

My first slurm job

Examples

Submit a simple MPI job

- On this example we run a small MPI application doing the following steps:
 - Create a submission file
 - Submit the job to the default partition
 - Execute a simple MPI code
 - Check the status of the job
 - Read the output
- Download source code

```
wget --no-check-certificate https://wiki.incd.pt/attachments/71 -O cpi.c
```

- Create a submission file

```
vi my_first_slurm_job.sh
```

- Edit the file

```
#!/bin/bash

#SBATCH --job-name=MyFirstSlurmJob
#SBATCH --time=0:10:0
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=16

# Be sure to request the correct partition to avoid the job to be held in the queue, furthermore
#   on CIRRUS-B (Minho) choose for example HPC_4_Days
#   on CIRRUS-A (Lisbon) choose for example hpc
#SBATCH --partition=hpc
```

```
# Used to guarantee that the environment does not have any other loaded module
module purge

# Load software modules. Please check session software for the details
module load gcc63/openmpi/4.0.3

# Prepare
src='cpi.c'
exe="./cpi.$SLURM_JOB_ID"

# Compile application
echo "=== Compiling ==="
mpicc -o $exe $src

# Run application. Please note that the number of cores used by MPI are assigned in the SBATCH directives.
echo "=== Running ==="
if [ -e $exe ]; then
    chmod u+x $exe
    mpiexec -np $SLURM_NTASKS $exe
    rm -f $exe
fi

echo "Finished with job $SLURM_JOBID"
```

- Submit the job

```
sbatch my_first_slurm_job.sh
```

- Check status of the job

```
$ squeue
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
1171	HPC_4_Days	MyFirstS	username	PD	0:00	1	wn075

- Check further details about your job (very long output)

```
scontrol show job 1171
```

- Read the output of the job:

If name is not specified slurm will create by default a file with the output of your run

slurm-{job_id}.out

e.g. slurm-1171.out

- Cancel your job

```
$ scancel 1171
```

MPI examples:

Hello World:

```
#include <mpi.h>
#include <stdio.h>

int main(int argc, char** argv) {
    // Initialize the MPI environment
    MPI_Init(NULL, NULL);

    // Get the number of processes
    int world_size;
    MPI_Comm_size(MPI_COMM_WORLD, &world_size);

    // Get the rank of the process
    int world_rank;
    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

    // Get the name of the processor
    char processor_name[MPI_MAX_PROCESSOR_NAME];
    int name_len;
    MPI_Get_processor_name(processor_name, &name_len);

    // Print off a hello world message
    printf("Hello world from processor %s, rank %d out of %d processors\n",
        processor_name, world_rank, world_size);

    // Finalize the MPI environment.
    MPI_Finalize();
}
```

```
}
```

PI calculation

```
/* -*- Mode: C; c-basic-offset:4 ; -*- */
/*
 * (C) 2001 by Argonne National Laboratory.
 * See COPYRIGHT in top-level directory.
 */

#include "mpi.h"
#include <stdio.h>
#include <math.h>

int main(int argc, char *argv[])
{
    long int    n, i;
    int    myid, numprocs;
    double PI25DT = 3.141592653589793238462643;
    double mypi, pi, h, sum, x;
    double startwtime = 0.0, endwtime;
    int    namelen;
    char    processor_name[MPI_MAX_PROCESSOR_NAME];

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD, &myid);
    MPI_Get_processor_name(processor_name, &namelen);

    n = 100000000000; /* default # of rectangles */
    if (myid == 0) {
        startwtime = MPI_Wtime();
    }

    MPI_Bcast(&n, 1, MPI_INT, 0, MPI_COMM_WORLD);

    h = 1.0 / (double) n;
    sum = 0.0;

    /* A slightly better approach starts from large i and works back */
    for (i = myid + 1; i <= n; i += numprocs)
```

```

{
    x = h * ((double)i - 0.5);
    sum += 4.0 / (1.0 + x*x);
}

mypi = h * sum;

MPI_Reduce(&mypi, &pi, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);

if (myid == 0) {
    endwtime = MPI_Wtime();
    printf("pi=%.16f, error=%.16f, ncores %d, wall clock time = %f\n", pi, fabs(pi - PI25DT), numprocs, endwtime-
startwtime);
    fflush(stdout);
}

MPI_Finalize();
return 0;
}

```

Revision #16

Created 13 December 2019 15:57:10 by Joao Machado

Updated 16 March 2023 16:29:49 by João Pina