

SLURM介绍与作业提交

目录

01 整体概况

02 用户登录及提交作业

03 软件及环境设置

04 作业管理

05 作业脚本示例

整体概况-集群

集群是利用通信网络将一组计算机（节点）按某种结构连接起来，在并行化设计及可视化人机交互集成开发环境支持下，统一调度、协调处理，实现高效并行处理的系统。所有计算机节点一起工作如同一个单一集成的系统资源，实现单一系统映像。

- 包含计算、存储、网络等各种资源实体且彼此联系的资源集合；
- 在物理上，一般由计算处理、互联回话、I/O 存储、操作系统、编译器、运行环境、开发工具等多个软硬件子系统组成；
- 节点是集群的基本组成单位，从角色上一般可以划分为管理节点、登录节点、计算节点、存储节点等。

整体概况-集群

adios
blas
boost
esfm

ferret
fftw
geos
GotoBlas2

grads
grib_api
gsl
hdf

hdf5
hdfeos
hdfeos5
hypre

ioapi
jasper
lapack
libpng16

ncl_ncarg
nco
ncview
netcdf

nlopt
openblas
parallel-netcdf
petsc

plapack
plasma
proj
scalapack

udunits
wgrib
wgrib2



整体概况-应用领域



整体概况-集群节点规划

角色类型	节点范围	软件路径	运行服务
登录节点	login01~login10	1. 调度软件安装目录/opt/gridview，主要包括munge、slurm两个软件。	munge.service
管理节点	admin01 (主用) admin02 (备用)	1. 调度软件安装目录/opt/gridview，包括munge、slurm。 2. 数据库软件目录/opt/mysql-5.7.25-linux-glibc2.12-x86_64	munge.service slurmdbd.service slurmctld.service
计算节点	node1~node666	1. 调度软件安装目录/opt/gridview，包括munge、slurm	munge.service slurmd.service
MySQL节点	admin01 (主用) admin02 (备用)	1. 数据库软件/opt/mysql-5.7.25-linux-glibc2.12-x86_64	my_mysqld

目录

01 整体概况

02 用户登录及提交作业

03 软件及环境设置

04 作业管理

05 作业脚本示例

用户登录及提交作业

通过国产计算云登录

<https://ac.sugon.com/>

The screenshot shows a web browser window for the URL <https://ac.sugon.com/sso/login?service=https%3A%2F%2Fac.sugon.com%2Fac%2Fapi%2Fauth%2FloginSsoRedirect.action>. The page title is "计算云统一认证中心" (Computing Cloud Platform Unified Authentication Center). The top navigation bar includes links for "应用" (Application), "曙光" (Sugon), "novnc", "ldap", "使用图形界面" (Use Graphics Interface), "slurm", "IBM Spectrum LS...", "AC-测试" (AC-Test), and "天机". The main content area features a large graphic of a person standing on a laptop screen, which is itself inside a cloud, symbolizing cloud computing. To the right is a "密码登录" (Password Login) form with fields for "用户名" (Username) and "密码" (Password), a "登录" (Login) button, and links for "手机验证码登录" (Mobile Verification Code Login), "立即注册" (Register Now), and "找回密码" (Forgot Password). The bottom of the page includes a copyright notice: "版权所有©2018-2021 曙光信息产业股份有限公司 津ICP备10200542号".

用户登录及提交作业

查看用户信息

及可用资源情况

The screenshot shows the曙光计算云 Compute Center - Kunshan dashboard. At the top, it displays account information for 'slurmtest' (余额: -5284.96元, 状态: 正常), available resources (CPU total cores: 54944, DCU total cards: 5436), and job status (no data). Below this, there are sections for services (AI深度学习服务, HPC高性能计算服务, 大数据服务 (公测), 天机气象预报系统), accessible queues (iap_cas, shanshan_test, ssct, ssct_c), storage resources (/public, /public/home/slurmtest), and machine statistics (bar chart showing usage over time).

曙光计算云 计算中心-昆山 概览 作业 E-Shell 文件传输 资源 账户 费用 工单 帮助 | slurmtest |

账户名
slurmtest
余额 -5284.96元 状态 欠费
计算用户 7 slurmtest 正常
所属组 slurmtest
主目录 /public/home/slurmtest

可用资源

CPU总核数 54944
DCU总卡数 5436

作业状态

无数据

我的服务

AI深度学习服务 HPC高性能计算服务 大数据服务 (公测) 天机气象预报系统

联系 我们

可访问队列

iap_cas 节点数(占用 576 | 其它 59 | 空闲 69)
shanshan_test 节点数(占用 33 | 其它 7 | 空闲 0)
ssct 节点数(占用 469 | 其它 22 | 空闲 82)
ssct_c 节点数(占用 309 | 其它 4 | 空闲 45)

数据更新时间： 2021-05-22 14:18:21 占用 7 其它 7 空闲 7

存储资源

/public 存储配额为700GB
/public/home/slurmtest 已使用 153.02 GB / 700 GB(21.86%)

机时统计

核*时

4,000
3,000
2,000
1,000
0

2021-01 2021-02 2021-03 2021-04 2021-05 2021-06 2021-07 2021-08 2021-09 2021-10 2021-11 2021-12

联系 我们

用户登录及提交作业

文件传输E-File

计算中心: 计算中心-昆山 用户: slurmtest

新建 ▾ 上传 ▾ 快传 ▾ 下载 删除 复制 移动

查看上传列表 刷新 搜索...

主目录 | 返回上一级 | 根目录 > public > home > slurmtest

<input type="checkbox"/>	文件名	文件类型	大小	修改时间	操作
<input type="checkbox"/>	00-CASE-0329	-		2021-05-10 16:51:38	↓ 下载
<input type="checkbox"/>	00-TEST-STAR	-		2021-04-02 15:31:37	↓ 下载
<input type="checkbox"/>	0203dnew	-		2021-02-03 13:39:05	↓ 下载
<input type="checkbox"/>	0206a	-		2021-02-06 17:41:05	↓ 下载
<input type="checkbox"/>	0309a	-		2021-04-29 23:17:11	↓ 下载
<input type="checkbox"/>	0320a	-		2021-04-14 22:02:13	↓ 下载
<input type="checkbox"/>	0324a	-		2021-03-24 20:16:53	↓ 下载
<input type="checkbox"/>	0329_zg	-		2021-04-29 10:54:25	↓ 下载
<input type="checkbox"/>	0329_zg_me	-		2021-03-29 17:33:13	↓ 下载
<input type="checkbox"/>	0412a	-		2021-04-12 21:40:28	↓ 下载
<input type="checkbox"/>	2017	-		2021-04-04 23:07:31	↓ 下载
<input type="checkbox"/>	20210304	-		2021-03-04 10:16:30	↓ 下载

1 /2 页(共 156 条) ▶ ⟲ ⟳ 🔍

用户登录及提交作业

作业模板：\$HOME/slurm_template

计算中心：计算中心-昆山 计算用户：slurmtest

```
[slurmtest@login03 ~]$ pwd
/public/home/slurmtest
[slurmtest@login03 ~]$ cd slurm_template/
[slurmtest@login03 slurm_template]$ ls
autodyn.slurm  fire_openmpi_slurm  List.txt      sleep.slurm      slurmFdtd.slurm    slurmLammps.slurm   su2.slurm     test.openmpi
cfx-b.slurm    fireV2.c           logs          slurmAnsys.slurm  slurmFluent.slurm  slurmStarccm.slurm  test.mpi      zhangtao.mpirun
fire.c         FLUENT.slurm       LS-DYNA.slurm  slurmCfx.slurm   slurmGromacs.slurm slurmVasp.slurm   testmpi.slurm  zhangtao.srun
[slurmtest@login03 slurm_template]$
```

用户登录及提交作业

E-Shell提交作业

计算中心：计算中心-昆山 计算用户：slurmtest

```
[slurmtest@login03 ~]$ pwd
/public/home/slurmtest
[slurmtest@login03 ~]$ cd slurm_template/
[slurmtest@login03 slurm_template]$ ls
autodyn.slurm  fire_openmpi_slurm  List.txt      sleep.slurm      slurmFDTD.slurm   slurmLammps.slurm  SU2.slurm    test.openmpi
cfx-b.slurm    fireV2.c           logs          slurmAnsys.slurm  slurmFluent.slurm  slurmStarccm.slurm  test.mpi     zhangtao.mpirun
fire.c         FLUENT.slurm       LS-DYNA.slurm  slurmCfx.slurm   slurmGromacs.slurm  slurmVasp.slurm   testmpi.slurm  zhangtao.srun
[slurmtest@login03 slurm_template]$ vim test.mpi
[slurmtest@login03 slurm_template]$ sbatch test.mpi
Submitted batch job 8013571
[slurmtest@login03 slurm_template]$ squeue
      JOBID PARTITION      NAME      USER ST      TIME  NODES NODELIST(REASON)
      8013571    sugon      FIRE slurmtes R      0:03      1 j09r3n04
[slurmtest@login03 slurm_template]$
```

用户登录及提交作业

作业查看 (当前作业, 历史作业, 作业详情)

Sugon Compute Cloud 计算中心-昆山 概览 作业 E-Shell 文件传输 资源 账户 费用 帮助 | slurmtest

当前作业 历史作业

重新运行 取消 全部状态 所有队列 请输入作业名 请输入作业ID 搜索 重置 自动刷新

作业ID	作业名	应用名	队列名	状态	开始时间	已运行时长	操作
8013571	FIRE	WRF	sugon	运行	2021-05-22 14:38:40	1分钟28秒	

当前作业 历史作业

按结束时间 2021-05-01 00:00:00 - 2021-05-22 14:46:45 全部 请输入作业ID 请输入作业名 搜索 重置 高级

作业ID	作业名	应用名	队列名	状态(退出码)	节点数	入队时间	结束时间	运行时长	操作
8013571	FIRE	WRF	sugon	正常(0)	1	2021-05-22 14:38:37	2021-05-22 14:40:23	00:01:43	
7955188	FLUENT_0519_1746...	FLUENT	sugon	正常(0)	3	2021-05-19 17:43:40	2021-05-19 18:09:30	00:25:45	
7954777	sleep	/usr/bin/sleep	sugon	正常(0)	1	2021-05-19 16:57:37	2021-05-19 17:06:19	00:08:30	
7944834	FLUENT_0519_1145...	FLUENT	sugon	取消(10004)	3	2021-05-19 11:42:14	2021-05-19 13:38:42	01:56:11	
7926999	FLUENT_0518_1918...	FLUENT	sugon	取消(10004)	4	2021-05-18 19:15:26	2021-05-18 19:46:25	00:30:40	
7926201	STDIN_0518_170521	BASE	sugon	正常(0)	1	2021-05-18 17:05:25	2021-05-18 17:14:12	00:08:31	
7923236	MECHANICAL_0518...	MECHANICAL	sugon	取消(10004)	2	2021-05-18 13:59:15	2021-05-18 14:01:03	00:01:38	
7900639	STDIN_0517_163148	BASE	sugon	正常(0)	1	2021-05-17 16:31:52	2021-05-17 16:40:25	00:08:33	
7890650	sleep	/usr/bin/sleep	sugon	正常(0)	1	2021-05-17 09:56:59	2021-05-17 09:58:56	00:01:49	
7890406	STDIN_0517_085127	BASE	sugon	正常(0)	1	2021-05-17 08:51:26	2021-05-17 09:00:48	00:09:06	
7890405	STDIN_0517_085102	BASE	sugon	正常(0)	1	2021-05-17 08:51:04	2021-05-17 08:59:54	00:08:32	
7809950	FLUENT.slurm	FLUENT	lap_cas	正常(0)	16	2021-05-13 15:20:47	2021-05-13 15:29:48	00:08:48	

作业详情

作业ID: 8013571 应用: WRF 队列名: sugon 申请CPU数量(核): 32 占用CPU数量(核): 32
作业名: FIRE 提交时间: 2021-05-22 14:38:37 申请GPU数量(卡): 0 占用GPU数量(卡): 0
所有者: slurmtest 开始时间: 2021-05-22 14:38:40 申请内存(M): 91200
作业状态: 完成 运行时间: 2天
重新运行: 该作业已重新运行 0 次

申请节点数量(个): 1 占用节点数量(个): 1
执行主机: j093n04
作业脚本: /public/home/slurmtest/slurm_template/test_mpi
工作路径: /public/home/slurmtest/slurm_template
标准输出: /public/home/slurmtest/slurm_template/logs/8013571.joop
错误输出: /public/home/slurmtest/slurm_template/logs/8013571.joop

文件列表 [/public/home/slurmtest/slurm_template]
文件名: list.txt 大小: 29.0 KB 修改时间: 2018-06-01 12:57:14 权限: rwxr-xr-x

目录

01 整体概况

02 用户登录及提交作业

03 软件及环境设置

04 作业管理

05 作业脚本示例

软件存储目录

存储目录及用途

目录	挂载节点	备注
/public	管理节点、 登录节点、 计算节点	用户家目录为/public/home目录。 应用软件目录为/public/software目录。

可用软件

计算系统安装了多种编译环境及应用，主要包括编译器、调试器、mpi并行开发环境及数学库等四部分。

软件安装目录为/public/software。

为方便使用，集群软件环境通过modules工具管理环境变量。

```
[slurmtest@glogin software]$ ls
apps  benchmark  compiler  mathlib  modules  mpi  profile.d
[slurmtest@glogin software]$
```

环境变量设置

“Environment **module**” (环境模块)是一组环境变量设置的集合。module可以被加载(load)、卸载(unload)、切换(switch)，这些操作会改变相应的环境变量设置，从而让用户方便地在不同环境间切换。

```
[root@glogin init]# rpm -qa|grep environment
environment-modules-3.2.10-10.el7.x86_64
[root@glogin init]# cat /usr/share/Modules/init/.modulespath
/public/software/modules
[root@glogin init]#
```

- ◆ 查看命令帮助-H
- ◆ 卸载模块unload
- ◆ 查看可用模块avail
- ◆ 切换模块switch
- ◆ 查看已加载模块list
- ◆ 卸载全部模块purge
- ◆ 加载模块load
- ◆ 显示模块内容show

环境变量设置

■ 查看可用模块avail

```
[slurmtest@login01 ~]$ module avail

----- /public/software/modules -----
apps/abinit/8.10.3/hpcx-2.4.1-intel2017
apps/amber/2018/hpcx-2.4.1-gcc-7.3.1
apps/anaconda2/4.3.0
apps/anaconda3/4.9.2
apps/anaconda3/5.2.0
apps/AutoDock-DCU/4.2.6/gcc-7.3.1
apps/automake/1.16.1
apps/barracuda-DCU/0.6.2_beta/hpcx-2.4.1-gcc-7.3.1
apps/bedtools/2.29.1/gcc-7.3.1
apps/bigDFT/1.8.3/hpcx-2.4.1-intel2017
apps/blast/2.10.0
apps/blastn-DCU/2.2.28/gcc-7.3.1
apps/blastp-DCU/2.2.28/gcc-7.3.1
apps/boost/intel/1.67.0
apps/bwa/0.7.17/gcc-7.3.1
apps/Calculix/2.16/hpcx-2.4.1-gcc-7.3.1
apps/canu/1.8/gcc-7.3.1
apps/CMAQ/5.3/hpcx-2.4.1-intel2017
apps/Code_Aster/13.8/hpcx-2.4.1-gcc-7.3.1
apps/cp2k/6.1.0/hpcx-2.4.1-intel2017
apps/DUNE/2.8/hpcx-2.4.1-gcc-7.3.1
apps/Elmer/8.4/hpcx-2.4.1-intel2017
apps/FastQC/0.11.9
apps/FlameGraph/master

apps/SPECFEM3D/rocm2.9/hpcx-v2.4.1-gcc-7.3.1
apps/SU2/7.0.1/hpcx-2.4.1-gcc-7.3.1
apps/TensorFlow/1.14.0/hpcx-2.4.1-gcc-7.3.1
apps/TensorFlow/1.14.0-hierarchical/hpcx-2.4.1-gcc-7.3.1
apps/TensorFlow/2.2.0/hpcx-2.4.1-gcc-7.3.1
apps/Trimmomatic/0.3.8
apps/vasp/5.2.2/intel2017
apps/vasp/5.4.4/hpcx-2.4.1-intel2017
apps/vaspkit/1.2.4/vaspkit1.2.4
apps/vtst/5.4.4/hpcx-2.4.1-intel2017
apps/WRF/3.8/hpcx-2.4.1-intel2017
apps/WRF/4.2/hpcx-2.4.1-intel2017
compiler/cmake/3.15.6
compiler/cuda/10.2
compiler/rocm/3.3
compiler/rocm/3.5
compiler/rocm/3.9
compiler/rocm/3.9.1
compiler/rocm/4.1.0
compiler/rocm/socprof
mathlib/antlr/2.7.7/intel
mathlib/cdo/1.7.2/intel
mathlib/fftw/3.3.8/double-gnu
mathlib/fftw/3.3.8/double/intel
```

环境变量设置

■ 查看已加载模块list

```
[slurmtest@login01 ~]$ module list
Currently Loaded Modulefiles:
 1) compiler/devtoolset/7.3.1    2) compiler/rocm/2.9          3) mpi/hpcx/2.4.1/gcc-7.3.1
[slurmtest@login01 ~]$ which gcc
/opt/rh/devtoolset-7/root/usr/bin/gcc
[slurmtest@login01 ~]$ which mpirun
/opt/hpc/software/mpi/hpcx/v2.4.1/gcc-7.3.1/bin/mpirun
[slurmtest@login01 ~]$
[slurmtest@login01 ~]$ module show compiler/devtoolset/7.3.1
-----
/opt/hpc/software/modules/compiler/devtoolset/7.3.1:

module-whatis      Name: Developer Toolset
module-whatis      Version: 7.3.1
module-whatis      Category: compiler, runtime support
prepend-path        PATH /opt/rh/devtoolset-7/root/usr/bin
prepend-path        MANPATH /opt/rh/devtoolset-7/root/usr/share/man
prepend-path        INFOPATH /opt/rh/devtoolset-7/root/usr/share/info
prepend-path        PERL5LIB /opt/rh/devtoolset-7/root/usr/lib64/perl5/vendor_perl:/opt/rh/devtoolset-7/root/usr/share/perl5/vendor_perl
prepend-path        LD_LIBRARY_PATH /opt/rh/devtoolset-7/root/usr/lib64:/opt/rh/devtoolset-7/root/usr/lib
prepend-path        C_INCLUDE_PATH /opt/rh/devtoolset-7/root/usr/include/c++/7:/opt/rh/devtoolset-7/root/usr/include/c++/7/x86_64-redhat-linux
prepend-path        CPLUS_INCLUDE_PATH /opt/rh/devtoolset-7/root/usr/include/c++/7:/opt/rh/devtoolset-7/root/usr/include/c++/7/x86_64-redhat-linux
setenv             LDFLAGS -Wl,-rpath=/opt/rh/devtoolset-7/root/usr/lib64 -Wl,-rpath=/opt/rh/devtoolset-7/root/usr/lib
-----
```

■ 查看模块配置show

环境变量设置

■ 加载新模块load

■ 卸载模块unload

```
[slurmtest@login01 ~]$ module list
Currently Loaded Modulefiles:
 1) compiler/devtoolset/7.3.1  2) compiler/rocm/2.9          3) mpi/hpcx/2.4.1/gcc-7.3.1
[slurmtest@login01 ~]$ module load compiler/intel/2017.5.239
[slurmtest@login01 ~]$
[slurmtest@login01 ~]$ which icc
/opt/hpc/software/compiler/intel/intel-compiler-2017.5.239/bin/intel64/icc
[slurmtest@login01 ~]$
[slurmtest@login01 ~]$ module unload compiler/intel/2017.5.239
[slurmtest@login01 ~]$
[slurmtest@login01 ~]$ which icc
/usr/bin/which: no icc in (/opt/hpc/software/mpi/hpcx/v2.4.1/gcc-7.3.1/bin:/opt/hpc/software/mpi/hpcx/v2.4.1/hcoll/bin:/opt/hpc/software/mpi/hpcx/v2.4.1/ucx_without_rocm/bin:/opt/rocm/bin:/opt/rocm/hcc/bin:/opt/rocm/hip/bin:/opt/rh/devtoolset-7/root/usr/bin:/usr/lib64/qt-3.3/bin:/opt/hpc/software/mpi/pmix/bin:/opt/hpc/software/mpi/pmix/sbin:/opt/hpc/software/mpi/hpcx/v2.4.1/ucx_without_rocm//bin:/opt/hpc/software/mpi/hpcx/v2.4.1/ucx_without_rocm//sbin:/public/home/slurmtest/perl5/bin:/opt/gridview/slurm/bin:/opt/gridview/slurm/sbin:/opt/gridview/munge/bin:/opt/gridview/munge/sbin:/opt/clusconf/sbin:/opt/clusconf/bin:/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/opt/ibutils/bin:/public/home/slurmtest/.local/bin:/public/home/slurmtest/bin)
[slurmtest@login01 ~]$
[slurmtest@login01 ~]$
```

环境变量设置

■ 置换模块switch

```
[slurmtest@login01 ~]$ module list
Currently Loaded Modulefiles:
 1) compiler/devtoolset/7.3.1  2) compiler/rocm/2.9
[slurmtest@login01 ~]$                                         3) mpi/hpcx/2.4.1/gcc-7.3.1      默认加载的3个模块
[slurmtest@login01 ~]$ which gcc
/opt/rh/devtoolset-7/root/usr/bin/gcc
[slurmtest@login01 ~]$ which icc
[slurmtest@login01 ~]$ which mpicxx
/usr/bin/which: no mpicxx in (/opt/hpc/software/mpi/hpcx/v2.4.1/gcc-7.3.1/bin:/opt/hpc/software/mpi/hpcx/v2.4.1/hcall/bin:/opt/hpc/software/mpi/4.1/ucx_without_rocm/bin:/opt/rocm/bin:/opt/rocm/hcc/bin:/opt/rocm/hip/bin:/opt/rh/devtoolset-7/root/usr/bin:/usr/lib64/qt-3.3/bin:/opt/hpc/mpi/pmix/bin:/opt/hpc/software/mpi/pmix/sbin:/opt/hpc/software/mpi/hpcx/v2.4.1/ucx_without_rocm//bin:/opt/hpc/software/mpi/hpcx/v2.4.1/ucx_wi
cm//sbin:/public/home/slurmtest/perl5/bin:/opt/gridview/slurm/bin:/opt/gridview/slurm/sbin:/opt/gridview/munge/bin:/opt/gridview/munge/sbin:/conf/sbin:/opt/clusconf/bin:/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/opt/ibutils/bin:/public/home/slurmtest/.local/bin:/public/home
st/bin)
[slurmtest@login01 ~]$ module switch compiler/devtoolset/7.3.1 compiler/intel/2017.5.239    切换模块，加载intel编译器，卸载
[slurmtest@login01 ~]$                                         gcc7.3.1
[slurmtest@login01 ~]$ which gcc
/usr/bin/gcc
[slurmtest@login01 ~]$ which mpicxx
[slurmtest@login01 ~]$ which mpirun
/opt/hpc/software/mpi/hpcx/v2.4.1/gcc-7.3.1/bin/mpirun
[slurmtest@login01 ~]$ module list
[slurmtest@login01 ~]$ module list
Currently Loaded Modulefiles:
 1) compiler/intel/2017.5.239  2) compiler/rocm/2.9
[slurmtest@login01 ~]$                                         3) mpi/hpcx/2.4.1/gcc-7.3.1      list查看切换结果
[slurmtest@login01 ~]$ module purge
[slurmtest@login01 ~]$ module list
[slurmtest@login01 ~]$ module list
No Modulefiles Currently Loaded.                                         清空所有加载模块
```

环境变量设置

module 命令可以直接写在.bashrc中

```
[slurmtest@login01 ~]$ cat .bashrc
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions
export OMPI_TIMING_ENABLE=1
export SLURM_PMIX_DIRECT_CONN=true
export SLURM_PMIX_DIRECT_CONN_UCX=false
export SLURM_PMIX_DIRECT_CONN_EARLY=true
#export SLURM_PMIX_FENCE=tree
export UCX_TLS=dc

#module purge
#module load compiler/rocm/2.9
#module load compiler/intel/2017.5.239
#module load mpi/hpcx/2.4.1/intel-2017.5.239

[slurmtest@login01 ~]$
```

自定义用户默认环境

目录

01 整体概况

02 用户登录及提交作业

03 软件及环境设置

04 作业管理

- 调度系统
- 查看分区
- 查看节点
- 作业提交
- 作业查看
- 作业控制

05 作业脚本示例

调度系统-基本概念

集群

- ✓ 包含计算、存储、网络、操作系统、编译器、运行环境、开发工具等各种资源实体且彼此联系的资源集合；

资源

- ✓ 作业运行过程中使用的可量化实体都是资源；
- ✓ 硬件资源（节点、内存、CPU、类GPU加速卡等）和软件资源（如License）；

作业

- ✓ 物理构成，一组关联的资源分配请求，以及一组关联的处理过程；
- ✓ 交互方式，可以分为交互式作业和非交互式作业；

队列

- ✓ 组织资源的一种形式
- ✓ 带名称的作业容器、用户访问控制、资源使用限制；

调度系统

- ✓ 负责监控和管理集群中资源和作业的软件系统；
- ✓ 通常由资源管理器、调度器、任务执行器，以及用户命令和API组成；

调度系统-主要作用

多任务管理

系统资源整合

单一系统镜像

资源访问控制

集群操作系统

调度系统-SLURM主要特点

架构设计优秀

- 扩展性 (功能/规模)
- 高性能 (提交/调度)
- 灵活性 (自定义插件)
- 容错性 (服务/节点/作业)

基础功能完善

- 优先级策略 (Multi-Factor)
- 作业调度 (FairShare/Backfill/ FCFS, etc)
- 资源竞争 (Exclusive/Preempt/Gang, etc)
- 多级限制 (Cluster/Account/User/Partition, etc)
- 资源预留 (CPU/MEM/类GPU加速卡/License, etc)

特色功能突出

- 资源绑定 (CPU、MEM、类GPU加速卡等各类资源)
- 关联调度 (如类GPU加速卡关联CPU)
- 能耗管理 (限频、开关机、记账等)
- 拓扑调度 (基于拓扑结构调度, 限定交换机数等)

兼容性/交互性良好

- MPI兼容 (IntelMPI/MVAPICH/OpenMPI,etc)
- LSF兼容 (bsub/bjobs/bqueue/bhosts, etc)
- PBS兼容 (qsub/qstat/pbsnodes/pestat, etc)
- 分析工具 (融合Influxdb/ES/HDF5等, 方便分析)

作业管理-查看分区sinfo

sinfo命令参数 (1/2)

-p <partition>, --partition=<partition>	查看指定分区的状态，对应环境变量SINFO_PARTITION。 示例：sinfo -p caspra
-n <nodes>, --nodes=<nodes>	查看指定节点的信息。 示例：sinfo -p caspra -n e02r4n[00-19]
-N, --Node	以面向节点（默认为分区）的格式打印信息，每行一个节点。 示例：sinfo -N -p caspra -n e02r4n[00-19]
-t <states>, --states=<states>	查询指定节点状态的分区或节点的信息。 常见状态包括：alloc idle drain down drng。
-s, --summarize	查看分区中节点状态的摘要信息。 主要是限制分区中节点的状态统计，如已分配数alloc、空闲数idle、总数total等。

作业管理-查看分区sinfo

sinfo参数介绍 (2/2)

-o <output_format>,

--format=

<output_format>

按照指定的格式显示信息，对应变量SINFO_FORMAT。

格式为 “%[.]size]<type> %[.]size]<type> ...” 。

其中，点号（.）表示右对齐，size表示字段长度，type为代表特定字段的字符（或字符串）。

示例如下：

sinfo -o %all 以逗号分隔的形式显示所有字段。

sinfo -o "%9P %.5a %.10l %.6D %.6t %N"

显示内容：分区名-分区状态-最大运行时间-节点数-节点状态-节点列表。

-O <output_format>,

--Format=

<output_format>

按照指定的字段显示状态信息。

格式为 “type[:.]size],type[:.]size],...” 。

其中，type为代表特定字段的字段名，点号（.）表示右对齐，size表示字段长度。示例如下：

sinfo -O all

sinfo -O Partition:9,available:.6,time:.11,nodes:.6,statecompact:.6,nodelist:.12

作业管理-查看分区

示例2: 查看分区的摘要信息 (--summarize) 。

等价于 sinfo -o "%9P %.5a %.10l %.16F %N"

```
[slurmtest@glogin ~]$ sinfo --summarize
PARTITION AVAIL  TIMELIMIT   NODES(A/I/O/T)  NODELIST
debug      up    infinite     1/3/0/4  node[1001-1004]
fat1t      up    3-00:00:00   1/11/4/16 node[1001-1016]
fat3t      up    3-00:00:00   0/6/0/6  node[1021-1026]
v100       up    3-00:00:00   10/2/0/12 node[1031-1042]
hcpu       up    3-00:00:00   1/250/1/252 node[2001-2252]
gpu        up    3-00:00:00   0/36/1/37 node[151-173,175-182,184-186,188-190]
gpu4       up    3-00:00:00   0/31/0/31 node[201-211,213,215-220,222-224,226-230,232-236]
```

示例3: 查看分区详情。

scontrol show partition

```
[slurmtest@glogin ~]$ scontrol show partition debug
PartitionName=debug
AllowGroups=ALL AllowAccounts=bujd,haowj,jiangzhilin,liyuan,root,sugon,test02,wangrongpt0224,liurx,wangrongpt02241,zoubs,lyuzh,im_ysd,panzhang
,sunpeng,aiiss,liurx01,houfy,sgzhou,tianlh,zhousen,guozk,yangg,chenxs,wangyt,jshu,ybyang,wangshancai,yuliangjin,anna_itp,cairg,xianzy,oycz
,huangqq,wenghongming,ktitimbo AllowQos=ALL
AllocNodes=ALL Default=NO QoS=N/A
DefaultTime=02:00:00 DisableRootJobs=NO ExclusiveUser=NO GraceTime=0 Hidden=NO
MaxNodes=UNLIMITED MaxTime=UNLIMITED MinNodes=0 LLN=YES MaxCPUsPerNode=UNLIMITED
Nodes=node[1001-1004]
PriorityJobFactor=1 PriorityTier=6000 RootOnly=NO ReqResv=NO OverSubscribe=NO
OverTimeLimit=NONE PreemptMode=OFF
State=UP TotalCPUs=224 TotalNodes=4 SelectTypeParameters=NONE
JobDefaults=(null)
DefMemPerNode=UNLIMITED MaxMemPerNode=UNLIMITED
```

作业管理-查看节点

节点详情： scontrol show node node151

```
[slurmtest@glogin ~]$ scontrol show node node151
NodeName=node151 Arch=x86_64 CoresPerSocket=6
    CPUAlloc=0 CPUTot=12 CPULoad=0.01
    AvailableFeatures=(null)
    ActiveFeatures=(null)
    Gres=gpu:TeslaK20m:2
    NodeAddr=node151 NodeHostName=node151 Version=19.05.5
    OS=Linux 3.10.0-957.el7.x86_64 #1 SMP Thu Nov 8 23:39:32 UTC 2018
    RealMemory=31983 AllocMem=0 FreeMem=16966 Sockets=2 Boards=1
    State=IDLE ThreadsPerCore=1 TmpDisk=0 Weight=1 Owner=N/A MCS_label=N/A
    Partitions=gpu
    BootTime=2020-05-21T14:57:56 SlurmdStartTime=2020-07-06T16:11:06
    CfgTRES=cpu=12,mem=31983M,billing=12,gres/gpu=2
    AllocTRES=
    CapWatts=n/a
    CurrentWatts=0 AveWatts=0
    ExtSensorsJoules=n/s ExtSensorsWatts=0 ExtSensorsTemp=n/s
```

作业管理-作业提交

srun

交互式作业提交

sbatch

批处理作业提交

salloc

节点资源获取

作业管理-作业提交参数

选项	含义	类型	示例
-J	作业名称，使用squeue看到的作业名	字符串	-J wrf; 表示作业名称为 “wrf”
-n	作业申请的总任务数	数值	-n 32; 表示作业申请32个任务。若不指定--cpus-per-task，则表示作业申请的总核数为32个核心。
-N	作业申请的节点数	数值	-N 10 表示作业申请10个计算节点
-p	指定作业提交的队列	字符串	-p silicon表示将作业提交到silicon队列
--ntasks-per-node	指定每个节点运行进程数	数值	--ntasks-per-node=32表示每个节点运行32个进程（任务）
--ntasks-per-socket=<count>	指定在每个Socket启动的进程数	数值	--ntasks-per-node=2标识每个Socket上运行2个进程
--cpus-per-task=<count>	指定任务需要的处理器数目	数值	--cpus-per-task=8 表示每个任务占用8个处理器核
-t	指定作业的执行时间，若超过该时间，作业将被杀死	数值	-t 30 表示作业的执行时间不超过30分钟，格式可以为：时:分:秒
-w, --nodelist=hosts...	指定分配特定的计算节点	字符串	-w t0100,t0101 表示使用t0100 t0101等2个节点
--mem=<size[units]>	指定作业在每个节点使用的内存限制。	数字	--mem=2G 限定作业在每个节点最多占用2G的最大内存。
-d, --dependency=<dependency_list>	作业依赖关系设置	字符串	-d after:123 表示本作业须待作业123开始以后再执行
--gres=<list>	指定每个节点使用通用资源名称及数量	字符串	--gres=gpu:2 表示本作业使用gpu卡，且每个节点使用2卡

作业提交-sbatch

```
[sugon@gpunode1 ~]$ sbatch -n 4 sleep.job //sbatch 只接收脚本  
Submitted batch job 19
```

```
[sugon@gpunode1 ~]$ cat sleep.job //脚本格式示例
```

```
#!/bin/bash  
#SBATCH -J sleep          //指定作业名  
#SBATCH -p debug          //指定队列  
#SBATCH --time=00:01:00      //指定运行时间（分钟）注：需要设定为比较准确的时间，否  
则调度系统会超时强杀作业。若不设置该参数，继承队列的默认运行时长。请在程序中设置断点，保存中间结果，  
防止程序中断或者异常导致中间结果丢失。  
#SBATCH -N 2                //请求节点数  
#SBATCH -n 2                //请求核心数  
#SBATCH --gres=gpu:2         //请求gpu数  
#SBATCH -o logs/%j.sleep    //标准输出文件 注：若为相对路径，则必须存在该目录，否则作业会提  
交失败，且无日志输出  
#SBATCH -e logs/%j.sleep    //错误输出文件 注：若为相对路径，则必须存在该目录，否则作业会提交  
失败，且无日志输出  
echo ${SLURM_JOB_NODELIST}  作业占用节点列表  
echo start on $(date)       开始时间  
sleep 100                   执行命令  
echo end on $(date)         结束时间
```

作业提交-srun

例：提交请求2个节点2个核心的并且指定作业的名称为job1

```
srun -J job1 -N 2 -n 2 sleep 10
```

提交命令

作业名称

申请节点数

申请核心数

运行命令

作业管理-作业查看

squeue:查询排队和运行状态的作业

参数	解释
-A, --account=account(s)	查询指定账号的作业, 默认全部账号下的作业
-j, --jobs <job_id_list>	查看指定JOB IDS的作业信息, 默认显示全部
-n, --name=<name>	查看指定名称的作业信息
-p, --partition=<names>	查看指定分区的作业信息
--state=<names>	指定状态查看作业信息
--users=<names>	指定用户名查看作业信息

```
[slurmtest@glogin slurm_template]$ squeue
      JOBID PARTITION      NAME      USER ST      TIME  NODES NODELIST (REASON)
        43248    hcpu      test slurmtes R      0:01      10 node[2005-2014]
        43247   debug      OMP slurmtes R      0:05       2 node[1001,1003]
[slurmtest@glogin slurm_template]$
```

作业管理-作业查看

查询指定作业详情:

scontrol show job [-d] [<jobid>]

```
[slurmtest@login05 newtest]$  
[slurmtest@login05 newtest]$ scontrol show jobs -d 1434869  
JobId=1434869 JobName=DTCP  
  UserId=slurmtest(1232) GroupId=slurmtest(1225) MCS_label=N/A  
  Priority=1084 Nice=0 Account=slurmtest QOS=normal  
  JobState=COMPLETED Reason=None Dependency=(null)  
  Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0  
  DerivedExitCode=0:0  
  RunTime=00:00:21 TimeLimit=00:20:00 TimeMin=N/A  
  SubmitTime=2020-03-29T17:55:18 EligibleTime=2020-03-29T17:55:18  
  AccrueTime=2020-03-29T17:55:18  
  StartTime=2020-03-29T17:55:18 EndTime=2020-03-29T17:55:39 Deadline=N/A  
  SuspendTime=None SecsPreSuspend=0 LastSchedEval=2020-03-29T17:55:18  
  Partition=caspra AllocNode:Sid=login05:126240  
  ReqNodeList=e02r4n[18-19] ExcNodeList=(null)  
  NodeList=e02r4n[18-19]  
  BatchHost=e02r4n18  
  NumNodes=2 NumCPUs=16 NumTasks=16 CPUs/Task=1 ReqB:S:C:T=0:0:0:0  
  TRES=cpu=16,mem=45600M,node=2,billing=16,gres/dcua=8  
  Socks/Node=** NtasksPerN:B:S:C=8:0:2:0 CoreSpec=*  
    Nodes=e02r4n[18-19] CPU IDs=0-1,8-9,16-17,24-25 Mem=22800 GRES=  
  MinCPUsNode=8 MinMemoryCPU=2850M MinTmpDiskNode=0  
  Features=(null) DelayBoot=00:00:00  
  OverSubscribe=OK Contiguous=0 Licenses=(null) Network=(null)  
  Command=/public/home/slurmtest/zhangtao/cpubind/newtest/sgsrun.slurm  
  WorkDir=/public/home/slurmtest/zhangtao/cpubind/newtest  
  StdErr=/public/home/slurmtest/zhangtao/cpubind/newtest/1434869  
  StdIn=/dev/null  
  StdOut=/public/home/slurmtest/zhangtao/cpubind/newtest/1434869  
  Power=  
  TresPerNode=dcu:4  
  
[slurmtest@login05 newtest]$
```

作业资源分布

作业管理-作业删除

scancel: 删除作业命令

COMMAND	解释
scancel <jobid>	删除指定作业
scancel -t ST	删除指定状态的作业
scancel --account=<name>	删除指定账号的作业
scancel --name=<name>	删除指定名称的作业
scancel --partition=<names>	删除指定分区的作业
scancel --reservation=<name>	删除指定预约名称的作业
scancel --state=<names>	删除指定状态的作业
scancel --user=<name>	删除指定用户的作业
scancel --nodelist=<names>	删除指定节点的作业

作业控制 - scontrol

scontrol: 控制作业命令

COMMAND	解释	备注
scontrol suspend <jobid>	挂起作业	运行作业可挂起
scontrol resume <jobid>	恢复作业	挂起作业可恢复
scontrol requeue <jobid>	作业重新排队	运行作业可重新排队
scontrol hold <id/name>	保留作业	排队作业可保留
scontrol release <id/name>	释放作业	保留作业可释放

目录

01 整体概况

02 用户登录及提交作业

03 软件及环境设置

04 作业管理

05 作业脚本示例

进程绑定：SLURM绑定参数

绑定资源

CPU MEM 类GPU加速卡

常用参数：

- ◆ --ntasks-per-
<node|socket|core..>
- ◆ --mem/--mem-per-
cpu
- ◆ --cpu-bind=
<socket|core||doms|verbose
...>

https://slurm.schedmd.com/mc_support.html

<u>Low-level (explicit binding)</u>	
--cpu-bind=...	Explicit process affinity binding and control options
<u>High-level (automatic mask generation)</u>	
--sockets-per-node=S	Number of sockets in a node to dedicate to a job (minimum)
--cores-per-socket=C	Number of cores in a socket to dedicate to a job (minimum)
--threads-per-core=T	Number of threads in a core to dedicate to a job (minimum)
-B S[:C[:T]]	Combined shortcut option for --sockets-per-node, --cores-per_cpu, --threads-per_core
<u>Task Distribution Options</u>	
-m / --distribution	Distributions of: arbitrary block cyclic plane=x [block cyclic]:[block cyclic fcyclic]
<u>Memory as a consumable resource</u>	
--mem=mem	amount of real memory per node required by the job.
--mem-per-cpu=mem	amount of real memory per allocated CPU required by the job.
<u>Task invocation control</u>	
--cpus-per-task=CPUs	number of CPUs required per task
--ntasks-per-node=ntasks	number of tasks to invoke on each node
--ntasks-per-socket=ntasks	number of tasks to invoke on each socket
--ntasks-per-core=ntasks	number of tasks to invoke on each core
--overcommit	Permit more than one task per CPU
<u>Application hints</u>	
--hint=compute_bound	use all cores in each socket
--hint=memory_bound	use only one core in each socket
--hint=[no]multithread	[don't] use extra threads with in-core multi-threading
<u>Resources reserved for system use</u>	
--core-spec=cores	Count of cores to reserve for system use
--thread-spec=threads	Count of threads to reserve for system use (future)

作业脚本示例1—串行作业示例

串行作业的提交示例：

```
#!/bin/bash
#SBATCH -o log/%j
#SBATCH -J SERIAL
#SBATCH -p debug
#SBATCH -t 00:10:00
#SBATCH --mem-per-cpu=3G
#SBATCH -n 1
#SBATCH -w e16r1n00
#SBATCH --tasks-per-node=1

module load compiler/devtoolset/7.3.1
module load compiler/rocm/2.9
module load mpi/hpcx/2.4.1/gcc-7.3.1

# 4x32, about 60 sec
export LOOPMAX=1000000

time srun --mpi=pmix_v3 ./open_fire_v5 $LOOPMAX
# 或者直接运行
#time ./open_fire_v5 $LOOPMAX
```

提交串行作业时，在脚本中直接调用可执行程序

作业脚本示例2—MPI作业

简化的MPI作业的提交示例：

```
#SBATCH -o log/%j
#SBATCH -J MPI
#SBATCH -p debug
#SBATCH -t 00:10:00
#SBATCH --mem-per-cpu=3G
#SBATCH -N 20
#SBATCH --tasks-per-node=32

module load compiler/devtoolset/7.3.1
module load compiler/rocm/2.9
module load mpi/hpcx/2.4.1/gcc-7.3.1

echo =====
env | grep "SLURM"
echo =====

ulimit -a
which mpirun

# 20x32, 60 sec
export LOOPMAX=5000000

echo "use srun, loop=$LOOPMAX" && time srun --mpi=pmix_v3 ./open_fire_v5 $LOOPMAX
```

作业脚本示例3—单节点OpenMP作业

简化的OpenMP作业脚本示例：

```
#!/bin/bash
#SBATCH -J OPENMP
#SBATCH -p debug
#SBATCH -N 1
#SBATCH -n 4
#SBATCH --cpus-per-task=2
#SBATCH -o log/%j.loop
```

```
module load compiler/devtoolset/7.3.1
module load compiler/rocm/2.9
module load mpi/hpcx/2.4.1/gcc-7.3.1
```

```
export OMP_NUM_THREADS=2
srun ./calc_openmp_init
```

设置变量OMP_NUM_THREADS， 控制单进程的并发线程数

作业脚本示例4—MPI+OpenMP

简化的MPI+OpenMP作业脚本示例：

```
#!/bin/bash
#SBATCH -J OPENMP
#SBATCH -p debug
#SBATCH -N 2
#SBATCH -n 8
#SBATCH --ntasks-per-node=4
#SBATCH --cpus-per-task=8

module load compiler/devtoolset/7.3.1
module load compiler/rocm/2.9
module load mpi/hpcx/2.4.1/gcc-7.3.1

export OMP_NUM_THREADS=8
srun --mpi=pmix_v3 ./calc_openmp_mpi
#或用mpirun运行
#mpirun <options> ./ calc_openmp_mpi
```

设置变量OMP_NUM_THREADS， 控制单进程的并发线程数

谢 谢