控制作业

Slurm

常用控制命令

命令	功能
`squeue --me`（或 `squeue -u $USER`）	查看自己正在排队或运行的未结束作业
`scontrol show job JOBID`	查看某个未结束作业的详细配置（如分配节点、环境变量、资源限制等）
`sacct`	查看自己历史作业的简要信息
`sacct -j JOBID -l`	查看某个已结束历史作业的详细统计信息（包括内存峰值、CPU 使用效率、退出状态等）
`tail -f slurm-JOBID.out`	查看正在运行作业的标准输出（注：Slurm 没有直接对应的 `bpeek`，通常直接查看其定义的输出文件即可）
`scancel JOBID`	终止某个作业
`scancel -u $USER`	终止属于该用户的所有作业（排队与运行中的）
`scontrol top JOBID`	请求将处于排队状态的作业置于队列顶部（需系统配置允许，仅影响自己的作业优先级排序）
`scontrol hold JOBID`	挂起作业（暂停排队）
`scontrol release JOBID`	解除挂起（允许作业继续排队调度）

作业状态

`squeue` 或 `sacct` 命令输出中的状态标识（通常简写为两个字母）可能包括以下常见值：

状态（简写）	描述
PENDING (PD)	作业正在等待资源分配。即作业仍在排队中尚未开始。

状态 （简写 ）	描述
CONFIGURING (CF)	资源已被分配，但计算节点正在启动或配置环境（例如从节能模式唤醒）。
RUNNING (R)	作业当前正在计算节点上运行。
COMPLETING (CG)	作业正在结束，系统正在回收资源和清理进程。
COMPLETED (CD)	作业以状态 0 成功终止（所有进程均正常退出）。
FAILED (F)	作业以非零状态终止，或其他严重错误导致运行失败。
CANCELLED (CA)	作业被用户或管理员手动取消（如使用 `scancel` 命令）。
TIMEOUT (TO)	作业运行时间超过了申请的限制时间 (`-t`)，被调度器强制终止。
OUT_OF_MEMORY (OOM)	作业使用的内存超出了申请的上限，被系统内核 OOM Killer 杀掉。
SUSPENDED (S)	作业在运行期间被挂起。通常是因为高优先级作业抢占资源，或管理员手动操作。
NODE_FAIL (NF)	作业所在节点发生硬件故障或失联导致作业非正常终止。
UNKNOWN (UN)	调度系统无法获取作业状态。一般是因为计算节点长时间故障或与控制节点断开连接。此时如果登不上计算节点，可以直接 `scancel` 杀掉作业。

作业等待与依赖控制

在 LSF 中存在独立的 `bwait` 工具。在 Slurm 中，严禁在 **Shell** 脚本中使用 **`while`** 循环加 **`squeue`** 的方式高频轮询作业状态，这会对调度器数据库造成灾难性的压力。

请使用以下优雅的标准替代方案：

提交并阻塞等待 (`--wait`)

如果你希望在终端或外部脚本中提交作业后挂起，直到该作业执行完毕才继续往下走，请在提交时添加 `-W, --wait` 选项：

```
sbatch -W job.slurm
```

此命令会一直挂起，直到 `job.slurm` 执行完毕（无论成功或失败）才会返回终端提示符。

利用调度器依赖树 (`--dependency`)

如果你有一个作业 B 必须在作业 A 完成后才能运行，应在提交作业 B 时声明依赖关系交由系统管理：

```
# 假设提交作业 A 后返回的 JOBID 为 10001
sbatch --dependency=afterok:10001 job_B.slurm
```

作业 B 会被调度器以 `Dependency` 的原因置于 `PENDING` 状态。作业 A 成功结束后，B 才会开始参与资源调度分配。

脚本中等待已经运行的作业（伪作业阻塞法）

如果确实需要在一个独立的 Shell 脚本中阻塞等待一个已经提交并正在运行的作业（例如 JOBID `10001`）结束，可以向系统提交一个零操作的极轻量级伪作业，并配合 `--wait`：

```
sbatch -W --dependency=afterany:10001 --wrap="exit 0"
```

此时当前 Shell 脚本将暂停。一旦 `10001` 结束，这个包装作业会被立刻触发执行并瞬间结束，Shell 脚本随之恢复运行。这完美替代了 `bwait` 功能且对集群完全无害。

LSF

常用控制命令

命令	功能
`bjobs`	查看自己未结束的作业
`bjobs -l JOBID`	查看某个未结束作业的详情
`bhist`	查看自己已结束的历史作业
`bhist -l JOBID`	查看某个已结束历史作业的详情
`bpeek JOBID`	查看正在运行某个作业的 stdout/stderr
`bkill JOBID`	终止某个作业
`btop JOBID`	设置作业最先运行
`bbot JOBID`	设置作业最后运行

作业状态

`bjobs`命令的作业状态可能值包括:

状态	描述
PEND	作业正在等待中。也就是说，作业尚未开始。
PROV	作业已被派发到一个正在唤醒的节能状态主机。在作业可以发送到 sbatchd之前，它处于 PROV状态。
PSUSP	作业在等待期间被挂起，可能是作业所有者或 LSF管理员操作的。
RUN	作业当前正在运行。
USUSP	作业在运行期间被挂起，可能是作业所有者或 LSF管理员操作的。
SSUSP	作业被 LSF挂起。
DONE	作业以状态 0终止。
EXIT	作业以非零状态终止。
UNKWN	一般是两种情况之一，如果作业状态长时间处于 UNKWN 状态，一般来 说就是计算节点坏了可以直接杀掉作业。 ① 因为计算节点负载过高，未 能及时获取作业状态导致状态未知，这种情况一般只需要等待即可 ，待 负载下降获取状态后就正常了。 ② 因为计算节点出现故障且长时间未恢 复，调度系统无法获取作业状态，此时如果登录不到相应的计算节 点， 可以直接杀掉作业。
WAIT	对于提交到块作业队列的作业，块作业中的成员正在等待运行。
ZOMBI	①当 sbatchd在执行主机上不可达时，非可重新运行的作业被 `bkill`杀死， 并且作业显示为 UNKWN。 ② 运行可重新运行作业的主机不可用，并且 LSF 已将作业重新排队，分配了新的作业 ID，就像提交了新作业一样。 ③在执 行主机可用之后， LSF尝试杀死 ZOMBI作业。 ZOMBI 作业成功终止后，作业 的状态将更改为 EXIT。使用 MultiCluster 时，当在远程执行群集上运行的作 业变为 ZOMBI作业时，执行群集将像本地 ZOMBI 作业一样处理该作业。此外， 它还会通知提交群集作业处于 ZOMBI 状态，并且提交群集将重新排队作业。

作业等待

```
bwait -w "wait_condition" [-t timeout]
```

暂停并等待作业条件满足，不满足一直暂停等待，满足则执行完毕返回。

典型用法：在脚本中不要循环使用 bjobs判断作业状态，而用 bwait 等待作业运行完成，这样更优雅且能显著降低对集群的压力。

- `-w wait_condition`：要满足的等待条件，此表达式与上述 `bsub -w` 选项的格式相同。
- `-t timeout`：等待条件的超时，范围为 1-525600分钟，默认为一年。

Revision #24
Created 2021-05-09 22:29:10 CST by Yao Ge
Updated 2026-10-09 19:52:08 CST by Jiang Meng