Condor: Managing Computationally Intensive Jobs from a Computer Science Perspective

733 M-AM

Andrew S Fox[1,2,4], Matthew Farrellee[3], Alain Roy[3], Richard J Davidson[1,2,4], Terrence R Oakes[1,2]

Waisman Laboratory for Brain Imaging and Behavior[1], Departments of Psychiatry[2], Computer Science[3] and Psychology[4], at the Universtiy of Wisconsin-Madison

Introduction: What is Condor?

Condor is a distributed job scheduler. Each day a productive neuroimaging analyst wishes to perform tremendous amounts of computation with as little user intervention as possible. Many neuroimaging analysis protocols require completing identical procedures on data from each subject collected. These procedures can be visualized as a series of modules (or nodes), each of which represents a single procedure with inputs and outputs. The inputs and outputs of different modules can be linked together to form a pipeline (or directed graph), which represents the procedures that each subject's data will undergo and the dependency relationships between the different procedures. Some analysis pipelines can take hours or days to execute, and increasingly require minimal user intervention. Programmers of individual neuroimaging software packages typically include a mechanism for executing a series of non-interactive jobs in a given sequence, which is called batch processing. However, it is frequently desirable to utilize multiple software packages in an optimal analysis protocol (Oakes et al., In Press; Rex et al., 2003). The need to maximize computational efficiency and manage computational processes is at the heart of the field of computer science, which has developed a variety of approaches with these aims. Here we introduce Condor, a program originally developed by Computer Scientists at the University of Wisconsin-Madison in 1988, as a free and well-established distributed job scheduler, which is currently implemented on over a thousand computer pools across six continents. Condor will manage distributed processing across multiple computers, ensure appropriate execution order, and is well suited to the neuroimaging community.

What can Condor do? (http://www.cs.wisc.edu/condor)

The primary role of Condor is to provide a framework for controlling complex processes on one or more computers. Condor provides a system for creating a pipeline and can distribute jobs in parallel across multiple computers or computer systems. Condor also gives neuroimaging labs the ability to harness personal computers that are temporarily idle. Condor supports all major operating systems, and has been effectively implemented on high-speed distributed computing clusters. The Condor software suite allows users to create modules that can be linked together to manage data analysis. Condor also allows for failure isolation and provides a framework for error checking. Expert Condor applications can further extend its powers to distribute single processes across multiple computers, using both basic checkpointing and grid computing.

What Condor can offer basic users:

- Stable and dependable process management, including dependency management.
- Parallel execution of non-dependent processes.
- The ability to harness the processing power of unused workstations.
- Failure isolation within processing pipelines, as well as Condor computer pools.
- Effective matching of process requirements and machine capabilities.
- Free download, and multi-platform execution.

What Condor can offer advanced users:

- Checkpointing - By recompiling existing programs using Condor libraries, procedures can begin on one computer and finish on another.
- Parallelization of individual programs - Condor provides explicit support for the MPI, PVM, and Master-Worker programming environments.
- Support for grid computing (Condor-G).
- Combining independent Condor pools (Flocking).

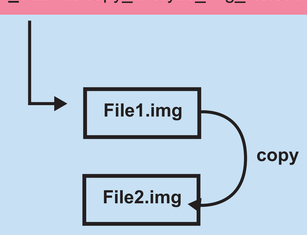
Practical Condor:

- Condor is a set of commandline programs.
- Condor jobs are added to the process queue using condor using **condor_submit**.
- Submiting a job to condor requires a **Submit File** as shown below.
- Condor jobs can be linked toghether using a Directed Acyclic Graph (DAG) which is equivalent to a pipeline.
- Submiting a DAG requires a **DAG File**, and corresponding **Submit Files** as shown below.

Example Condor Module (Submit File):

```
####################################
# SUBMIT FILE: copy_analyze file.submit
#    Copy file1.img to file2.img
####################################
 universe = vanilla
 log = copy_analyze_img_file.log
 error = copy_analyze_img_file.error
 output = copy_analyze_img_file.output

 Executable = /bin/cp
 Arguments = file1.img file2.img
 queue
```

condor_submit copy_analyze_img_file.submit



Example Condor Pipeline (Directed Acyclic Graph or DAG):

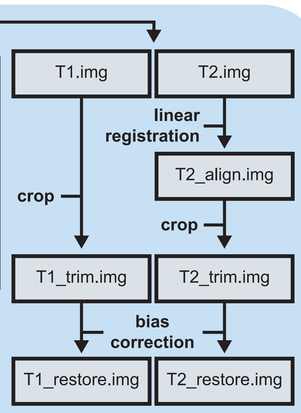
```
########################################
# DAG FILE: align_t2_to_t1.dag
#    Align T2.img to T1.img
#    then Crop and Bias Correct both Images.
########################################
JOB linear_registration linear_registration_-_fsl_simple.sumbit
JOB crop_T1 crop_16si_avwroi_-_fsl.sumbit
JOB crop_T2 crop_16si_avwroi_-_fsl.sumbit
JOB multispectral_bias_correction multispectral_fast_-_fsl_simple.sumbit

PARENT linear_registration CHILD crop_T1 cropT2
PARENT crop_T1 crop_T2 CHILD multispectral_bias_correction

VARS linear_registration dof="-dof 6" cost="-cost mutualinfo" input="-in /study/MRI/T2.hdr"
   ref="-ref /study/MRI/T1.hdr" omat="-omat /study/MRI/T2_to_T1.mat" out="-out
   /study/MRI/aligned_to_T1/T2_aligned.img"
VARS crop_T1 input=" /study/MRI/T1.img" output=" /study/MRI/T1_trim.img" xmin=" 156"
   xsize=" 216" ymin=" 194" ysize=" 202" zmin=" 34" zsize=" 168"
VARS crop_T2 input=" /study/MRI/T2_aligned.img" output=" /study/MRI/T2_trim.img" xmin="
   156" xsize=" 216" ymin=" 194" ysize=" 202" zmin=" 34" zsize=" 168"
VARS multispectral_bias_correction input_t1=" /study/MRI/T1_trim.img" input_t2="
   /study/MRI/T2_trim.img" no_segments="-n " output_restored="-or "
```

condor_submit_dag align_t2_to_t1.dag

```
##################################
# linear_registration_-_fsl_simple.sumbit
#
##################################
universe = vanilla
log = /study/MRI/condorFiles/linear_registration_-_fsl_simple.log
error = /study/MRI/condorFiles/linear_registration_-_fsl_simple.error
output = /study/MRI/condorFiles/linear_registration_-_fsl_simple.output

Executable = /usr/local/bin/flirt
Arguments = $(dof) $(cost) $(input) $(ref) $(omat) $(out)

queue
```

```
Executable = /usr/local/bin/avwroi_16SI
Arguments = $(output) $(input) $(xmin) $(xsize) $(ymin) $(ysize) $(zmin) $(zsize)

queue
```

```
Executable = /usr/local/bin/mfast
Arguments = $(input_t1) $(input_t2) $(no_segments) $(output_restored)

queue
```



Methods: Condor and LONI Pipeline

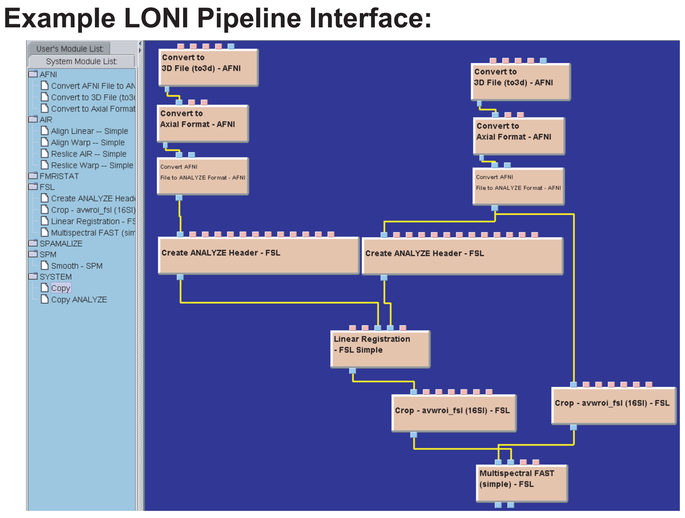
Neuroimaging researchers have recently proposed pipeline or batch processing approaches for streamlining the execution of complex analysis protocols, e.g. LONI (Rex et al, 2003) and FISWIDGETS (Fissel et al, 2003). These pipeline programs provide an excellent mechanism for managing data-flow from one program to another, and executing the same protocol for multiple subjects, but are less proficient at managing program execution across multiple computers. In contrast, Condor excels at managing program execution while maintaining a utilitarian interface for pipeline creation. Since LONI is currently the most extensible and accessible utility for visualizing the execution patterns of multiple processes, we decided to build on the existing LONI framework. LONI Pipeline uses a simple Extensible Markup Language (XML) wrapper to enclose command-line arguments, and provides a cross-platform graphical user interface to create and manipulate these modules. To translate LONI pipelines into Condor scripts, we created a simple utility that reads the LONI Pipeline XML files and creates the necessary Condor files for execution. We provide this LONI-to-condor pipeline conversion utility freely to the neuroimaging community.

Open-source python code facilitating the conversion from LONI Pipeline XML to Condor scripts can be downloaded from:
http://brainimaging.waisman.wisc.edu/~fox

Example LONI Pipeline XML Module:

```
<?xml version="1.0"?>
<Module name="Copy ANALYZE"
    helpfile="http://cs-www.bu.edu/help/unix/copying_files_with_cp.html"
    description="Copy file1.img to file2.img"
    command="/bin/cp -InFile1 -OutFile1">
  <InFile index="1"
        description="Input File"
        type=" file"
        filename="file1.img"
        isOptional="true"
        />
  <OutFile index="1"
        description="Output File"
        type=" file"
        filename="${3:r} - 1 + 2 + ${1:e}"
        />
</Module>
```

Example LONI Pipeline Interface:



For more information on the LONI Pipeline Processing Environment visit:
http://www.loni.ucla.edu/Software/

Results

Condor has been demonstrated to run on a variety of different clusters, and is currently running on over sixty-thousand computers in over fifteen-hundred pools.

Condor can be used on both small-scale and large-scale computer installations, and can easily link multiple software packages together in a single pipeline. Condor executes across servers and workstations alike, effectively matching computer capabilities with individual program requirements.

Within our lab, we have successfully implemented Condor over 20 computers. We have demonstrated Condor's capability to execute processes from different neuroimaging software packages, namely AIR, SPM, FSL, and AFNI, and manage their execution. Furthermore, we have created a utility for converting LONI Pipeline XML pipelines into Condor DAG files for efficient execution.

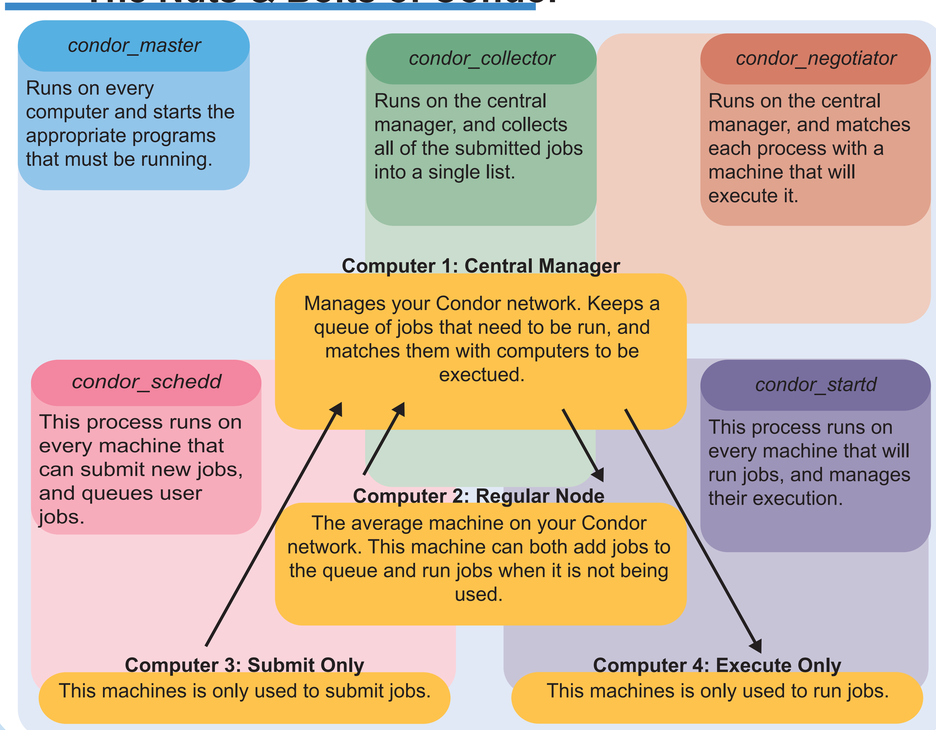
Examples of simple LONI Pipeline XML Pipelines and the LONI to Condor conversion utility is available online at http://brainimaging.waisman.wisc.edu/~fox

Discussion

Condor is ideally suited for use in neuroimaging process scheduling. By demonstrating a conversion utility that will allow LONI Pipeline XML pipelines to be translated into Condor scripts, we believe Condor can be efficiently implemented in large and small-scale neuroimaging labs around the world.

Condor is a package that can be easily implemented and applied to complex and varied neuroimaging data analyses. Furthermore, Condor can easily adapt to implement analysis protocols that require the use of different software packages, and its pipelines can

The Nuts & Bolts of Condor



Supported by NIH Grant MH67167-01.

- Condor references can be found at: http://www.cs.wisc.edu/condor/publications.html
- Fissell K, Tseytlin E, Cunningham D, Iyer K, Carter CS, Schneider W, Cohen JD. Fiswidgets: a graphical computing environment for neuroimaging analysis. Environment. Neuroimage. 2003; 19(3):1033-48.
- Oakes TR, Johnstone T, Ores-Walsh KS, Greischar LL, Alexander AL, Fox AS, Davidson RJ (2005) Comparison of fMRI motion correction software tools. Neuroimage. (in press).
- Rex DE, Ma JQ, Toga AW. The LONI Pipeline Processing Environment. Neuroimage. 2003; 19(3):1033-48.