Adaptive COMPUTING

Intelligent Workload Management Software for HPC and Cloud Environments

Moab Reservations

Shawn Hoopes
Director of Training
shawn@adaptivecomputing.com

May 1, 2014

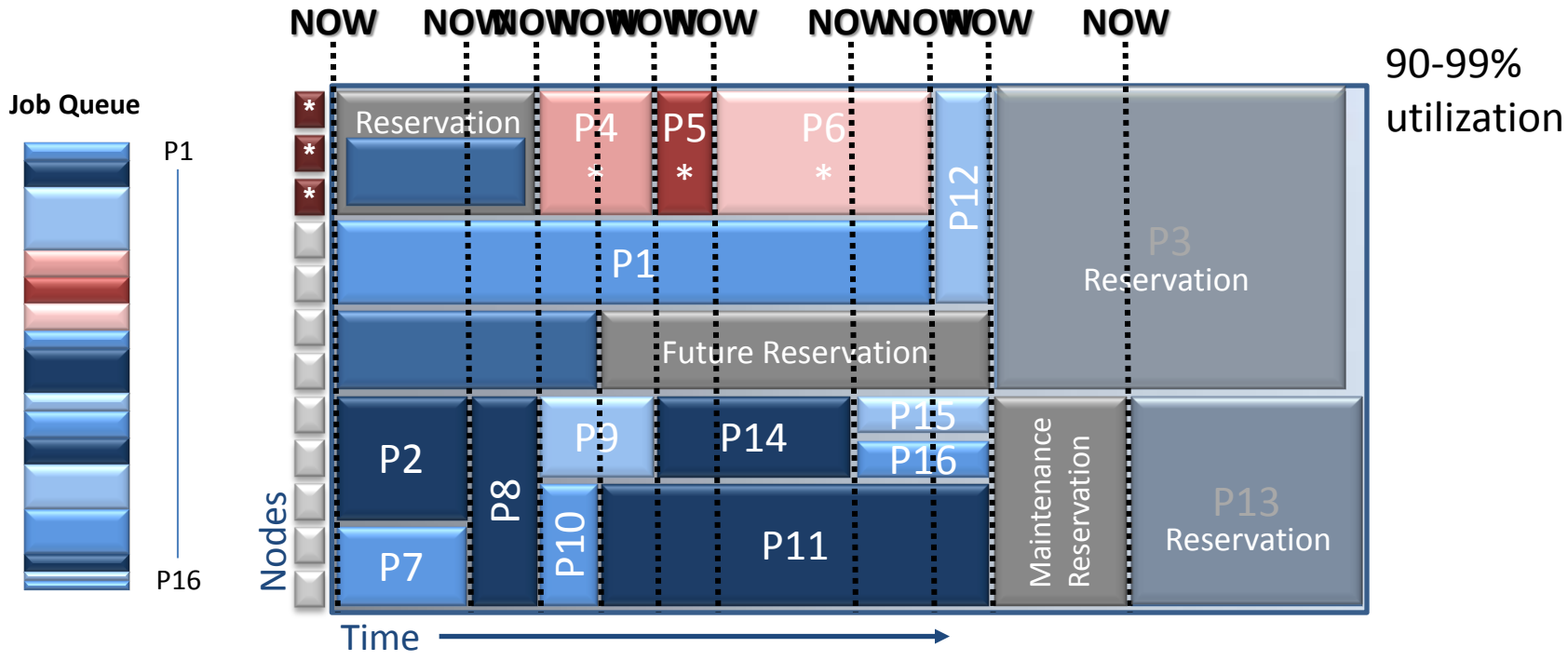
Reservation Overview

- **A reservation is the mechanism by which Moab guarantees the availability of a set or resources at a particular time**
- **Space and time considerations affect all decisions**
- **Associate ownership, policies, actions, and environments to specific resources at specific times**
- **Dynamically scheduled and optimized**

Reservation Types

- **Idle Job Reservations**
- **Active Job Reservations**
- **System Reservation**
- **System Reservation with ACL**
- **Standing Reservation**
- **Rolling Reservation (Rollback Offset)**

Reservation Types



Adaptive COMPUTING

Idle Job Reservation

- **Reservation Depth**
  - RESERVATIONDEPTH = [VALUE]
  - Reasons to Increase RESERVATIONDEPTH
    - Estimated start time (`showstart`) is heavily used and the accuracy needs to be increased
    - Users are more interested in knowing when their job will run than in having it run sooner
  - Reasons to Decrease RESERVATIONDEPTH
    - Scheduling efficiency and job throughput need to be increased
  - Reservation Policy (RESERVATIONPOLICY)
    - Highest, CurrentHighest, Never

Active Job Reservation

- **showq**

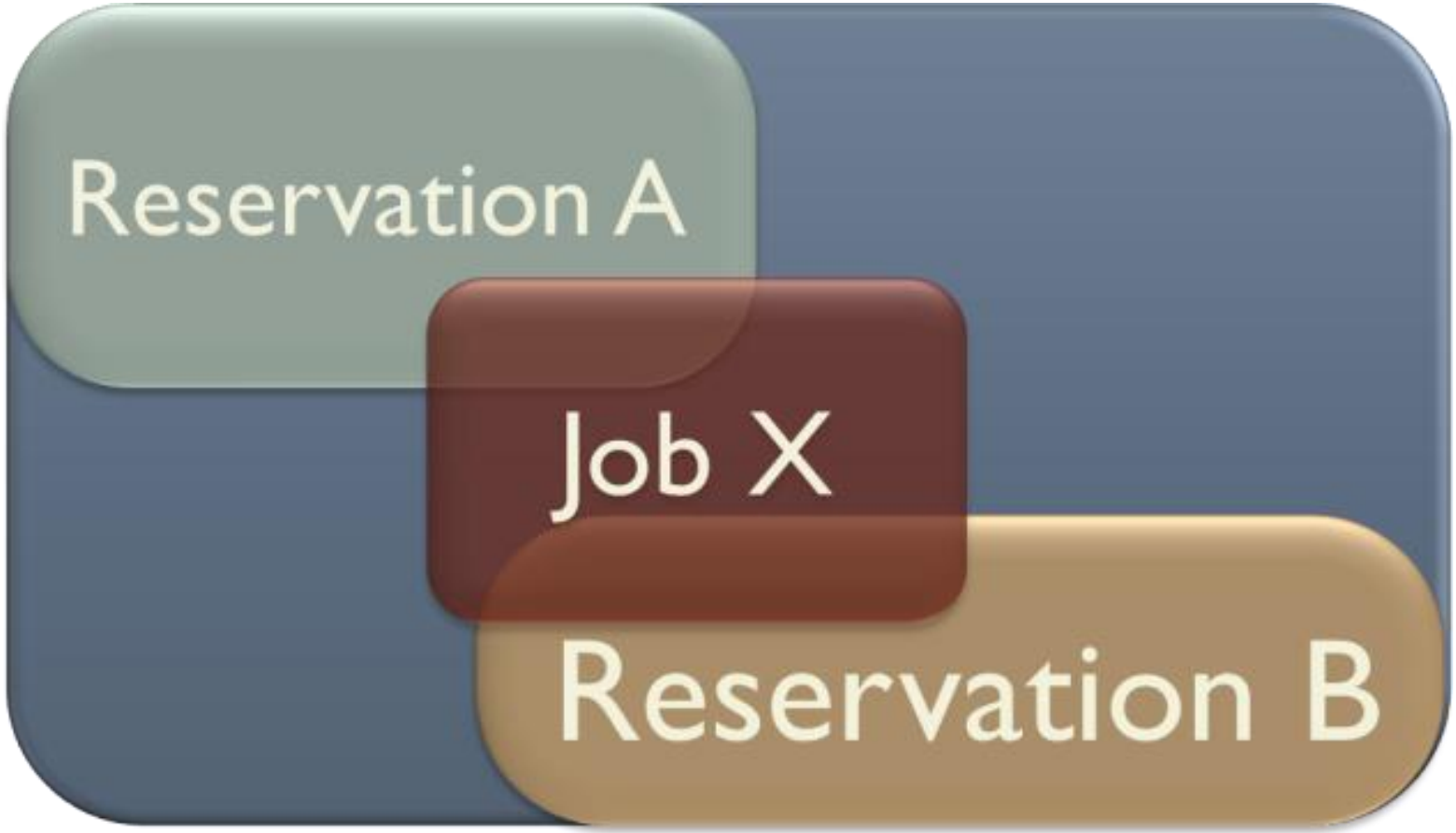
```
[root@it-mc05 ~]# showq
active jobs------------------------
JOBID              USERNAME      STATE PROCS   REMAINING            STARTTIME
116                 shoopes    Running    12    00:59:58  Mon Apr 28 09:45:21
1 active job               12 of 36 processors in use by local jobs (33.33%)
                             1 of 3 nodes active      (33.33%)
```

- **mdiag –r –v**

```
[root@it-mc05 ~]# mdiag -r
Diagnosing Reservations
RsvID                    Type  Partition    StartTime     EndTime     Duration Node Task Proc
-----                    ----  ---------    ---------     -------     -------- ---- ---- ----
116                       Job     torque   -00:00:15    00:59:45      1:00:00    1   12   12
    Flags: ISACTIVE
    ACL:   JOB==116=
    CL:    JOB==116 USER==shoopes GROUP==company CLASS==batch JATTR==checkpoint JPRIORITY<=0
DURATION==1:00:00 PROC==12 PS==43200 CLUSTER==Moab TASKSPERNODE==12
    SubType: JobReservation
    Nodes='it-mc08.ac:1'
    Rsv-Group: 116

Active Reserved Processors: 12
```

Active Job Reservation



Administrative Reservations

- **System Reservations**
- **Create using `mrsvctl -c`:**

```
mrsvctl -c -t 4 -s +2:00:00 -d 1:00:00
```

creates a 4-task, 1-hour system reservation that starts in 2 hours

```
mrsvctl -c -h node0[1-2] -d 12:00:00
```

creates a 12-hour reservation on nodes node01 and node02

Administrative Reservations

- **System Reservation**

```
# mrsvctl -c -t ALL
# reservation system.1 created

# mrsvctl -c -t ALL -s +1:00:00
# reservation system.2 created

# mrsvctl -c -h node0[0-9][0-9] -d 24:00:00
# reservation system.3 created

# mrsvctl -c -h node0[0-9][0-9] -T Action="/tmp/update.pl \
$HOSTLIST",atype=exec,etype=start -s 23:50:00_6/15 -d 15:00
# reservation system.4 created
```

Administrative Reservations

- **System Reservation w/ACL**

```
# mrsvctl –c –h node0[0-9][0-9] -d 24:00:00 -a user==fred
# reservation fred.5 created

# mrsvctl -c -a account==acme -t 6
# reservation acme.6 created
```

Standing Reservations

- **Periodic, repeating reservations**

```
# moab.cfg

SRCFG[special] USERLIST=fred,barney  OWNER=user:fred
SRCFG[special] MAXTIME=1:00:00
SRCFG[special] TASKCOUNT=16
SRCFG[special] STARTTIME=9:00:00
SRCFG[special] ENDTIME=17:00:00
SRCFG[special] DAYS=Mon,Tue,Wed,Thu,Fri
SRCFG[special] PERIOD=DAY DEPTH=2
```

Reservation Profile

- **Avoid manually and repetitively inputting standard reservation attributes**

```
#moab.cfg:
RSVPROFILE[special] USERLIST=fred
RSVPROFILE[special] HOSTLIST=node01,node02
```

```
mrsvctl -c -P special -d 1:00:00 -s +1:00:00
```

Host Expressions

- **Regular Expressions**

```
mrsvctl -c -h node[0-9][0-9] ...
```

- **Ranges**

```
mrsvctl -c -h R:node00-99 ...
```

- **Comma Delimited**

```
mrsvctl -c -h node00,node01,node02 ...
```

- **Features and Classes**

```
mrsvctl -c -h node[0-0] -f fast ...
```

Task-Based

- **Default definition of a task**
  - All the processors on a node
- **Define your own task**

```
mrsvctl -c -R procs:1
```

or

```
SRCFG[test]  RESOURCES=procs:1
```

- **Define the task count**

```
mrsvctl -c -R procs:1,mem:1000mb -t 20
```

or

```
SRCFG[test]  RESOURCES=procs:1,mem:1000mb
SRCFG[test]  TASKCOUNT=20
```

Flags and Exclusivity

- **Ignore State (`-F ignstate`)**
  - The default when using host expressions
- **Ignore Reservations (`-F ignres`)**
  - Forces reservation onto available resources even if it conflicts with other reservations
- **Ignore Job Reservations (`-F ignjobrsv`)**
  - Ignores existing job reservations
- **Ignore Idle Jobs (`-F ignidlejobs`)**
  - Reservation can be placed on top of idle job reservations

Flags and Exclusivity

- **Exclusive access**
  - `mrsvctl -c -E`
  - `SRCFG[test]  FLAGS=DEDICATEDACCESS`
- **Reservations are NOT exclusive by default**

Combining Tasks and Host Expressions

```
# mrsvctl -c -R procs:2,mem:1000mb -t 20 -h n0[1-9]
# mrsvctl -c -R procs:2,mem:1000mb -t 20 -h ALL -f fastio
# mrsvctl -c -R procs:2,mem:1000mb -t 20 -h class:batch
# mrsvctl -c -R gres=matlab:2 -a user==fred -d 40:00 -t 1
```

Adaptive COMPUTING

Reservation Timeframe

- **Reoccurring**
  - Occurs at specified times
    `SRCFG[test] PERIOD=DAY DAYS=Mon,Tue,Wed`
- **Infinite**
  - Never ends (until admin removes it)
- **One-Time**
  - One time use, for projects or special jobs
- **Rollback**
  - To guarantee SLA within specified time

Access Control List

- **Access Control List Flags**
- **Credential Lock**
- **Hard Policy Enabled**
- **Affinity**
- **Job Attributes**

Adaptive COMPUTING

ACL Flags

- **Required (ACL is normally OR'd)**
  - All required ACLs must be satisfied for requestor access to be granted:
    ```
    SRCFG[test] QOSLIST=*high MAXTIME=*2:00:00
    ```
- **Not**
  - If attribute is met, the requestor is denied access regardless of any other satisfied ACLs:
    ```
    mrsvctl -c -a user==!john
    ```

Credential Lock

- **Matching jobs will be required to run on the resources by this reservation**

```
SRCFG[test] CLASSLIST=&batch
mrsvctl -c -a user==&john -a class==batch
```

- **Lock is applied to all jobs in the queue upon creation of the reservation**
- **Lock is removed from all jobs when the reservation is removed**
- **All new jobs inherit the lock**

Hard Policy Enabled

- **ACLs marked with this modifier are ignored during soft policy scheduling and are only considered for hard policy scheduling once all eligible soft policy jobs start:**

```
SRCFG[johnspace] USERLIST=john
SRCFG[johnspace] CLASSLIST=~debug
```

- **All of John's jobs are allowed to run in the reservation at any time. Debug jobs are also allowed to run in this reservation but are only considered after all of John's jobs are given an opportunity to start. John's jobs are considered before debug jobs regardless of job priority**

Affinity

- **Positive, Neutral, Negative**
  - Jobs “gravitate” to positive
  - Jobs “retreat” from negative
- **Granted on a per access object basis**
- **Default is Positive:**

```
mrsvctl -c -a user=john- -a class==batch+
```

Job Attributes

- **Walltime:**

```
SRCFG[test] MAXWALLTIME=2:00:00
```

- **Processor Seconds:**

```
SRCFG[test] PSLIMIT<=86400
```

- **Job Features:**

```
SRCFG[test] JOBATTRLIST=PREEMPTEE

mrsvctl -c -a jattr=blue

qsub -l nodes=1,walltime=600 -W x=GATTR:blue
```

Reservation Commands

- **mrsvctl**
  - `mrsvctl -c` (create)
  - `mrsvctl -r` (release/remove)
  - `mrsvctl -q` (query)
- **showres**
  - `showres -n` (show reservations on each node)
  - `showres -f` (show nodes with no reservations)
- **mdiag**
  - `mdiag -r` (diagnose reservation problems)

mshow

- **It's all about the "What" and "When"**

- **`mshow -a` identifies the spaces**
- **Shows "Earliest Start Times"**
- **Can "Carve Out" the spaces for reservations**
- **Useful for "carving out" space for one-time use**
  - Defines the resources and the timeframe
  - Does NOT define who has access

Adaptive COMPUTING

mshow

- **Using `mshow -a`**

- **Always exclusive**
- **Automatically searches for the earliest spot**
- **Query for resources and commit:**

```
# mshow -a -i -o -x -- \
flags=tid,future -w \
minnodes=1,duration=600

# mrsvctl -c -R 4
```


Adaptive COMPUTING

Using "mshow -a"

- **Level 4 Admin and higher can run the command by default**

  ADMINCFG[5] SERVICES=mshow

- **Default shows resources that are available now and for how long**

```
$ mshow -a

Partition        Tasks    Nodes    Duration StartOffset       StartDate

---------        -----    -----    -------- -----------       ---------

ALL              74       5        INFINITY 00:00:00          09:57:23_05/05
```

- **Default "Task" is 1 processor**

Adaptive COMPUTING

Additional `mshow` Flags

- **Flags to tune behavior and results**
  - `mshow -a [--flags=<tid><,future>]`
- **"TID" Flag**
  - Transaction ID
  - Bundles up the results for later use
  - Result will only show what was asked for
- **"Future" Flag**
  - Shows results available now and in the future

The "where" Clause

- **Use it to define "what" you are looking for**
- **Default will return as many as possible**
  - Minnodes
  - Minprocs
  - Mintasks
- **Recommended to use "mintasks" in majority of use cases**
  - Define the taskcount and task shape

```
$ mshow -a -w mintasks=4@procs:2
Partition     Tasks  Nodes      Duration   StartOffset       StartDate
---------     -----  -----  ------------  ------------  --------------
ALL               4      4      INFINITY      00:00:00  09:57:23_05/05
```

“where” Clause Examples

```
$ mshow -a -w minnodes=1 --flags=tid
Partition     Tasks  Nodes      Duration   StartOffset       StartDate
---------     -----  -----  ------------  ------------  --------------
ALL               1      1      00:00:01      00:00:00  10:56:57_05/05  TID=7
```

```
$ mshow -a -w minprocs=4 --flags=tid
Partition     Tasks  Nodes      Duration   StartOffset       StartDate
---------     -----  -----  ------------  ------------  --------------
ALL               4      1      00:00:01      00:00:00  10:58:24_05/05  TID=10
```

```
$ mshow -a -w mintasks=4@procs:2+mem:2gb,duration=4:00:00 --flags=tid
Partition     Tasks  Nodes      Duration   StartOffset       StartDate
---------     -----  -----  ------------  ------------  --------------
ALL               4      1       4:00:00      00:00:00  11:02:31_05/05  TID=13
```

```
$ mshow -a -w hostlist=proxy,duration=6:00:00 --flags=tid
Partition     Tasks  Nodes      Duration   StartOffset       StartDate
---------     -----  -----  ------------  ------------  --------------
ALL              16      1       6:00:00      00:31:28  11:36:46_05/05  TID=16
```

Transactions

- **Allow you to bundle the results:**

```
$ mshow -a -w hostlist=proxy,duration=6:00:00 --flags=tid
Partition     Tasks  Nodes      Duration    StartOffset       StartDate
---------     -----  -----  ------------  ------------  --------------
ALL              16      1       6:00:00      00:31:28  11:36:46_05/05  TID=16
```

- **View transactions:**

```
$ mschedctl -l trans 16 --xml
<Data><trans DRes="PROCS=1" Duration="21600" Flags="TID" IsValid="TRUE" Name="16"
NodeList="proxy:16" StartTime="1304617006"></trans></Data>
```

- **Commit transactions:**

```
$ mrsvctl -c -R 16
NOTE:     reservation system.1 created
$ mdiag -r system.1
Diagnosing Reservations
RsvID                    Type Par   StartTime     EndTime     Duration Node Task Proc
-----                    ---- ---   ---------     -------     -------- ---- ---- ----
system.1                 User pbs    00:23:25     6:23:25      6:00:00    1   16   16
    Task Resources: PROCS: 1
    Attributes (HostExp='^proxy$')
```

Using Transactions

```
$ mshow -a --flags=tid,future -w minnodes=2,duration=2:00:00:00
Partition     Tasks  Nodes      Duration    StartOffset       StartDate
---------     -----  -----  ------------  ------------  --------------
ALL               2      2    2:00:00:00      00:00:00  11:16:21_05/05  TID=23

$ mschedctl -l trans 23 --xml
<Data><trans DRes="PROCS=[ALL]" Duration="172800" Flags="FUTURE,TID" IsValid="TRUE" Name="23"
NodeList="kahi:1,maka:1" StartTime="1304615781"></trans></Data>

$ mrsvctl -c -R 23 -a user=wightman
NOTE:     reservation wightman.2 created
WARNING:  specified starttime is in the past and will be changed to now

$ mdiag -r wightman.2
Diagnosing Reservations
RsvID                    Type Par   StartTime     EndTime     Duration Node Task Proc
-----                    ---- ---   ---------     -------     -------- ---- ---- ----
wightman.2               User pbs   -00:00:04  1:23:59:29   1:23:59:33    2    2   32
    ACL:   RSV==wightman.2= USER==wightman+
    Task Resources: PROCS: [ALL]
    Attributes (HostExp='^kahi$|^maka$')
```

Additional Notes on Transactions

- **Can be combined with any ACL**
  - Customize ACL at time of reservation creation time
- **Transaction expiration**
  - Creation of reservations voids transactions
  - Best practice:
    - Stop the scheduler
    - Run mshow -a followed by mrsvctl -c -R
    - Resume the scheduler
- **Transactions cannot be modified**
- **Placeholder reservations can be used**