Supporting Anti-cancer Drug Discovery by a Private SZTAKI Desktop Grid System

József Kovács
smith@sztaki.hu

ISGC 2009, Taipei, Taiwan, 23/04/2009
SZTAKI Desktop Grid: BOINC project

Number of users: 22947  Number of hosts: 31004
Active hosts in last 48 hours: 1479
Estimated performance of last 48 hours: 755.531 GFlop/s  Peak performance: 1.5 TFlop/s
Workunits processed in last 48 hours: N/A

Join SZTAKI Desktop Grid
Rules and policies [read this first]
Getting started
More information on desktopgrid.hu
Wikipedia article about SZTAKI Desktop Grid
Frequently Asked Questions (FAQ)
Create account
Applications
Returning participants
Your account - view stats, modify preferences

Application current
SZTAKI Desktop Grid
number systems
Description of the a
User of the day

Choose a project
To choose a project, click its name or type its URL below.
World Community Grid
Itelligence
Grid Computing
SAGE
Project URL:

http://szdg.lpds.sztaki.hu/szdg
SZTAKI Desktop Grid is a collection of various developments towards Desktop Grid direction, based on BOINC:

- Debian package of the BOINC server
- application programming interface: DC-API
- integration with various backends: e.g.: Condor
- supporting various application types on the client side: e.g.: Java, MPI
- aggregating the power of different BOINC projects: hierarchically connected DGs
- ease the application porting under BOINC client: genwrapper
- improving security: introducing certificates in BOINC, sandboxing under BOINC client
- WU creation/forwarding jobs to boinc: 3GBridge on BOINC server

Most of them can be downloaded from www.desktopgrid.hu, others are under development, ⇒ desktopgrid@lpds.sztaki.hu
The CancerGrid project

• EU Framework Program 6
• Title: Grid Aided Computer System For Rapid Anti-Cancer Drug Design
• Project period
• Goals:
  – Developing focused libraries with a high content of anti-cancer leads, building models for predicting various molecule properties
  – Developing a computer system based on grid technology, which helps to accelerate and automate the in silico design of libraries for drug discovery processes
The CancerGrid consortium

- TargetEx, Inc., Hungary (coordinator) [chemistry]
- AMRI Hungary, Inc., Hungary [chemistry]
- Inte:Ligand, Austria [chemoinformatics]
- Tallinn University of Technology, Estonia [chemoinformatics]
- University of Helsinki, Finland [biotechnology]
- GKI Economic Research, Hungary [economics]
- SZTAKI, Hungary [computer science]
- University of Jerusalem, Israel [chemoinformatics]
- DAC, Italy [biotechnology]
- University of Bari, Italy [chemistry]
- University of Pompeu Fabra, Spain [chemoinformatics]
Wide variety of apps in a workflow

Applications: cmol3d, mopac, mdc, fmt, fma, etc.

- Fortran, C, C++
- Processing/Memory requirements
- Multi-binary applications, Libraries
- For some apps source is not available
- Config file preparation before execution
- Pure logging/debugging information
Basic workflows developed in CancerGrid

Molecule Descriptor Calculation I.

Property Prediction

Model building

Virtual Screening
The CancerGrid portal (gUSE & SZTAKI DG)

Integrated components of CancerGrid portal:
- Structure viewer
- Molecule database browser
- Algorithms configuration
- Workflow development & configuration
- Workflow execution

Workflow name: CG-wf2-DB
Note: 2008-7-9
Workflow Graph: CG-wf2-DB-graph
Workflow Template: Optional selection of a new Template: CG
The CancerGrid architecture

DG Server Components

BOINC Server

Job Database

(Job Database (Description of Jobs: Apps, Args, I/O files))

Queue Manager

Scheduling policy

Batch creation

DC-API master

BOINC Task DB

BOINC client

DC-API cli

GenWrapper for batch execution

Legacy Application

WS-PGRADE (User IF)

(gUSE Storage)

(WF representation)

gUSE Local Submitter

(gUSE WS Submitter)

(gUSE)

(gUSE-DG integration)

DesktopGrid

BOINC client

DC-API cli

GenWrapper for batch execution

Legacy Application

The CancerGrid architecture
Generic Wrapper (GenWrapper)

- **Why did we develop?**
  - The features of **BOINC wrapper is not enough** (e.g. patching config files on client machines, generating extra messages, independent jobs in a WU, etc.)
  - Wanted to **be prepared for unknown requirements** might be raised by future Cancergrid applications
  - We did not want to extend the BOINC wrapper to make it an XML-based programming language, we choose to **BOINCify an existing language** -> Bourne shell

- **How does it work?**
  - A shell interpreter (gitbox - a variant of busybox) is started instead of the real application
  - It executes an application script, that
    - realizes boincification through script commands
    - may run legacy applications in any way (e.g.: multiple input process)
    - may perform any preparation on input-, output files, environment, etc.
    - may do **whatever you can do by a script**
1. unzips %BASENAME%.zip to slot dir
2. executes `dc_init()` or `boinc_init()` [ and `dc_finish()` or `boinc_finish()` at the end ]
3. creates a script that:
   - sources the Profile Script
   - starts the Application Script
4. starts GenWrapper
5. executes the generated script
   - compound BOINC application
   - handles architecture/ platform dependent pre-run tasks for the application
6. exit status becomes the exit status of the WU
Sample GenWrapper Script

1. IN=`boinc resolve_filename in`  
2. OUT=`boinc resolve_filename out`  
3. NUM=`cat ${IN}`  
4. PERCENT_PER_ITER=$((100000 / NUM))  
5. for i in `seq $NUM`; do  
6.   PERCENT_COMPLETE=$((PERCENT_PER_ITER * i / 1000))  
7.   boinc fraction_done_percent ${PERCENT_COMPLETE}  
8.   echo -e "I am ${PERCENT_COMPLETE}% complete." >> ${OUT}  
9.   sleep 1;  
10. done

- shell script contains the BOINC commands  
- every filename needs to be resolved  
- status: on-going development, still missing some features (signal handling, background process, etc.)
Integration of the web-portal to DG by the 3G Bridge

**Source Plug-in**
- WS-PGRADE/gUSE web based portal plugin

**Destination Plug-in**
- BOINC plug-in
  - Batch creation
  - DC-API master

**Job Database**
(Description of Jobs: Apps, Args, I/O files)

**Queue Manager**
- Scheduling policy
Batching in 3GBridge

- Substrings like "{%<word>}" are substituted with the appropriate value. Unknown substitutions are left alone and copied as-is.
- 3 template scripts must be prepared for every app.
- **Head template:** extracts %{inputs}
- **Per-job templates:**
  - All input files are under %{input_dir} (relative to the directory where the script is started)
    - Moves the input files to appropriate location if necessary
  - Calls "application %{args}"
  - Moves all output files to %{output_dir} (relative to the directory where the script is started)
- **Tail template:** packs the directory %{output_dir} as %{outputs}
Example templates

• Example head template:
  
  set +e
tar xzf %{inputs}
BASEDIR=`pwd`

• Example per-job template:
  
  cd $BASEDIR/%{input_dir}
$BASEDIR/app %{args} >stdout 2>stderr
mv out_file stdout stderr $BASEDIR/%{output_dir}
  
  cd –

• Example tail template:
  
  cd $BASEDIR/outputs
tar czf $BASEDIR/%{outputs} *
Ongoing work – Virtualization of resources

- VM Base Image
- VM Instance Image
- VM Manager
  - Start
  - Create
  - Disk I/O
- VM Instance
  - suspend
  - resume
  - start
  - stop
  - checkpoint
  - continue
- BOINC core client
- VM API
- VM Base Image
- VM Instance Image
- VM Manager
- Start
- Create
- Disk I/O
- VM Instance
- Communication Daemon (HTTP)
- Message Handler
- Data Handler
- Execution Environment
  - Application
  - Input Files
  - Output Files
  - stdin/ stdout/ stderr/ argv
  - ...

We run Linux in VM instances, components are implemented in C/C++

- We use QEMU
  - fulfills every requirement
  - opens source
  - runs as single a single process
  - requires KQEMU for its full speed (optional kernel module or driver)
Conclusion

• Any community that has a class of workflow type applications requiring bag of task type of components can easily use a BOINC system:
  – the community can create its own institutional BOINC project
  – can easily map the bag of task components into BOINC applications
  – can easily combine these components into more complex workflow applications

• Such a system
  – has been prototyped for the Cancer Research community within the CancerGrid projects
  – can have its Portal specialized for any community requirements

• We are ready to support other communities with this technology
If you need more detailed (technical) information, email to desktopgrid@lpds.sztaki.hu or visit www.desktopgrid.hu

Thank you for your attention!

Questions?

Acknowledgement:
CancerGrid EU FP6 project (FP6-2005-LIFESCTHTALTH-7)
http://www.cancergrid.eu