Slot Multiplier for Calculating CPU Usage in Gridengine

Contents

Where "cpus" are mentioned, read "cores". I'll try to make this clearer at some point.

Introduction

When Gridengine (e.g., 8.1.9) calculates cpu usage information, it is done based on slots used and either cpu time or wallclock time. This works if each slot corresponds to a single cpu. But, if more than one cpu is allocated to a slot, this calculation does not give the desired usage information.

For the purposes of this article:

- `execd_params` is configured with `ACCT_RESERVED_USAGE=true` and `SHARETREE_RESERVED_USAGE=true` so that wallclock time rather than cpu time is used
- cpus are a dedicated resource managed by a consumable
- jobs request the number of cpus for each slot

with the goal of calculating cpu usage as:

```
cpu = wallclock * nslots * ncpus_per_slot
```

Usage

The complex used to track the number of cpus used per slot is specified in the `execd_params` with `SLOT_MULTIPLIER_NAME`. E.g.

```
execd_params        SLOT_MULTIPLIER_NAME=ncpus
```

The complex would be defined as:

```
ncpus               ncpus       INT       <=    FORCED      YES        0        1000
```

A queue would be configured with:

```
complex_values        ncpus=4
```

- 4 cpus allocatable from this queue

So, a job request for 4 slots X 16 cpus would be:

```
#$ -pe dev 4
#$ -l ncpus=16
```

and a run of 30s would amount to:

cpu = 30 * 4 *16 = 1920

versus what is currently returned:

cpu = 30 *4 = 120

Implementation

Changes

`source/libs/sgeobj/sge_conf.h`:

- add `mconf_get_slot_multiplier_name()` declaration

`source/libs/sgeobj/sge_conf.c`:

- set `slot_multiplier_name` static variable to hold value set in `execd_params SLOT_MULTIPLIER_NAME`
- implement `mconf_get_slot_multiplier_name()`

`source/daemons/execd/load_avg.h`:

- augment `build_reserved_usage()` declaration to accept reference to job

`source/daemons/execd/load_avg.c`:

- augment `build_reserved_usage()` to accept reference to job
- enhance `build_reserved_usage()` to get slot multiplier and use it when calculating cpu usage
- add `get_slot_multiplier()` static function get the multiplier value if defined, or 1.0 otherwise
- update `calculate_reserved_usage()` to get job reference and provide it when calling `build_reserved_usage()`

`source/daemons/execd/reaper_execd.c`:

- update `build_derived_final_usage()` to provide job reference when calling `build_reserved_usage()`

Patch files (based off of https://arc.liv.ac.uk/downloads/SGE/releases/8.1.9/):

- 0001-Support-for-slot-multiplier.patch

Summary

Changes to the code are minimal. The highlights are:

- change the signature of the `build_reserved_usage()` function to take a job reference so that the consumable information can be obtained
- update `build_reserved_usage()` to use the slot multiplier value

Conclusion

These changes make it possible to account for the number of cpus per slot which is critical to calculating cpu usage by wallclock.