**launcher_creator.py**

Create SGE or SLURM launchers for Stampede and Lonestar

*launcher_creator.py* is a convenience utility that automates creating a launcher for TACC clusters based on parameters (and commands) you give it. It outputs a either a SLURM or SGE file that can be submitted to Stampede or Lonestar using `sbatch` or `qsub`. (It detects which machine it is being executed on, so the same code should work on either cluster.) It can run both Bash commands on a single node and/or a parametric list of commands to be dispatched to multiple nodes.

**How To Use It**

At the command line:

```
$ launcher_creator.py <options listed below>
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-n</code></td>
<td>name</td>
</tr>
<tr>
<td><code>-a</code>, <code>-A</code></td>
<td>allocation</td>
</tr>
<tr>
<td><code>-q</code></td>
<td>queue</td>
</tr>
<tr>
<td><code>-w</code></td>
<td>wayness</td>
</tr>
<tr>
<td><code>-N</code></td>
<td>number of nodes</td>
</tr>
<tr>
<td><code>-t</code></td>
<td>time</td>
</tr>
<tr>
<td><code>-e</code></td>
<td>email</td>
</tr>
<tr>
<td><code>-l</code></td>
<td>launcher</td>
</tr>
<tr>
<td><code>-m</code></td>
<td>modules</td>
</tr>
<tr>
<td><code>-b</code></td>
<td>Bash commands</td>
</tr>
<tr>
<td><code>-j</code></td>
<td>Command list</td>
</tr>
<tr>
<td><code>-s</code></td>
<td>stdout</td>
</tr>
</tbody>
</table>

Note that while the `-b` and `-j` options are both *optional*, using neither would pass no work to Lonestar, so you want to use at least one of them!

**Manage Modules**
Use the \texttt{-m} option to load and swap modules for the script. Use the exact same syntax to load or swap module as you would at the command line.

\textit{Example:} \texttt{-m "module load luatools"}

To run several module commands in the launcher, just separate the commands with a semicolon, as you would in the shell.

\textit{Example:} \texttt{-m "module swap intel gcc; module load bedtools"}

Or you can use the shortcut \texttt{ml} command to perform several module actions in one command.

\textit{Example:} \texttt{-m "ml -intel gcc bedtools"}

\textbf{Run multi-line shell commands}

The \texttt{-b} option lets you insert shell commands into the launcher script. For a short number of commands, separating them with a semicolon may be convenient. For longer commands you may want to write the commands on several lines. To insert a multi-line bash script into a \texttt{launcher_creator.py} command, you can use a \texttt{heredoc} like so:

\begin{Verbatim}
Use a heredoc to assign multiple lines to a variable, then use this variable as input to the \texttt{-b} option.

\begin{verbatim}
jelly_bash=$(cat <<-JELLY
ulimit -s unlimited
Trinity.pl --seqType $seqtype --JM 900G $input_data --output $trinity_dir
--CPU $cpu_per_node --no_run_chrysalis
JELLY
)

launcher_creator.py -n jelly -t 6:00:00 -b "$jelly_bash"
\end{verbatim}
\end{Verbatim}

\textbf{Create and submit a launcher in one line using \texttt{-s}}

The \texttt{-s} option outputs the name of the launcher to STDOUT. With command substitution you can immediately submit the new launcher file.

\begin{Verbatim}
Immediately submit a launcher file on Stampede

\begin{verbatim}
sbatch $(launcher_creator.py -n variant_calling -t 3:00:00 -q normal -b "freebayes -f mapping/bwa_TAIR10/TAIR10_all.fasta alignments.sorted.bam > alignments.vcf" -w 1 -s)
\end{verbatim}
\end{Verbatim}

\textbf{Examples}

\textbf{Distributing Shrimp to four nodes}

- Make a list of the Shrimp commands to run. You can do this in a text editor, algorithmically with a Bash loop, whatever. Let's pretend this text is saved as \texttt{mapping_commands.list}
### List of Shrimp commands

- `gmapper-ls -N 6 --qv-offset 33 -p opp-in --fastq -1 Sample_R1-00.fq -2 Sample_R2-00.fq reference.fasta > Sample_mapped-00.sam`
- `gmapper-ls -N 6 --qv-offset 33 -p opp-in --fastq -1 Sample_R1-01.fq -2 Sample_R2-01.fq reference.fasta > Sample_mapped-01.sam`
- `gmapper-ls -N 6 --qv-offset 33 -p opp-in --fastq -1 Sample_R1-02.fq -2 Sample_R2-02.fq reference.fasta > Sample_mapped-02.sam`
- `gmapper-ls -N 6 --qv-offset 33 -p opp-in --fastq -1 Sample_R1-03.fq -2 Sample_R2-03.fq reference.fasta > Sample_mapped-03.sam`

- Generate a launcher using `launcher_creator.py`.

  ```
  launcher_creator.py -q normal -a MyAllocation -n MapSample -j mapping_commands.list -w 2 -t 6:00:00 -l map_sample.sge -m "module load shrimp"
  ```

- Submit the newly-created launcher.

  ```
  qsub map_sample.sge
  ```

  ```
  sbatch map_sample.slurm
  ```