SLURM调度系统的进阶使用

目录

目录

使用场景

系统环境：

- 登陆节点**大量资源被占用、CPU负载高**
- 有空闲计算节点

使用场景：

- **作业程序源码编译**
- **作业程序测试运行**
- **作业程序BUG调试**
- **数据处理（高负载）**
- **大文件拷贝**

此时以上场景在登陆节点可能执行**异常缓慢**甚至**无法执行。**
可通过salloc命令申请空闲的计算节点来执行高负载任务。

场景实战

```
[user4@login01 ~]# uptime
 15:48:40 up 74 days,  4:11, 80 users,  load average: 92.99, 93.37, 93.78
[user4@login01 ~]$ sinfo -t idle
PARTITION     AVAIL  TIMELIMIT  NODES  STATE NODELIST
normal           up   infinite     62  drain a3103n[01,03],a3104n[06,11],a3106n04,a3107n18,a3111n02,a3213n12,a3310n[15-16],
411n[08,15-19],a3412n[01-16],b2103r5n1,b2103r6n3,b2104r5n4,b3201r7n7,b3202r1n5,b3203r1n8,b3203r3n2,b3204r7n6,b3207r6n1,b320
4r5n4,b3306r3n6,b3308r3n6,b3308r8n5
normal           up   infinite     46   idle a3105n14,a3108n[13-16],a3314n13,b2103r7n[2-3],b3208r7n[1-8],b3208r8n[1-6],b330
[user4@login01 ~]$ salloc -p normal -N 1
salloc: Pending job allocation 4451433
salloc: job 4451433 queued and waiting for resources
salloc: job 4451433 has been allocated resources
salloc: Granted job allocation 4451433
salloc: Waiting for resource configuration
salloc: Nodes a3411n05 are ready for job
[user4@login01 ~]$ ssh a3411n05
[user4@a3411n05 ~]$
[user4@a3411n05 ~]$ tar -xhf blis-2.2.tar.gz
[user4@a3411n05 ~]$ cd blis-2.2/
[user4@a3411n05 blis-2.2]# ./configure --prefix=/opt/hpc/software/mathlib/blas/amd/blis-2.2 -t openmp --enable-cblas zen2
.
.
.
[user4@a3411n05 blis-2.2]# make -j
.
.
.
[user4@a3411n05 blis-2.2]# make install
.
.
.
[user4@a3411n05 ~]$ exit
logout
Connection to a3411n05 closed.
[user4@login01 ~]$ exit
exit
salloc: Relinquishing job allocation 4451447
salloc: Job allocation 4451447 has been revoked.
```

登陆节点login01负载高

使用salloc申请一个计算节点a3411n05

在当前终端登陆计算节点a3411n05

执行程序编译安装

退出计算节点

退出当前终端，释放申请的计算资源

目录

使用场景

- 很多时候我们需要运行一组作业，这些作业所需的资源和内容非常相似，只是一些参数不相同。这个时候借助作业数组（***Job Array***）就可以很方便地批量提交这些作业。
- **作业数组提供了一种快速、轻松地提交和管理类似作业集合的机制**; 具有成百上千个任务的作业数组可以在毫秒内提交(受配置的大小限制的限制)。
- 作业数组中的所有作业必须具有**相同的初始选项**(如大小、时间限制等) 。
- 作业数组中的**每一个作业在调度时都被视为独立的作业**，仍然受到队列以及服务器的资源限制。
- 作业数组**只支持批处理作业**，不支持交互式作业。

Job Array

在 SLURM 脚本中使用 `#SBATCH -a <range>` 即可指定 Job Array 的数字范围，即**作业编号（任务ID）**，其中的 ***range*** 参数需要是连续或不连续的整数。下面几种指定方式都是合法的。

- `#SBATCH -a 0-9` 作业编号范围是 0 到 9，**均含边界**。
- `#SBATCH -a 0,2,4` 作业编号是 0,2,4。
- `#SBATCH -a 0-9,20,40` 混合指定也是可以的。
- `#SBATCH -a 0-5:2` 同上，作业编号为 0,2,4，这个写法的意义相当于 MATLAB 中 的 0:2:5，即间隔变成 2。

Job Array

在脚本运行中，SLURM 使用环境变量来表示作业数组，具体为

- **SLURM_ARRAY_JOB_ID** 作业数组中**第一个作业的 ID**。
- **SLURM_ARRAY_TASK_ID** 该作业在数组中的**索引**。
- **SLURM_ARRAY_TASK_COUNT** 作业数组中**作业总数**。
- **SLURM_ARRAY_TASK_MAX** 作业数组中**最后一个作业**的索引。
- **SLURM_ARRAY_TASK_MIN** 作业数组中**第一个作业**的做引。

可用以上变量来区分不同组内的任务，以便于处理不同的输入参数。
对于数组内的每个作业，它的默认输出文件的命名方式为：
slurm-<作业JOBID>_<任务TASKID>.out。

作业数组示例

下面是一个很小的 SLURM 脚本例子，它使用 Job Array 来返回一些预设数组中的不同元素。 在实际应用中，这些不同的字符串或许就是程序所需输入的文件名。当然**你也可以使用一个脚本来包装你的程序，然后在这个脚本中获取这个环境变量。**

```
#!/bin/bash
#SBATCH -J array
#SBATCH -p normal
#SBATCH -N 1
#SBATCH --cpus-per-task=1
#SBATCH -a 0-2

input=(foo bar baz)
echo "This is job #${SLURM_ARRAY_JOB_ID}, with parameter ${input[$SLURM_ARRAY_TASK_ID]}"
echo "There are ${SLURM_ARRAY_TASK_COUNT} task(s) in the array."
echo "  Max index is ${SLURM_ARRAY_TASK_MAX}"
echo "  Min index is ${SLURM_ARRAY_TASK_MIN}"
sleep 5
```

作业数组示例

提交以上脚本并使用 **squeue** 命令查看可以看到下面的结果：

```
[user4@admin ~]$ sbatch array.slurm
Submitted batch job 4451708
[user4@admin ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
         4451708_0    normal    array    user4  R       0:01      1 a3310n13
         4451708_1    normal    array    user4  R       0:01      1 a3310n13
         4451708_2    normal    array    user4  R       0:01      1 a3310n13
```

作业 ID 是 **4451708**，而作业数组中的每个作业的JOBID命名方式为 **JOBID_TASKID**，非常 容易分辨。以下是其中某个作业的输出：

```
[user4@admin ~]$ cat slurm-4451708_1.out
This is job #4451708, with parameter bar
There are 3 task(s) in the array.
  Max index is 2
  Min index is 0
```

作业数组示例

取消作业数组的部分或全部任务可以使用 **scancel** 命令。

- **scancel 4451708_[1-2]:** 取消 4451708 号作业数组 TASK_ID 为 1 和 2 的作业。
- **scancel 4451708_0 4451708_2:** 取消 4451708 号作业数组 TASK_ID 为 0 和 2 的作业。
- **scancel 4451708:** 取消 4451708 号作业数组的全部作业。

目录

使用场景

申请独占资源（一个节点只能运行一个作业）时，如果申请的节点资源还有相当部分的空闲，此时就会造成计算资源浪费。为了避免这种不必要的浪费，可以**一次提交作业，（同时）计算多个任务**（包括串行、mpi等任务）。

最简单的方法就是通过**SLURM**的作业步来实现。

SLURM支持**作业**（job）和**作业步**（step）两层的资源调度。作业步通过**srun**启动。可以在同一个作业脚本中**并行或串行执行多个作业步**。

作业步运行有如下的限制条件：

- 任何作业步申请的资源，不能超过作业申请的总资源数；
- 作业步只能使用其它作业步未使用的剩余作业资源；
- 并发运行的作业步的资源之和不能超过作业申请的总资源数。

多任务示例

下面是一个很小的 SLURM 脚本例子，它使用srun提交简单的echo命令来表示一个任务，并通过输出重定向获取任务输出日志信息。最后通过wait命令等待两个后台的srun命令结束。

```
#!/bin/bash
#SBATCH -J multi-tasks
#SBATCH -p high
#SBATCH -N 1
#SBATCH -n 2

srun -N 1 -n 1 echo $SLURM_JOB_ID > slurm-${SLURM_JOB_ID}_1.out &
srun -N 1 -n 1 echo $SLURM_JOB_ID > slurm-${SLURM_JOB_ID}_2.out &
wait
```

注意这里的资源分配，脚本总共申请两个进程资源，每个srun申请一个进程资源，这样每个srun提交的作业才不会造成资源竞争

提交以上脚本并使用 **squeue** 命令查看可以看到下面的结果：

```
[user4@login02 ~]$ sbatch multi-tasks.slurm
Submitted batch job 4451906
[user4@login02 ~]# squeue
Sat Feb 11 17:54:32 2023
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           4451906      high multi-ta     user4  R       0:01      1 e2309r6n1
```

多任务示例

作业 ID 是 **4451906**，以下是各个作业步的输出：

```
[user4@login02 ~]$ cat slurm-4451906_1.out
4451906
[user4@login02 ~]$ cat slurm-4451906_2.out
4451906
[user4@login02 ~]$ cat slurm-4451906.out
```

作业结束后可以通过 sacct 命令看到各个作业步信息，其实在作业运行节点使用 **squeue -s** 也可看到作业步信息

```
[user4@login02 ~]$ sacct -j 4451906
       JobID    JobName  Partition    Account  AllocCPUS      State ExitCode
------------ ---------- ---------- ---------- ---------- ---------- --------
4451906      multi-tas+       high      user4          1  COMPLETED      0:0
4451906.bat+      batch                 user4          1  COMPLETED      0:0
4451906.ext+     extern                 user4          1  COMPLETED      0:0
4451906.0          echo                 user4          1  COMPLETED      0:0
4451906.1          echo                 user4          1  COMPLETED      0:0
```

目录

使用场景

- 有时候作业1与作业2有点关系，我们希望等作业1结束后再运行作业2
- 如果将作业1和作业2都提交了，而且它们之间没有关系，会在获得计算资源后都运行，这不是我们想要的
- 因此，我们可以正常提交作业1，然后在提交作业2的时候，使用sbatch的 **-d** 参数设置依赖关系，让作业2等待作业1结束后再运行，即便有空闲的计算资源也等着
- 关联作业也被叫做任务依赖

参数说明

-d 参数表示**推迟此作业的启动，直到满足指定的依赖项完成，**其用法如下：

```
-d <依赖条件>
```

如果有多个依赖条件，使用分隔符隔开即可。
使用分隔符是**逗号**，表示**所有条件**都满足。如果分隔符是**问号**，表示**任意条件**满足。
依赖条件的语法是：

```
<条件类型>:<作业ID>
```

如果有多个作业ID，用**冒号**分割。条件类型表示**当前作业只有达到等待条件时才会开始执行**，有以下4种常用选项：

选项	说明
after	等待指定的作业开始或者取消，当前作业才会开始运行
afterok	等待指定的作业成功运行（运行结束，退出码是0）
afterany	等待指定的作业终止（结束），此为默认选项，**可以省略**
afternotok	等待指定的作业非正常结束（非0退出码、节点异常，超时等）

关联作业示例 - after

```
[user4@login03 ~]$ sbatch job1.slurm
Submitted batch job 4474554
[user4@login03 ~]$ sbatch -d after:4474554 job2.slurm
Submitted batch job 4474555
[user4@login03 ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           4474555      high     test    user4  R       0:01      1 e2211r2n1
           4474554      high     test    user4  R       0:11      1 e2204r6n1
[user4@login03 ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
[user4@login03 ~]$ sacct  -j 4474554,4474555 -X -o jobid,jobname,partition,account,start,end
       JobID    JobName  Partition    Account               Start                 End
------------ ---------- ---------- ---------- ------------------- -------------------
4474554            test       high      user4 2023-02-15T20:13:33 2023-02-15T20:13:53
4474555            test       high      user4 2023-02-15T20:13:43 2023-02-15T20:13:54
```

关联作业示例 - afterok

```
[user4@login03 ~]$ sbatch job1.slurm
Submitted batch job 4474621
[user4@login03 ~]$ sbatch -d afterok:4474621 job2.slurm
Submitted batch job 4474622
[user4@login03 ~]$ sbatch job3.slurm
Submitted batch job 4474623
[user4@login03 ~]$ sbatch -d afterok:4474623 job2.slurm
Submitted batch job 4474624
[user4@login03 ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           4474624      high     test    user4 PD       0:00      1 (Dependency)
           4474622      high     test    user4 PD       0:00      1 (Dependency)
           4474623      high     test    user4  R       0:04      1 e2204r6n1
           4474621      high     test    user4  R       0:13      1 e2204r6n1
[user4@login01 ~]$ sacct -j 4474621,4474622,4474623,4474624 -X -o jobid,jobname,partition,account,start,end,exitcode,state
       JobID    JobName  Partition    Account               Start                 End ExitCode      State
------------ ---------- ---------- ---------- ------------------- ------------------- -------- ----------
4474621            test       high      user4 2023-02-15T20:30:43 2023-02-15T20:31:13      0:0  COMPLETED
4474622            test       high      user4 2023-02-15T20:31:15 2023-02-15T20:31:45      0:0  COMPLETED
4474623            test       high      user4 2023-02-15T20:30:52 2023-02-15T20:31:27      1:0     FAILED
4474624            test       high      user4 2023-02-15T20:31:48 2023-02-15T20:31:48      0:0  CANCELLED
```

关联作业示例 - afterany

```
[user4@login01 ~]$ sbatch job1.slurm
Submitted batch job 4460976
[user4@login01 ~]$ sbatch -d 4460976 job2.slurm
Submitted batch job 4460977
[root@login01 ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           4460977      high     test    user4 PD       0:00      1 (Dependency)
           4460976      high     test    user4  R       0:17      1 e2413r2n3

[user4@login01 ~]$ sacct -j 4460976,4460977 -X -o jobid,jobname,partition,account,start,end
       JobID    JobName  Partition    Account               Start                 End
------------ ---------- ---------- ---------- ------------------- -------------------
4460976            test       high      user4 2023-02-13T09:20:45 2023-02-13T09:21:36
4460977            test       high      user4 2023-02-13T09:21:38 2023-02-13T09:21:48
```

关联作业示例 – afternotok

```
[user4@login03 ~]$ sbatch job1.slurm
Submitted batch job 4474638
[user4@login03 ~]$ sbatch -d afternotok:4474638 job2.slurm
Submitted batch job 4474640
[user4@login03 ~]$ sbatch job3.slurm
Submitted batch job 4474641
[user4@login03 ~]$ sbatch -d afternotok:4474641 job2.slurm
Submitted batch job 4474642
[user4@login03 ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           4474642      high     test    user4 PD       0:00      1 (Dependency)
           4474640      high     test    user4 PD       0:00      1 (Dependency)
           4474641      high     test    user4  R       0:06      1 e2204r6n1
           4474638      high     test    user4  R       0:19      1 e2204r6n1
[user4@login01 ~]$ sacct -j 4474642,4474640,4474641,4474638 -X -o jobid,jobname,partition,account,start,end,exitcode,state
       JobID    JobName  Partition    Account               Start                 End ExitCode      State
------------ ---------- ---------- ---------- ------------------- ------------------- -------- ----------
4474638            test       high      user4 2023-02-15T20:36:24 2023-02-15T20:36:55      0:0  COMPLETED
4474640            test       high      user4 2023-02-15T20:36:56 2023-02-15T20:36:56      0:0  CANCELLED
4474641            test       high      user4 2023-02-15T20:36:37 2023-02-15T20:37:13      1:0     FAILED
4474642            test       high      user4 2023-02-15T20:37:15 2023-02-15T20:37:45      0:0  COMPLETED
```

谢 谢