Quick Start Guide 快速入门指南

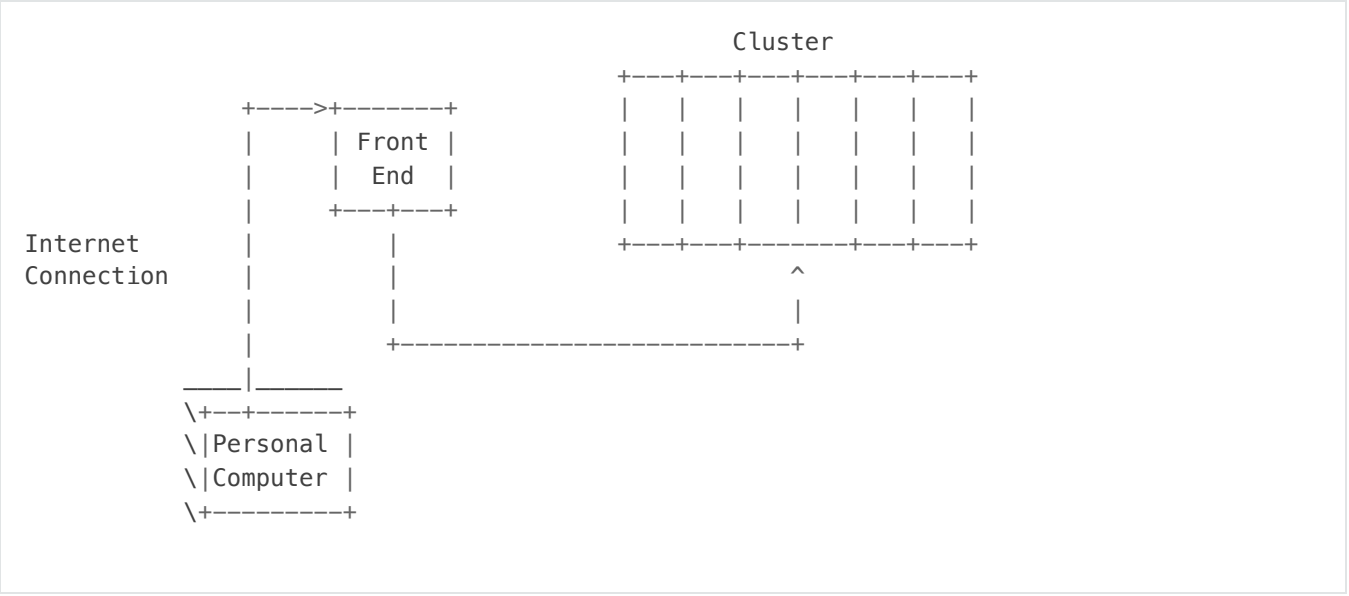
Detailed below are common methodologies and good practices for access and using the CRC's resources.

以下详细介绍了访问和使用 CRC 资源的常用方法和良好实践。

Cluster Computing Format
集群计算格式

The organization of a super-computer or cluster is quite different from traditional desktop computing. Instead of all commands executing on the machine in front of you, everything is executed on a remote server, at times with no user interaction. A typical connection will look like:

超级计算机或集群的组织结构与传统的桌面计算有很大不同。所有命令不是在你面前的机器上执行，而是在远程服务器上执行，有时甚至无需用户交互。典型的连接方式如下：



To access the CRC infrastructure, you will connect to a Front end or Head node where you can prepare, submit, check on, and delete your batch jobs. Most jobs executed on CRC servers are batch jobs, which are non-interactive well-defined jobs which are queued and executed on a

compute host when resources are available.

要访问 CRC 基础设施，您需要连接到前端或主节点，在那里您可以准备、提交、检查和删除批处理作业。大多数在 CRC 服务器上执行的作业都是批处理作业，这些作业是非交互式的、定义明确的作业，当计算资源可用时会排队并在计算主机上执行。

Front End Systems　前端系统

> **Note 注意**
>
> For users connecting from off campus, please review the section Connecting from off campus first.
>
> 对于从校外连接的用户，请先查看“从校外连接”部分。

In order to submit and run jobs on CRC servers, the first step is to connect to a front end machine. This machine will facilitate connections to the job manager which will handle your job from submission to execution.

为了在 CRC 服务器上提交和运行作业，第一步是连接到前端机器。该机器将促进与作业管理器的连接，作业管理器将从提交到执行处理您的作业。

The CRC provides the following front-end machines for compilation and job submission. Each machine is configured with *identical* software stacks.

CRC 提供以下前端机器用于编译和作业提交。每台机器都配置了相同的软件堆栈。

- crcfe01.crc.nd.edu (2 12 core Intel(R) Haswell processors with 256 GB RAM)

  crcfe01.crc.nd.edu（2 个 12 核 Intel(R) Haswell 处理器，256 GB 内存）
- crcfe02.crc.nd.edu (2 12 core Intel(R) Haswell processors with 256 GB RAM)

  crcfe02.crc.nd.edu（2 个 12 核 Intel(R) Haswell 处理器，256 GB 内存）

> **Warning 警告**
>
> Front-Ends are NOT for large long running (>1hr) jobs. For such jobs please using the queuing system and compute nodes. Any long running on a public front end machine may be killed.
>
> 前端不适用于大型长时间运行（超过 1 小时）的任务。对于此类任务，请使用排队系统和计算节点。任何在公共前端机器上长时间运行的任务可能会被终止。

To see how to connect to a front end from your favorite OS, see Connecting to CRC Servers.

要了解如何从您喜欢的操作系统连接到前端，请参见连接到 CRC 服务器。

Proper Front End Usage
正确的前端使用

The front end machines are the interface to the the rest of the computing resources of the CRC. There are both faculty owned and public front ends, the majority of information found here will pertain to public front ends. The public front ends are shared by all of campus and external collaborators to perform a wide variety of tasks related to research.

前端机器是 CRC 其余计算资源的接口。这里既有教职工拥有的前端，也有公共前端，本处大部分信息将涉及公共前端。公共前端由全校师生及外部合作者共享，用于执行与研究相关的各种任务。

Logging in　　登录

There are 2 general access front ends available.

有两台通用访问前端可用。

```
crcfe01.crc.nd.edu
crcfe02.crc.nd.edu
```

The primary interface to these machines is through SSH. To connect to either of these machines, you'd start either a terminal (Mac and Linux) or an ssh client on Windows. For Windows we recommend MobaXterm. You'd then either follow the instructions on the ssh client or enter the following command in a terminal, replacing "X" with the desired machine (1 or 2) and replacing "netid" with your Notre Dame NetID:

访问这些机器的主要方式是通过 SSH。要连接到任一机器，您可以启动终端（Mac 和 Linux）或 Windows 上的 SSH 客户端。对于 Windows，我们推荐使用 MobaXterm。然后，您可以按照 SSH 客户端上的说明操作，或在终端中输入以下命令，将"X"替换为所需机器编号（1 或 2），将"netid"替换为您的 Notre Dame NetID：

```
ssh netid@crcfe0X.crc.nd.edu
```

If you are having troubles logging in and have a new account, be sure to read the welcome email which was sent on the activation of your CRC account:

如果您在登录时遇到问题并且有一个新账户，请务必阅读在激活您的 CRC 账户时发送的欢迎邮件：

- First login to okta.nd.edu.

  首次登录 okta.nd.edu。
- If you continue to have trouble logging in:

  如果您仍然无法登录：

  - Check all spelling 检查所有拼写是否正确
  - Be sure your ssh client and OS are updated/patched.

    确保您的 ssh 客户端和操作系统已更新/打好补丁。

If all else fails, notify us via email at CRCSupport@nd.edu and please provide the following details:

如果所有方法都失败，请通过电子邮件通知我们，邮箱地址是 CRCSupport@ndedu，并请提供以下信息：

- ND netid
- Which machine you're trying to connect to

  您尝试连接的机器名称
- Your computer's Operating System (Windows, Mac, Linux, etc)

  您的计算机操作系统（Windows、Mac、Linux 等）
- The ssh client you're using (Putty, MobaXterm, terminal, etc).

  您使用的 ssh 客户端（Putty、MobaXterm、终端等）

> **Note 注意**
>
> If you intend on using `HTCondor` for job submissions, the front end machine to connect to is `condorfe.crc.nd.edu` If you need to compile code with infiniband support, connect to `epycfe.crc.nd.edu`.
>
> 如果您打算使用 `HTCondor` 提交作业，连接的前端机器是 `condorfe.crc.nd.edu` 。如果您需要编译支持 infiniband 的代码，请连接到 `epycfe.crc.nd.edu` 。

Utilizing Front Ends 利用前端

Once connected to a front end, you may begin preparing jobs for submission to either UGE or HTCondor (on `condorfe.crc.nd.edu`). There are a few important notes about operating within the front end environment:

一旦连接到前端，您就可以开始准备作业，提交到 UGE 或 HTCondor（在 `condorfe.crc.nd.edu` 上）。关于在前端环境中操作，有几点重要说明：

- There will be other users connected to this machine, any disruptive behaviour will be addressed.

  这台机器上会有其他用户连接，任何干扰行为都会被处理。

- Any task / process running longer than 1 (ONE) hour may be removed / killed by an administrator. Submit long running jobs to the batch system.

  任何运行时间超过 1（一个）小时的任务/进程可能会被管理员移除或终止。请将长时间运行的作业提交到批处理系统。

- Do not launch large mpi or smp processes on a front end.

  不要在前端启动大型的 mpi 或 smp 进程。

  - More information on the queues and UGE can be found on the UGE Environment page.

    有关队列和 UGE 的更多信息，请参见 UGE 环境页面。

- When logged in, you will be within your user AFS space. Initially, there will be a few directories there. **DO NOT** remove your `Public` directory, this contains important login scripts.

  登录后，您将进入您的用户 AFS 空间。最初，那里会有几个目录。请勿删除您的 `Public` 目录，该目录包含重要的登录脚本。

- The default shell is `BASH`. If you have an older account, your login shell may be `tcsh`. If you'd like to request bash as your default, email CRCSupport@nd.edu.

  默认的 shell 是 `BASH` 。如果您有较旧的账户，您的登录 shell 可能是 `tcsh` 。如果您想申请将 bash 设为默认，请发送邮件至 CRCSupport@ndedu。

- Software which is not included in an install of RHEL, can be accessed through the Modules.

  未包含在 RHEL 安装中的软件，可以通过模块访问。

- Small processing which is not disruptive/resource intensive can be done on the front ends. This is normally pre-processing or post-processing after completion of UGE jobs.

  不干扰且资源消耗较小的小型处理可以在前端完成。这通常是在 UGE 作业完成后的预处理或后处理。

- All CRC systems are a Linux variant. For tip on Linux commands, see Linux Coding Cheat Sheet

  所有 CRC 系统都是 Linux 变体。有关 Linux 命令的提示，请参见 Linux Coding Cheat Sheet

---

For submitting jobs to UGE, please see Submitting Batch Jobs. More sample scripts/submissions can normally be found on a module/software's page. Search for the module in question in the search bar on the top left of the page.

有关提交作业到 UGE，请参见提交批处理作业。更多示例脚本/提交通常可以在模块/软件页面找到。请在页面左上角的搜索栏中搜索相关模块。

Available Software　可用软件

The software environment on CRC servers is managed utilizing Modules. A module is a an easy way to manage the necessary environmental paths and changes to utilize different pieces of software. You can easily modify your programming and/or application environment by simply loading and removing the required modules. The most common module commands are:

CRC 服务器上的软件环境通过模块进行管理。模块是一种简便的方法，用于管理使用不同软件所需的环境路径和变更。您可以通过加载和移除所需模块，轻松修改您的编程和/或应用环境。最常用的模块命令有：

```
module avail

module list

module load module_name
```

More information on the CRC Modules can be found on the Modules page.

有关 CRC 模块的更多信息，请参见模块页面。

File Systems　文件系统

The CRC provides two complimentary file systems for storing programs and data.

CRC 提供了两种免费的文件系统用于存储程序和数据。

AFS

- Distributed file system 分布式文件系统
- Initial 100GB allocation 初始 100GB 分配
- Longer-term storage; backup taken daily

  长期存储；每日备份
- User file system, your home directory lives here

  用户文件系统，您的主目录位于此处
- You can check your current AFS usage with the following command:

  您可以使用以下命令检查当前的 AFS 使用情况：

```
quota
```

Panasas

- High-performance parallel file system

  高性能并行文件系统
- To obtain an allocation on /scratch365, send us a request at CRCSupport@nd.edu.

  要在 /scratch365 上获得分配，请发送请求至 CRCSupport@ndedu。
- Used for runtime working storage; no backup, purged yearly.

  用于运行时工作存储；无备份，每年清理一次。
- You can check your current /scratch365 usage with the following command:

  您可以使用以下命令检查当前的 /scratch365 使用情况：

```
pan_df –H /scratch365/netid
```

Further info: Storage 更多信息：存储

File Transfers　文件传输

To transfer files from your local desktop MacOS filesystem to your CRC filesystem space we recommend installing and using the following file transfer (GUI) client: Cyberduck

要将文件从本地桌面 MacOS 文件系统传输到您的 CRC 文件系统空间，我们建议安装并使用以下文件传输（图形界面）客户端：Cyberduck

For Windows users, we recommend using MobaXterm as both an SSH client and a file transfer client.

对于 Windows 用户，我们推荐使用 MobaXterm 作为 SSH 客户端和文件传输客户端。

If you would like to transfer data between the CRC servers and Google Drive, we recommend the Rclone tool.

如果您想在 CRC 服务器和 Google Drive 之间传输数据，我们推荐使用 Rclone 工具。

Job Scripts　作业脚本

A typical batch job is a simple script which contains the commands necessary to execute the job which needs to be accomplished whether this is an R, Matlab, or Python script or a compiled executable.

一个典型的批处理作业是一个简单的脚本，包含执行所需任务的命令，无论这是一个 R、Matlab 或 Python 脚本，还是一个编译后的可执行文件。

Jobs are submitted to the compute nodes via the Univa Grid Engine (UGE) batch submission system. Basic UGE batch scripts should conform to the following template:

作业通过 Univa Grid Engine（UGE）批处理提交系统提交到计算节点。基本的 UGE 批处理脚本应符合以下模板：

```
#!/bin/bash

#$ -M netid@nd.edu   # Email address for job notification
#$ -m abe            # Send mail when job begins, ends and aborts
#$ -pe smp 24        # Specify parallel environment and legal core size
#$ -q long           # Specify queue
#$ -N job_name       # Specify job name

module load xyz      # Required modules

mpirun -np $NSLOTS ./app # Application to execute
```

Further info: Submitting Batch Jobs

更多信息：提交批处理作业

Parallel Environments　并行环境

To use more than one core in a job, you must specify a parallel environment. A parallel environment is a way to specify whether to use multiple cores on one machine or to use multiple nodes (more than 1 server). The two parallel environments are below.

要在作业中使用多个核心，必须指定一个并行环境。并行环境是一种指定是在一台机器上使用多个核心，还是使用多个节点（多个服务器）的方法。以下是两种并行环境。

P.E. name 并行环境名称	Valid Core counts 有效核心数	Example 示例	Targeted machines 目
smp	1, 2, 3, ... N 1，2，3，..... N	smp 8	any node 任意节点
mpi-48	48, 96, 144, ... 48，96，144，...	mpi-48 96	d24cepycXXX
mpi-64	64, 128, 192, . .	mpi-64 192	d32cepycXXX

Note 注意

- If no parallel environment is requested (i.e. you do not specify a -**pe** parameter), then the default execution environment is a single-core serial job.

  如果未请求并行环境（即未指定 -pe 参数），则默认执行环境为单核串行作业。

- **Every machine has one thread per core!**

  每台机器每个核心都有一个线程！

Further info: UGE Environment

更多信息：UGE 环境

Queues　队列

CRC provides two general-purpose queues for the submission of jobs (using the **-q** parameter):

CRC 提供了两个通用队列用于提交作业（使用 -q 参数）：

Below you'll find a summarization of the available queues.

以下是可用队列的总结。

Queue Name 队列名称	Purpose 目的	Run time Limit 运行时
long 长	Long running jobs 长时间运行的作业	14 days* 14 天*
gpu	Long running GPU jobs 长时间运行的 GPU 作业	4 days* 4 天*

> **Note 注意**
>
> In the above table, the long queue and gpu queue can include any faculty / lab owned machines you have access to. The runtime limit and example machines will most likely be different from the table above. Speak to your lab-mates, PI, or email us at CRCSupport@nd.edu if you'd like to know specifics of those machines.
>
> 在上表中，long queue 和 gpu queue 可以包含您有权限访问的任何学院/实验室拥有的机器。运行时间限制和示例机器很可能与上表不同。如需了解这些机器的具体信息，请与您的实验室同事、导师联系，或发送邮件至 CRCSupport@ndedu。

If you wish to target a specific architecture for your jobs, then you can specify a `host group` instead of a general-purpose queue.

如果您希望为作业指定特定的架构，则可以指定一个 `host group` ，而不是通用队列。

Host Groups 主机组

If you wish to target machines in a finer granularity than queues, there are host groups. For example, the general access machines are the "@crc_d32cepyc" host group. If your research lab or faculty advisor has purchased a machine(s), there is most likely a host group you can target.

如果您希望以比队列更细的粒度定位机器，可以使用主机组。例如，一般访问机器属于"@crc_d32cepyc"主机组。如果您的研究实验室或导师购买了机器，通常会有一个您可以定位的主机组。

Monitoring the status and availability of resources at a glance can be done with the **free_nodes.sh** script.

可以使用 free_nodes.sh 脚本一目了然地监控资源的状态和可用性。

```
free_nodes.sh -G              # For general access
free_nodes.sh -D              # For Debug nodes
free_nodes.sh @crc_d32cepyc    # You can target host groups
```

For further resource monitoring, consult the help information: `free_nodes.sh --help`

有关更多资源监控信息，请参阅帮助信息： `free_nodes.sh --help`

Job Submission and Monitoring
作业提交与监控

Job scripts can be submitted to the SGE batch submission system using the `qsub` command:

作业脚本可以使用 `qsub` 命令提交到 SGE 批处理提交系统：

```
qsub job.script
```

Once your job script is submitted, you will receive a numerical `job id` from the batch submission system, which you can use to monitor the progress of the job.

一旦您的作业脚本提交成功，您将从批处理提交系统收到一个数字 `job id` ，您可以使用该数字来监控作业的进度。

Job scripts that are determined by UGE to have made valid resource requests will enter the queuing system with a queue-waiting `qw` status (once the requested resources become available, the job will enter the running `(r)` status). Job scripts that are determined not to be valid will enter the queuing system with an error queue-waiting `(Eqw)` status.

被 UGE 判定为资源请求有效的作业脚本将以排队等待 `qw` 状态进入排队系统（当请求的资源可用时，作业将进入运行 `(r)` 状态）。被判定为无效的作业脚本将以错误排队等待 `(Eqw)` 状态进入排队系统。

To see the status of your job submissions, use the `qstat` command with your username (netid) and observe the status column:

要查看您的作业提交状态，请使用带有您的用户名（netid）的 `qstat` 命令，并观察状态列：

```
qstat -u username
```

For a complete overview of your job submission, invoke the qstat command with the job id:

要全面查看您的作业提交情况，请使用作业 ID 调用 qstat 命令：

```
qstat –j job_id
```

To see all pending and running jobs, simply type `qstat` without any options

要查看所有待处理和正在运行的作业，只需输入 `qstat` ，无需任何选项

```
qstat
```

The main reasons for invalid job scripts (i.e. having Eqw status) typically are:

作业脚本无效（即状态为 Eqw）的主要原因通常是：

- Illegal specification of parallel environments and/or core size requests

  非法指定并行环境和/或核心大小请求
- Illegal queue specification

  非法队列规范
- Copying job scripts from a Windows OS environment to the Linux OS environment on the front-end machines (invisible Windows control code-blocks are not parsed correctly by UGE). This can be fixed by running the script through the `dos2unix` command

  将作业脚本从 Windows 操作系统环境复制到前端机器的 Linux 操作系统环境（不可见的 Windows 控制代码块无法被 UGE 正确解析）。可以通过使用 `dos2unix` 命令运行脚本来解决此问题。

Why is my job waiting in the Queue?
为什么我的任务在队列中等待?

The first reason a job could be waiting in a queue is simply because there are no resources available yet for your job to begin running. Note that for the most part (especially if you are using the general access machines), you are sharing resources with every other CRC user.

作业可能排队等待的第一个原因，通常是因为还没有可用的资源让你的作业开始运行。请注意，大多数情况下（尤其是当你使用通用访问机器时），你是与所有其他 CRC 用户共享资源的。

The first item to check if if there are available resources at the moment for your job, you can use the `free_nodes.sh` script to check, for example:

首先要检查的是当前是否有可用资源供你的作业使用，你可以使用 `free_nodes.sh` 脚本来检查，例如：

```
free_nodes.sh -G
```

Note that even if there resource shown as available, the scheduler could be freeing up space for a larger job with higher priority than yours. There is also a maximum number of concurrent jobs to avoid overloading the systems with one person's jobs, this number fluctuates during the year depending on system utilization.

请注意，即使资源显示为可用，调度器也可能正在为比您的作业优先级更高的更大作业释放空间。系统还设有最大并发作业数，以避免单个人的作业过多而导致系统过载，这个数值会根据系统利用率在一年中波动。

- If you need to push a large amount of jobs consisting of less than 4 cores each, perhaps HTCondor is a good fit for you.

  如果您需要提交大量每个作业使用少于 4 个核心的作业，HTCondor 可能非常适合您。

Parallel Jobs　并行作业

If you requested a parallel environment with your job, be sure that requested environment adheres to the target resources.

如果您在提交作业时请求了并行环境，请确保所请求的环境符合目标资源的要求。

For example, be sure the machines you're targeting through `smp` can support the number of cores requested, a request of `smp 75` will never be served on the general access or `@crc_d32cepyc` host group.

例如，确保您通过 `smp` 定位的机器能够支持所请求的核心数， `smp 75` 的请求在普通访问或 `@crc_d32cepyc` 主机组上永远不会被处理。

If you're requesting an `mpi-X Y` environment, be sure the `Y` value is a multiple of `X`, and be sure `X` is a valid number for the targeted machines. For example, `mpi-64 128` is the correct requested environment of 2 machines out of the general access queue.

如果您正在请求一个 `mpi-X Y` 环境，请确保 `Y` 的值是 `X` 的倍数，并且确保 `X` 是目标机器的有效数字。例如， `mpi-64 128` 是从通用访问队列中请求 2 台机器的正确环境。

Job Deletion　　作业删除

To delete a running or queued (e.g. submissions with `Eqw` status) job, use the following command:

要删除正在运行或排队中的作业（例如，状态为 `Eqw` 的提交），请使用以下命令：

```
qdel -j job_id
```

Job Resource Monitoring
作业资源监控

To better understand the resource usage (e.g. memory, CPU and I/O utilization) of your running jobs, you can monitor the runtime behavior of your job's tasks as they execute on the compute nodes.

为了更好地了解正在运行的作业的资源使用情况（例如内存、CPU 和 I/O 利用率），您可以监控作业任务在计算节点上执行时的运行时行为。

To determine the nodes on which your tasks are running, enter the following `qstat` command along with your username. Record the machine names (e.g. d12chas400.crc.nd.edu) associated with each task (both MASTER and SLAVE):

要确定任务运行的节点，请输入以下 `qstat` 命令及您的用户名。记录与每个任务（包括 MASTER 和 SLAVE）相关联的机器名称（例如 d12chas400.crc.nd.edu）：

```
qstat -u username -g t
```

There are two methods for analyzing the behavior of tasks (once you have a machine name):

分析任务行为有两种方法（在你获得机器名称后）：

- Xymon GUI Tool (detailed breakdown per task on a given machine)

  Xymon 图形界面工具（针对指定机器的每个任务的详细分析）

  > Xymon is a GUI tool to analyze the behavior of processes on a given CRC machine. It can be accessed on CRC Xymon. Note that you will need to be on the ND network to see this site.
  >
  > Xymon 是一个图形用户界面工具，用于分析指定 CRC 机器上进程的行为。可以通过 CRC Xymon 访问。请注意，您需要连接到 ND 网络才能查看此站点。
  >
  > Use Xymon to navigate to the specific machine and then view the runtime resource usage of tasks on the machine.
  >
  > 使用 Xymon 导航到特定机器，然后查看该机器上任务的运行时资源使用情况。

- `qhost` command (aggregate summary across all tasks on a given machine)

  `qhost` 命令（汇总给定机器上所有任务的总体摘要）

  > You can summarize the resource utilization of all tasks on a given machine using the following qhost command:
  >
  > 您可以使用以下 qhost 命令汇总给定机器上所有任务的资源利用情况：

```
qhost -h machine_name
```

Job Arrays　作业数组

> **Note 注意**
>
> If you find that you need to frequently submit 50 or more different jobs, we request that you implement those tasks within a job array. Grid engine is able to handle arrays much more efficiently than tens or hundreds of individual scripts from a single user. Fewer individual tasks reduces load on the job scheduler and improves overall performance.
>
> 如果您发现需要频繁提交 50 个或更多不同的作业，我们建议您将这些任务实现为作业数组。网格引擎能够比单个用户提交的数十个或数百个独立脚本更高效地处理数组。较少的独立任务减少了作业调度器的负载，并提升整体性能。

If you have a large number of job scripts to run that are largely identical in terms of executable and processing tasks, (e.g. a parameter sweep where only the input changes per run) then you may want to use a `job array` to submit your job. You specify how many copies of the script that you need to run with the `-t` parameter. For example, specifying:

如果您有大量的作业脚本需要运行，这些脚本在可执行文件和处理任务方面基本相同（例如参数扫描，每次运行仅输入不同），那么您可能想使用 `job array` 来提交作业。您可以通过 `-t` 参数指定需要运行的脚本副本数量。例如，指定：

```
#$ -t 1-3
```

will run 3 identical job scripts. Job arrays use an internal environment variable, `$SGE_TASK_ID`, to dinstinguish between the different jobs and this can also be used as input to your job.:

将运行 3 个相同的作业脚本。作业数组使用一个内部环境变量 `$SGE_TASK_ID` 来区分不同的作业，这个变量也可以作为作业的输入。

```
#!/bin/bash

#$ -pe smp 1           # Specify parallel environment and legal core size
#$ -q long             # Specify queue
#$ -N job_name         # Specify job name
#$ -t 1-3              # Specify number of tasks in array

module load python

# Note that different tasks will use different input files.
python analyze.py < data.$SGE_TASK_ID
```

If your tasks do not map directly to consecutive integer numbers, you may use a shell array within the script to do the mapping for you. In this example, we specify three different input strings that are then passed as an argument to the Python script:

如果您的任务不能直接映射到连续的整数编号，您可以在脚本中使用 shell 数组来为您进行映射。在此示例中，我们指定了三个不同的输入字符串，然后将它们作为参数传递给 Python 脚本：

```
#!/bin/bash

#$ -pe smp 1           # Specify parallel environment and legal core size
#$ -q long             # Specify queue
#$ -N job_name         # Specify job name
#$ -t 1-3              # Specify number of tasks in array

module load python

fruits=( "orange" "apple" "pair" )

# Note that different tasks will use different input parameters.
python pie_recipe.py ${fruits[$SGE_TASK_ID-1]}
```

Warning 警告

To avoid extraneous emails, please do not use email notification options (-M and -m) when submitting an array job.

为了避免收到多余的电子邮件，请在提交数组作业时不要使用电子邮件通知选项（-M 和 -m）。

More detailed information about job arrays is available from the manual page:

有关数组作业的更详细信息，请参阅手册页：

```
man qsub
```

or from this website: qsub manual

或访问此网站：qsub manual