University
of Ljubljana

Faculty
of civil and geodetic engineering

Computer Mediated Communication (CE6014)

HTC - Condor Overview

October 2011

Matevž Dolenc
University of Ljubljana

Credits

- This presentation was adopted from the presentation “Using Condor: An Introduction”, presented at Condor Week 2005

http://www.cs.wisc.edu/condor/CondorWeek2005/presentations/user_tutorial.ppt

Available Computer Resources

- UNIX® servers are actually "serving" something less than 10% of the time.
- Most PCs do nothing for 95% of a typical day.
- Imagine ...
  - an airline with 90% of its fleet on the ground
  - an auto maker with 40% of its assembly plants idle
  - a hotel chain with 95% of its rooms unoccupied

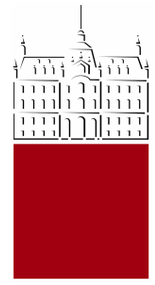
Available Computer Resources

- Almost every organisation is sitting atop enormous unused computing capacity.

HPC vs. HTC

	High-performance Computing HPC	High-throughput Computing HTC
FLOPS	very important, single computing unit	not that important, many computing units
environment	static	dynamic
problem	single large problem	parallel independent jobs
protocols / solutions	MPI, PVM	Condor, Torque, SETI@Home

HTC DRM Systems (a selection)

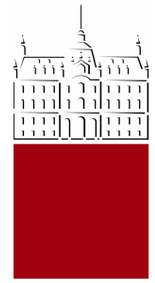


- Condor
  - http://www.cs.wisc.edu/condor
  - OS: Windows, Linux, Mac, ...
  - great support for dynamic environments, source available
- Sun Grid Engine
  - http://gridengine.sunsource.net
  - OS: Solaris, Linux, ...
  - open source

HTC DRM Systems (a selection)

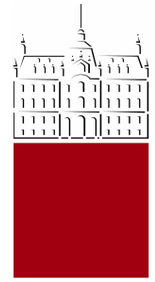
- Torque
  - http://www.clusterresources.com/products/torque-resource-manager.php
  - OS: UNIX, Linux
  - scalability, extends Open PBS, source available
- Xgrid
  - http://www.apple.com/acg/xgrid.html
  - OS: MacOS X
  - graphical user interface, easy configuration

Frida the Scientist



- Frieda has to perform a parametric study.
- The problem:
  - Run a Parameter Sweep of F(x,y,z) for 20 values of x, 10 values of y and 3 values of z (20*10*3 = 600 combinations)
  - F takes on the average 6 hours to compute on
  - a “typical” workstation (total = 3600 hours)
  - F requires a “moderate” (128MB) amount of memory
  - F performs “moderate” I/O - (x,y,z) is 5 MB and F(x,y,z) is 50 MB

Personal Condor



- Where to get Condor?
  - http://www.cs.wisc.edu/condor
- Download Condor for your operating system
  - Available for most UNIX (including Linux) platforms
  - Also for Windows NT / XP, MacOS X, ...
- What do we mean by a "Personal" Condor?
  - Condor on your own workstation, no root access required, no system administrator intervention needed
  - After installation, Frieda submits her jobs to her Personal Condor…

- Where is the benefit?
- Your Personal Condor will:
  - keep an eye on your jobs and will keep you posted on their progress
  - implement your policy on the execution order of the jobs
  - keep a log of your job activities
  - add fault tolerance to your jobs
  - implement your policy on when the jobs can run on your workstation

Submitting Jobs to Condor

- Make your job "batch-ready"
  - Must be able to run in the background: no interactive input, windows, etc.
  - Can still use STDIN, STDOUT, and STDERR, but files are used for these instead of the actual devices
  - Organise data files
- Creating a submit description file
  - A plain ASCII text file
  - Tells Condor about your job
  - Can describe many jobs at once (a "cluster"), each with

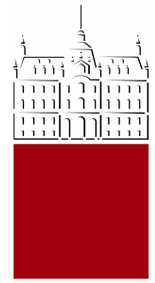
Submitting Jobs to Condor

```
# Simple condor_submit input file
# (Lines beginning with # are comments)
# NOTE: the words on the left side are not
#       case sensitive, but filenames are!

Universe   = vanilla
Executable = my_job

Queue
```

Submitting Jobs to Condor

- Run condor_submit on your submit description file
  - You give condor_submit the name of the submit file you have created: condor_submit my_job.submit
  - condor_submit parses the submit file, checks for it errors, and creates a "ClassAd" that describes your job(s)
  - ClassAds: Condor's internal data representation
    - Similar to classified ads (as the name implies)
    - Represent an object & it's attributes
    - Can also describe what an object matches with

Submitting Jobs to Condor

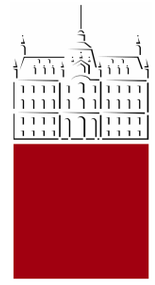


- The job queue
  - condor_submit sends your job's ClassAd(s) to the schedd
    - Manages the local job queue
    - Stores the job in the job queue
    - View the queue with condor_q

Accessing Data in Condor

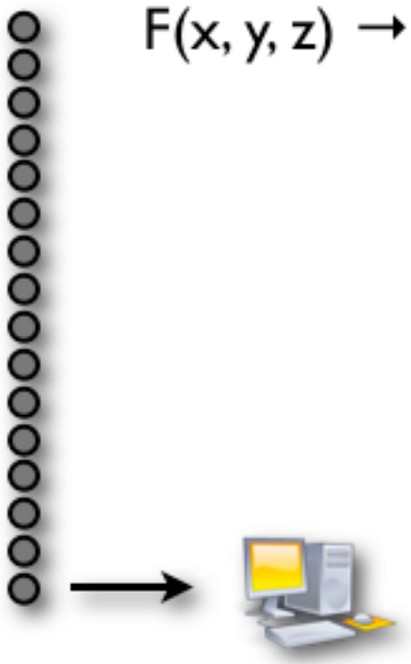
- Use Shared Filesystem if available
- No shared filesystem?
  - Condor can transfer files
    - Can automatically send back changed files
    - Atomic transfer of multiple files
    - Can be encrypted over the wire
  - Remote I/O Socket
  - Standard Universe can use remote system calls

Accessing Data in Condor



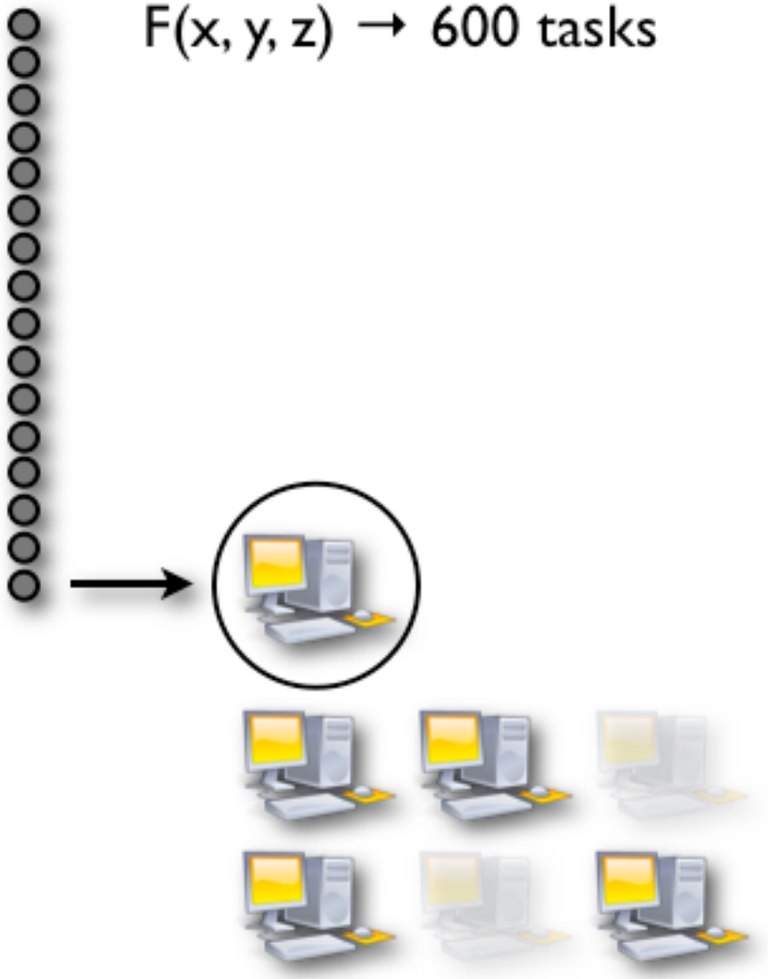
- Condor file transfer
  - ShouldTransferFiles = YES
  - ShouldTransferFiles = NO
  - ShouldTransferFiles = IF_NEEDED (Will automatically transfer the files if the submit and execute machine are not in the same FileSystemDomain)

Frida's Condor Pool



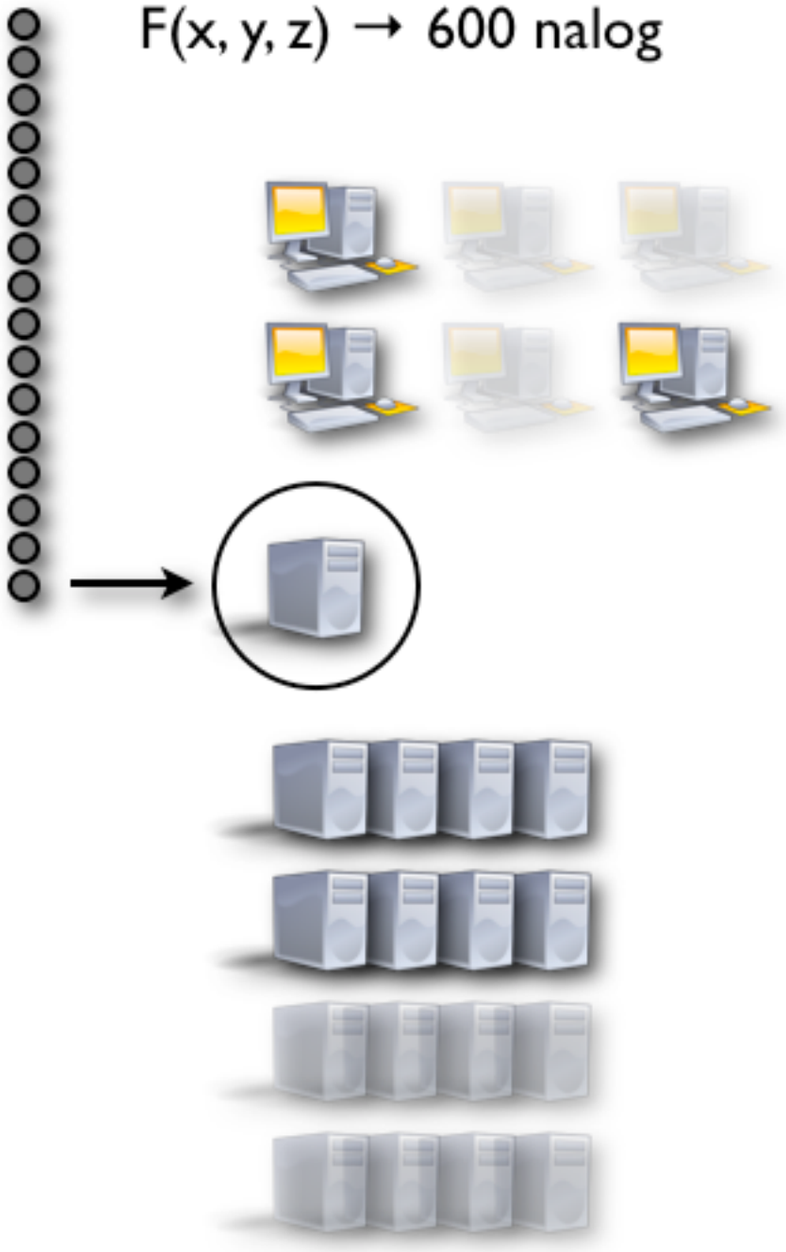
- Frieda can still only run one job at a time, however.
- There are good news.
- The Boss says Frieda can add her co-workers' desktop machines into her Condor pool as well… but only if they can also submit jobs.

Frida's Condor Pool



- Frieda installs Condor on the desktop machines, and configures them with her machine as the central manager
- These are "non-dedicated" nodes, meaning that they can't always run Condor jobs
- Now, Frieda and her co-workers can run multiple jobs at a time so their work completes sooner.

Frida's Condor Pool



- They need more!
- The organisation buys a dedicated cluster.
  - Frieda Installs Condor on all the dedicated Cluster nodes and adds a dedicated central manager
  - She configures her entire pool with this new host as the central manager…
  - With the additional resources, Frieda and her co-workers can get their jobs completed even faster.

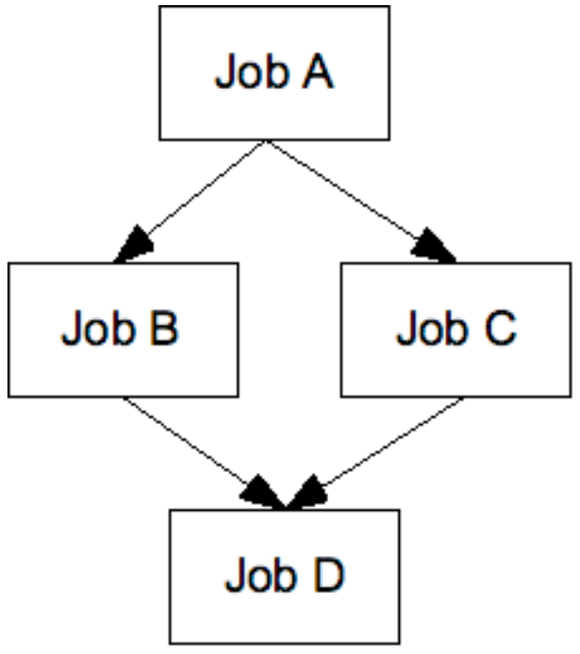
Job Dependencies

- Directed Acyclic Graph Manager (DAGMan)
  - DAGMan allows you to specify the dependencies between your Condor jobs, so it can manage them automatically for you. (e.g., "Don't run job "B" until job "A" has completed successfully.")
- What is DAG
  - A DAG is the data structure used by DAGMan to represent these dependencies.
  - Each job is a "node" in the DAG.
  - Each node can have any number of "parent" or "children" nodes – as long as there are no loops!

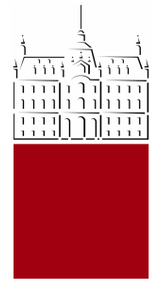
Job Dependencies

- Defining a DAG
  - A DAG is defined by a .dag file, listing each of its nodes and their dependencies
  - each node will run the Condor job specified by its accompanying Condor submit file

```
# diamond.dag
Job A a.sub
Job B b.sub
Job C c.sub
Job D d.sub
Parent A Child B C
Parent B C Child D
```



Job Dependencies



- Submitting a DAG

```
$ condor_submit_dag diamond.dag
```

Job Dependencies

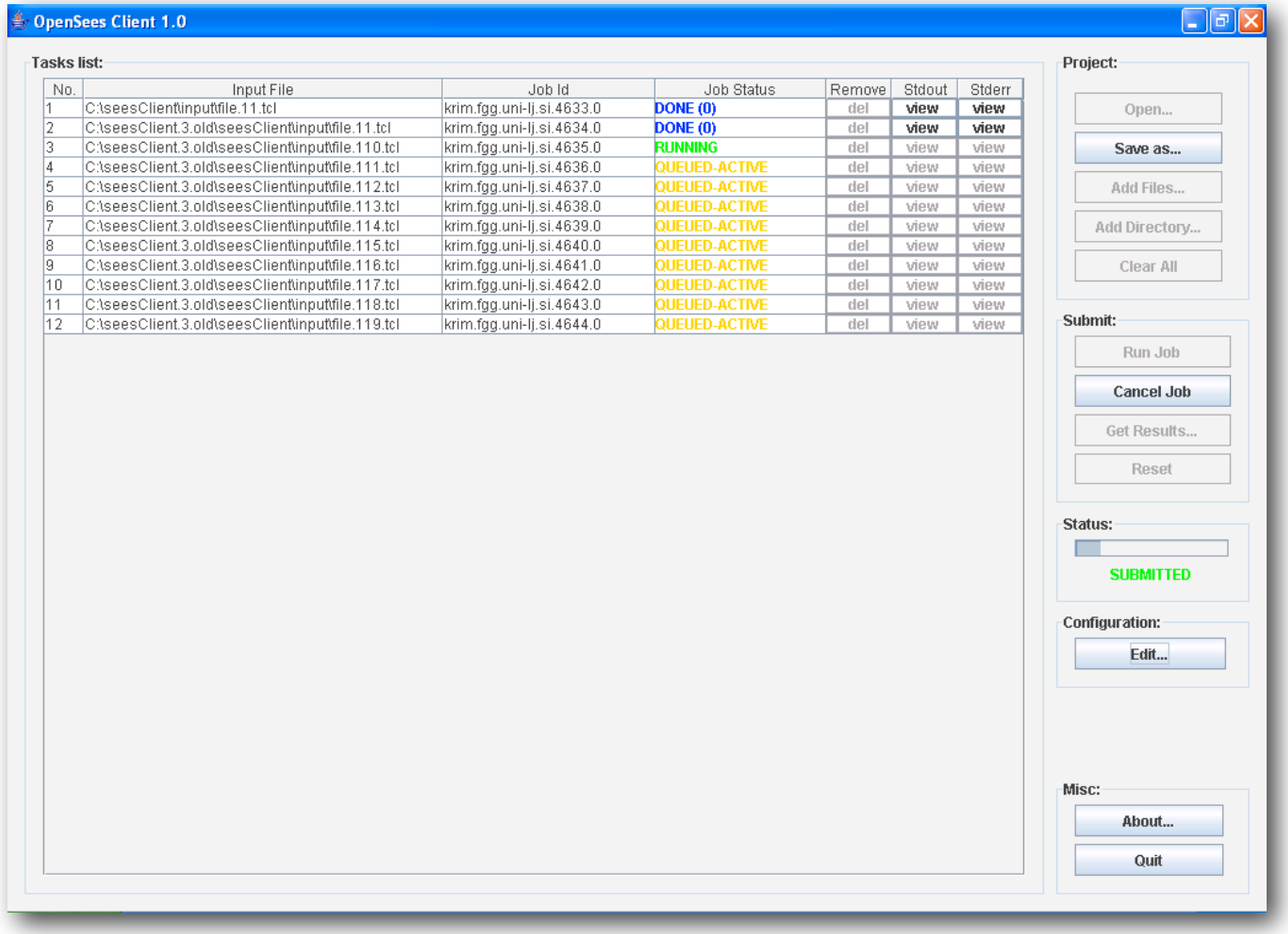
- Running a DAG
  - DAGMan acts as a “meta-scheduler”, managing the submission of your jobs to Condor based on the DAG dependencies.
  - DAGMan holds & submits jobs to the Condor queue at the appropriate times.
  - In case of a job failure, DAGMan continues until it can no longer make progress, and then creates a “rescue” file with the current state of the DAG.

Job Dependencies

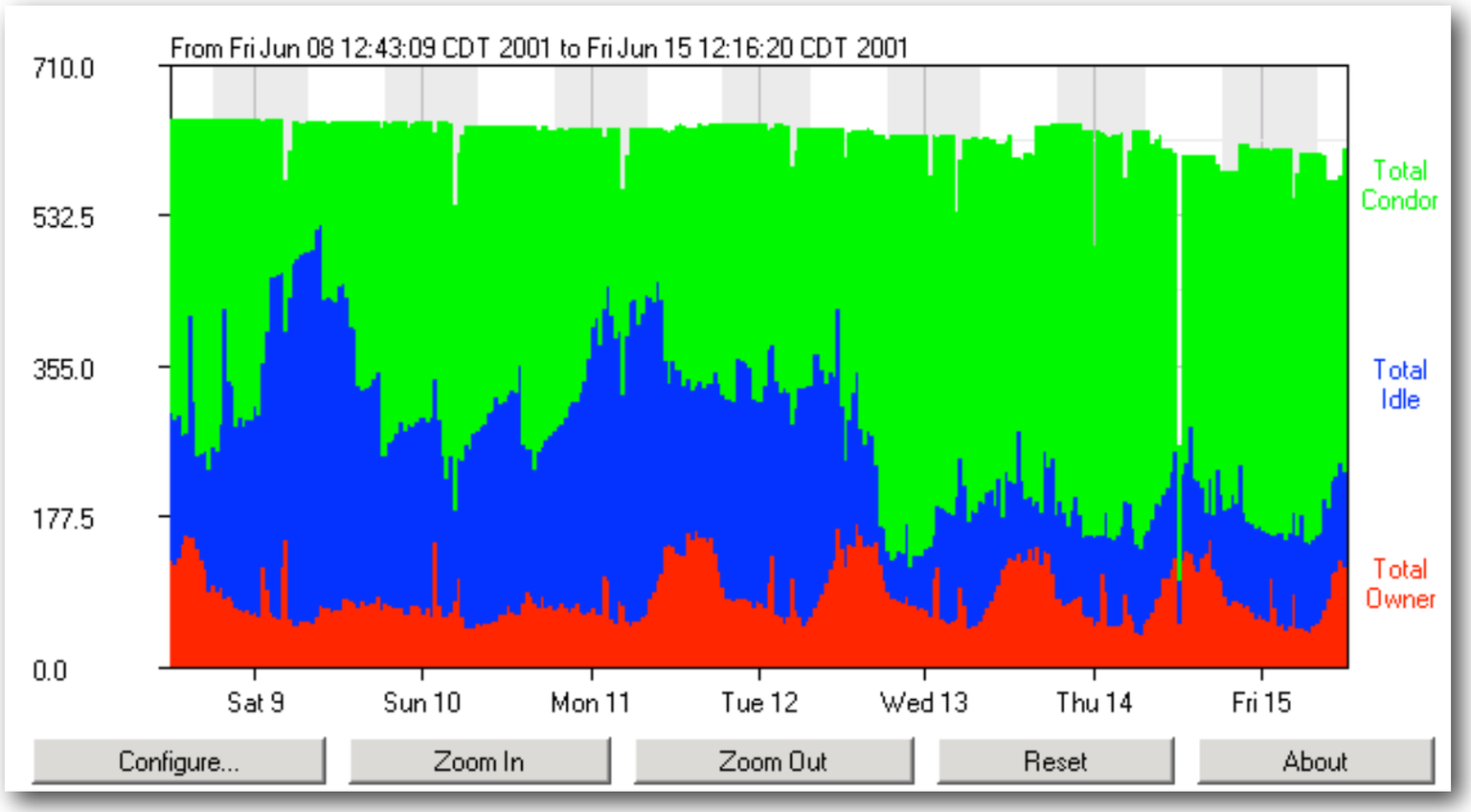
- Additional DAG features
  - Provides other handy features for job management…
  - nodes can have PRE & POST scripts
  - failed nodes can be automatically re-tried a configurable number of times
  - job submission can be "throttled"

DRMAA

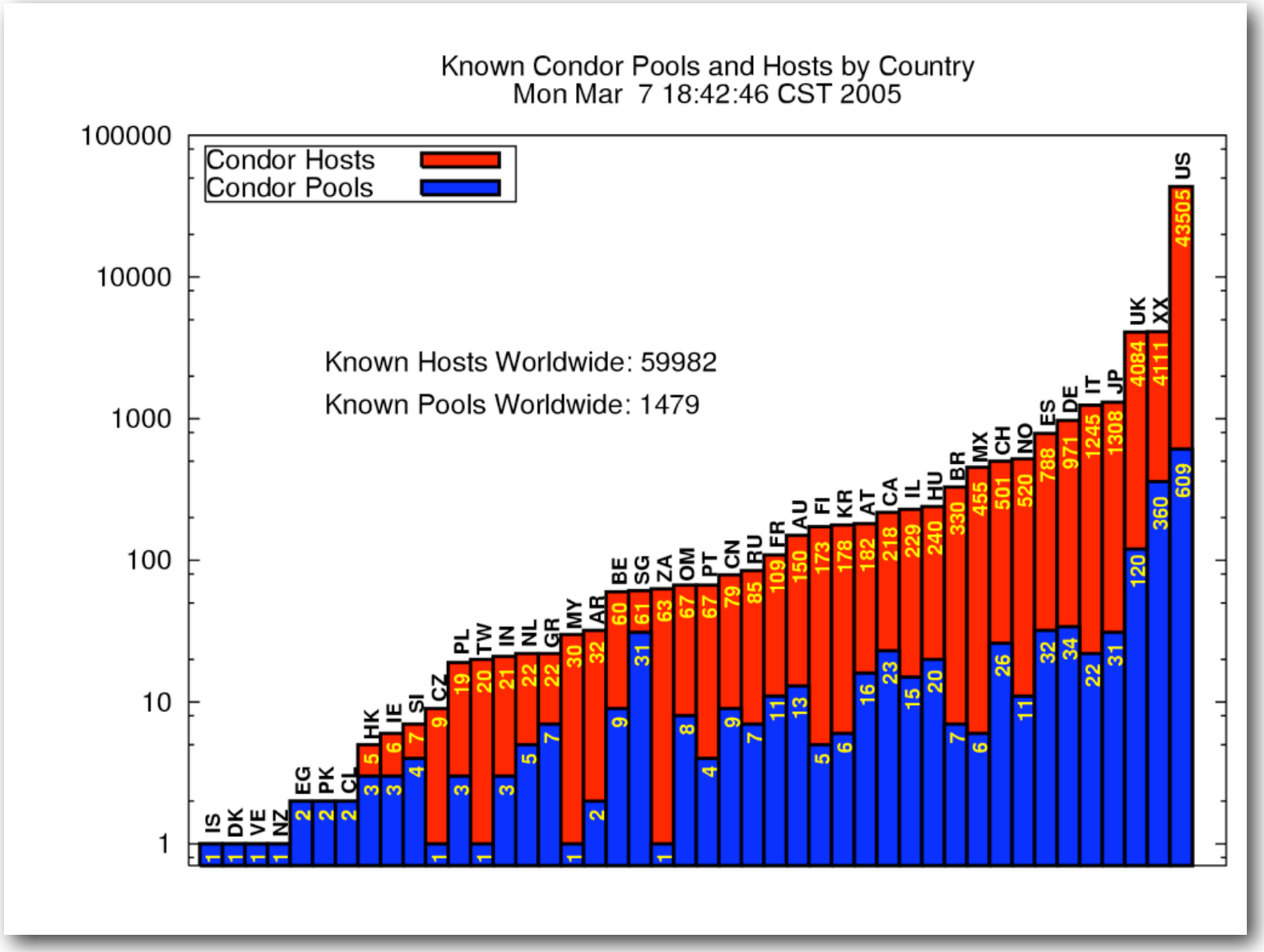
- Programming API (C/C++, Java, Python, Ruby, ...)
- OpenDSP: WS impl. of the DRMAA API



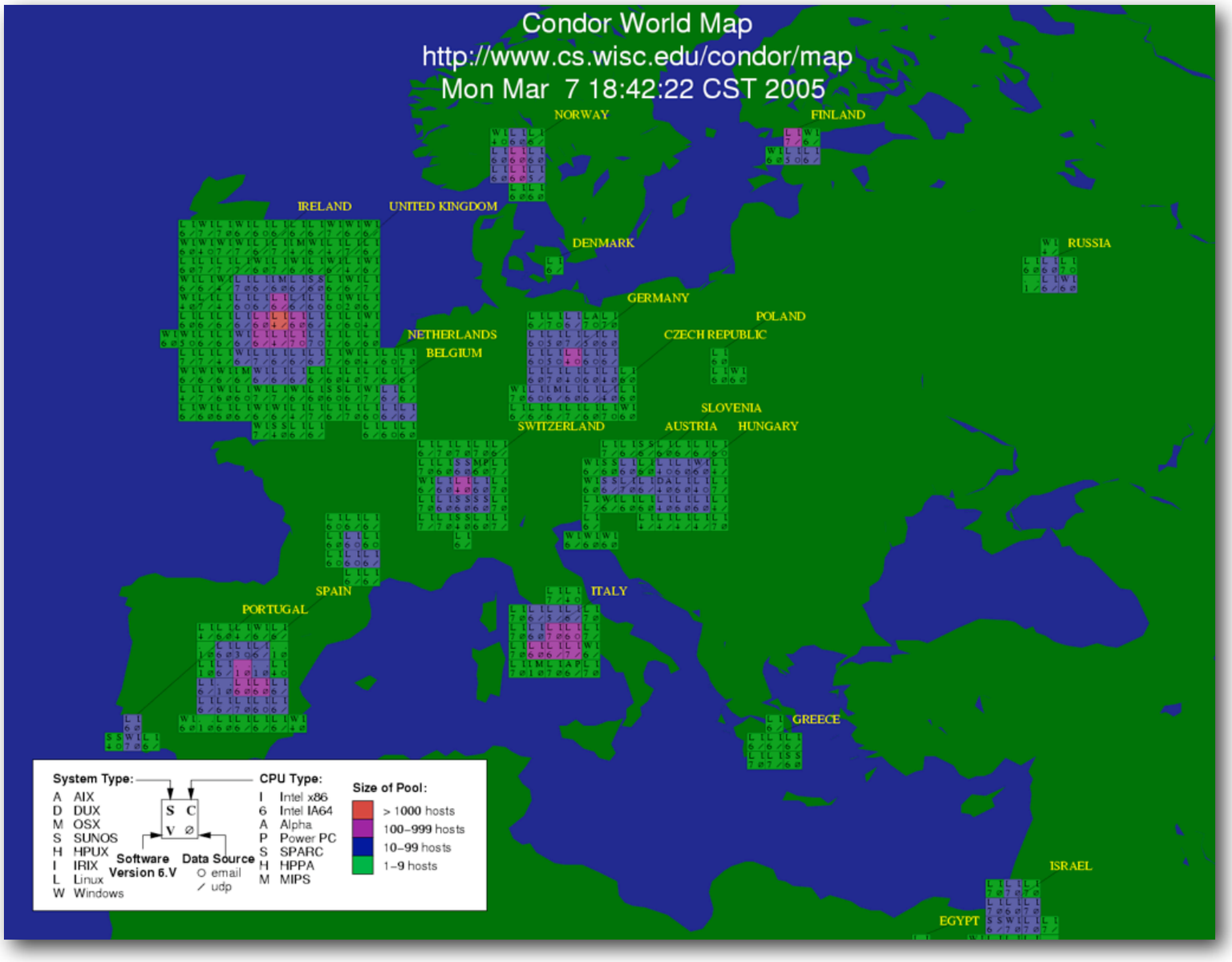
CondorView


From Fri Jun 08 12:43:09 CDT 2001 to Fri Jun 15 12:16:20 CDT 2001
710.0
532.5
355.0
177.5
0.0
Total Condor
Total Idle
Total Owner
Sat 9
Sun 10
Mon 11
Tue 12
Wed 13
Thu 14
Fri 15
Configure...
Zoom In
Zoom Out
Reset
About

Condor Pools

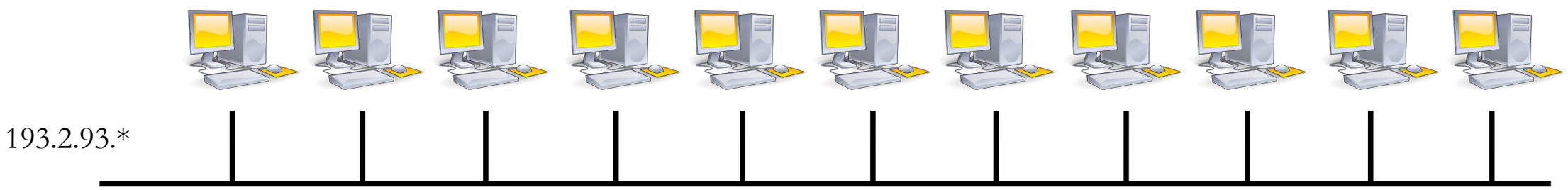


Condor Europe Map



Condor pool at UL-FGG

Computer labs:
75+ computers, Windows XP
can only execute Condor jobs



193.2.91.*
193.2.92.*

kgi-cl.fgg.uni-lj.si
193.2.91.28

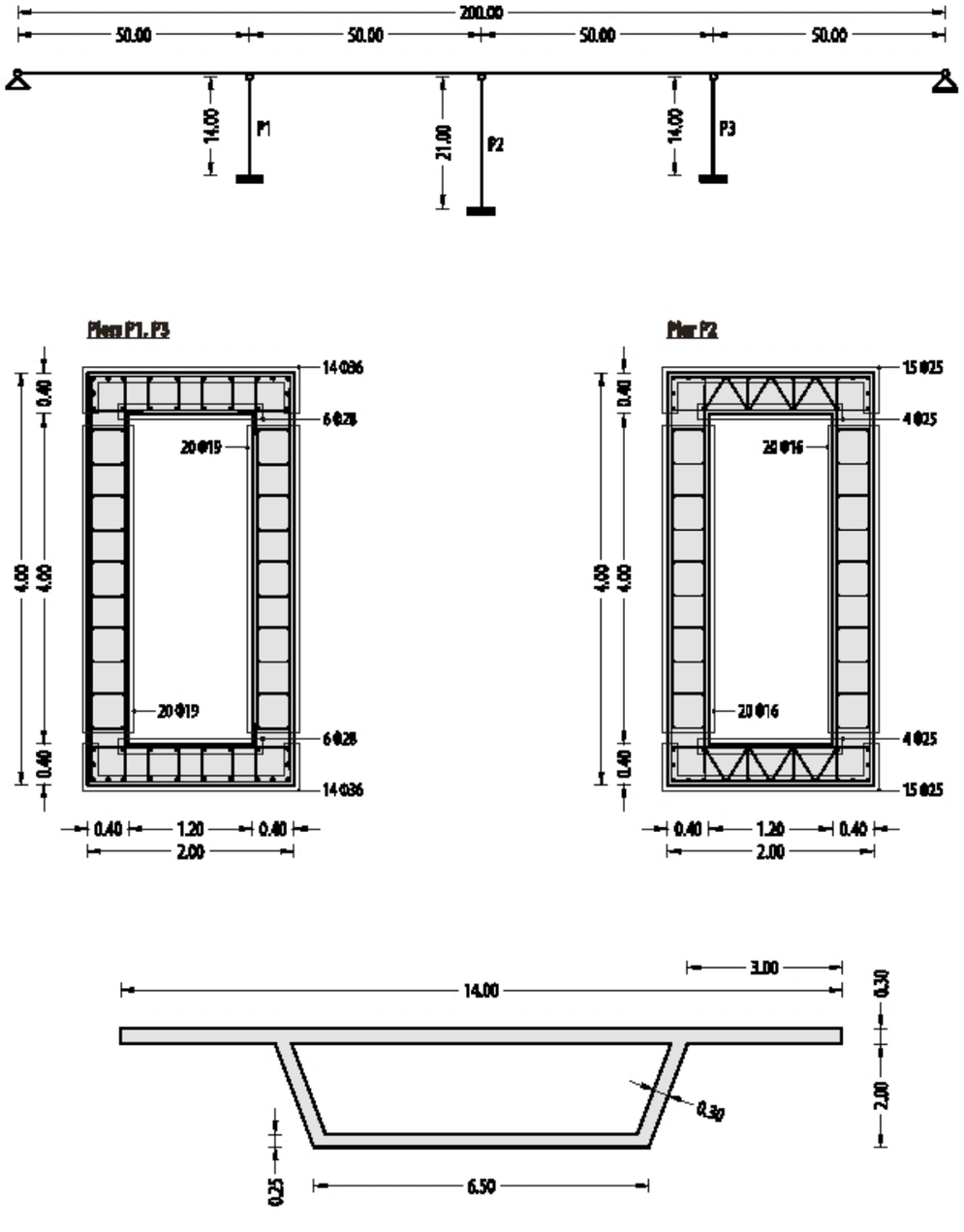
UL-FGG research computors:
can submit and execute (usually disabled) Condor jobs

UL-FGG static cluster

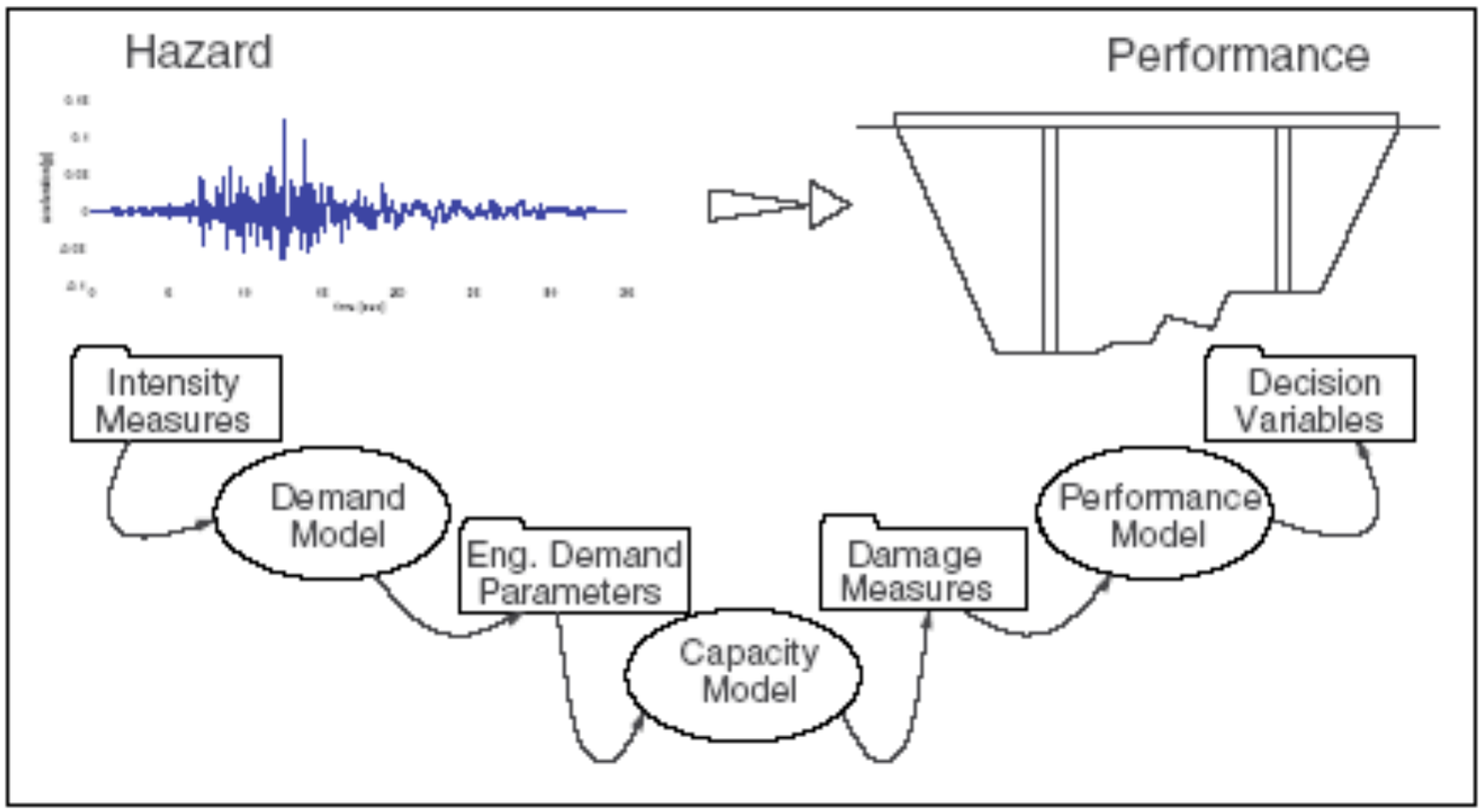
Case study: Parametric studies

- Past/present
  - single limit state deterministic techniques
- Future
  - multiple performance objectives
  - probabilities of different socio-economic decision variables
  - decision variables relate performance of a structure to decision making (involve: cost of repair, indirect cost of down time, loss of lives, ...)

Case Study: IDA Analysis

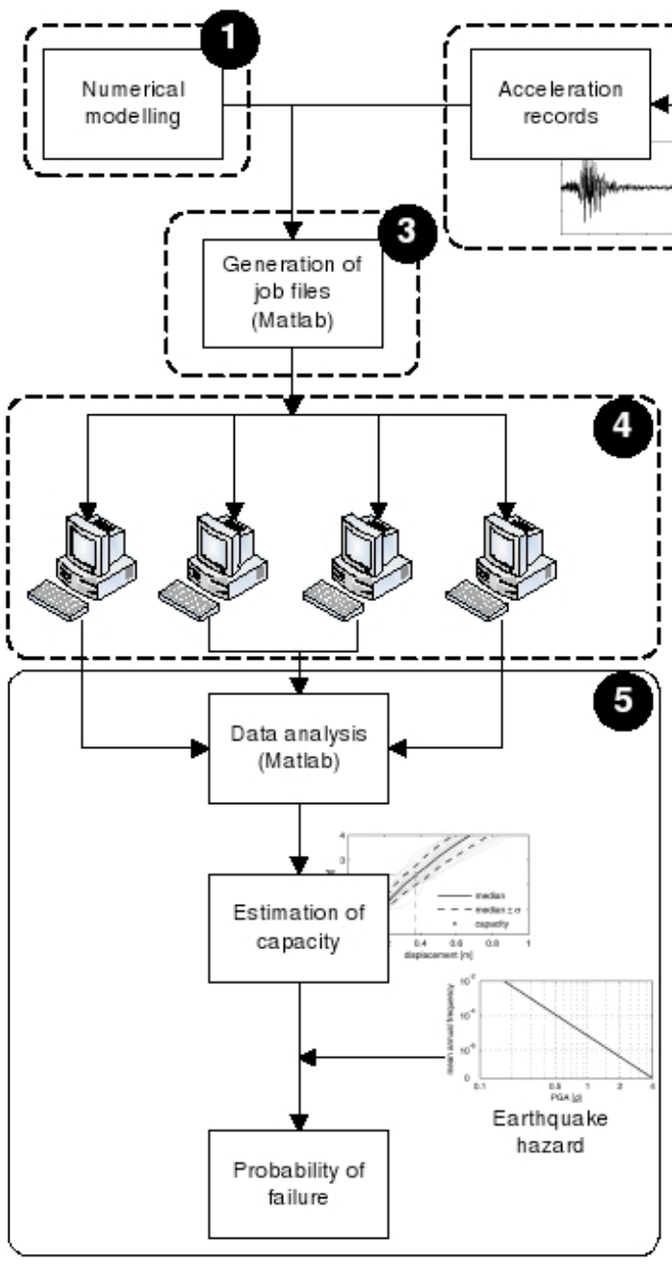


- IDA – Incremental Dynamic Analysis
- ... determining the relation between the engineering demand parameter and seismic intensity measure.



D. Vamavatsikos and C.A. Cornell, Incremental Dynamic Analysis, Earthquake Engineering and Structural Dynamics, 31(3), 491-514, 2002.

Case Study: IDA Analysis



- Number of analyses
  - different structural models (loss, capacity, demand, seismic hazard)
  - several ground motion records
  - multiple levels of intensity
- IDA workflow
  - numerical modelling
  - selection of earthquakes
  - job descriptions

Case Study: IDA Analysis

- Number of analyses: 280
- Average analysis time: ~13min

Number of computers	Analysis time [hours]	Speed-up factor
1	61.3	1
5	14.7	4.17
10	7.1	8.63
25	2.5	24.52