CU Research Computing Allocation Request

Project Title Here

Overview

Please fill out this template when constructing an allocation request for Summit. Be sure to address all of the bullet points in this worksheet. If you feel that some of the requested information is not applicable, please note why that is rather than leaving that section out of your request entirely.

Requests for larger allocations will need more detailed justification. As a general guideline, a request for 300K SU might require a two-page proposal, while a 1M+ SU request might require three to five pages.

If you have questions when preparing this allocation request, please contact rc-help@colorado.edu. We are happy to provide advice and answer any questions.

1 Introduction and Summary

Provide some background information on your proposal in this section. Particularly:

- Concise Description - Describe the portion of the Project that this computational work supports.
- Allocation goals - Describe the anticipated goals for this particular effort as a subset of the Project goals.
- Duration: indicate if this allocation is for one year or until completion of a nearer-term goal whichever is sooner.
- Expected follow-up - indicate if this is the final allocation for the Project or if work will likely continue.
- Supporting grants - indicate funding agencies and grant numbers that support this work (if any).

2 Computational Details

We ask that you use this section to provide an overview of the program(s) that you will be using.

- Describe, in general terms, the application(s) that you will be running.
- Detail how you have adapted your application to Summit:
  - If application is an RC-provided module please indicate so; in that case you don't need to provide further details about application optimization.
  - What compiler and compiler flags were used to compile your software on Summit, and why? Is this a community model, and are you unsure of its compiler flags? If so, please indicate the steps you took to build the model on Summit, and provide a link to the software's installation documentation.
  - Note whether any third-party libraries, such as Intel MKL or the C++ Boost library are used by your application.
  - Does your application require the use of container software (e.g., Singularity)?
- Work-flow Details
  - How many cores per node does your application use?
  - Does your application require large amounts of RAM, limiting the useable cores-per-node or forcing you onto the high-memory nodes?

  - How long will your typical job(s) to run for?
  - Does your application incorporate checkpointing, so that it can restart from a saved state? Describe.
  - Which partition do you typically run in, and why (e.g., shas, smem, sgpu)?

- **For parallel applications**, show how the total job time changes **on Summit** as more cores or nodes are used (i.e., provide scaling information). An overview of how to generate scaling data for your parallel application is provided in the companion document, *Scaling Analysis*, located on the RC website.

- **For serial applications**, provide a table of computation times for each calculation size that you plan to run. This table will be used to evaluate your request for resources in Section 2.1. Submit this table in lieu of scaling information.

2.1 CPU Time Request

Once you have characterized the performance of your code, indicate the number of service units (SU's) needed for your proposed work by filling out the SU-request table below. SUs are used in lieu of core hours to account for differences in the availability and capability of the different node types available on Summit. These billing weights are indicated in Table 1 below. The SUs required for any given job can be computed from the formula:

SU = (node weight) x (number of cores) x (hours job is run).

Node Type	Partition Name	Weight
Haswell	shas	1
GPU	sgpu	2.5
High Memory	smem	6
Knights Landing	sknl	0.1

Table 1: Relative billing weights for the different Summit node-types. See the Summit Partitions in our User Guide for more details on the different Summit nodes available.

Adapt the SU request worksheet in Table 2 to reflect your allocation request. Be sure to:

- Include a row for each *type* of job you plan to run. Different job types might run on different partitions, on different core counts, or run for differing amounts of time.

- Indicate the partition, billing weight, number of jobs, cores, and hours run for each job type.

- Multiply the weight, jobs, cores, and hours columns together to arrive at the SU total required for that job type

Finally, refer to your scaling data and provide some brief justification for your choice of core counts for each job type. For instance, given the example scaling data in Section 2, we see that performance gains diminish substantially when running with 96 cores. For such a performance profile, it makes sense to run jobs with up to 48 cores, but running with 96 cores would be wasteful.

Job Type	Node Type	Weight	Jobs	Cores	Hours	Total (SU)
Production (small)	shas	1	50	48	12	28,800
Production (large)	shas	1	10	384	24	92,160
Post-processing	smem	6	60	12	1	4,320
Grand Total (SU)	–	–	–	–	–	125,280

Table 2: SU request worksheet.

3 Data Management

Provide some information on the data generated by your application here.

- Describe the disk I/O by job type:
  1. Temporary files - indicate the size and number of job-specific temporary files and how/if they are removed.
  2. Local vs scratch - Usage of on-node local disk vs /scratch
  3. Output files - Describe the nature, size and number of output files that remain after job completion.

Please modify the table below as needed to indicate the storage requirements for your different job types. When calculating the required space and number of files, please also include both temporary files and files that remain after job completion.

Job Type	Required Space (GB)	Number of Files
Production (small)	10	1,000
Production (large)	1,000	500

Table 3: Scatch-disk-space requirements by job type.

How will you deal with the data you generate during the course of the project?

- How much will need to be migrated off scratch to safe storage. (Recall that scratch file systems are purged at regular intervals and thus can only be used for temporary storage.)
- Describe this "safe" storage (RC PetaLibrary, department file server, etc.)