slappt

Release 0.0.3

Computational Plant Science Lab

Dec 31, 2022

INTRODUCTION

CHAPTER ONE

INSTALLATION

1.1 Official release

`slappt` is available on the Python Package Index and can be installed with `pip`:

```
pip install slappt
```

1.2 Development version

The latest development version of `slappt` can be installed from GitHub:

```
pip install git+https://github.com/Computational-Plant-Science/slappt.git
```

CHAPTER

TWO

QUICKSTART

Say you have access to a Slurm cluster with `apptainer` installed, and you have permission to submit to the `batch` partition.

Copy the `hello.yaml` file from the `examples` directory to your current working directory, then run:

```
slappt hello.yaml > job.sh
```

Alternatively, without the configuration file:

```
slappt --image docker://alpine \
       --shell sh \
       --partition batch \
       --entrypoint "echo 'hello world'" > hello.sh
```

Your `hello.sh` script should now contain:

```
#!/bin/bash
#SBATCH --job-name=0477f4b9-e119-4354-8384-f50d7a96adad
#SBATCH --output=slappt.0477f4b9-e119-4354-8384-f50d7a96adad.%j.out
#SBATCH --error=slappt.0477f4b9-e119-4354-8384-f50d7a96adad.%j.err
#SBATCH --partition=batch
#SBATCH -c 1
#SBATCH -N 1
#SBATCH --ntasks=1
#SBATCH --time=01:00:00
#SBATCH --mem=1GB
module load apptainer  # only if you need, some clusters
apptainer exec docker://alpine sh -c "echo 'hello world'"
```

If already on the cluster, use the `--submit` flag to submit the job directly. (Standard Slurm commands must be available for this to work.) In this case the job ID is shown if submission was successful.

You can provide authentication information to submit the script to remote clusters over SSH. For instance, assuming you have key authentication set up and your key is `~/.ssh/id_rsa`:

```
slappt ... --host <cluster IP or FQDN> --username <username>
```

2.1 Caveats

There are a few things to note about the example above.

2.1.1 Shell

For most image definitions, specifying the `shell` is likely not necessary, as the default is `bash`. However, for images that don't have `bash` installed (e.g., `alphine` only has `sh`) a different shell must be selected.

2.1.2 Singularity support

If your cluster still uses `singularity`, pass the `--singularity` flag (or set the `singularity` key in the configuration file to `true`) to substitute `singularity` for `apptainer` in the command wrapping your workflow entrypoint.

2.1.3 Pre-commands

Note: if `apptainer` or `singularity` are not available by default on your cluster's compute nodes, you may need to add `--pre` commands (or a `pre` section to the configuration file), for instance `--pre "module load apptainer"`, or:

```
pre:
  - module load apptainer
```

CHAPTER

THREE

USAGE

3.1 Introspection

To show CLI usage help run `slappt --help`.

To show the current version run `slappt --version` (short form `-v`).

3.2 Basics

At minimum, `slappt` needs to know a few things before it can generate and/or submit a job script:

- `image`: the Docker image to use (e.g. `docker://ubuntu:latest` — note the `docker://` prefix is required — support for other registries is still in development)
- `partition`: the Slurm partition to submit the job to
- `entrypoint`: the command to run inside the container

To generate and/or submit a job script interactively, just run `slappt`. Parameters may also be provided as CLI options, or in a YAML configuration file. Config files are useful if the same workflow configurations is often reused — see the *YAML specification* for more details

To create a script for a job specified in `hello.yaml` (see for instance `examples/hello.yaml`), run:

```
slappt hello.yaml
```

Equivalently, using CLI arguments instead of a configuration file:

```
slappt --image docker://alpine \
       --shell sh \
       --partition batch \
       --entrypoint "echo 'hello world'"
```

3.3 Inputs

slappt is convenient not only for one-off container jobs, but for mapping a workflow over a list of inputs. This is accomplished with job arrays and can be configured via the `--inputs` options.

Note: your job's parallelism remains limited by the number of nodes allocated to it by the scheduler. To run containers in parallel, you must request multiple nodes.

The `--inputs` option's value must be the path to a text file containing a list of inputs, one on each line. This can be useful for parameter sweeps or to process a collection of files.

For instance, say we have some files:

```
$ cat f1
> hello 1
$ cat f2
> hello 2
```

We can create a file `inputs.txt`:

```
f1.txt
f2.txt
```

Assuming we have permission to submit to the `batch` partition, we can generate a script with:

~~~
```shell
slappt --image docker://alpine \
       --shell sh \
       --partition batch \
       --entrypoint "cat \$SLAPPT_INPUT" \
       --inputs inputs.txt > job.sh
~~~

This will generate a script to spawn a container, reading the input from the `SLAPPT_INPUT` environment variable.

It can be then submitted with, for instance:

```
sbatch --array=1-2 job.sh
```

3.4 Submissions

slappt can also submit jobs to a local or remote Slurm cluster via the `--submit` option. For instance, if you've cloned this repository to a cluster filesystem, standard Slurm commands (e.g. `sbatch`) are available:

```
slappt example/hello.yaml --submit
```

If successfully submitted, the job ID will be shown.

To submit to a remote cluster, use the `--host` and `--username` options, as well as the optional `--password` for password authentication, or `--pkey` to provide a path to a private key file.

Note: `sshlurm` is not compatible with multi-factor authentication.

Say you have a set of parameters:

```
1
2
3
```

Assuming you're on a Slurm cluster with permission to submit to the `batch` partition, you can generate and submit a parameter sweep job script with:

```
slappt --image docker://alpine \
       --shell sh \
       --partition batch \
       --entrypoint "echo $SLAPPT_INPUT" \
       --inputs inputs.txt  \
       --submit
```

You can also submit to a remote cluster. For instance, assuming you have key authentication set up, and your private key is the default location/name `~/.ssh/id_rsa`:

```
slappt --image docker://alpine \
       --shell sh \
       --partition batch \
       --entrypoint "echo $SLAPPT_INPUT" \
       --inputs inputs.txt \
       --submit \
       --host <your cluster IP or FQDN> \
       --username <your username>
```

The `--password` or `--pkey` options can be used to provide a password or a private key file, respectively.

CHAPTER

FOUR

YAML SPECIFICATION

slappt supports declarative YAML configuration to make container workflows reusable.

```
# standard attributes
image:          # the container image definition to use, e.g. docker://alpine (registry␣
↪prefix is required)
shell:          # the shell to use (default: bash)
partition:      # the cluster partition to submit to
entrypoint:     # the command to run inside the container
workdir:        # the working directory to use
email:          # the email address to send notifications to
name:           # the name of the job (default: slappt.<guid>)
pre:            # a list of commands to run before invoking the container (e.g. loading␣
↪modules)
inputs:         # a text file containing a newline-separated list of input files
environment:    # a dictionary of environment variables to set
bind_mounts:    # a list of bind mounts to use, in format <host path>:<container path>
no_cache:       # don't use the apptainer/singularity cache, force a rebuild of the image␣
↪(default: false)
gpus:           # the number of GPUs to request
time:           # the job's walltime
account:        # the account name to associate the job with
mem:            # the amount of memory to request (default: 1GB)
nodes:          # the number of nodes to request (default: 1)
cores:          # the number of cores to request (default: 1)
tasks:          # the number of tasks to request (default: 1)
header_skip:    # a list of header lines to skip when parsing the input file (can be␣
↪useful e.g. for clusters which have virtual memory and reject --mem headers)
singularity:    # whether to invoke singularity instead of apptainer (default: false)
# submission attributes
host:           # the hostname, IP or FQDN of the remote cluster to submit to
port:           # the port to use for the SSH connection (default: 22)
username:       # the username to use for the SSH connection
password:       # the password to use for SSH authentication
pkey:           # the path to the private key to use for SSH authentication
allow_stderr:   # don't raise an error if sshlurm encounters stderr output (default:␣
↪false)
timeout:        # the timeout for the SSH connection (default: 10)
```

CHAPTER FIVE

INDICES AND TABLES

- genindex
- modindex
- search