Use the local filesystem

Structure of SCITAS filesystems

- The structure and purpose of each filesystem is described here.
- $HOME$ and $WORK$ are shared across the site, while $SCRATCH$ is local to each cluster.
- On $SCRATCH$ automatic deletion of files older than 2 weeks may happen without notice.
- Production jobs should use $SCRATCH$.

What to do when CPU time is significantly less than WALL time?

- $SCRATCH$ is a GPFS parallel filesystem which is designed to perform well with parallel I/O.
- In certain cases a big number of files is produced at runtime. Such I/O patterns put stress on the $SCRATCH$ filesystem metadata service, and are generally much slower than using a local disk.
- A long term solution would require changing the code to use external libraries like HDF5 or ADIOS. Those libraries give more flexibility in the way data is saved/handled.
- A workaround is to use the local filesystem $TMPDIR$.
- Beware $TMPDIR$ is visible only once resources are allocated.

If you query the value of $TMPDIR$ in a login node:

```
[user@fidis ~]$ echo $TMPDIR
[user@fidis ~]$
```

However, within a job allocation:

```
[user@fidis ~]$ Sinteract
Cores:            1
Tasks:            1
Time:             00:30:00
Memory:           4G
Partition:        parallel
Account:          scitas-ge
Jobname:          interact
Resource:         
QOS:              normal
Reservation:      salloc: Pending job allocation 159671
salloc: job 159671 queued and waiting for resources
salloc: job 159671 has been allocated resources
salloc: Granted job allocation 159671
srun: Job step created
[nvarini@f061 ~]$ echo $TMPDIR
/tmp/159671
```

The variables $TMPDIR$, $WORK$ and $SCRATCH$ are set by the SLURM scheduler while preparing the environment for each job.

How to use the $TMPDIR$ in your simulations

Quantum Espresso

- The following example show how to use the $TMPDIR$ with Quantum-ESPRESSO (QE).
- QE relies on fortran namelists to read certain parameters used during the simulation.
- The only change that has to be done to a standard pw input is related to the outdir in the &CONTROL namelist. For example, in the input below the outdir is set equal to a placeholder `fakeoutdir`:

```
&CONTROL
  calculation = 'scf',
  restart_mode = 'from_scratch',
  prefix = 'lgps_diel'
  tstress = .false.
  tprnfor = .false.
  outdir = 'fakeoutdir'
  pseudo_dir = '/scratch/nvarini/pseudo'
  disk_io='low'
  max_seconds=1800
  
```
• The submission script would look like:

```bash
#!/bin/bash
#SBATCH --nodes 2
#SBATCH --time=1:00:00
#SBATCH -p debug

module purge
module load intel/16.0.3
module load intelmpi/5.1.3
module load fftw/3.3.4-mpi
module load mkl/11.3.3

sed "s|fakeoutdir|${TMPDIR}|g" temp_pw > ${TMPDIR}/${SLURM_JOB_ID}_pw
srun pw.x < ${TMPDIR}/${SLURM_JOB_ID}_pw>${TMPDIR}/${SLURM_JOB_ID}.tmp.out
tar cvf ${SLURM_JOB_ID}.archive.tar ${TMPDIR}/* .
```

• After the sed command the CONTROL namelist looks like:

```plaintext
&CONTROL
  calculation = 'scf',
  restart_mode = 'from_scratch',
  prefix = 'lgps_diel'
  tstress = .false.
  tprnfor = .false.
  outdir = '/tmp/1325324'
  pseudo_dir = '/scratch/marcolon/test_LGPS/pseudo'
  disk_io='low'
  max_seconds=1800
/
```

• For a single 100GB file, all results in MB/s, <write into TMPDIR> : <copy from TMPDIR to /scratch>:

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Architecture</th>
<th>write into TMPDIR</th>
<th>copy from TMPDIR to /scratch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deneb</td>
<td>E5v2</td>
<td>76</td>
<td>74</td>
</tr>
<tr>
<td>Deneb</td>
<td>E5v3</td>
<td>109</td>
<td>103</td>
</tr>
<tr>
<td>Fidis</td>
<td>E5v4</td>
<td>529</td>
<td>498</td>
</tr>
</tbody>
</table>