UCRL-SM-230043

LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

Understanding Job Priority Calculation on Livermore Computing's Moab-Scheduled Machines

Donald A. Lipari

February 6, 2008

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

Understanding Job Priority Calculation on Livermore Computing's Moab-Scheduled Machines

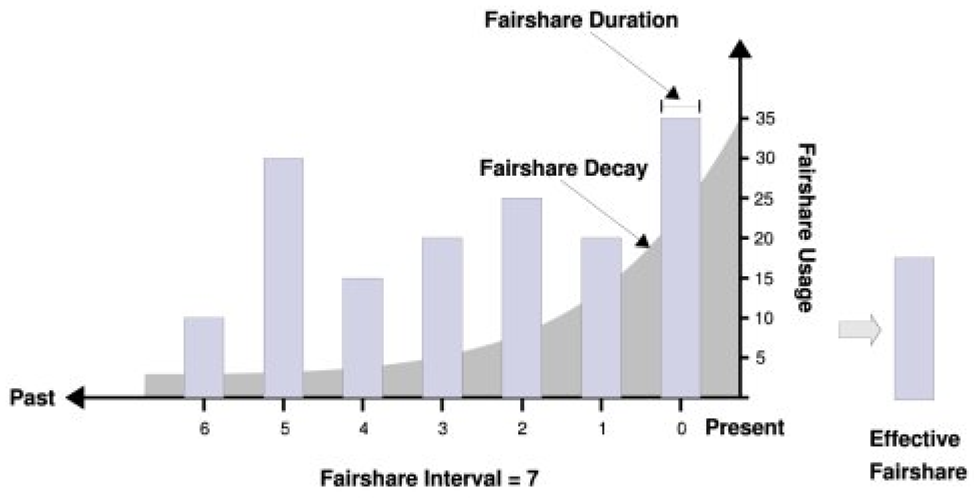
While Moab offers a large set of configuration parameters that can contribute to job priority, only four are enabled for jobs running on Livermore Computing machines: fair-share, node size, queue wait time, and quality-of-service. This perpetuates the same policies users are used to from years of running jobs under the Livermore Computing Resource Management (LCRM) system. This report explains how Moab calculates job priority based on the configured settings.

Fair-Share

The heart of the fair-share concept is the Moab account. Every job must be associated with an account. When one is not overtly specified, the user's default account will be assigned. All computing resources used by the job will be charged to that account. Currently, CPU time is the only computing resource charged.

Every account is configured with a share value that represents the portion of the machine that has been promised to the set of customers who are granted the privilege of charging to that account.

Every job waiting in the queue to run will contain a fair-share component to its priority. The value of the fair-share component represents the delta between the assigned share of the machine and the weighted usage. The usage is the sum of usage over the most recent, collection intervals, where each interval's usage is weighted by a decay factor. In this way, a user's most recent usage counts more than last week's usage. The configuration parameters have been chosen to achieve a half-life usage decay of two weeks.



The *mdiag -f -v* command is used to view the fair-share component of a job's priority:

The Upper Half of the *mdiag -f -v* Display

The upper half of the output displays usage by each of the five Moab credentials. The only credential that is configured to contribute to the fair-share component is the user credential. The user section of the *mdiag -f -v* output looks like this:

```
FairShare Information

Depth: 14 intervals   Interval Length: 1:00:00:00   Decay Rate: 0.90

FS Policy: DEDICATEDPS
System FS Settings:  Target Usage: 0.00    Flags: 0

FSInterval        %     Target       0       1       2       3       4       5       6...
FSWeight       ------- -------  1.0000  0.9000  0.8100  0.7290  0.6561  0.5905  0.5314
TotalUsage      100.00 -------  118718  192816  150678  104718  195038  201482  199520

USER
-------------
bill              0.01 -------    0.01    0.01    0.05 ------- ------- -------    0.00
mary              0.00 ------- ------- ------- ------- ------- ------- ------- -------
fred              1.88 -------    2.88    2.93    3.86    2.80    1.05 -------    0.01
dawn              0.00 ------- ------- -------    0.00 -------    0.00 -------    0.00
marcia            3.92 -------   17.96    2.76 -------    7.63    5.92 -------    0.01
starr             0.00 ------- ------- ------- ------- ------- ------- ------- -------
ted               9.83 -------    3.08   17.98   23.25   15.26    6.85 -------    0.39
```

The above display indicates that there are 14 fair-share intervals configured (though only the first 7 are shown) and that each interval lasts one day. The contribution from the current day’s usage interval is 100%. Yesterday’s usage is weighted by .9; the previous day, .81, etc. The weighted sum over the 14 day interval determines the user’s usage value. All usage values in the upper half are displayed as percentages of the total time delivered to batch jobs.

The day starts at 0 Hours, UTC. So, all of the interval values will shift to the right at 4pm PST / 5pm PDT. The TotalUsage row indicates the amount of CPU resources utilized in each interval by batch jobs and is stated in processor-hours.

While this information will be helpful, one should realize that the percentages reported for each user are aggregates across all the accounts their jobs may have charged. In the second half of the mdiag -f -v display, the usage for each user is separated out by account.

The Lower Half of the *mdiag -f -v* Display

The lower half of the *mdiag -f -v* output presents the fair-share factors for each account and includes the target and usage for each account used to derive the fair-share factors:

```
Share Tree Overview for partition 'ALL'
Name                 Usage    Target                (FSFACTOR)
-----                -----    ------              ------------
root             100000.00 100000.00  of 100000.00 (node: 4254839877.28) (0.00)
- projectA         3865.88  10109.00  of 100000.00 (acct: 158414661.65) (42839.61)
  - bill              2.18      1.00  of       3.00 (user: 115016709.59) (42840.61)
  - mary              0.00      1.00  of       3.00 (user: 0.00) (42840.61)
  - fred              0.82      1.00  of       3.00 (user: 43397951.46) (42840.61)
- projectB        16144.12   6740.00  of 100000.00 (acct: 661580368.94) (-64530.11)
  - dawn              1.10      1.00  of       4.00 (user: 181692403.46) (-64529.11)
  - marcia            0.00      1.00  of       4.00 (user: 0.00) (-64529.11)
  - starr              .28      1.00  of       4.00 (user: 46033911.02) (-64529.11)
  - ted               2.62      1.00  of       4.00 (user: 433854054.46) (-64529.11)
```

The Target column presents the portion of the machine that has been promised to the users of each account. In the example above, projectA was promised 10109/100,000 of the machine's CPU cycles per week. For a machine like atlas with 1104 nodes and 8 processors per node, projectA was promised:

$$10{,}109/100{,}000 * 1104 * 8 * 24 * 7 = 150000 \text{ CPU - Hours / week.}$$

The target values are goals for average use. Actual usage may be less (if not enough jobs were submitted) or more (if more of a project's jobs were scheduled to make use of idle cycles).

The second column from the right, presented in parentheses, conveys the weighted usage value (in processor-seconds) charged to each user for each account, i.e., the weighted sum of processor-seconds charged to that account over the last 14 intervals (days). This number, when divided by all the weighted CPU-seconds delivered (4254839877.28), and multiplied by the total number of shares (100000) determines the Usage value in the second column from the left.

The Usage column presents the portion of each target that was actually charged to running jobs and is expressed in the same units as the target value. Usage values that are less than the target represent an under-serviced account. Usage values greater than the target value indicate that more usage than promised has been charged to that account.

The rightmost column, also in parentheses, displays the fair-share factor for each account. This is calculated by taking the target, subtracting the usage and multiplying by a constant. Fair-share factors greater than zero represent cases where an account has yet to be charged for all of its allocated CPU cycles. Fair-share factors less than zero represent cases where more than the promised CPU cycles have been charged to an account.

All users in an account will typically be assigned a target (share) of 1.0. The fair-share factor assigned to a user in an account is based largely on the account to which their job is charged. However, different usage for users within a given account will result in minor differences in fair-share factors.

Job Priority

As stated above, a job's priority will be based on four factors, the quality-of-service (QoS), fair-share, queue wait, and resource utilization. The priority determination for each queued job can be displayed by invoking *mdiag -p*:

```
diagnosing job priority information (partition: ALL)

Job                    PRIORITY*   Cred(  QOS)     FS( User)  Serv(QTime:UPrio)   Res( Node)
           Weights     --------       1(    1)     60(    1)     1(  100:    1)     30(   10)

5846                   -3434995     0.0(  1.0)  98.7(-5801)   0.8(297.6:  0.0)   0.5( 54.0)
5848                   -3450825     0.0(  1.0)  99.1(-5801)   0.8(271.3:  0.0)   0.1( 10.0)
5849                   -3459319     0.0(  1.0)  99.4(-5801)   0.5(180.4:  0.0)   0.1( 12.0)
5862                   -3466589     0.0(  1.0)  99.6(-5801)   0.3( 95.7:  0.0)   0.1( 16.0)
```

The priority value for each job in the above display is the sum of all four factors. Each of the above jobs is a normal job and receives a QoS factor of one. Expedited jobs receive a QoS factor of 100000000 while standby jobs' QoS factors are -100000000.

The fair-share factors displayed for each account in the second half of the *mdiag -f -v* output appear in the parentheses under the FS (User) heading. These values are truncated to 5 places.

The Serv-QTime heading presents the portion of the job priority based on the number of minutes the job has been waiting in the queue. When users request a decreased priority for their jobs (using the *msub -p* option), this priority will appear under the Serv-UPrio heading.

The Res(Node) heading presents the portion of the job priority based on the size of the job (number of nodes).

Each factor is given a nominal weighting and this appears in the heading. The nominal weighting of each of the factors are established to implement the policy of LC management. In this case, the fair-share factor counts for the majority of the job's priority. However, the queue wait factor will help ensure that a job with a low fair-share factor will eventually run, the longer it sits in the queue. Similarly, larger jobs are favored on a capability machine and the weighting of the resource component is set to favor larger jobs.

The actual percentage contribution of each of these factors to a job's priority value is shown just to the left of the parenthetical values under each factor's heading.

In the example below, job 5816 has been in the queue so long that its queue wait factor is beginning to contribute more to its job priority. Similarly, jobs 5838, and 5840 are larger sized jobs and are getting more of a boost from their resource component:

```
diagnosing job priority information (partition: ALL)

Job                  PRIORITY*   Cred(  QOS)    FS( User)  Serv(QTime:UPrio)   Res( Node)
           Weights   --------       1(    1)    60(    1)     1(  100:    1)    30(   10)

5882                  1247765     0.0(  1.0)  98.2(20431)   1.6(195.0:  0.0)   0.2(  8.0)
5838                 -1004887     0.0(  1.0)  89.2(-1904)   4.8(612.9:  0.0)   6.0(256.0)
5840                 -1007389     0.0(  1.0)  89.4(-1904)   4.6(587.9:  0.0)   6.0(256.0)
5816                 -1023987     0.0(  1.0)  90.6(-1904)   7.9(997.9:  0.0)   1.5( 64.0)
5818                 -1119869     0.0(  1.0)  98.0(-1904)   0.3( 39.1:  0.0)   1.6( 64.0)
5779                 -4296328     0.0(  1.0)  96.4(-7436)   3.2(1465.:  0.0)   0.4( 64.0)
5817                 -4438990     0.0(  1.0)  99.5(-7436)   0.1( 39.1:  0.0)   0.4( 64.0)
5510                 -5897900     0.0(  1.0)  97.2(-1012)   1.6(996.6:  0.0)   1.2(256.0)
5861                 -5963120     0.0(  1.0)  98.2(-1012)   0.5(308.4:  0.0)   1.3(268.0)
5744                -10957186     0.0(  1.0)  98.0(-1863)   1.8(2048.:  0.0)   0.2( 64.0)
Percent Contribution --------     0.0(  0.0)  97.2( 97.2)   2.1(  2.1:  0.0)   0.7(  0.7)

* indicates absolute/relative system prio set on job
```