



Status of Standalone Inspiral Code

Duncan Brown
University of Wisconsin-Milwaukee

LIGO Scientific Collaboration
Inspiral Working Group

LIGO-G030596-00-Z

Standalone Inspiral Code Status

- Standalone inspiral code has been written
 - » `lalapps_tmpltbank` based on LAL bank package
 - » `lalapps_inspiral` based on LAL findchirp package
- Flat search inspiral code debugged, tested and validated
- Bank package tested by Cardiff
- Can run on any machine with LAL and LALapps installed
 - » Requires FFTW, FrL, dataflow
 - » Available in LIGOtools
- Code is in `lalapps/src/inspiral/` directory

Enhancements to LAL Code

- Actual inspiral engine based on findchirp package is essentially unchanged
 - » 2 pN stationary phase template matched filtering
- Modification to χ^2 code
 - » Code reports χ^2 with $2p-2$ degrees of freedom (p is number of bins)
 - » Internally thresholds on $\chi^2 < (\text{threshold})(p + r^2 d^2)$
- Template bank generation code has been enhanced to reduce overcoverage
[Sathya and Cardiff group]
- All called from new standalone code

Standalone Code Engine

- Based on hierarchical search engine developed at UWM in spring 2000
[Allen, Anderson, Brady, Brown, J & T Creighton]
- Reads data from frames, templates from XML
- Writes output triggers to LIGO lightweight XML
- Write output data to frames, if requested
 - » input data, filtered data, PSD, calibration, r , x^2 , etc.
- Writes output triggers to LIGO lightweight XML

New LAL Functionality Used by Engine

- Streamlined frame reading/writing code
[J Creighton]
- Code to compute a response from frame calibration data
[Brady, Brown, J Creighton]
- Resampling code for integer downsampling
[Brown, T Creighton]
- Median PSD estimation code (mean also available)
[Brady, Brown, Williamsen]
- LAL structures that correspond to database tables
[Brown]
- Functions to write database structures to XML
[Brown]

Grid Infrastructure Used (Koranda)

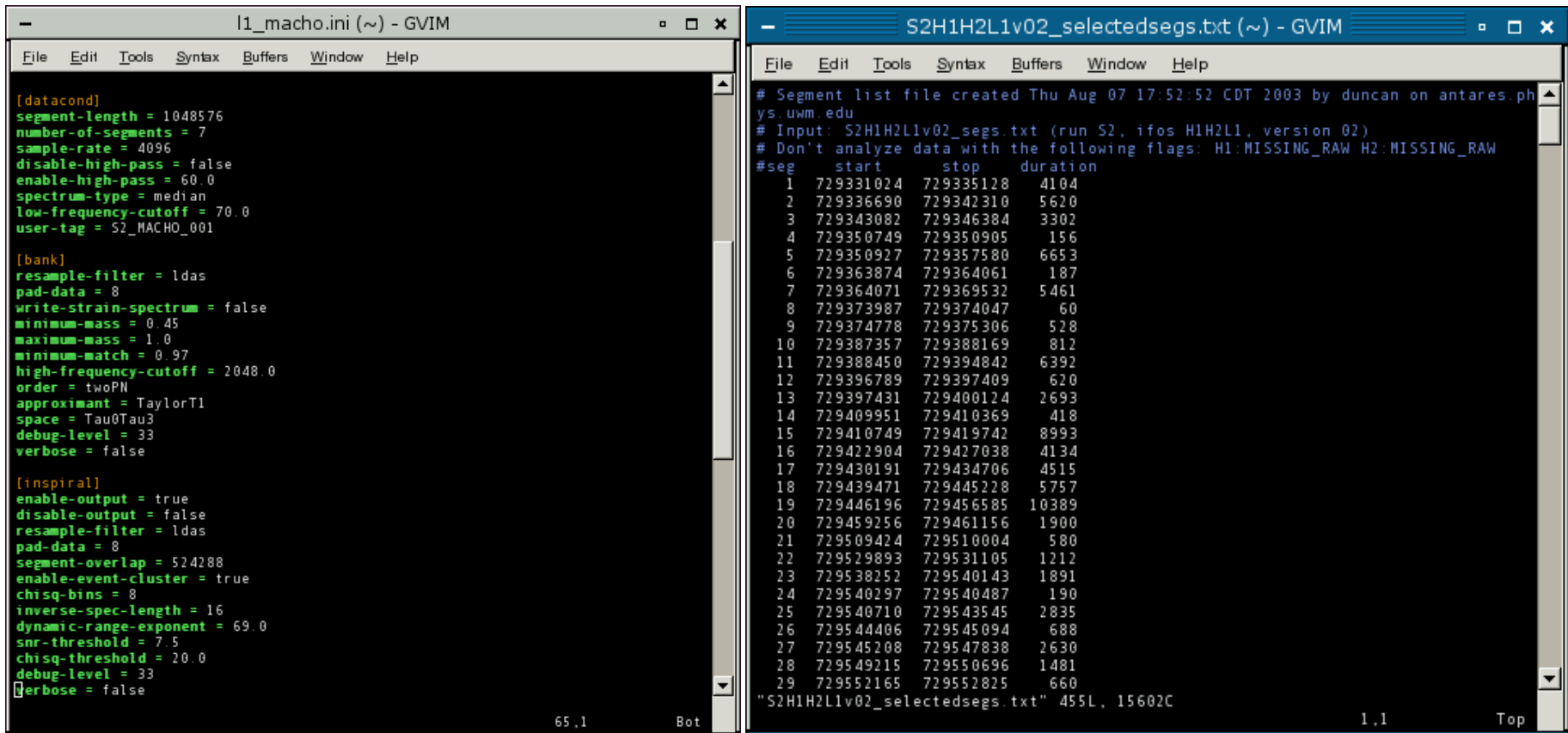
- A condor DAG (workflow description) executes the search
- Uses LALdataFind to query LDR to discover data
- Runs lalapps_tmpltbank to generate banks
- Runs lalapps_inspiral to perform search
- Code available to insert XML files into LIGO metadata database

What is a Condor DAG?

- Human readable description of workflow (text file)
- Tells condor what needs to be done and in what order
- Condor executes the DAG to run the code on available computing resources
- On exit, condor writes a “rescue” DAG containing instructions to run any failed jobs

Use a Simple Script to Generate DAGs

- Input is a parameter file and a list of science segments



The image shows two side-by-side screenshots of the GVIM text editor. The left window, titled 'l1_macho.ini (~) - GVIM', displays a configuration file with sections for [datacond], [bank], and [inspiral]. The right window, titled 'S2H1H2L1v02_selectedsegs.txt (~) - GVIM', displays a segment list file with a header and a table of segments.

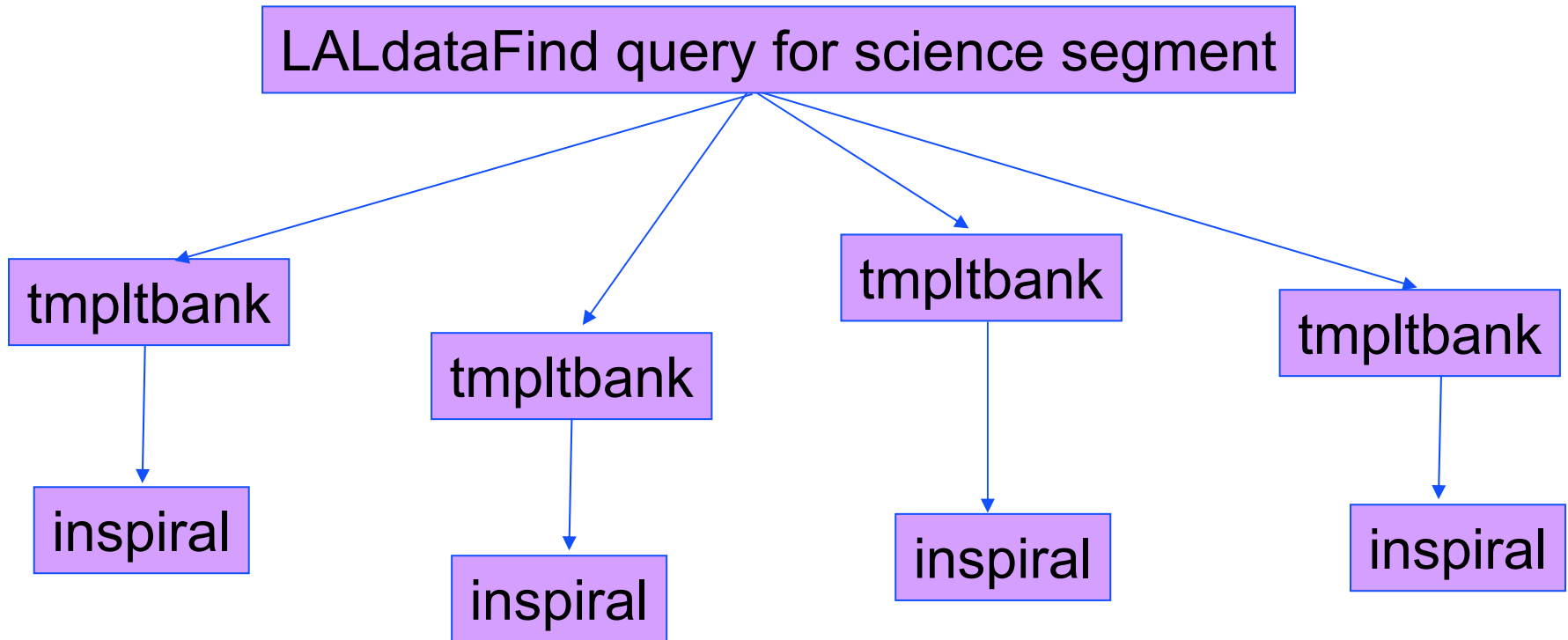
```
[datacond]
segment-length = 1048576
number-of-segments = 7
sample-rate = 4096
disable-high-pass = false
enable-high-pass = 60.0
spectrum-type = median
low-frequency-cutoff = 70.0
user-tag = S2_MACHO_001

[bank]
resample-filter = ldas
pad-data = 8
write-strain-spectrum = false
minimum-mass = 0.45
maximum-mass = 1.0
minimum-match = 0.97
high-frequency-cutoff = 2048.0
order = twoPN
approximant = TaylorT1
space = Tau0Tau3
debug-level = 33
verbose = false

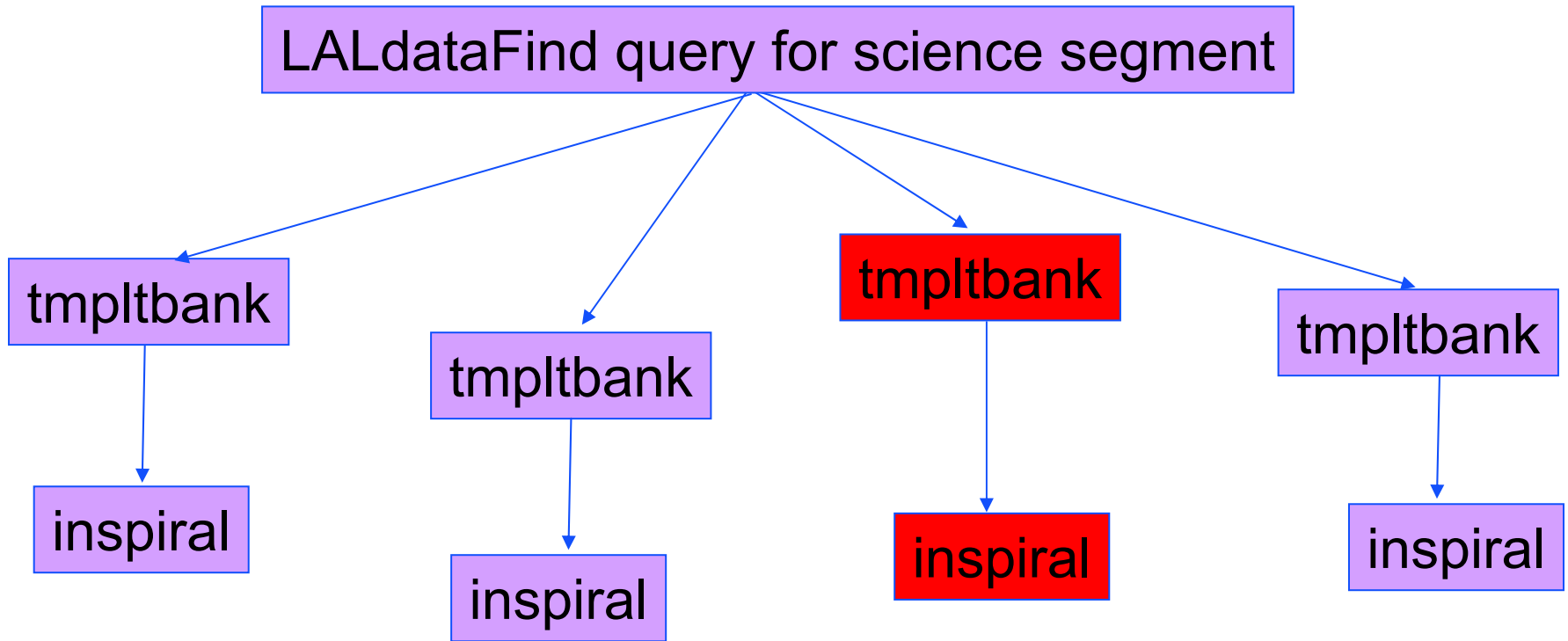
[inspiral]
enable-output = true
disable-output = false
resample-filter = ldas
pad-data = 8
segment-overlap = 524288
enable-event-cluster = true
chisq-bins = 8
inverse-spec-length = 16
dynamic-range-exponent = 69.0
snr-threshold = 7.5
chisq-threshold = 20.0
debug-level = 33
verbose = false
```

```
# Segment list file created Thu Aug 07 17:52:52 CDT 2003 by duncan on antares.ph
ys.uwm.edu
# Input: S2H1H2L1v02_segs.txt (run S2_ifos_H1H2L1, version 02)
# Don't analyze data with the following flags: H1:MISSING_RAW H2:MISSING_RAW
#seg start stop duration
1 729331024 729335128 4104
2 729336690 729342310 5620
3 729343082 729346384 3302
4 729350749 729350905 156
5 729350927 729357580 6653
6 729363874 729364061 187
7 729364071 729369532 5461
8 729373987 729374047 60
9 729374778 729375306 528
10 729387357 729388169 812
11 729388450 729394842 6392
12 729396789 729397409 620
13 729397431 729400124 2693
14 729409951 729410369 418
15 729410749 729419742 8993
16 729422904 729427038 4134
17 729430191 729434706 4515
18 729439471 729445228 5757
19 729446196 729456585 10389
20 729459256 729461156 1900
21 729509424 729510004 580
22 729529893 729531105 1212
23 729538252 729540143 1891
24 729540297 729540487 190
25 729540710 729543545 2835
26 729544406 729545094 688
27 729545208 729547838 2630
28 729549215 729550696 1481
29 729552165 729552825 660
```

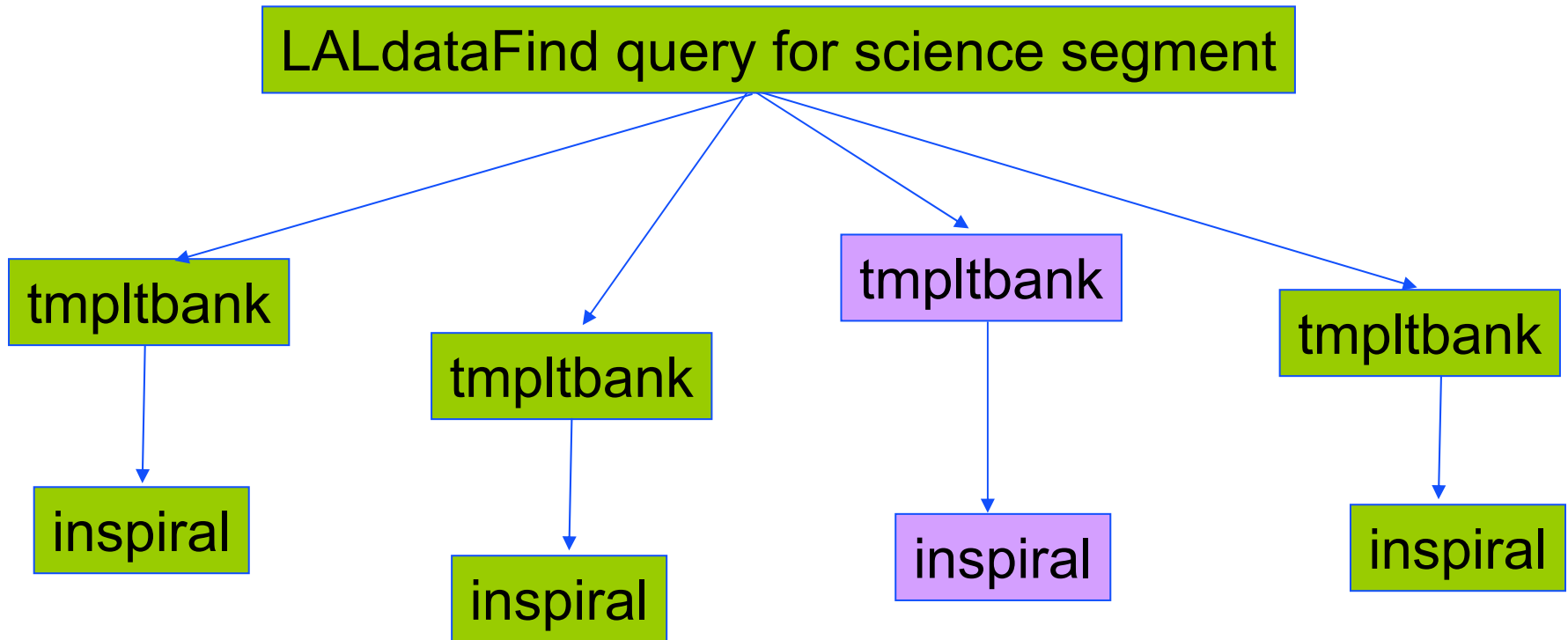

Script writes DAG to describe workflow



Script writes DAG to describe workflow

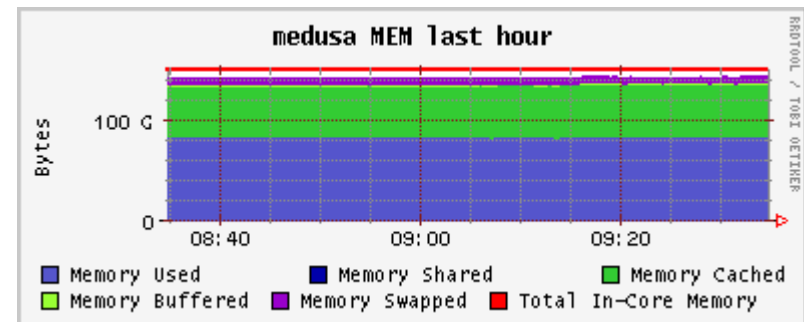
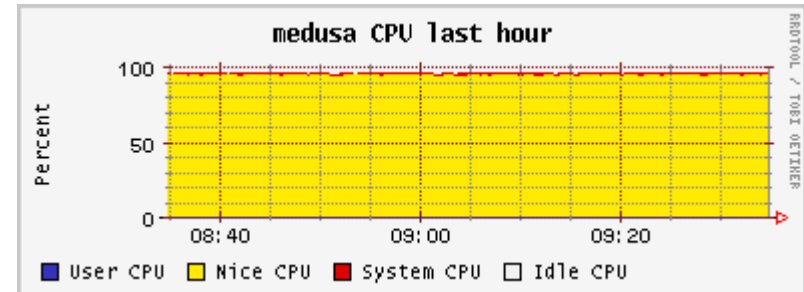
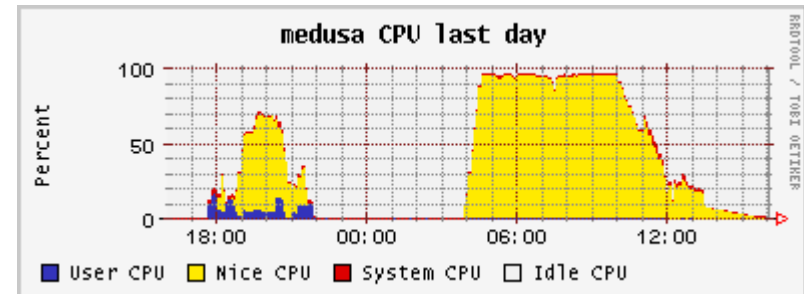


Script writes DAG to describe workflow



Timing of Inspiral Code

- Have run all S2 searches on triple coincident **playground** data since the weekend
- Have also run S2 NS search on **all** triple coincident data (255 hours of usable data, will improve usable data)
- Search takes 14 hours to execute on 256 node cluster at UWM (1GHz P3 nodes)
- Can run the playground fast!



Further Developments

- Write a small piece of code to turn triggers into templates
- Write a small piece of code to check triggers for coincidence
- Update the DAG generation script for the full S2 pipeline
- Start testing scientific enhancements to pipeline
- Add additional code to search for BCV templates
- Add injection interface to detection waveform family