**Adrian Balint** Daniel Gall
Gregor Kapler Robert Retz

July 10, 2010

# Experiment Design and Administration for Computer Clusters for SAT-solvers (EDACC), system description

# Introduction

## SAT solvers

1. Have a wide application range
2. Are developed by a large community
3. Are easy to use
4. Are getting better, faster and more robust

# Introduction

## SAT solvers

1. Have a wide application range
2. Are developed by a large community
3. Are easy to use
4. Are getting better, faster and more robust

## The design of solvers

1. Starts with an idea, an intuition or a theoretical finding
2. Is followed by
   - An implementation phase
   - An intensive test-phase which can be very time-consuming
   - An analysis phase

# Testing of solvers - The main tasks

1. Choose a parameter setting for your solver
2. Choose the competing solvers (Which one is state of the art?)
3. Choose the instances to test on
4. Choose a (fast) computing system where to run the tests
5. Write scripts to manage the jobs for the tests
6. Collect results and analyse them $\rightarrow$ perhaps repeat point 1

# Testing of solvers

## The problems

1. Parameters: Many parameters $\rightarrow$ many tests
2. Competing solvers: Is the code/binary available?
3. Instances: Where to get them? Freely available?
4. Computing system: Multi CPU? Cluster? Grid?
5. Scripts: Optimal usage of resources?
6. Results: Manage, merge, analyse, choose representation.

# Testing of solvers

## The problems

1. Parameters: Many parameters $\rightarrow$ many tests
2. Competing solvers: Is the code/binary available?
3. Instances: Where to get them? Freely available?
4. Computing system: Multi CPU? Cluster? Grid?
5. Scripts: Optimal usage of resources?
6. Results: Manage, merge, analyse, choose representation.

## EDACC - The solution

1. Management-tool for solvers, instances, jobs and results (GUI, DB)
2. Design-tool for complex and large tests (experiments)
3. Analyse-tool for results supporting graphical representation

# EDACC - Overview

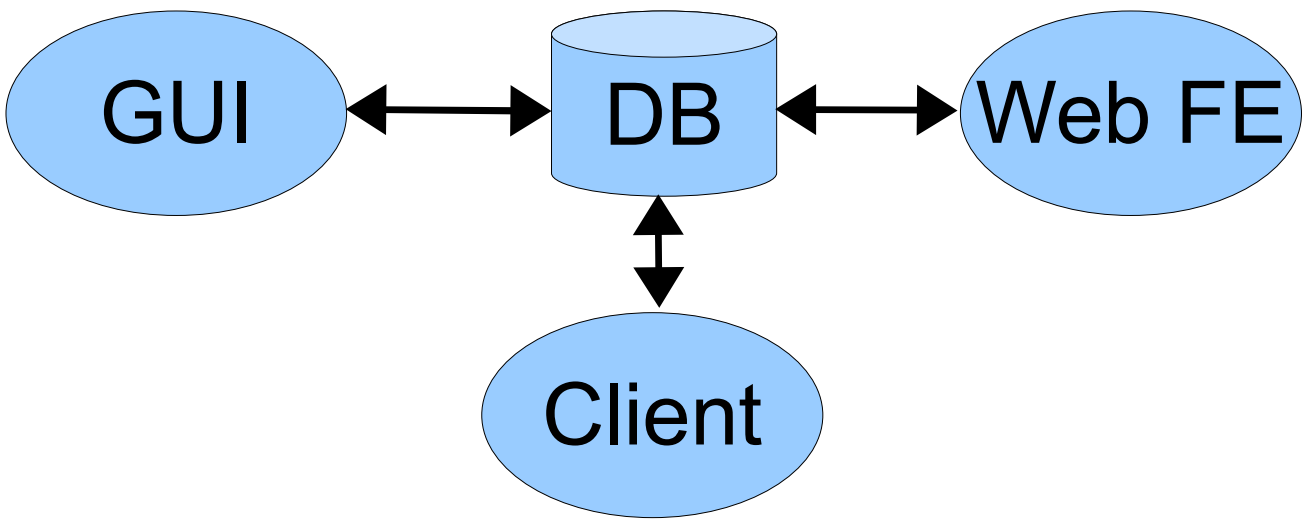
## Components of EDACC

1. Database
2. Graphical user interface
3. Client
4. Web front end

# EDACC - Overview

## Components of EDACC

1. Database
2. Graphical user interface
3. Client
4. Web front end




The interaction between the components of EDACC.

# EDACC - Database

## The information stored in the DB

- Solvers with their parameters
- Instances and instance classes (categorization of instances)
- Experiments = $\{\text{solvers}\} \times \{\text{parameters settings}\} \times \{\text{instances}\} \times \{\text{run-time information}\}$
- Information about computing systems (clusters, grid queues)

# EDACC - Database

## The information stored in the DB

- Solvers with their parameters
- Instances and instance classes (categorization of instances)
- Experiments = $\{\text{solvers}\} \times \{\text{parameters settings}\} \times \{\text{instances}\} \times \{\text{run-time information}\}$
- Information about computing systems (clusters, grid queues)

## Technical data

- Can be hosted on an arbitrary MySQL-server
- All files (code and binary of solvers, instances) are saved in the DB
  $\rightarrow$ avoids file-system inconsistencies
- All read/write operations to the DB are checked with MD5-sums (GUI, Client, WEB-FE)

# EDACC - GUI

## Manage DB Mode

Provides CRUD and export operations for solvers, parameters, instance classes, instances, computing system settings

# EDACC - GUI

## Manage DB Mode

Provides CRUD and export operations for solvers, parameters, instance classes, instances, computing system settings

## Experiment Mode

- Choose solvers to test and configure their parameters
- Choose instances to test on
- Configure the run-time properties and generate the jobs
- Real-time monitoring of the jobs
- Export or analysis of the results

# EDACC - GUI

## Manage DB Mode

Provides CRUD and export operations for solvers, parameters, instance classes, instances, computing system settings

## Experiment Mode

- Choose solvers to test and configure their parameters
- Choose instances to test on
- Configure the run-time properties and generate the jobs
- Real-time monitoring of the jobs
- Export or analysis of the results

## Technical data

- Java (JRE6)- independent of the operating system
- Graphical analysis with R

# EDACC - Client

- Loads information about an experiment such as solvers, parameters, instances and computing system
- Starts and manages multiple jobs on a computer (or node in a computer cluster)
- Monitors the jobs and writes the results back to the database
- Can be started on different computing systems (time comparison - only possible if the systems are homogeneous)
- Unfinished or crashed jobs can be recomputed by other clients - in some sense fail-proof with regards to the computing system

# EDACC - Client

- Loads information about an experiment such as solvers, parameters, instances and computing system
- Starts and manages multiple jobs on a computer (or node in a computer cluster)
- Monitors the jobs and writes the results back to the database
- Can be started on different computing systems (time comparison - only possible if the systems are homogeneous)
- Unfinished or crashed jobs can be recomputed by other clients - in some sense fail-proof with regards to the computing system

## Technical data

- Implemented in C
- Uses Linux built-in commands for monitoring

# EDACC - Web-front-end

## Features

- Real-time monitoring of jobs for running experiments
- Analysis of results for finished experiments
- Export possibilities for solvers and instances
- Supports a sort of SAT Competition modus

# EDACC - Web-front-end

## Features

- Real-time monitoring of jobs for running experiments
- Analysis of results for finished experiments
- Export possibilities for solvers and instances
- Supports a sort of SAT Competition modus

## Technical data

- Python application using cross-platform modules $\rightarrow$ runs on Linux, Windows, Mac OS, ...
- Can be deployed on any web server implementing the WSGI interface, e.g. Apache (mod_wsgi), lighttpd (FastCGI), MS IIS, ...
- Connects to an arbitrary MySQL server hosting EDACC databases
- Uses an existing R interpreter to render graphs

# Related-work

## SAT Competition System

- The functionalities of the DB, GUI and WEB-FE are partially or better supported by the SAT Competition system by O. Rousell , D. Le Berre and L. Simon
- Disadvantage - the system is not freely available

# Related-work

## SAT Competition System

- The functionalities of the DB, GUI and WEB-FE are partially or better supported by the SAT Competition system by O. Rousell , D. Le Berre and L. Simon
- Disadvantage - the system is not freely available

## Condor - High Throughput Computing (HTC)

- Can start and manage jobs spread over different computing systems
- Disadvantage - the user has to specify scripts for each job - not necessary in EDACC
- Disadvantage - installation necessary

# Outlook

- Integration of the runsolver-program (SAT-Competition) in the client
- Extended integrated graphical analysis-tools
- Integration of automated parameter tuning
- Plug in for Condor - generate Condor jobs from EDACC-jobs
- No limitation to SAT: arbitrary programs with arbitrary inputs

# EDACC - Further details

- License: open-source : MIT License
- Can be downloaded from

  `http://sourceforge.net/projects/edacc/`
- Under full development
- Further features are planned and assigned

# Demo

- Comparing two solvers TNM and gNovelty+2
- Small set of instances from the SAT Competition 2009
- Computing system : BW-Grid Ulm (2$\times$ Quad-core CPUs per node)