“Array syntax” for LSF job submission

Craig Schwartz
NCAR/NESL/MMM
Array syntax is an efficient way of submitting multiple jobs simultaneously.

Useful when submitting a large number of jobs:
- Ensemble forecasting
- Forecast evaluation

Most batch-submission platforms have this feature.
How to submit an array of jobs

- Typical job submission submits just one job:

  >> bsub –J "myjob" < myjob.csh
  >> bjobs

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USER</th>
<th>STAT</th>
<th>QUEUE</th>
<th>FROM_HOST</th>
<th>EXEC_HOST</th>
<th>JOB_NAME</th>
<th>SUBMIT_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>642847</td>
<td>schwart</td>
<td>PEND</td>
<td>caldera</td>
<td>caldera02-i</td>
<td>caldera05-i</td>
<td>myjob</td>
<td>Oct 7 12:31</td>
</tr>
</tbody>
</table>

- With array syntax, multiple jobs are submitted:

  bsub –J "myjob[1-10]" < myjob.csh submits "myjob.csh" 10 times

Job array
## Output of `bjobs`

```bash
>> bsub -J "myjob[1-10]" < myjob.csh
```
```bash
>> bjobs
```

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USER</th>
<th>STAT</th>
<th>QUEUE</th>
<th>FROM_HOST</th>
<th>EXEC_HOST</th>
<th>JOB_NAME</th>
<th>SUBMIT_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>642847</td>
<td>schwart</td>
<td>RUN</td>
<td>caldera</td>
<td>caldera02-i</td>
<td>caldera05-i</td>
<td>myjob[1]</td>
<td>Oct 7 12:31</td>
</tr>
<tr>
<td>642847</td>
<td>schwart</td>
<td>RUN</td>
<td>caldera</td>
<td>caldera02-i</td>
<td>caldera05-i</td>
<td>myjob[8]</td>
<td>Oct 7 12:31</td>
</tr>
<tr>
<td>642847</td>
<td>schwart</td>
<td>RUN</td>
<td>caldera</td>
<td>caldera02-i</td>
<td>caldera05-i</td>
<td>myjob[9]</td>
<td>Oct 7 12:31</td>
</tr>
<tr>
<td>642847</td>
<td>schwart</td>
<td>RUN</td>
<td>caldera</td>
<td>caldera02-i</td>
<td>caldera05-i</td>
<td>myjob[10]</td>
<td>Oct 7 12:31</td>
</tr>
</tbody>
</table>

- All elements have same JOBID, but individual ones can be specified using brackets (e.g., 642847[5])

10/11/13 Yellowstone workflow meeting
Scripting

• When a job array is submitted, each array value is stored as variable `$LSB_JOBINDEX`

• This is how a script knows which array value is being processed

• Thus, an ensemble of forecasts can be run by submitting one job and having the ensemble member equal `$LSB_JOBINDEX`
Scripting

#! /bin/csh
#This script is myjob.csh
#BSUB -P P64000510
#BSUB -n 1
#BSUB -R "span[ptile=32]"
#BSUB -J fcst
#BSUB -o wrf.%I
#BSUB -e wrf.%I
#BSUB -W 1
#BSUB -q caldera

set mem = $LSB_JOBINDEX

```
touch -f mem_${mem}
exit 0
```

"%I" refers to the job element. In this example, output will be wrf.1, wrf.2, wrf.3 ... wrf.10

Will vary between 1 and 10

Output of the program will be files mem_1, mem_2, mem_3, ... mem_10

```
>> bsub -J "myjob[1-10]" < myjob.csh
```

10/11/13 Yellowstone workflow meeting
Array elements

• Do not have to submit “continuous numbers”:
  
  • `bsub –J “myjob[1-10]” < myjob.csh` passes values of 1,2,3,4,5,6,7,8,9,10 to `$LSB_JOBINDEX`
  
  • `bsub –J “myjob[1,3-6,42,90]” < myjob.csh` passes values of 1,3,4,5,6,42, and 90 to `$LSB_JOBINDEX`
  
  • `bsub –J “myjob[1,10:2]” < myjob.csh` passes values of 1,3,5,7, and 9 to `$LSB_JOBINDEX`
Killing an array of jobs

• To kill the whole array, you only need to specify the single job number:

    \texttt{bkill 123} kills all jobs in the array

• You can also kill a single entry of an array:

    \texttt{bkill “123[5]”} kills job array element 5
Limiting the number of jobs that can run

• Sometimes it may be desirable to limit how many array members can run simultaneously

• This can be limited by adding “%val” in the submission

```bash
bsub -J "myjob[1-50]%17" < myjob.csh
```

• In this example, a maximum of 17 jobs will run simultaneously
Dependency conditions

Arrays can easily be used in dependency conditions:

```
>> bsub –J “myjob[1-10]” < myjob.csh
```

```
>> bsub –w “ended(myjob[1-10])” –J “myjob2[1-10]” < myjob2.csh
```

- In this case, “myjob2” won’t execute until all elements of “myjob” have ended
Conclusion

- Array syntax is a concise method of submitting multiple jobs
- All array elements referenced by a single job number
- Avoids “looping” and unnecessarily submitting a large number of jobs with different numbers
- For more information, google “array syntax LSF” and you’ll find many relevant pages