Implementing and Evaluating Multi-Resource Scheduling Design in Slurm

Report Dated: 02-17-2019

1. Read Slurm Scheduler API Plugin from : https://slurm.schedmd.com/schedplugins.html
2. Focussed mainly on directory structure of Slurm in Github.
3. Understood how to add a new plugin and which files to add/ change. (Ref: https://slurm.schedmd.com/add.html)
4. As most of the code uses multithreading and as I am not familiar with pthreads(Since I worked on Java), I have spent some time in understanding them using the following links:
    a. https://timmurphy.org/2010/05/04/pthreads-in-c-a-minimal-working-example/
    b. https://www.cs.nmsu.edu/~jcook/Tools/pthreads/library.html
    c. https://computing.llnl.gov/tutorials/pthreads/

5. Every new Scheduler plugin needs to have:
   Variables:

```
const char plugin_name[] – Any meaningful name
const char plugin_type[] – Major type needs to be sched, minor type defines the
functionality of plugin. For Example: sched/backfill
const uint32_t plugin_version – Identifies the version of slurm used to build this
plugin. It's a good approach to specify plugin version rather than not specifying it.
```

   Functions:
   - **int init(void)** – Called when plugin is loaded, Need to do any global initialization here and this returns SLURM_ERROR, SLURM_SUCCESS.
   - **void fini(void)** – Called when plugin needs to be removed. Need to clear any allocated Storage here.
   - **int slurm_sched_p_reconfig(void)** – Re-read any configuration files. Needs to return Slurm Success error code if the operation is successful else returns error code with error num set to an appropriate value to indicate reason for failure.
   - **int slurm_sched_p_schedule(void) –** For passive schedulers, it invokes a scheduling pass(Not clear on what are passive schedulers and what is scheduling pass?)
   - **int slurm_sched_p_newalloc (struct job_record *job_ptr) –** It notes successful allocation of resources to a job.
     ***Arguments*:** pointer to the slurmctld job structure. It gives information regarding partition, allocated resources and time limit, etc.,
   - **int slurm_sched_p_freealloc (struct job_record *job_ptr) -** It notes the successful release of resources of a job. Arguments same as previous function.
   - **uint32_t slurm_sched_p_initial_priority (uint32_t last_prio, struct job_record *job_ptr)** – It establishes the initial priority of a new job. So the return value is the priority of this Job.
     ***Arguments**:* last_prio: implies default priority of the previously submitted job. What is the need?

- Used to provide First in first out scheduling by assigning the new job a priority less than this value.
- It can also be used to establish an initial priority of zero to all jobs and allow a scheduler plugin to decide job's priority.
- Pointer to the slurmctld job structure. This can be used to get partition, resource requirements, time limit, etc.

➢ **void slurm_sched_p_job_is_pending (void):** Notes that some job is pending execution.
➢ **void slurm_sched_p_partition_change (void):** Notes that some partition change has happened such as time or size limits
➢ **char *slurm_sched_p_get_conf (void):** Returns scheduler specific configuration information as a string. This is reported for scontrol show configuration command. Note: the return value is released using **xfree()** function
➢ **int slurm_sched_p_get_errno (void) :** Gives the number of scheduler specific error
➢ **const char *slurm_sched_p_strerror(int errnum):** Returns the string description of scheduler specific error code.

6. All the Scheduler plugins have a header file, a wrapper file and an actual C file (For example: back fill scheduler has backfill.h, backfill.c and backfill_wrapper.c). Usually wrapper has init, fini and functions some of the functions specified above.
7. Summary of First In and First out scheduler:
    a. builtin.h has: functions builtin_agent, stop_builtin_agent and builtin_reconfig which are just declared and implemented in builtin.c
    b. builtin_wrapper.c has: init, fini, slurm_sched_p_reconfig, slurm_sched_p_initial_priority which are mentioned as in API summary.
    c. builtin.c has: Implements functions in builtin.h.
8. Back fill Scheduler has the similar structure with the files as backfill.h, backfill.c and backfill_wrapper.c.
9. I am now focused on understanding the implementation in Backfill scheduler i.e backfill.c as well as First In First out scheduler i.e., builtin.c. There are around 75 functions totally and I am half way through understanding each function. I should be in a pretty good shape in a week.
10. The slides in the following link shared by Yuping are very interesting but there are a number of them. So, I am going through the ones related to scheduling. https://slurm.schedmd.com/publications.html
11. Read the Slurm design from https://slurm.schedmd.com/slurm_design.pdf