Performance analysis with Periscope

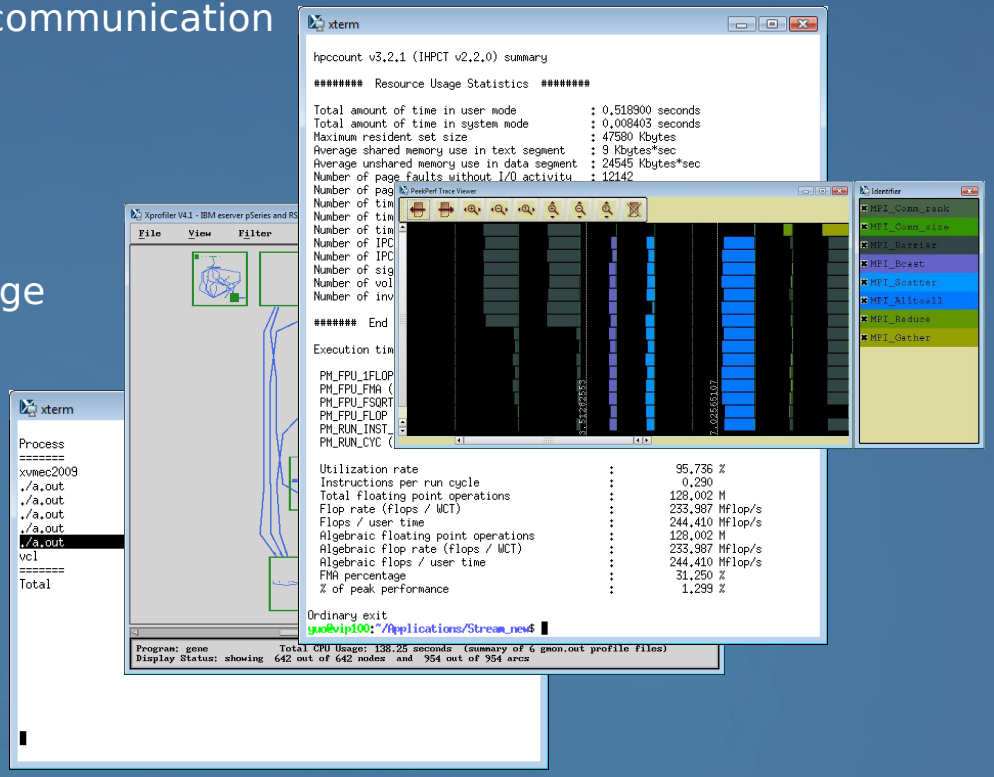
M. Gerndt, V. Petkov,
Y. Oleynik, S. Benedict
Technische Universität München
petkovve@in.tum.de
March 2010

Outline

- Motivation
- Periscope (PSC)
- Periscope performance analysis model
- Tool architecture
- Performance analysis automation
- Periscope GUI

Motivation

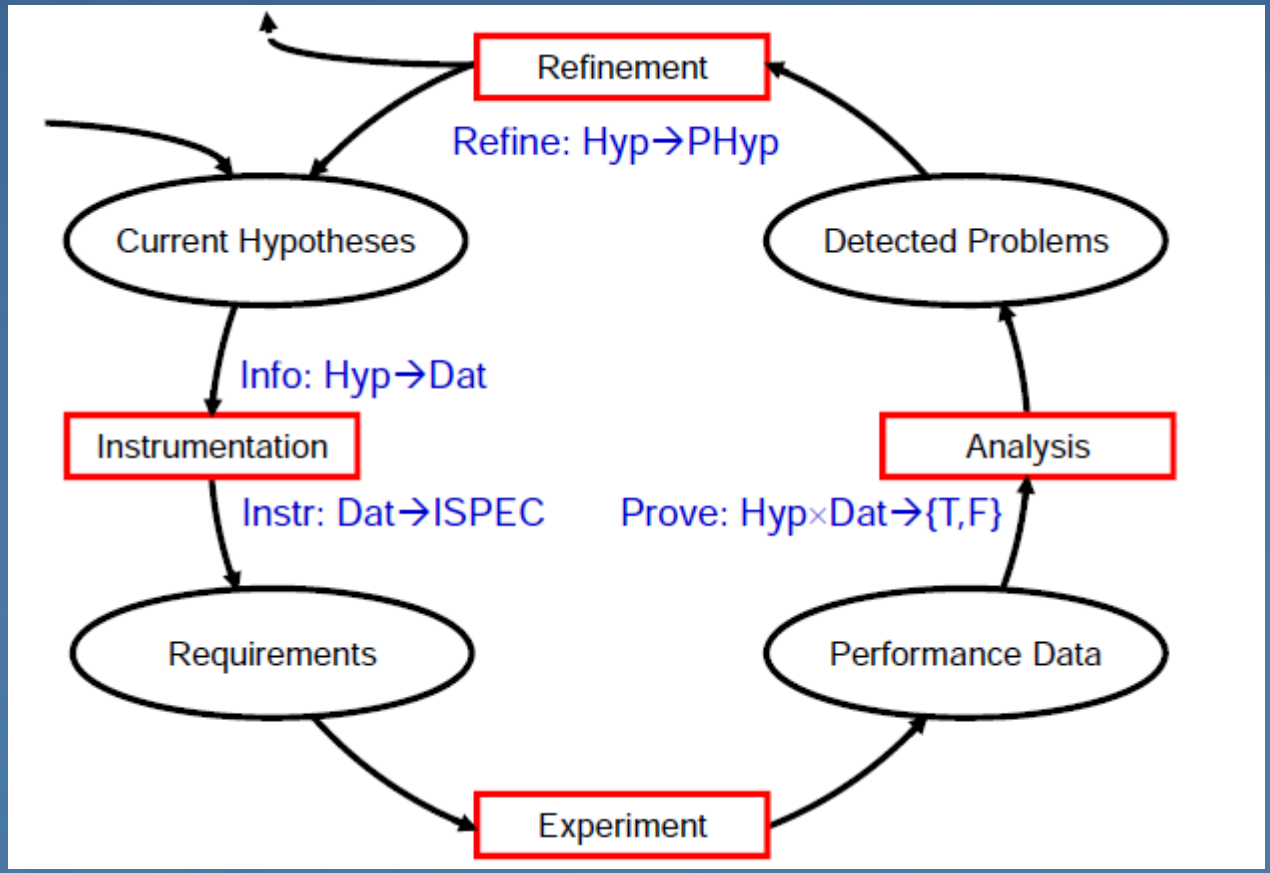
- Currently performance analysis procedure on POWER6 example:
  - Use *Tprof* to pinpoint time consuming subroutines
  - Use *Xprofiler* (GUI for gprof) to understand call graph
  - Use *hpmcount* (libhpm) to measure Hardware Counters
  - Use *mpitrace* to investigate mpi communication
- Problems:
  - Routine, time consuming
  - Error prone
  - Not scalable
  - Requires deep hardware knowledge
- Solution:
  - Performance analysis automation



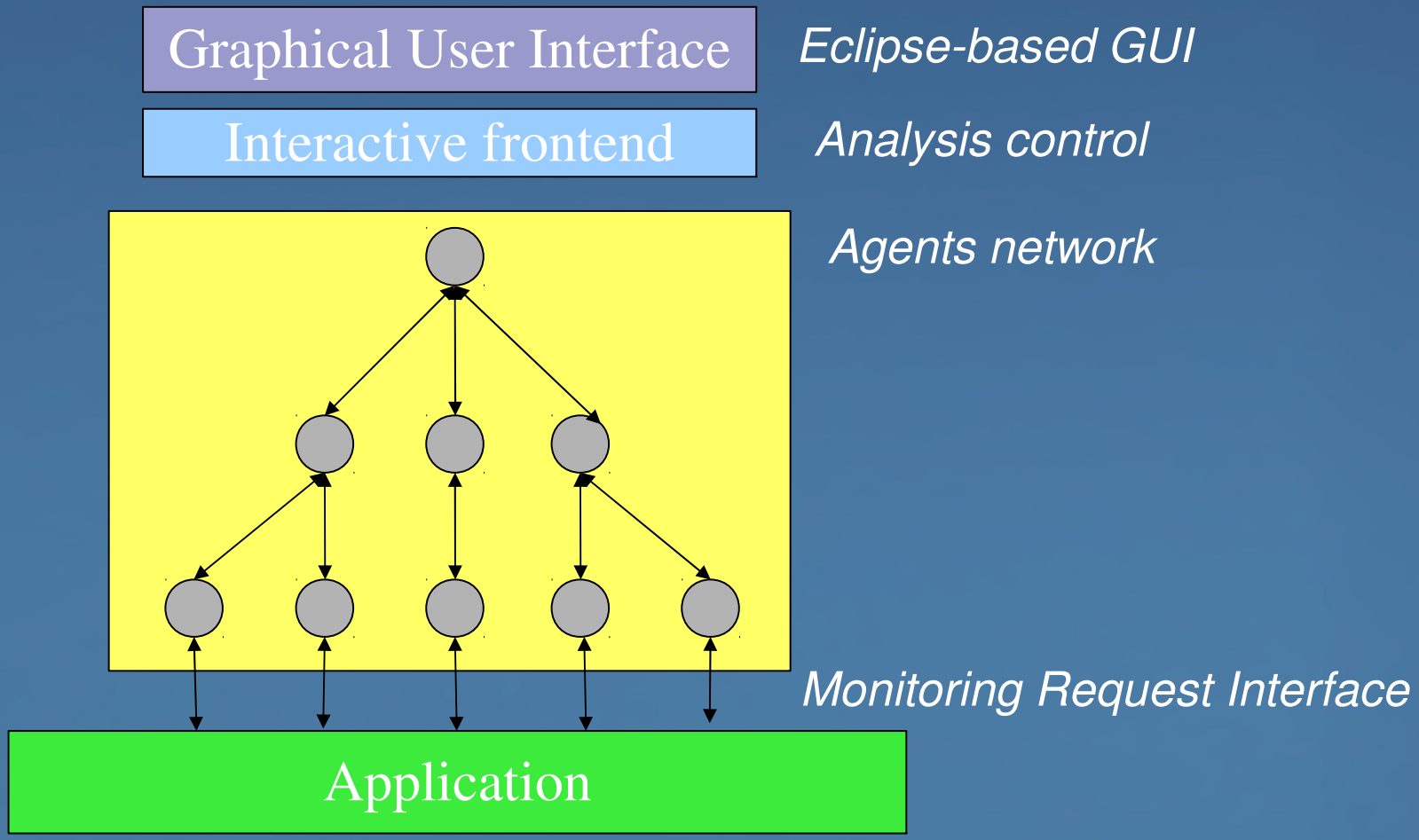
Periscope

- Iterative online analysis
  - Measurements are configured, obtained and evaluated on the fly
  - no tracing!
- Distributed architecture
  - Analysis performed by multiple distributed hierarchical agents
- Automatic bottlenecks search
  - Based on performance optimization experts' knowledge
- Enhanced GUI
  - Eclipse based integrated development and performance analysis environment
- Instrumentation
  - Fortran, C
- Supported architectures
  - SGI Altix 4700 Itanium2
  - IBM Power575 POWER6
  - x86 architectures
- Current version 1.0.1
  - Available under: http://www.lrr.in.tum.de/periscope

Iterative online analysis model



Distributed architecture



Automatic search for bottlenecks

- Automation based on formalized expert knowledge:
  - Potential performance problems → properties
  - Efficient search algorithm → search strategies
- Performance property
  - Condition
  - Confidence
  - Severity
- Performance analysis strategies
  - Itanium2 Stall Cycle Analysis
  - IBM POWER6 Single Core Performance Analysis
  - MPI Communication Pattern Analysis

POWER6 Single Core Performance Properties

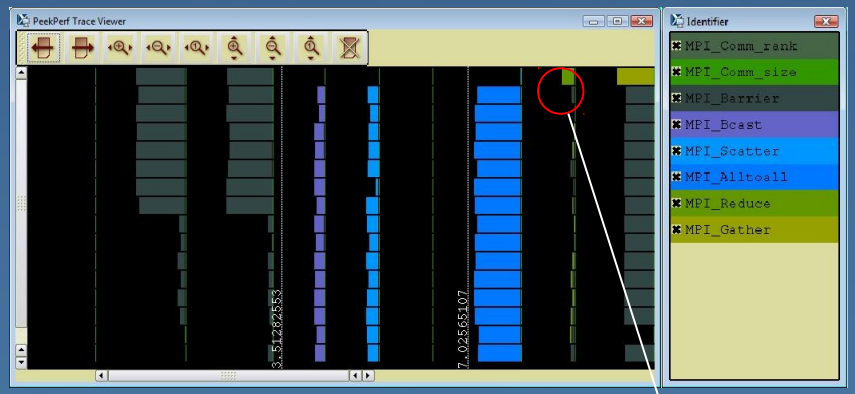
- Hot spot of the application
  - Cycles lost due to cache misses
    - Average amount of cycles lost per L1 miss
    - High L1 demand load miss rate
    - High L2 demand load miss rate
    - High L3 demand load miss rate
  - Cycles lost due to address translation misses
  - Cycles lost due to store instructions
  - Cycles lost due to Floating Point instructions inefficiencies
  - Cycles lost due to Integer multiplications and divisions
  - Cycles lost due to no instruction to dispatch

...

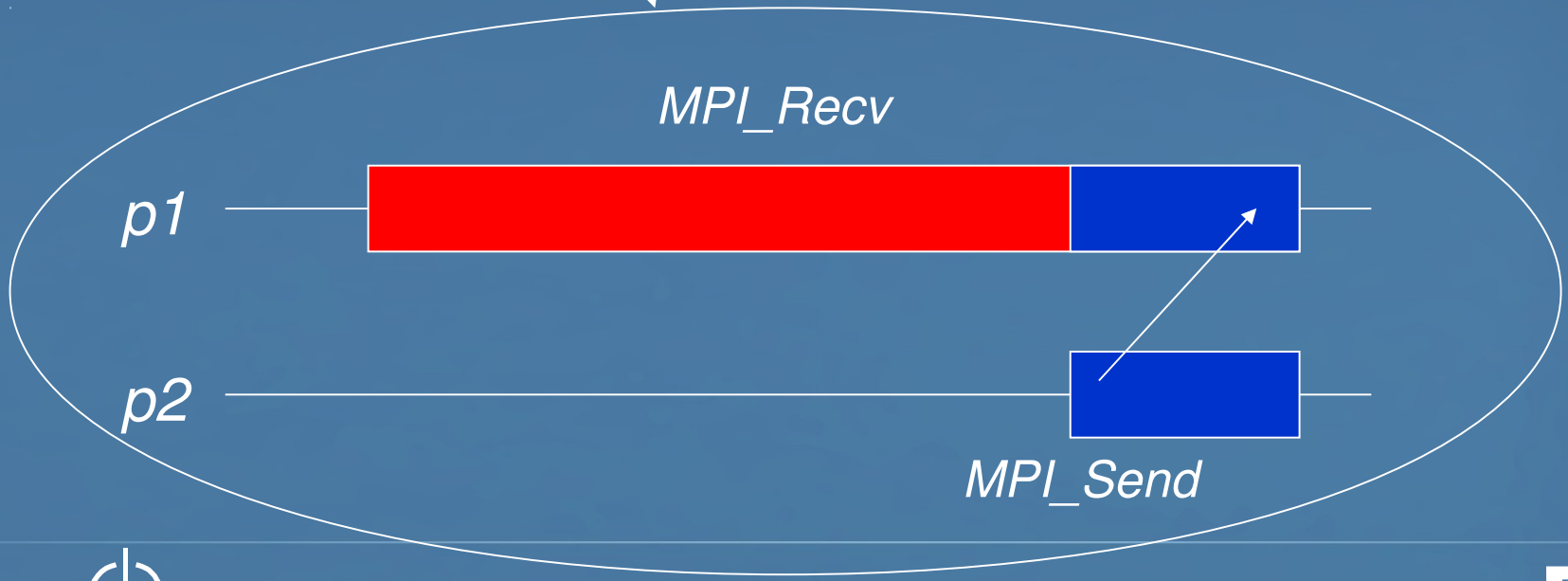
Itanium2 Stall Cycle properties

- IA64 Pipeline Stall Cycles
  - Stalls due to pipeline flush
    - Stalls due to branch misprediction flush
    - Stalls due to exception flush
  - Stalls due to floating point exceptions or L1D TLB misses
    - Stalls due to Flush to zero or SIR stalls
    - Stalls due to L1D TLB misses ...
  - Stalls due to waiting for data delivery to register
    - Stalls due to waiting for integer register
    - Stalls due to waiting for integer results
    - Stalls due to waiting for FP register
    - Stalls due to waiting for integer loads
    - L3 misses dominate data access
    - L2 misses
    - L3 misses
  - Stalls due to register stack engine

MPI Communication Patterns Analysis



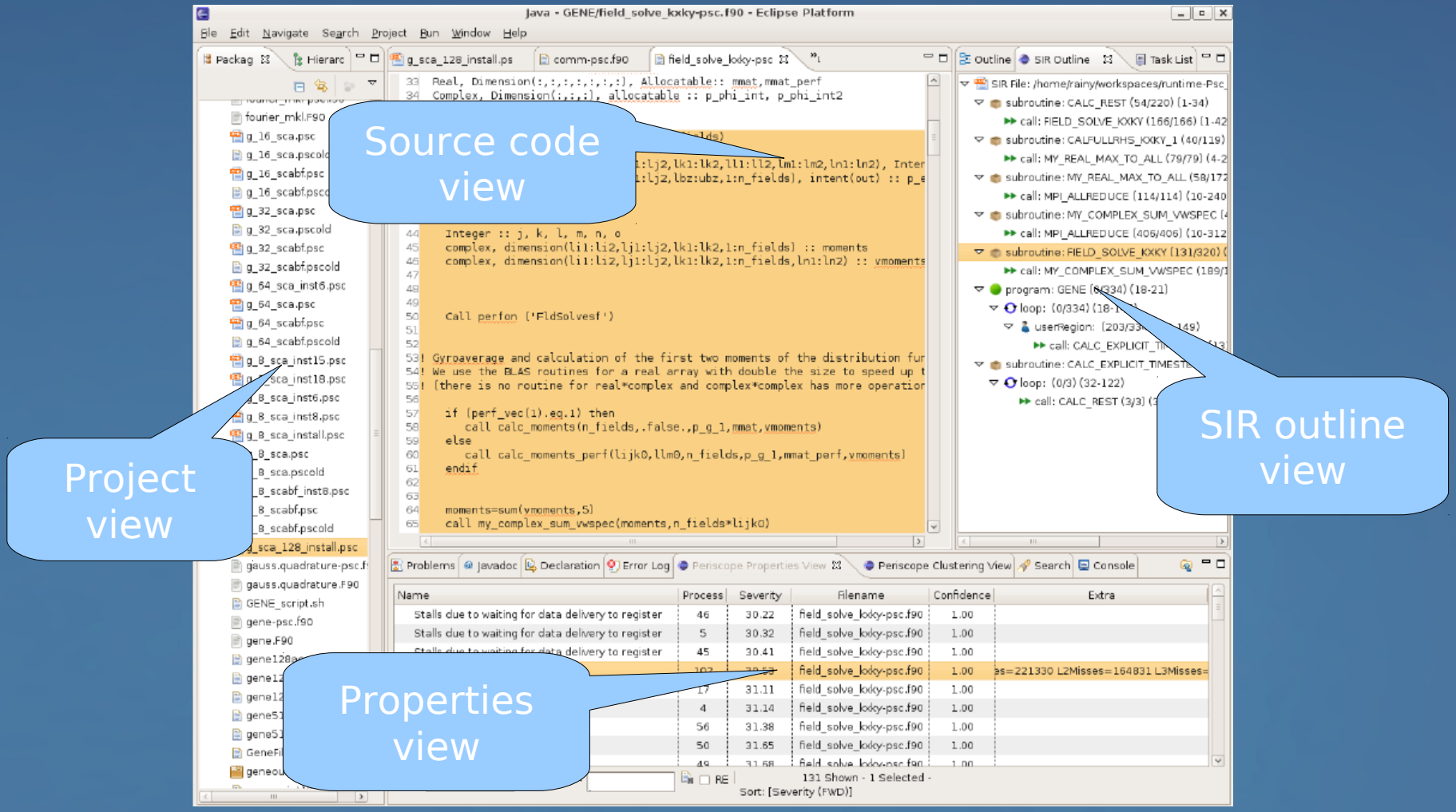
- Automatic detection of wait patterns
- Measurement on the fly
- **No tracing required!**



MPI Performance Properties

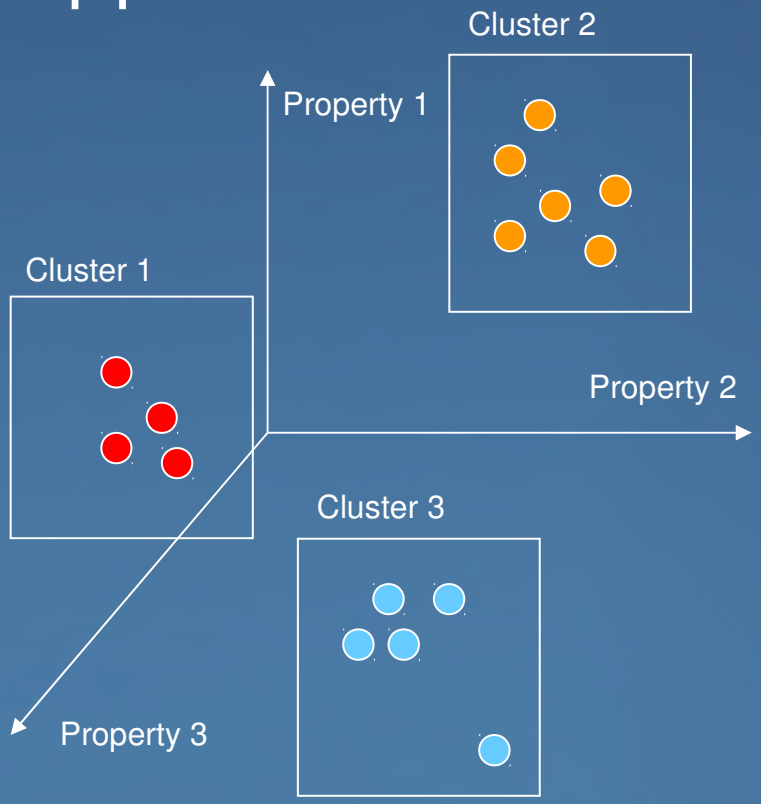
- Excessive MPI time in receive due to late sender
- Excessive MPI time due to late root in broadcast
- Excessive MPI time in root due to late process in reduce
- Excessive MPI time in ... (`1xN`, `Nx1`, `1x1`, `NxN`)
- Excessive MPI time due to many small messages
- Excessive MPI communication time

Graphical User Interface



Clustering support

- Properties summarization
  → Metaproperties
- Needed for peta-scale PA
- Identify *hidden* behavior



Cluster 1 CPUs: 7-10,16

Cluster 2 CPUs: 2-3,5,11,13-14

Cluster 3 CPUs:1,4,6,12,15

Thank you for your attention!

- Further information:
  - Periscope web page: http://www.lrr.in.tum.de/periscope