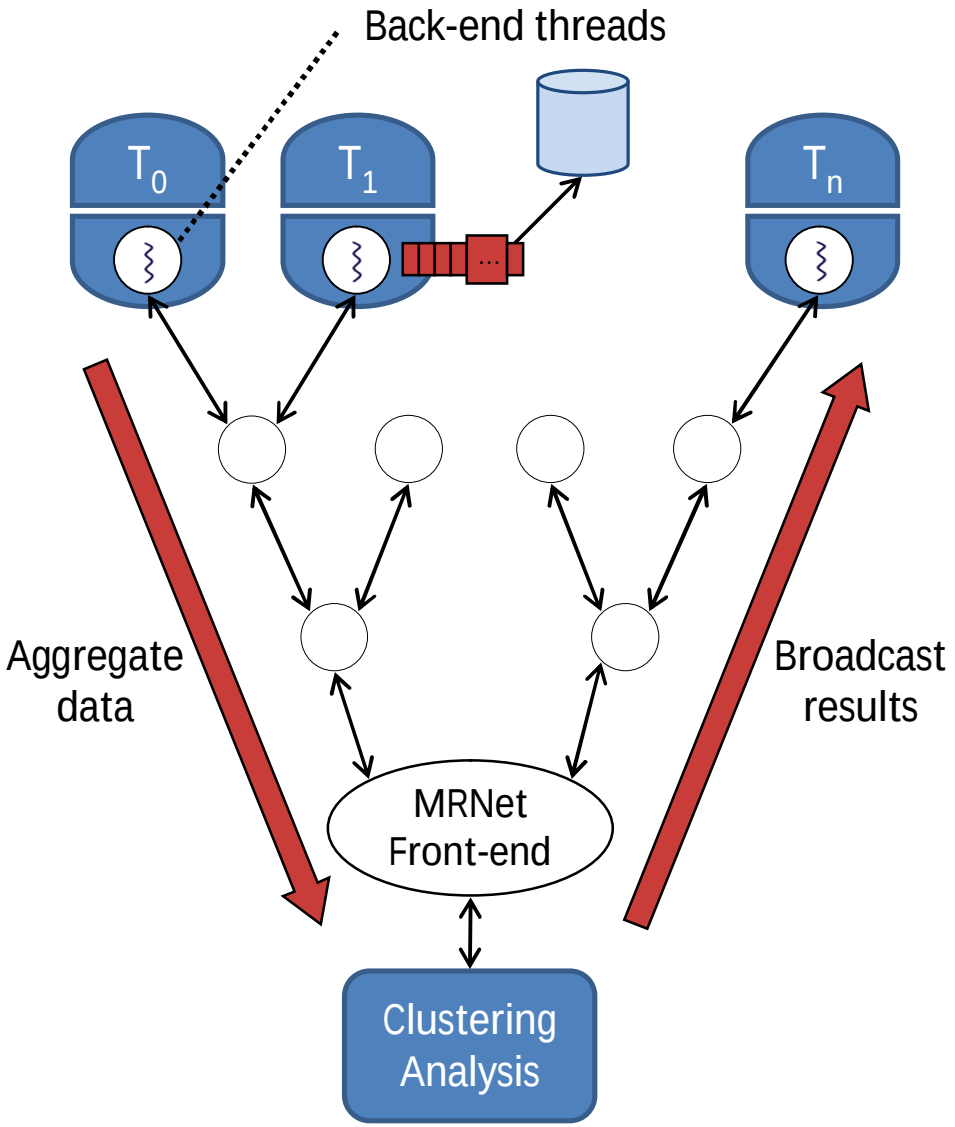
Modules integration

- Data acquisition
  - MPItrace (BSC)
    - PMPI wrappers
- Data transmission
  - MRNet (U. of Wisconsin)
    - Scalable master / worker
    - Tree topology
- Data analysis
  - Clustering (BSC)
    - Find structure of computing regions



Modules interaction

- Local trace buffers
- BE threads blocked
- FE periodically collects data
  - Automatic / fixed interval
  - Reduction on tree
- Global analysis
- Propagate results
- Locally emit trace events



Clustering analysis

- Density-based clustering algorithm
  - J. Gonzalez, J. Gimenez, J. Labarta – IPDPS'09

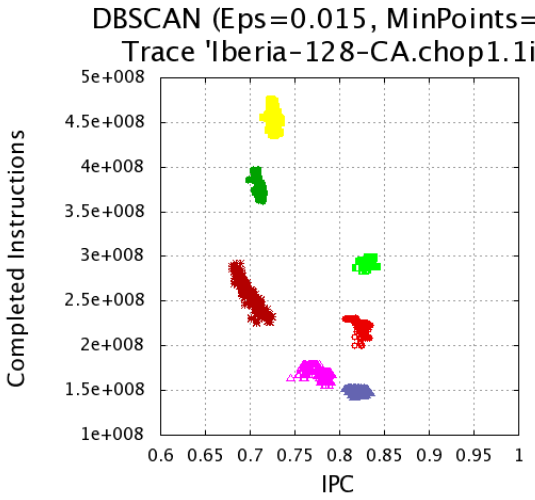
  "Automatic detection of parallel applications computation phases"

- Characterize structure of computing regions

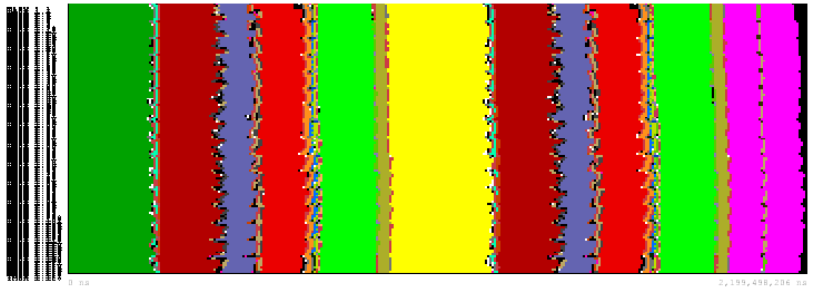
- Using hardware counters data
  - Instructions + IPC
    - Complexity & Performance
  - Any other metric
    - i.e. L1, L2 cache misses

Clustering results

Scatter Plot of Clustering Metrics



Clusters Distribution Over Time



Clusters Performance

Cluster	1	2	3	4	5	6
% Time	36.29	29.52	10.13	9.68	3.73	1.71
Avg. Burst Dur. (ms)	220.46	177.70	60.81	29.09	38.71	44.83
IPC	0.53	0.50	0.62	0.77	0.66	0.59
MIPS	1210.07	1164.36	1403.19	1743.32	1499.47	1338.24
L1M/KInstr	22.72	32.63	12.65	8.39	16.12	6.86
L2M/KInstr	0.59	1.23	1.08	0.61	1.23	1.73
Mem.BW (MB/s)	90.77	182.65	193.32	136.33	236.15	295.71

Code Linking

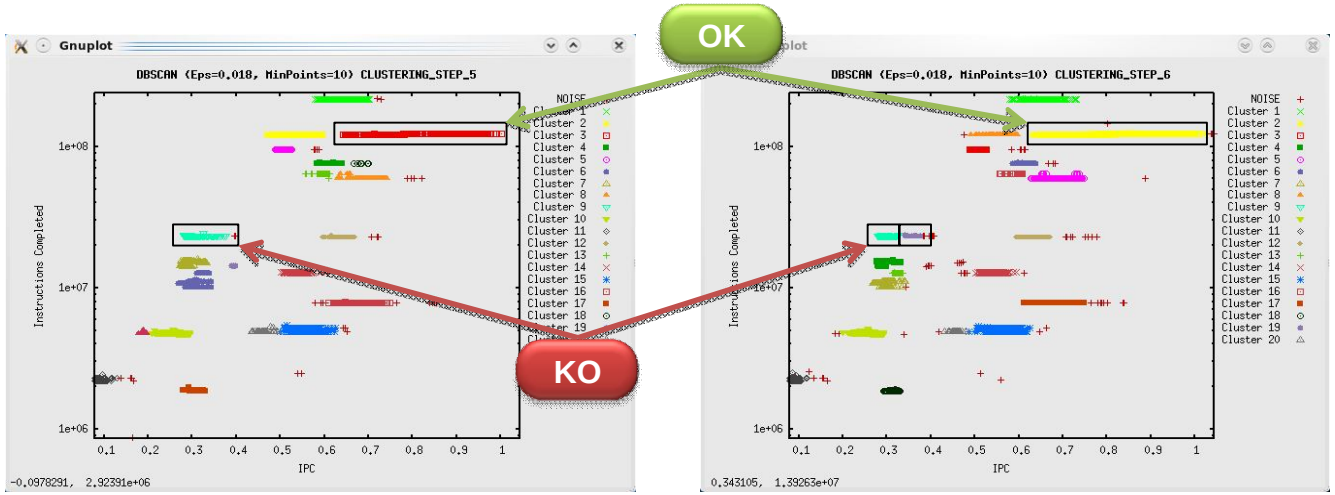
Cluster	Code Section
1	solve_nmm.f:[2037 - 2310]
2	solve_nmm.f:[1478 - 1782] solve_nmm.f:[2030 - 1782]
3	solve_nmm.f:[1241 - 1345]
4	solve_nmm.f:[2771 - 2865] solve_nmm.f:[2388 - 2489]
5	solve_nmm.f:[1478 - 1569]
6	solve_nmm.f:[1607 - 1633]

Tracking evolution

- Trigger clustering analysis periodically
  - Sequence of structure snapshots
- Compare subsequent clusterings
  - See changes in the application behavior
- Find a representative region
  - Most applications are highly iterative

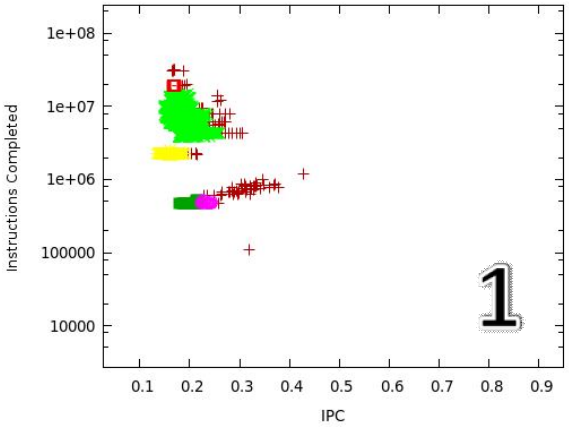
Select representative

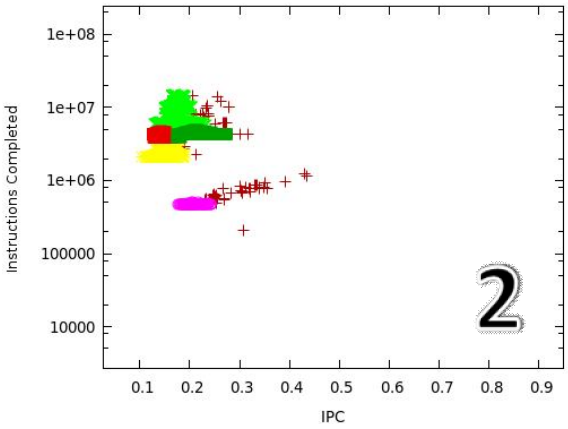
- Compare 2 clusterings, cluster per cluster
  - Inscribe clusters into a rectangle
  - Match those that overlap with a 5% variance
  - Sum of the matched clusters cover the 85% of total computing time

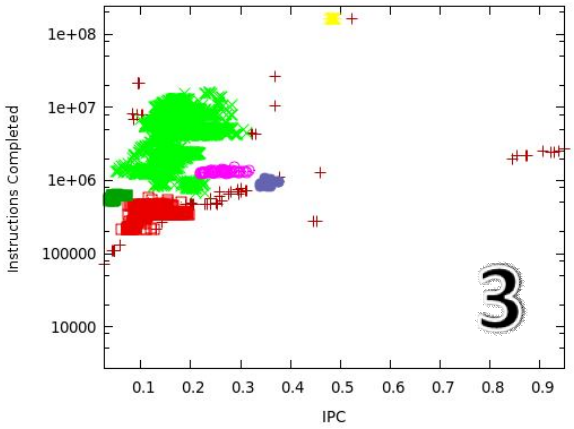


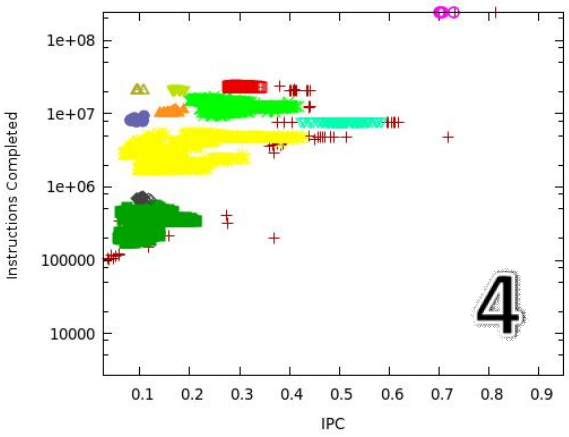
- Stability = N equivalent clusterings “in-a-row”
  - Keep on looking for differences
- Gradually lower requisites if can not be met
  - Best possible region based on “seen” results

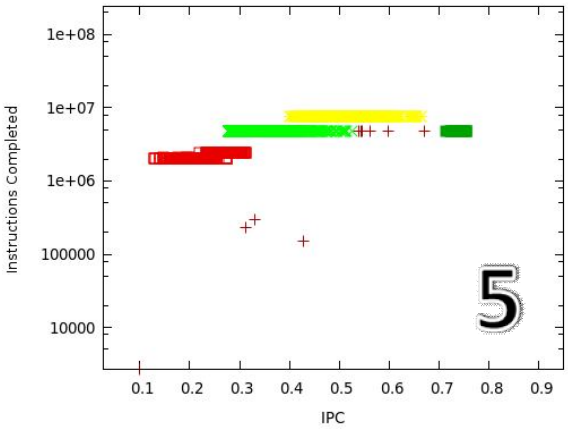
Evolution of MILC

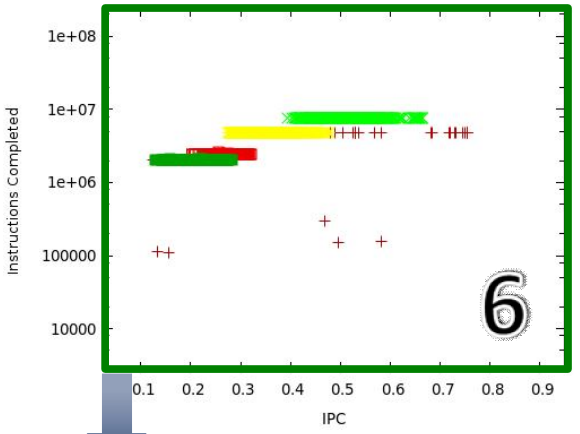


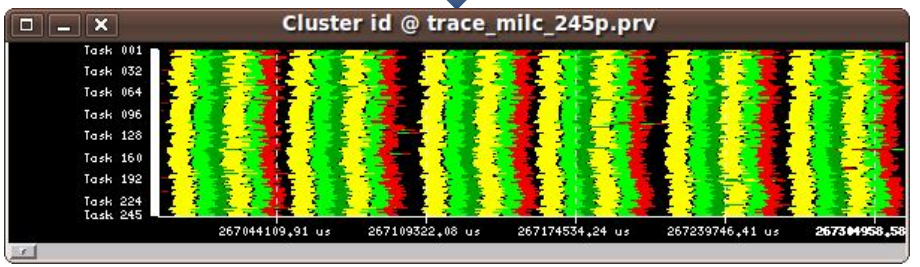












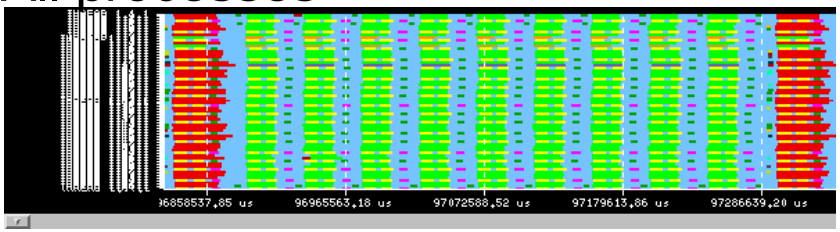
- 60 Mb, 6 iterations

Clustering vs. Classification

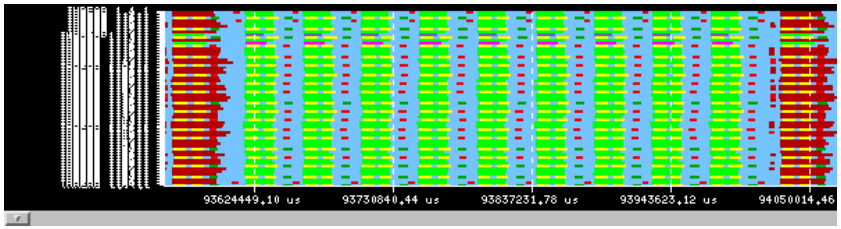
- Clustering time grows with the number of points
  - 5k pts → 10 sec, 50k pts → 10 min
- Sample a subset of data to cluster (SDBScan)
  - Space: Select a few processes. Full time sequence.
  - Time: Random sampling. Wide covering.
- Classify remaining data
  - Nearest neighbor algorithm
    - Reusing clustering structures

Clustering vs. Classification

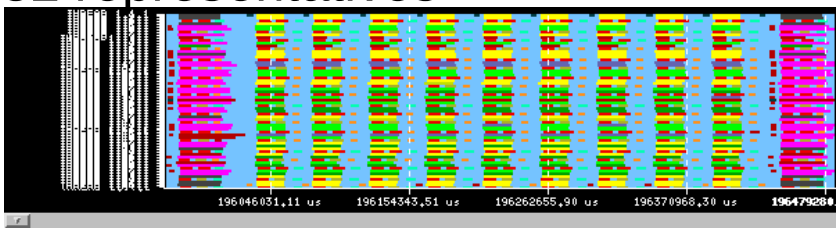
All processes



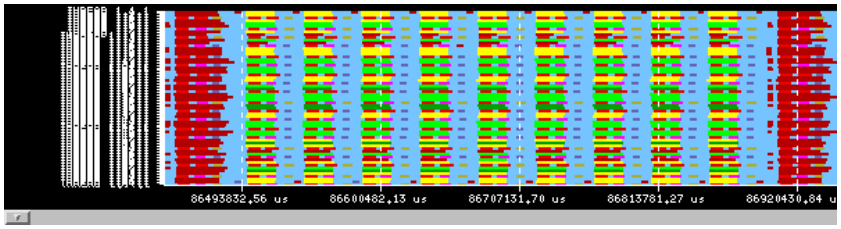
25% random records



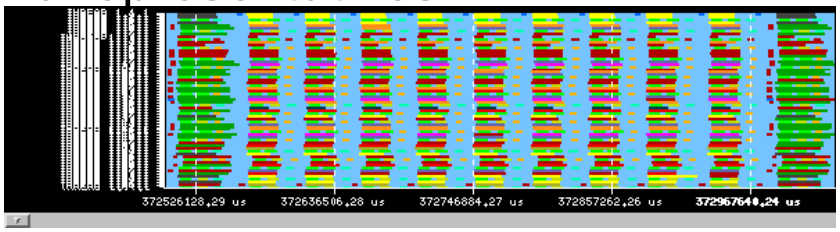
32 representatives



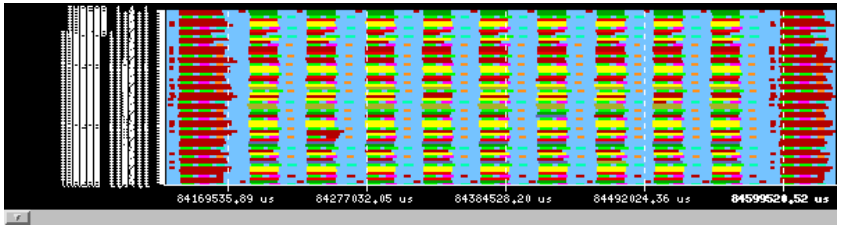
15% random records



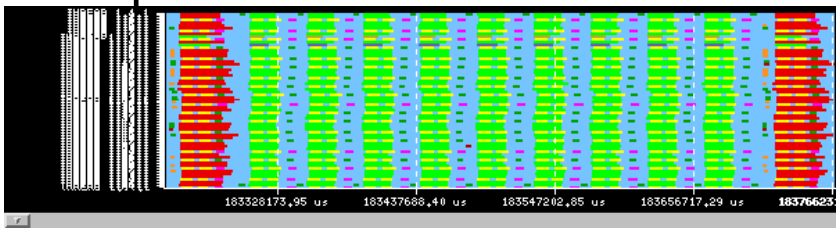
16 representatives



10% random records



8 representatives + 15% random



75% less data
6s down from 2m

Good quality
Fast analysis

Experiments

Metric	SPECFEM3D	Gromacs	MILC	Zeus-MP	Leslie3D
Number of tasks	64	64	245	256	512
Requested trace size	100 Mb	200 Mb	200 Mb	350 Mb	600 Mb
Full run time	58 m	8 m	5.5 m	10 m	56 m
Full trace size	3 Gb	20 Gb	5.5 Gb	22 Gb	82 Gb
Analysis steps	6	8	7	4	8
Clustering time/step	0.6 s	4.5 s	5.85 s	1 s	60 s
Classification time	1 s	1 s	1 s	1 s	3 s
Time to get results	15 m	2.5 m	3 m	3 m	12 m
Traced iterations	35	10	20	4	8
Time/iteration	3 s	500 ms	60 ms	1 s	100 ms

- Important trace size reductions
- Results before the application finishes
- Final trace is representative

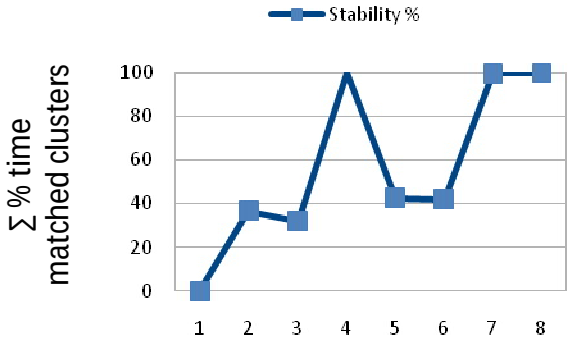
Quality of the trace segment

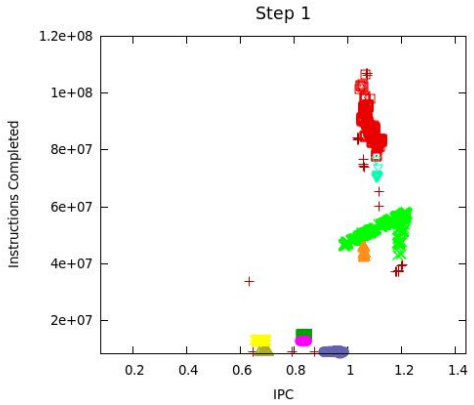
- Compared vs. Profiles for the whole run
  - TAU Performance System (U. of Oregon)

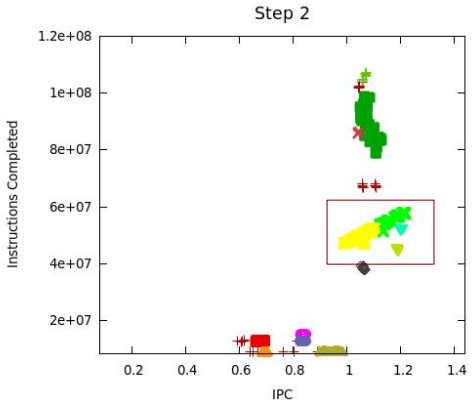
- Same overall structure
  - Same relevant functions, Avg. HWC's & Time %
  - Most measurement differences under 1%

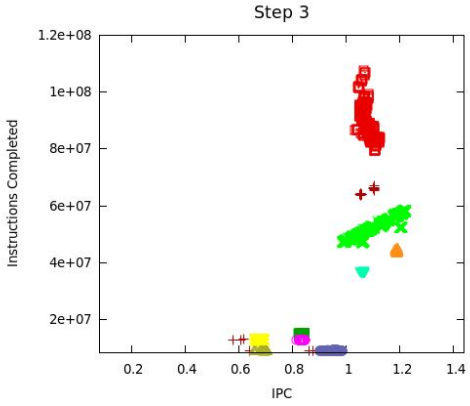
	Full run profile (TAU)			Trace segment (MPItrace)		
GROMACS user functions	% Time	Kinstr	Kcycles	% Time	Kinstr	Kcycles
do_nonbonded	23.72%	24,709	22,349	23.94%	24,700	22,533
solve_pme	10.47%	6,795	9,913	10.52%	6,776	9,898
gather_f_bsplines	5.69%	5,286	5,387	5.64%	5,248	5,302

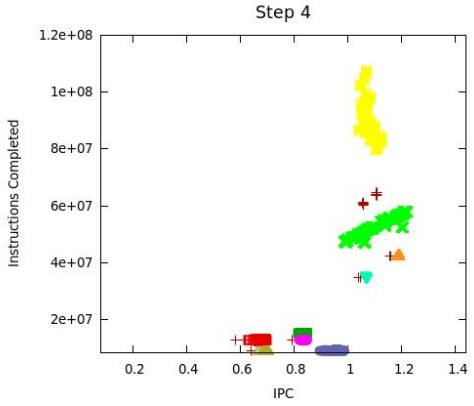
Example: GROMACS evolution

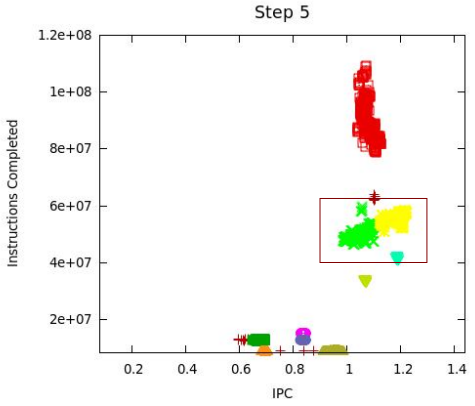


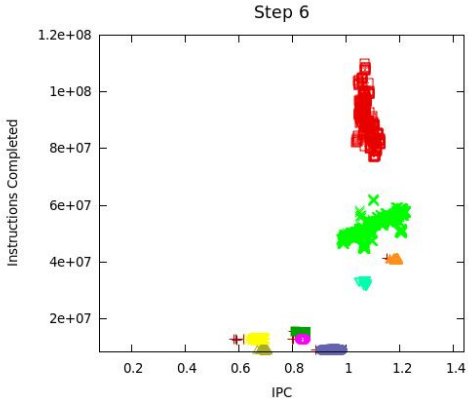


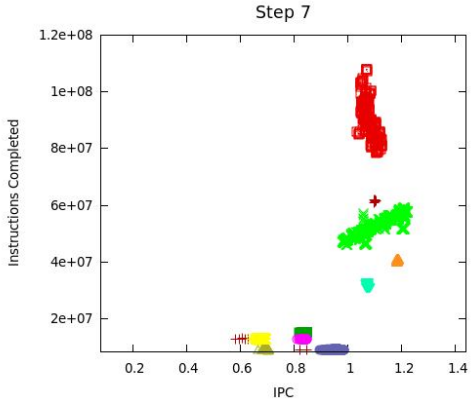


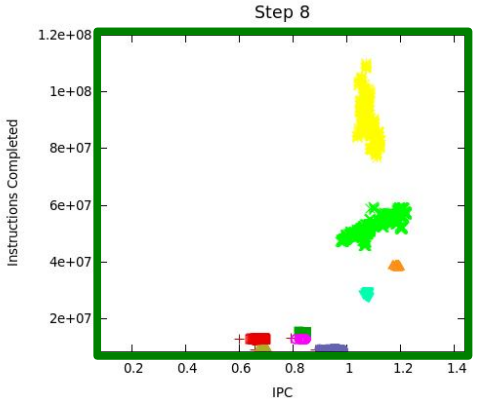




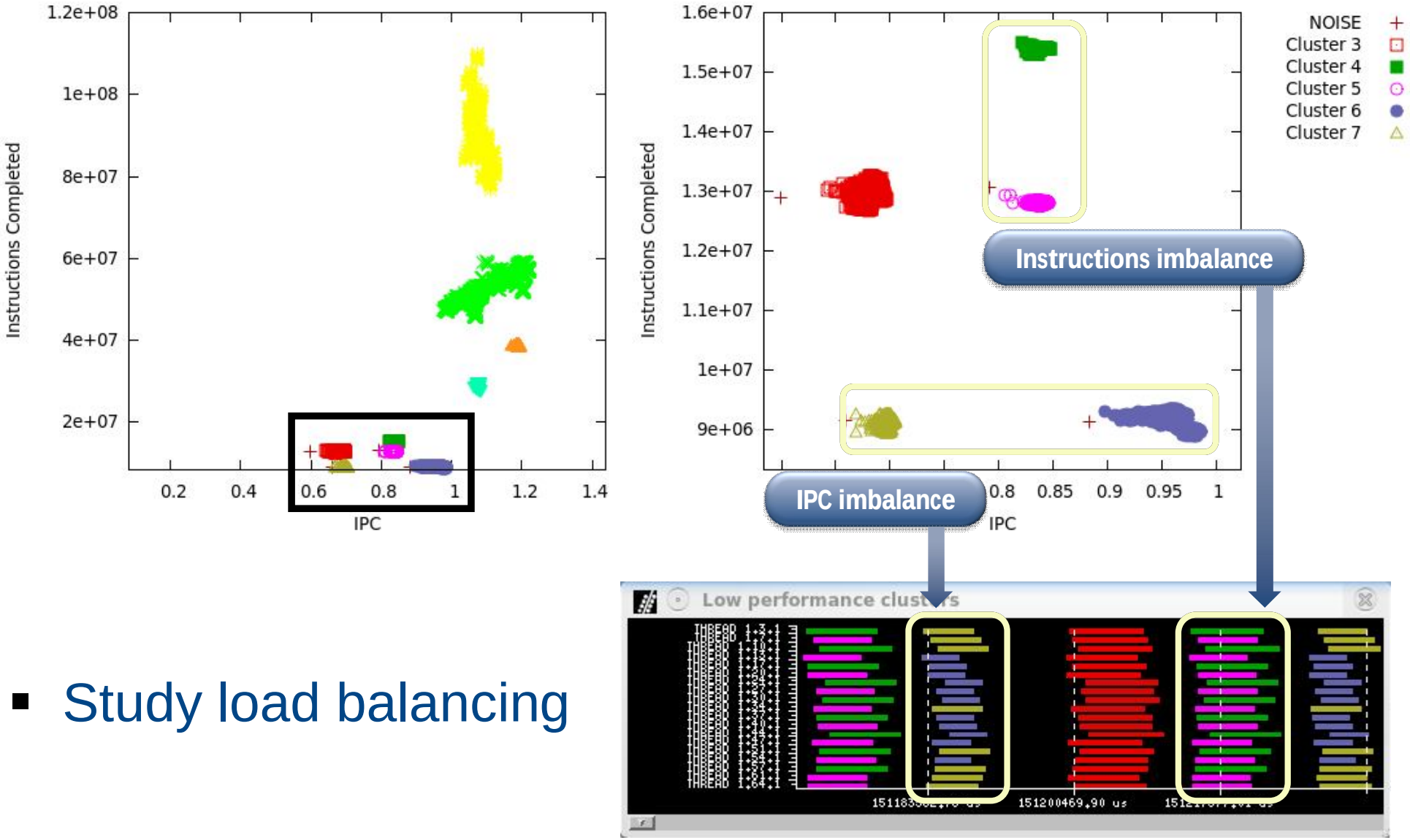








Example: GROMACS structure



- Study load balancing

Increasing scale

- Initial development
  - All data centralized
  - Sampling, clustering & classification at front-end
  - Bad scaling at large processor counts

- >10k tasks
  - Sampling at leaves
  - Only put together the clustering set
  - Broadcast clustering results, classify at leaves

Conclusion

- On-line automatic analysis framework
- Identify structure and see how evolves
- Determine a representative region
- Detailed small trace + Periodic reports
- Reductions in the time dimension
- Scalable infrastructure supports other analyses

- Current work
  - Spectral analysis (M. Casas): Better delineate the traced region
  - Parallel clustering in the tree
  - Finer stability heuristic


Thank you for
your attention!