

BIM A+

European Master in
Building Information Modelling

HTCondor computing environment

Topic 2: Fundamentals of programming

BIM A+3: Parametric Modelling in BIM

Matevž Dolenc

Univerza v Ljubljani



Universidade do Minho



POLITECNICO
MILANO 1863



© 2019 by authors

This work is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](#).

	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	HTCondor, Torque, SETI@Home

A problem to solve

- 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 ($20 \times 10 \times 3 = 600$ 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

HTCondor to the rescue

- Where to get HTCondor
 - <http://research.cs.wisc.edu/htcondor/>
- Download HTCondor for your OS
 - HTCondor runs on all modern operating systems: Windows, Mac, Linux
- Install HTCondor
 - You can start by installing “Personal HTCondor”

The screenshot shows the official HTCondor website at research.cs.wisc.edu. The page features a logo with a penguin and the text "HTCondor High Throughput Computing". It includes a navigation bar with links to Home, News, Download, Publications, and Contact Us. A search bar is also present. The main content area has sections for "Computing with HTCondor™", "Latest News" (RSS feed), and three columns for Software, Community, and Research and Development.

Computing with HTCondor™

Our goal is to develop, implement, deploy, and evaluate mechanisms and policies that support [High Throughput Computing \(HTC\)](#) on large collections of distributively owned computing resources. Guided by both the technological and sociological challenges of such a computing environment, the [Center for High Throughput Computing](#) at UW-Madison has been building the open source [HTCondor distributed computing software](#) (pronounced "itch-tee-condor") and related technologies to enable scientists and engineers to increase their computing throughput.

Note: The HTCondor software was known as 'Condor' from 1988 until its name changed in 2012. If you are looking for Phoenix Software International's software development and library management system for z/VSE or z/OS, click [here](#).

Latest News [RSS](#)

- HTCondor 8.9.3 released! [September 17, 2019](#)
- HTCondor 8.8.5 released! [September 4, 2019](#)
- HTCondor 8.8.4 released! [July 9, 2019](#)
- HTCondor 8.9.2 released! [June 4, 2019](#)
- HTCondor 8.8.3 released! [May 28, 2019](#)
- HTCondor 8.9.1 released! [April 17, 2019](#)

[More News >](#)

Software	Community	Research and Development
<ul style="list-style-type: none"> ▪ What is the HTCondor software? ▪ Downloads including HTCondor source and binaries for Linux, Windows, and Mac. ▪ Documentation includes the Quick Start Guide and the HTCondor Manual. Also see MONITOR 	<ul style="list-style-type: none"> ▪ Email Lists - ask questions and share experiences with users worldwide. ▪ HTCondor Week is our annual community meeting in Madison, and we often have an annual meeting in Europe as well. Materials from past meetings include talks from 	<ul style="list-style-type: none"> ▪ Research Publications ▪ HTCondor Technologies including the widely used ClassAd Library ▪ Information for Developers

Personal HTCondor



$F(x, y, z) \rightarrow 600$ tasks



- 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 HTCondor

- 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 different input, arguments, output, etc.

```
# 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

Frida's HTCondor pool

$F(x, y, z) \rightarrow 600$ tasks

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



HTCondor to the rescue

$F(x, y, z) \rightarrow 600$ tasks

- Frieda installs HTCondor 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 HTCondor jobs
- Now, Frieda and her co-workers can run multiple jobs at a time so their work completes faster.



HTCondor to the rescue

$F(x, y, z) \rightarrow 600$ nalog

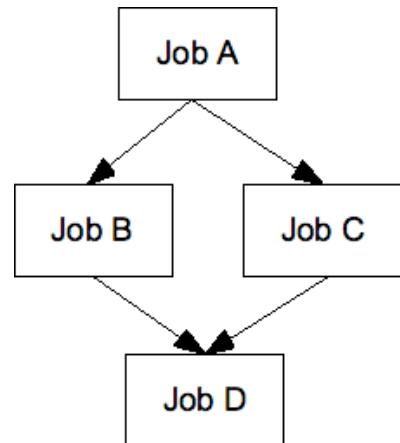


- They need more!
- The organisation buys a dedicated cluster.
 - Frieda Installs HTCondor on all the dedicated Cluster nodes and adds a dedicated central manager
 - She configures her entire pool with this new host as the central manager...
 - With the additional resources, Frieda and her co-workers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

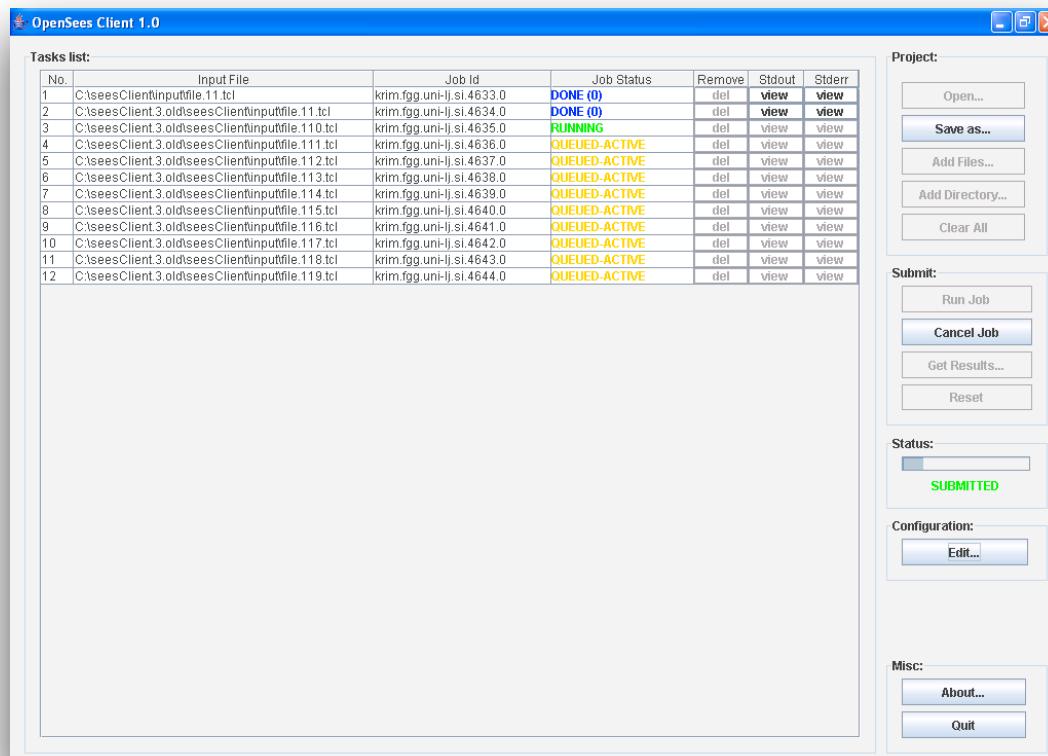
Job dependencies

- Directed Acyclic Graph Manager (DAGMan)
 - DAGMan allows you to specify the dependencies between your HTCondor 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!
- 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
```



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



History of HTCondor @ UL FGG

BIMA+

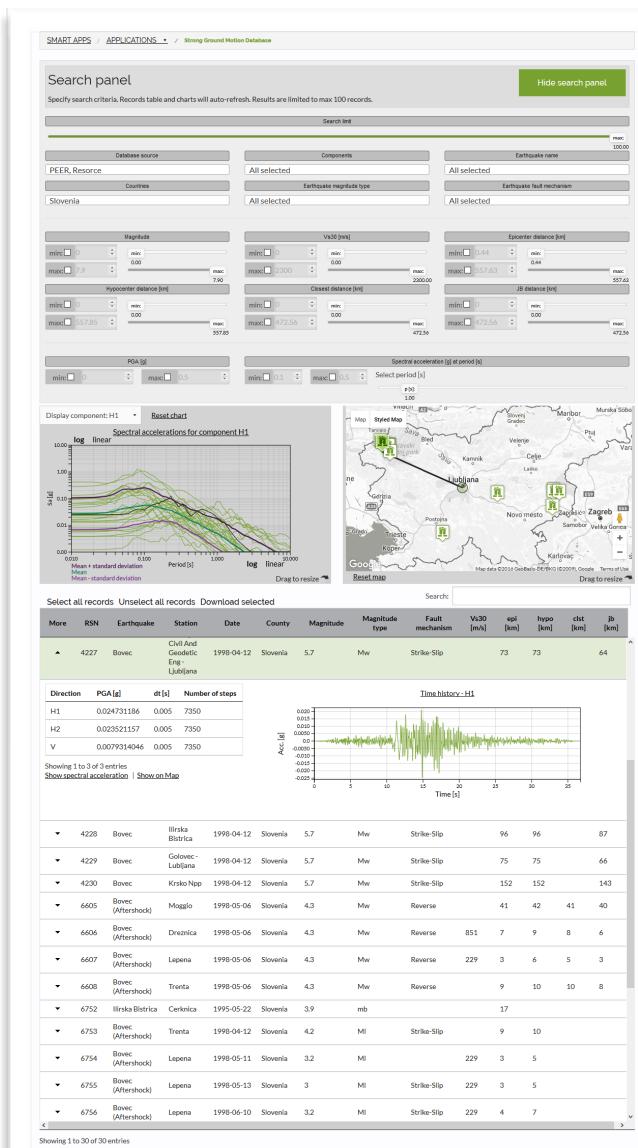




- Hardware
 - Intel Xeon CPUs
 - 8/16/32 GB RAM (total 316 GB)
 - 316 CPU
- Software
 - Ubuntu Server 18.04 LTS
 - HTCondor, MPI enabled
 - General applications: MATLAB, BLAS, LINPACK, ...
 - Specific applications: OpenSees, Abaqus, ...

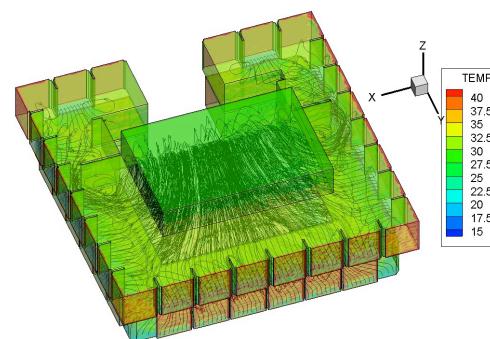
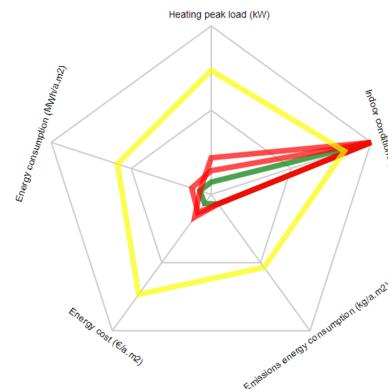
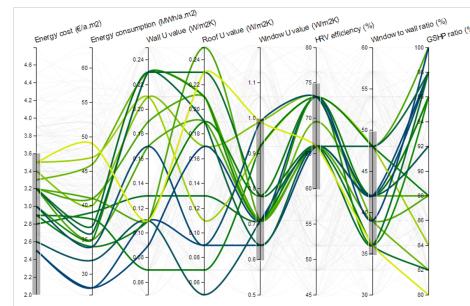
Example: IDA analysis (OpenSees)

- Earthquake analyses (IDA, 3R, Qfactor)
- Parametric studies - OpenSees, MATLAB
- Energy efficient buildings
- Parametric studies - Nandrad (TUD-IBK), Therakles (TUD-IBK), Riuska (Gradlund)
- CFD analyses
- Parallel programs (MPI) - Sofistik CFD, 96 CPU
- 3D numerical analyses
- Parallel programs (MPI) - Abaqus, 16 CPUs

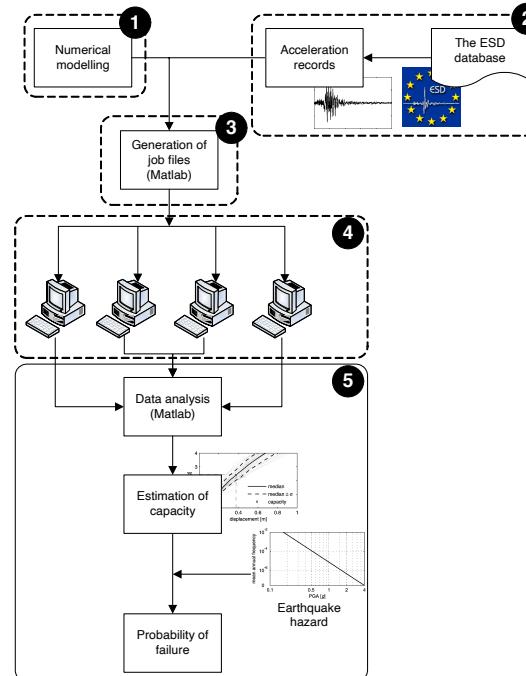
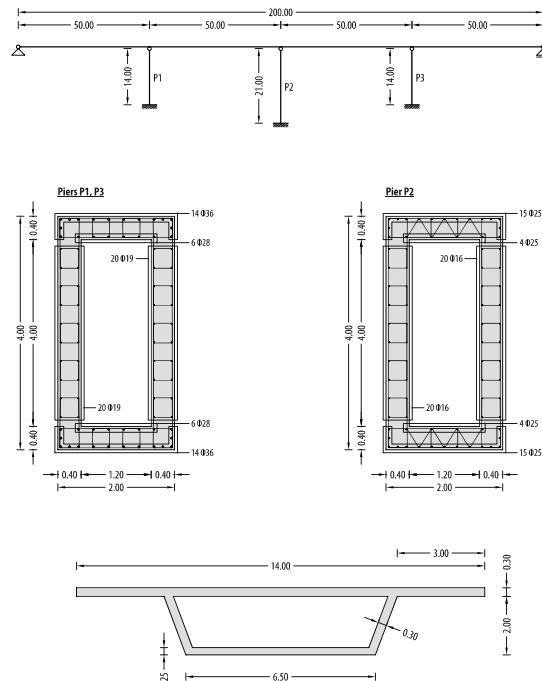


Examples

- Earthquake analyses (IDA, 3R, Qfactor)
 - Parametric studies - OpenSees, MATLAB
- Energy efficient buildings
 - Parametric studies - Nandrad (TUD-IBK), Therakles (TUD-IBK), Riuska (Gradlund)
- CFD analyses
 - Parallel programs (MPI) - Sofistik CFD, 96 CPU
- 3D numerical analyses
 - Parallel programs (MPI) - Abaqus, 16 CPUs



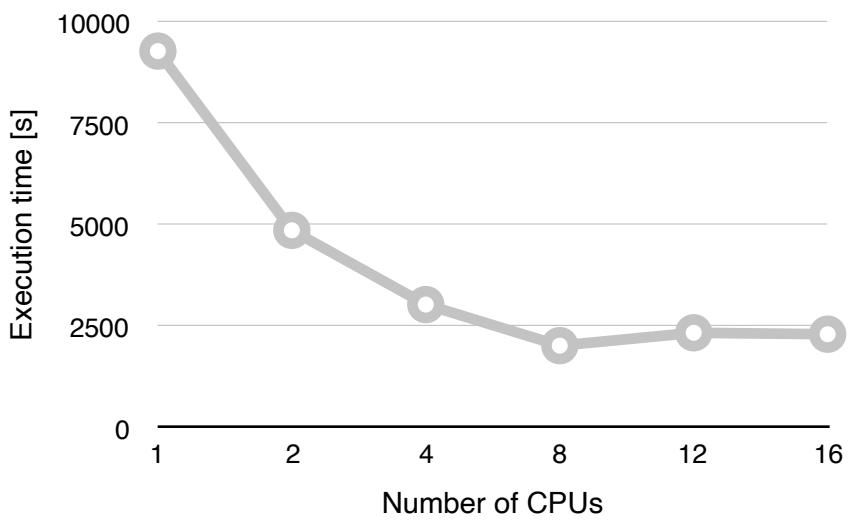
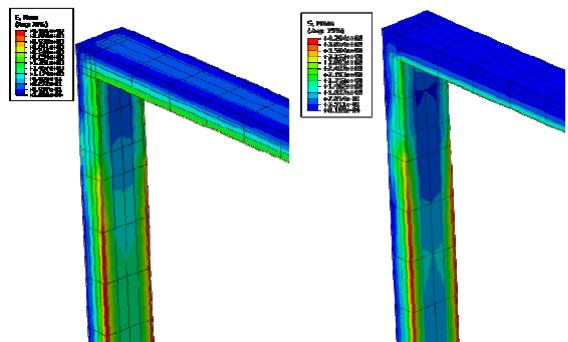
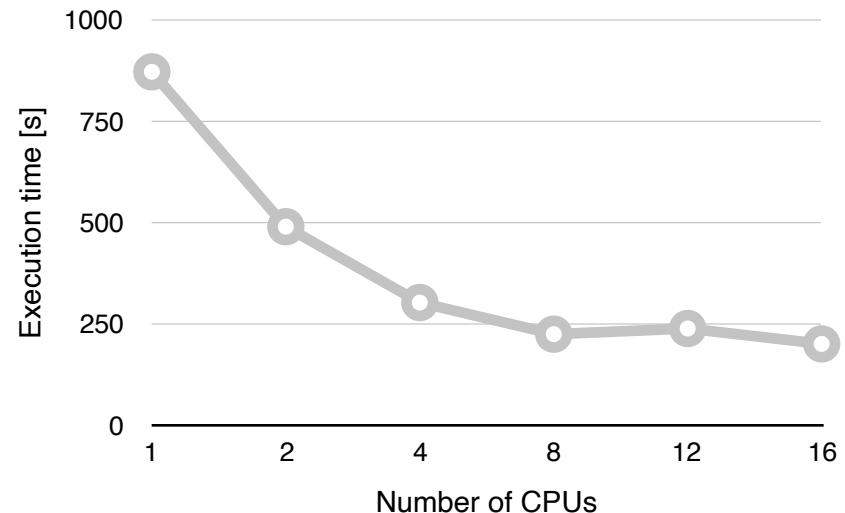
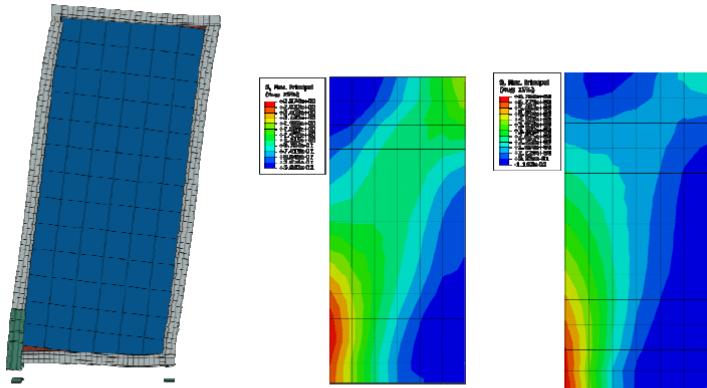
Examples: IDA analysis (OpenSees)



Number of CPUs	Execution time [h]	Speedup factor
1	61.3	1
5	14.7	4.17
10	7.1	8.63
25	2.5	24.52

Examples: IDA analysis (OpenSees)

BIM A+



Demo