Build-and-Test Workloads for Grid Middleware
Problem, Analysis, and Applications

PDS Group, EEMCS, TU Delft

Alexandru Iosup and
Dick H.J. Epema

CS Dept., U. Wisconsin-Madison

Peter Couvares,
Anatoly Karp, and
Miron Livny

Grids are far from being reliable job execution environments

Server	•	99.99999% reliable
Small Cluster	•	99.999% reliable

In today's grids, reliability is more important than performance!

DAS-2	•	>10% jobs fail [Ios06]
TeraGrid	•	20-45% failures [Kha06]
Grid3	•	27% failures, 5-10 retries [Dum05]

Sources of failure in grids and what to do about them

1. User applications
   - Failures [HLi07]
   - Error propagation [Tha02]
2. Grid + User Middleware
   - **No large-scale study**
   - Adopt industry practices [NMI06]
3. Machine + OS
   - Desktop grids [Kon04]
   - Resource availability [Ios07]
   - Availability-aware scheduling

Outline

1. Introduction
2. The Build-and-Test problem
3. The NMI Build-and-Test Environment
4. Build-and-Test workloads
    1. Arrival patterns
    2. Workflow structure
    3. Observed failures
5. Applications
6. Conclusion

2. The Build-and-Test problem

- Industry practice for complex software development:

Industry practice:
do (build-and-test environments)
or die!

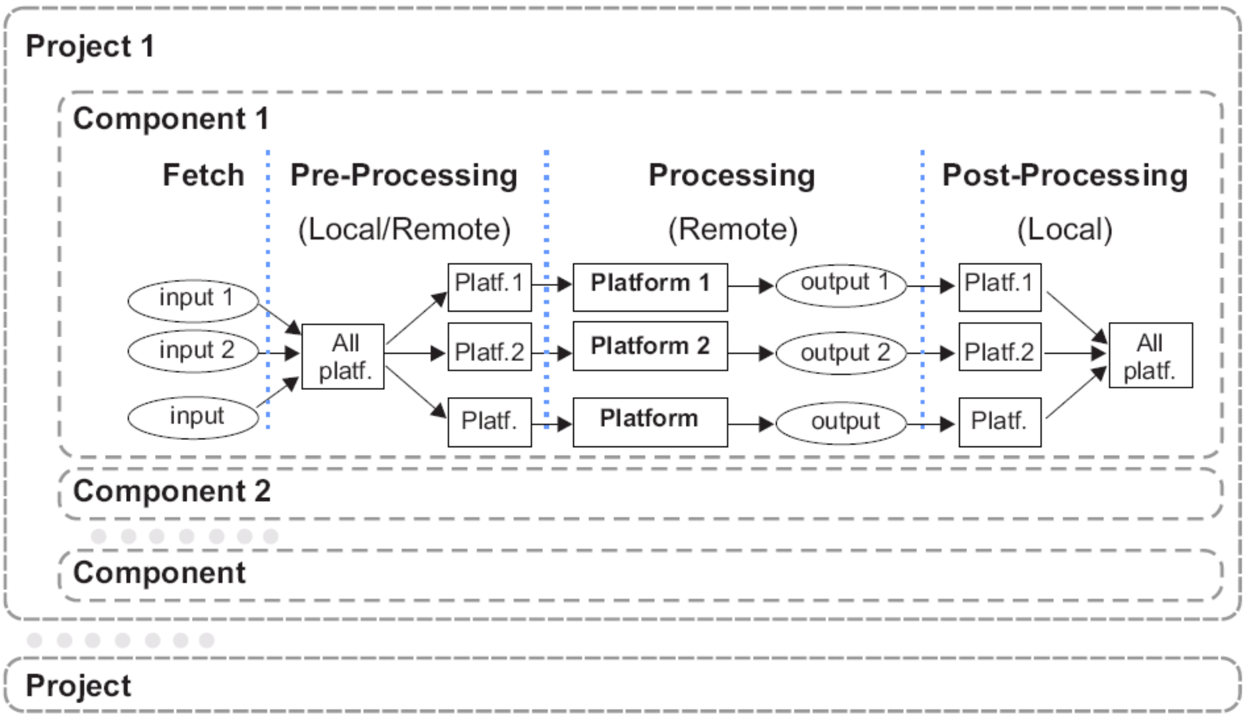
- **Problem**: following industry practice, build a build-and-test environment for grid software development
  - Unknowns (build): workload characteristics, size
  - Unknowns (operation): response time, accuracy

Outline

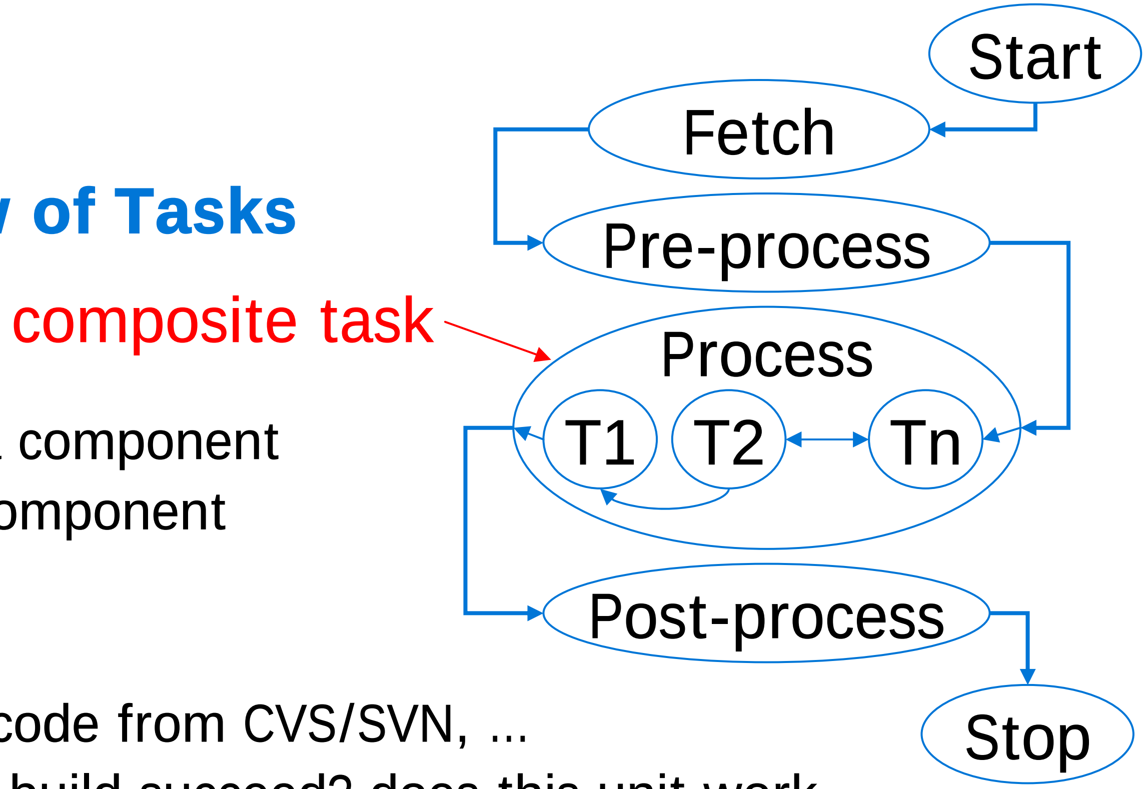
1. Introduction
2. The Build-and-Test problem
3. **The NMI Build-and-Test Environment**
4. Build-and-Test workloads
    1. Arrival patterns
    2. Workflow structure
    3. Observed failures
5. Applications
6. Conclusion

3. NMI, U. Wisc.-Madison: Sample Build-and-Test Environment (1)

- NMI U.Wisc.-Madison: 112 hosts, >40 platforms (e.g., X86/RH/9)
- Serves >50 **projects,** >100 **components**: Condor, Globus, VDT, gLite, GridFTP, RLS, NWS, INCA(-2), APST, NINF-G, BOINC ...



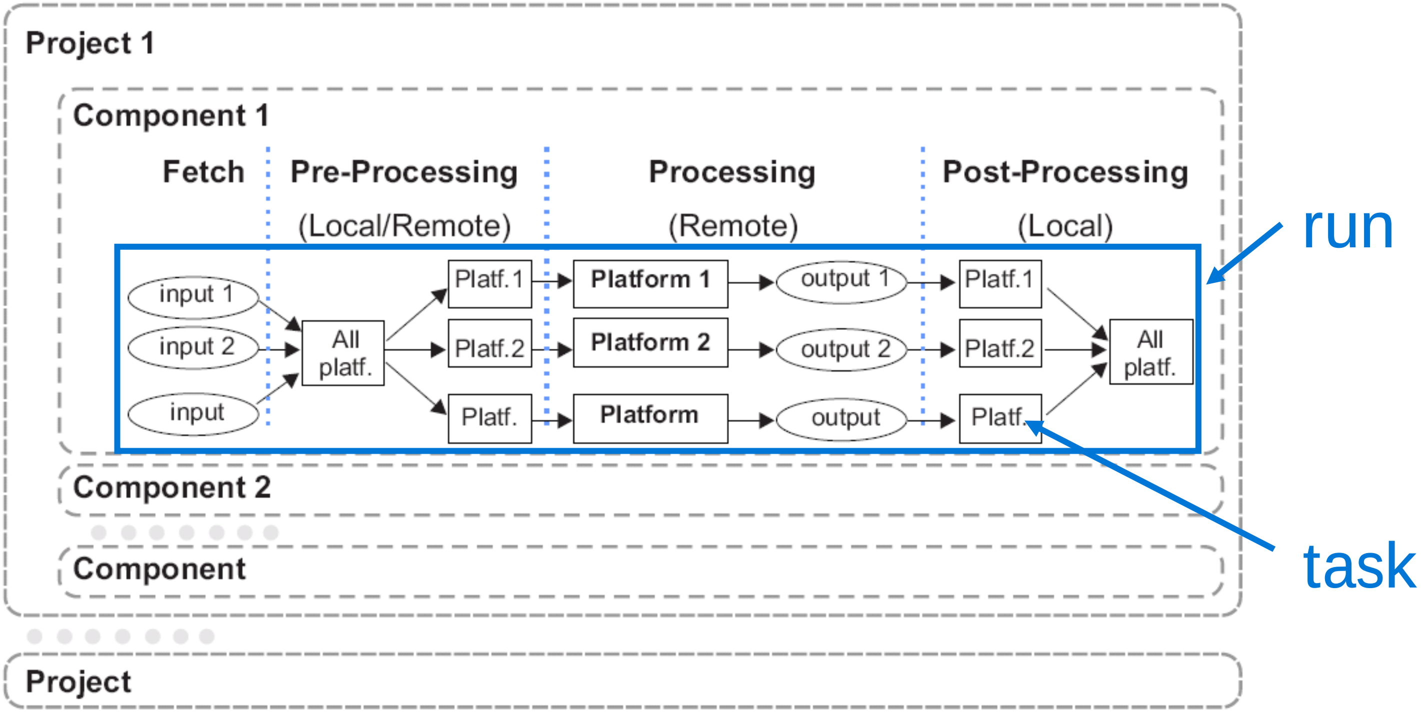
3. NMI, U. Wisc.-Madison: Sample Build-and-Test Environment (2)



- **Run = workflow of Tasks**
- Two Run types
  1. BUILD: building a component
  2. TEST: testing a component
- Many Task types
  - Setup Job: fetch code from CVS/SVN, ...
  - Test Job: did this build succeed? does this unit work according to the specifications?
  - Composite: a suite of setup and/or test jobs

3. NMI, U. Wisc.-Madison: Sample Build-and-Test Environment (3)

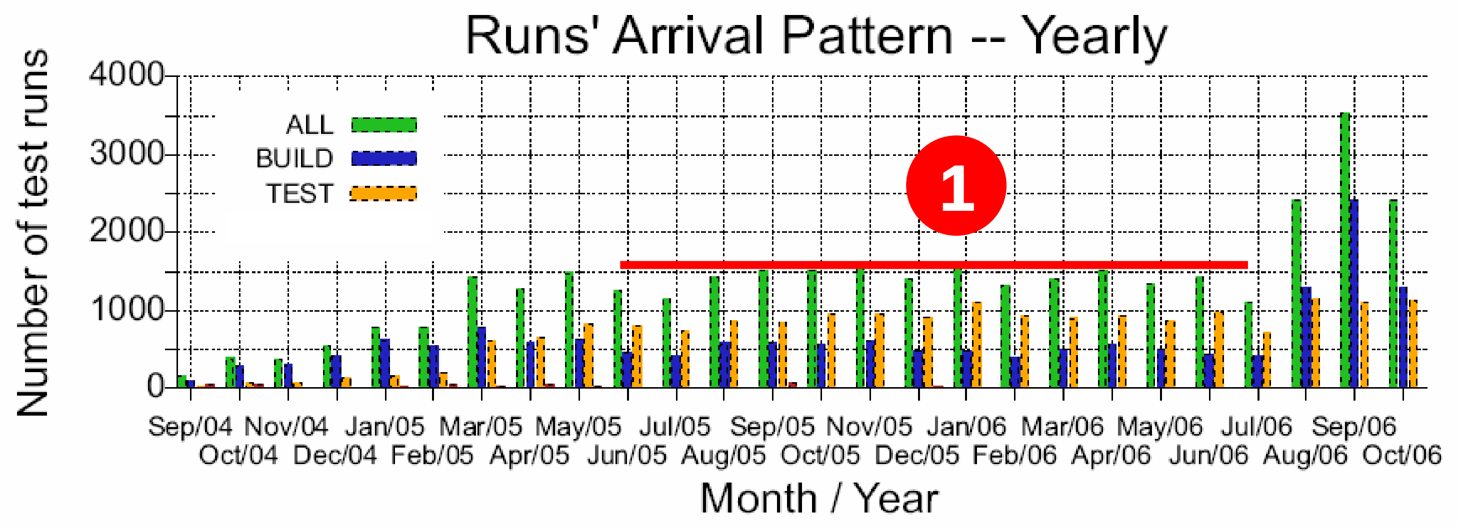
- Traces: 2 years (10/04-11/06), 90 CPUYrs (50% load), 35,000 **runs**, 2,400,000 **tasks** (73% in TEST runs)



Outline

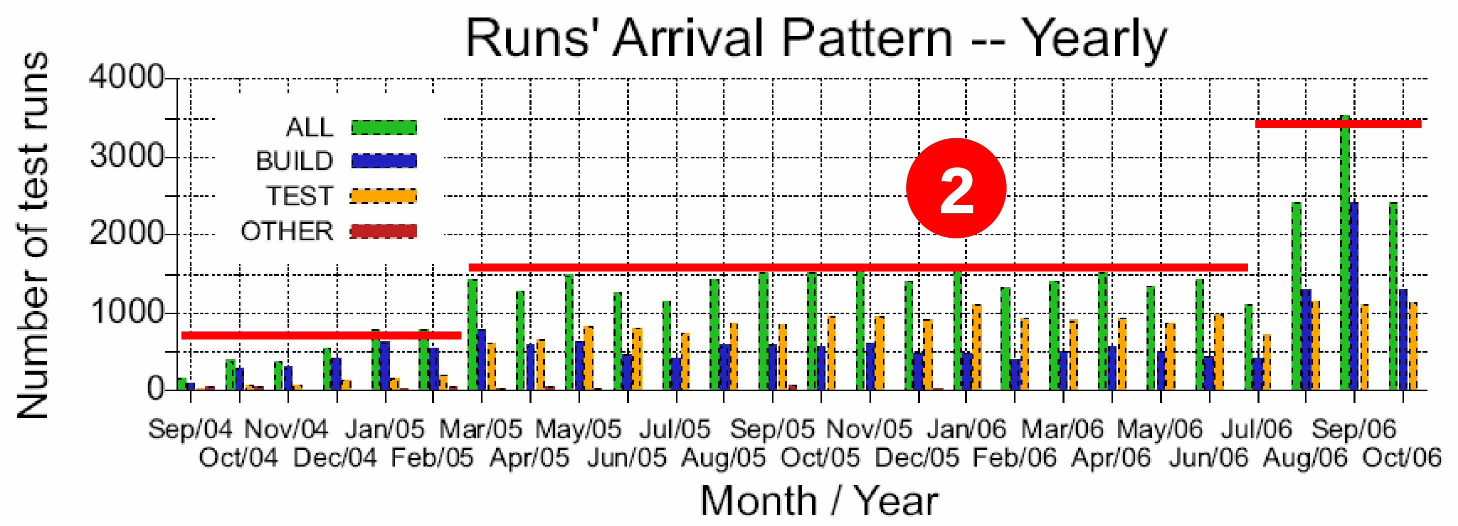
1. Introduction
2. The Build-and-Test problem
3. The NMI Build-and-Test Environment
4. **Build-and-Test workloads**
    1. **Arrival patterns**
    2. **Workflow structure**
    3. **Observed failures**
5. Applications
6. Conclusion

4.1. Build-and-Test Workloads: Arrival Patterns



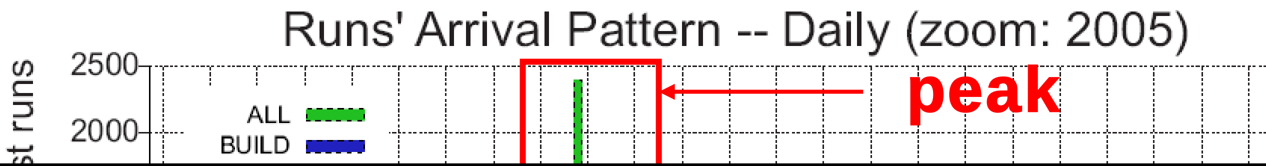
1. Arrival rate is constant throughout the year
   - 1500 runs per month in Jun'05-Jul'06
   - 6-months shift: year starts in July (dead production month)
   - Slow month: -20%, July both in 2005 and 2006

4.1. Build-and-Test Workloads: Arrival Patterns



1. Arrival rate is constant throughout the year
2. System evolution in 12-18 months cycles
   - Level 1: 800 runs per month
   - Level 2: 1500 runs per month
   - Level 3: 3500 runs per month

4.1. Build-and-Test Workloads: Arrival Patterns

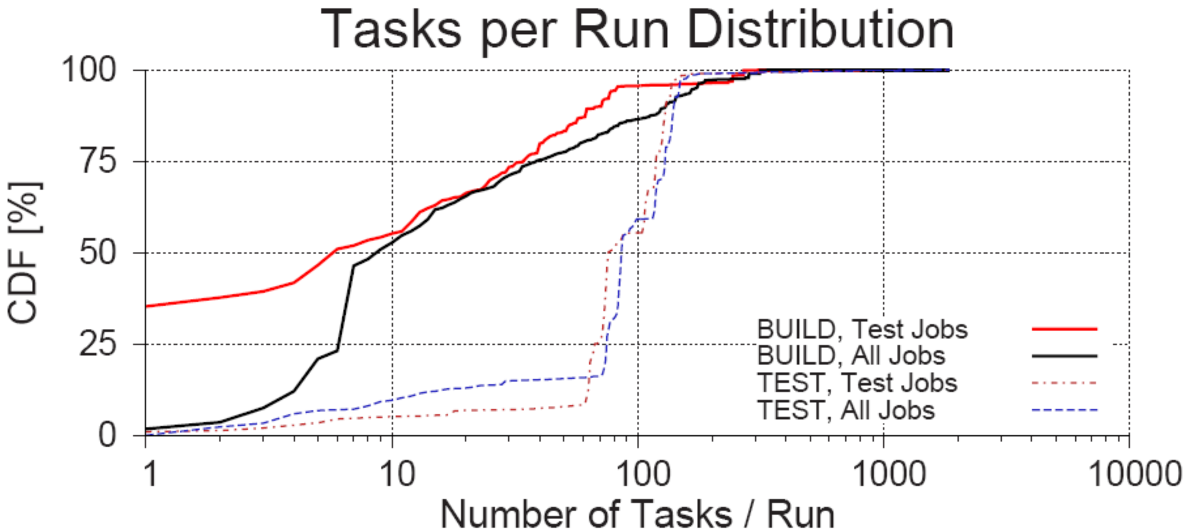


Build-and-Test workloads: heavy load, constantly.

1. Arrival rate is constant throughout the year
2. System evolution in 12-18 months cycles (4x in 2 yrs)
3. Daily pattern (Madison, WI base time: GMT-8)
   - High level of intensity 16h/day
   - Peak hours 09-10GMT

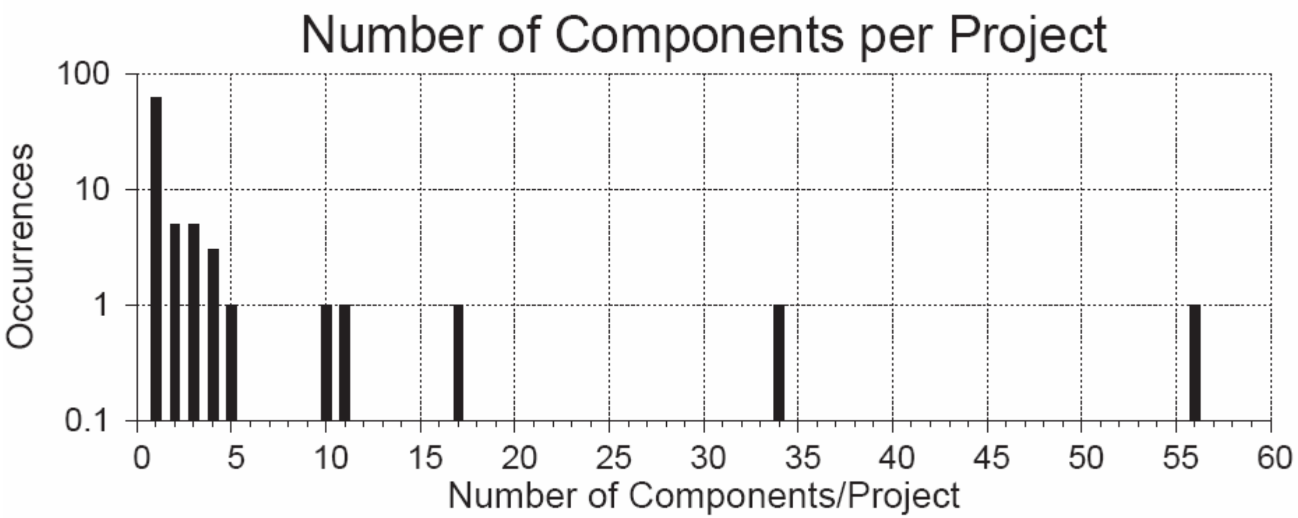
4.2. Build-and-Test Workloads: Workflow structure

- Reminder: **Run = workflow of Tasks**



1. Number of Tasks per Run
   - 39±61 Tasks / BUILD Run
   - 96±75 Tasks / TEST Run

4.2. Build-and-Test Workloads: Workflow structure



1. Number of Tasks per Run: 10-100 BUILD, 50-120 TEST
2. Number of Components per Project
   - 1 component most often
   - Few projects over 5 components

4.2. Build-and-Test Workloads: Workflow structure

Build-and-Test workloads are complex, and require a heterogeneous environment

1. Number of Tasks per Run: 10-100 BUILD, 50-120 TEST
2. Number of Components per Project: mostly 1
3. Number of Platforms per Component
   - 1-15 platforms most often
   - Few projects over 15 platforms

4.3. Build-and-Test Workloads: Observed failures

37% failures:
Build-and-Test environments critical for grid development!

- BUILD Runs fail more than TEST Runs (so is our software mature enough to create The Grid?)
- Take into account the failures due to the environment! (10-20% failures due to environment)

Outline

1. Introduction
2. The Build-and-Test problem
3. The NMI Build-and-Test Environment
4. Build-and-Test workloads
    1. Arrival patterns
    2. Workflow structure
    3. Observed failures
5. **Applications**
6. Conclusion

5. Applications
Test optimization

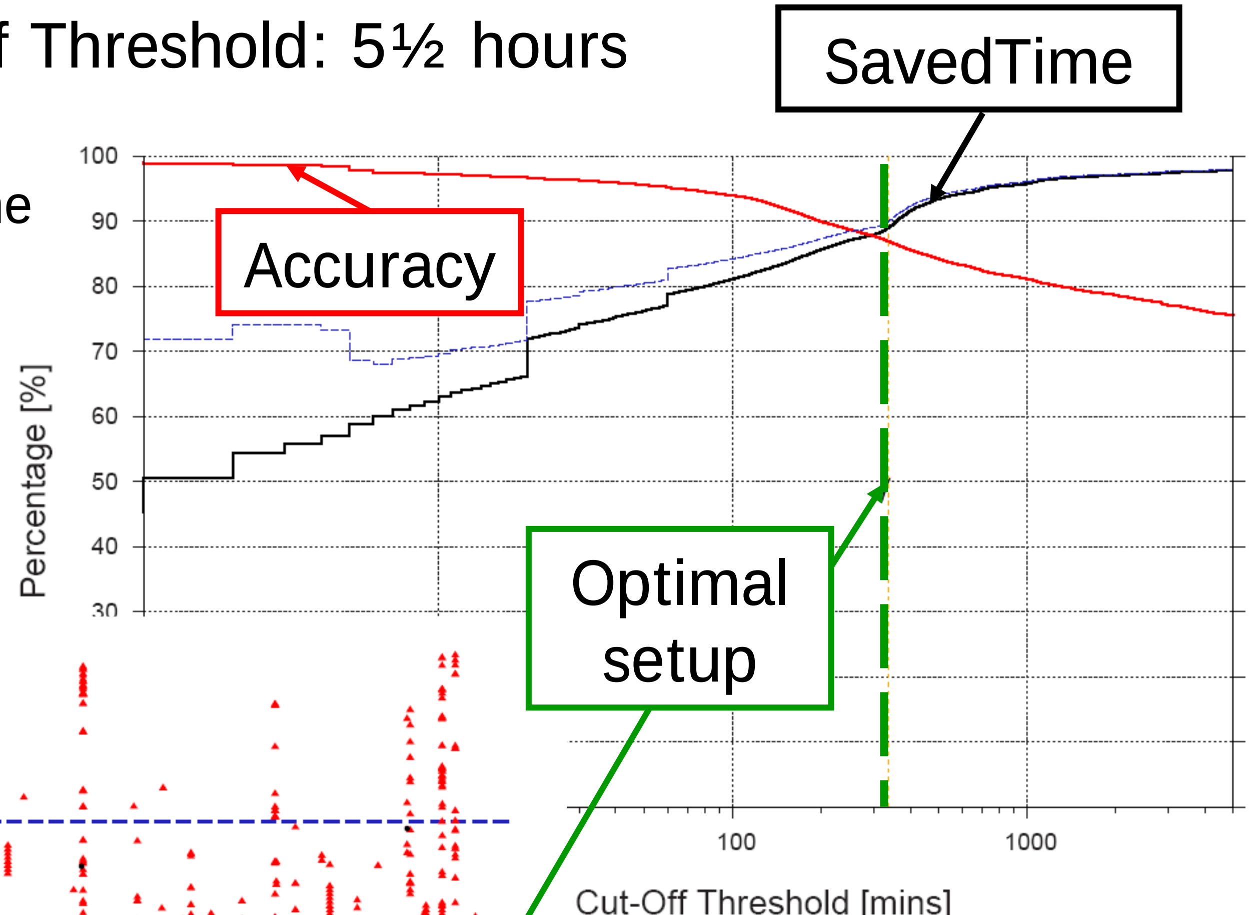
- Testing: trade-off between accuracy and runtime

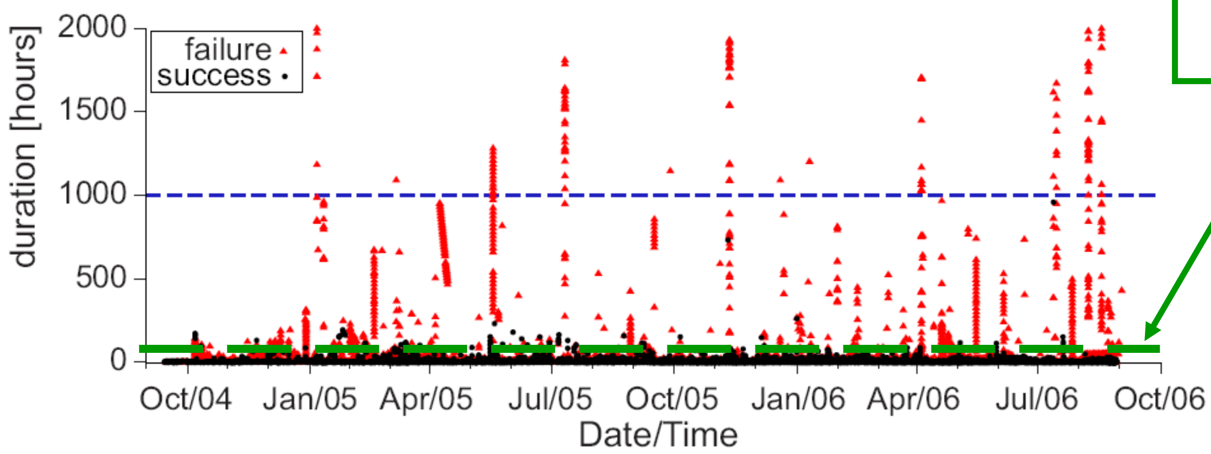
- **How long should we let the test jobs run?** Consider stopping jobs with runtime exceeding a cut-off threshold (administrator or automatic setup). Then, **What is the optimal cut-off threshold?**

- Metrics: accuracy (fraction of errors actually diagnosed), saved time (time saved because of cutting of jobs)

5. Applications
Test Optimization

- Optimal Cut-Off Threshold: 5½ hours
  - 95% accuracy
  - 93% saved time





Take home message

Create your B&T environment or perish!

- **Build-and-Test environments**
  - critical for grid software development
  - high load 16h/day, >1M tasks / year
  - build-and-test workflows, 20-150 tasks
  - 1 environment can serve 100s of projects

- **Applications**
  - Team / Project management [paper]

In today's grids, reliability is more important than performance!

- **Future research**
  - Test optimization
  - Test environment management / provisioning [paper]

Thank you! Questions? Remarks? Observations?

[**Dum05**] C. Dumitrescu, I. Raicu, and I. T. Foster. Experiences in running workloads over Grid3. In *GCC*, vol. 3795 of *LNCS*, pp. 274–286, 2005.

[**HLi07**] Hui Li, David Groep, Lex Wolters, Jeff Templon. Job Failure Analysis and Its Implications in a Large-scale Production Grid. In *e-Science*. IEEE CS, 2006.

[**Ios06**] A. Iosup and D. H. J. Epema. GrenchMark: A framework for analyzing, testing, and comparing grids. In *CCGRID*, pp. 313–320. IEEE CS, 2006.

[**Ios07**] A. Iosup, M. Jan, O. Sonmez, and D. H. J. Epema. On the Dynamic Resources Availability in Grids. INRIA RR-6172, 2007 (submitted).

[**Kha06**] O. Khalili, Jiahue He, et al. Measuring the performance and reliability of production computational grids. In *GRID*. IEEE CS, 2006.

[**Kon04**] D. Kondo, M. Taufer, C.L. Brooks III, H. Casanova, A. Chien, Characterizing and Evaluating Desktop Grids: An Empirical Study. In *IPDPS*, IEEE CS, 2004.

[**NMI06**] A. Pavlo, P. Couvares, R. Gietzel, A. Karp, I. D. Alderman, and M. Livny. The NMI build and test laboratory: Continuous integration framework for distributed computing software. In *USENIX LISA*, Dec 2006.

[**Tha02**] D. Thain, M. Livny, Error Scope on a Computational Grid: Theory and Practice. In *HPDC*, pp. 199--208, IEEE CS, 2002.