SLURM调度系统的常见问题和解决方案

常见命令执行问题

- **sinfo** 报错 **Zero Bytes were transmitted or received**

```
[user4@login01 ~]$ sinfo
sinfo: error: slurm_receive_msg: Zero Bytes were transmitted or received
slurm_load_partitions: Zero Bytes were transmitted or received
```

通常是当前节点（服务器）与管理节点的系统时间不同步导致的，联系集群管理员处理。

作业提交报错

- **提交作业后，提示 Unable to allocate resources: No partition specified or system default partition**

```
[user4@login01 ~]$ srun -N 1 hostname
srun: error: Unable to allocate resources: No partition specified or system default partition
```

用户提交作业时通常需要加“**-p <分区名>**”这一参数，同时该参数应写在程序名前，并可用 *sinfo* 来查看所在分区。

- **提交作业后，提示 Unable to allocate resources: Invalid account or account/partition combination specified**

```
[user4@login01 ~]$ srun -p normal -n 1 date
srun: error: Unable to allocate resources: Invalid account or account/partition combination specified
```

用户在该分区没有权限提交作业，改为合适分区即可。

作业提交报错

- **sbatch 报错 `error: Invalid directive found in batch`**

```
[user4@login01 ~]$ sbatch job1.slurm
sbatch: error: Invalid directive found in batch script: defined
```

检查 *sbatch* 脚本，是否有参数书写错误。

- **提交作业后，提示 `CPU count per node can not be satisfied`**

```
[user4@login03 ~]$ srun -p normal -N 2 --ntasks-per-node=2 --cpus-per-task=64 date
srun: error: CPU count per node can not be satisfied
srun: error: Unable to allocate resources: Requested node configuration is not available
```

用户在提交作业时指定的资源分配不合理，作业需要的CPU核心数超过了系统配置的核心数。如图中作业需要的CPU核心数是2*2*64=256，但是实际上一个节点只有64核CPU。

作业运行的其他问题

- **作业程序运行缓慢、性能低**

调度系统只会监控作业的运行状态，原则上并不会参与作业程序的运行过程。所以作业运行出现报错或者性能慢等问题，请根据日志排查节点、存储、网络或作业输入与参数等。

- **作业程序无法并行执行**

作业任务是否能够并行执行并不是由调度系统来决定的，是由作业程序本身是否采用了并行计算设计并进行编程决定的。一个单进程任务即使分配了多个节点也无法实现并行计算。

- **作业暂停或中断后无法从断点继续计算**

通过调度系统暂停计算的作业任务是否能够由中断点继续计算同样取决于作业程序本身，如果作业程序本身不支持断点续算，则作业程序可能会重头开始计算，调度系统仅负责作为控制作业任务开始或暂停的开关。

作业不被运行

用户提交作业后，是否运行取决于**用户申请的资源情况和当前系统的情况**。建议使用 **squeue** 命令来查看所有已经提交和正在运行的作业。其中 **NODELIST(REASON)** 一栏包含非常有用的信息，在作业未运行时，它会显示未运行的原因；当作业在运行时，它会显示作业是在哪个节点运行的。

```
[root@login01 ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           4461120      high     date    user4 PD       0:00      1 (ReqNodeNotAvail, Unavaila
bleNodes:e2202r6n4)
           4460977      high     test    user4 PD       0:00      1 (Dependency)
           4460976      high     test    user4  R       0:17      1 e2413r2n3
           4461040 cpu_paral  real_yx    user3 PD       0:00     20 (Resources)
           4460959 cpu_paral    OMIP2    user5 PD       0:00     12 (Priority)
```

作业不被运行

- **红色加粗**部分需要修改脚本或者联系管理员处理
- **黑色加粗**部分是最常见的。

原因代码	详细说明
BeginTime	未到用户所指定的任务开始时间
Dependency	该作业所依赖的作业尚未完成
InvalidAccount	**用户的 SLURM 账号无效**
InvalidQOS	**用户指定的 QoS 无效**
ParitionTimeLimit	**用户申请的时间超过该分区时间上限**
QOSMaxCpuPerUserLimit	超过当前 QoS 用户最大 CPU 限制
QOSMaxGRESPerUser	超过当前 QoS 用户最大 GRES(GPU) 限制
Priority	**存在一个或多个更高优先级的任务，该任务需要等待**
ReqNodeNotAvail	**所申请的部分节点不可用**
Resources	**暂无闲置资源，该任务需等待其他任务完成**

作业不被运行

- 对于**Resources**和**Priority**这两个最常见的状况，用户可以从以下两个方面考虑来改善所提交作业的运行机会。
  - 尽可能申请**较少的计算资源**，如节点数、CPU数等；
  - 尽可能指定**较短的运行时间**（***-t*** 参数），便于提高作业优先级。
- 对于**ReqNodeNotAvail**这种情况，需要用户重新提交作业，否则只能等待节点重新恢复上线才会被调度执行。

交互式作业

- **采用 *srun* 提交作业，关闭界面后，再次登录时发现作业被killed**

srun 是交互式提交作业模式，一旦作业提交的界面关闭作业就会被 **killed**。若需要较长时间运行的作业，建议用户采用 *sbatch* 批处理提交方式。*sbatch* 负责资源分配，获取资源后会在获取资源的**第一个节点**运行提交的脚本，当前登录**shell** 断开后，加载作业仍可正常运行。

- ***salloc* 分配资源，退出 *salloc* 后发现作业断掉**

salloc 与 ***sbatch*** 最主要的区别是，*salloc* 命令资源请求被满足时，直接在提交作业的节点执行相应任务，适合需要指定运行节点和其他资源限制，并有特定命令的作业。当前 *shell* 断开后，申请获得的资源以及加载作业任务会退出。

谢 谢