Parallel Recon Status

Problem:

Due to the overhead in data taking for taking short enough runs that make moderate demands on CPU and filesizes, it is desirable to allow for longer runs (hence larger digi files) and reconstruct pieces of them in parallel.

The pieces of the puzzle are:

1. read in input digi file and find number of events to be processed.
2. create a list of event ranges for each job to munch on
3. set up the run conditions (via jobOptions, environment variables) for Gleam
4. fork off separate jobs for each piece waiting for them all to finish
5. concatenate the output recon and merit files

State of Prototyping

The pieces of this have been prototyped using python by Richard and Warren, and the code can be found on SLAC afs in /afs/slac/u/ey/richard/GLAST /Pipeline/parallel/. The work so far is done in recon.py. It opens a digi file and gets the number of events, calculating the event ranges to work on.

It makes up a fake set of batch jobs and runs them, waiting for the last to finish.

It then sets up to concatenate the output files.

There are 2 ouput modules suppling tools: reconPM.py and runMany.py (suppled by Warren).

What it does not do

It doesn't create the shell/jobOptions files to run Gleam with. At the moment, it does not get the return codes from the batch jobs, so it cannot tell if they failed.

Presumably it should also delete the intermediate files if all went well.

TODO list

- fix the rc handling of the threads
- prepare the shell and jobOptions files for each job
- report the rc at end of the script and clean up intermediate files if all went well

As things stand, I'm hoping for Warren to pick up these pieces to construct a parallel recon task that handles his recon needs.