GridMap Documentation

Release 0.14.0

Daniel Blanchard **Cheng Soon Ong**
Christian Widmer

Apr 23, 2020

Contents

A package to allow you to easily create jobs on the cluster directly from Python. You can directly map Python functions onto the cluster without needing to write any wrapper code yourself.

This is the ETS fork of an older project called PythonGrid. Unlike the older version, it is Python 2/3 compatible.

For some examples of how to use it, check out `map_reduce.py` (for a simple example of how you can map a function onto the cluster) and `manual.py` (for an example of how you can create list of jobs yourself) in the examples folder.

CHAPTER 1

Documentation

1.1 Installation

GridMap can easily be installed via pip:

```
pip install gridmap
```

It can also be downloaded directly from Github.

1.2 License

GridMap is distributed under version 3 of the GNU Public License (GPLv3).

1.3 `gridmap` Package

The most useful parts of our API are available at the package level in addition to the module level. They are documented in both places for convenience.

1.3.1 From `job` Module

class gridmap.**Job**(*f*, *args*, *kwlist=None*, *cleanup=True*, *mem_free=u'1G'*, *name=u'gridmap_job'*, *num_slots=1*, *queue=u'all.q'*, *interpreting_shell=None*, *copy_env=True*, *add_env=None*, *par_env=u'smp'*, *gpu=0*, *h_vmem=None*, *h_rt=None*, *resources=None*)

Bases: `object`

Central entity that wraps a function and its data. Basically, a job consists of a function, its argument list, its keyword list and a field "ret" which is filled, when the execute method gets called.

Note: This can only be used to wrap picklable functions (i.e., those that are defined at the module or class level).

---

args

cause_of_death

cleanup

copy_env

environment

execute()

Executes function f with given arguments and writes return value to field ret. If an exception is encountered during execution, ret will contain a pickled version of it. Input data is removed after execution to save space.

function

gpu

h_rt

h_vmem

heart_beat

home_address

host_name

id

interpreting_shell

kwlist

log_stderr_fn

log_stdout_fn

mem_free

name

native_specification
define python-style getter

num_resubmits

num_slots

par_env

path

queue

resources

ret

timestamp

traceback

track_cpu

track_mem

uniq_id

white_list

working_dir

exception gridmap.**JobException**
New exception type for when one of the jobs crashed.

gridmap.**process_jobs**(*jobs*, *temp_dir=u'/scratch/'*, *white_list=None*, *quiet=True*, *max_processes=1*, *local=False*, *require_cluster=False*)
Take a list of jobs and process them on the cluster.

Parameters

- **jobs** (*list of Job*) – Jobs to run.
- **temp_dir** (*str*) – Local temporary directory for storing output for an individual job.
- **white_list** (*list of str*) – If specified, limit nodes used to only those in list.
- **quiet** (*bool*) – When true, do not output information about the jobs that have been submitted.

- **max_processes** (*int*) – The maximum number of concurrent processes to use if processing jobs locally.
- **local** (*bool*) – Should we execute the jobs locally in separate processes instead of on the the cluster?
- **require_cluster** (*bool*) – Should we raise an exception if access to cluster is not available?

Returns List of Job results

gridmap.**grid_map**(*f*, *args_list*, *cleanup=True*, *mem_free=u'1G'*, *name=u'gridmap_job'*, *num_slots=1*, *temp_dir=u'/scratch/'*, *white_list=None*, *queue=u'all.q'*, *quiet=True*, *local=False*, *max_processes=1*, *interpreting_shell=None*, *copy_env=True*, *add_env=None*, *gpu=0*, *h_vmem=None*, *h_rt=None*, *resources=None*, *completion_mail=False*, *require_cluster=False*, *par_env=u'smp'*)

Maps a function onto the cluster.

Note: This can only be used with picklable functions (i.e., those that are defined at the module or class level).

Parameters

- **f** (*function*) – The function to map on args_list
- **args_list** (*list*) – List of arguments to pass to f
- **cleanup** (*bool*) – Should we remove the stdout and stderr temporary files for each job when we're done? (They are left in place if there's an error.)
- **mem_free** (*str*) – Estimate of how much memory each job will need (for scheduling). (Not currently used, because our cluster does not have that setting enabled.)
- **name** (*str*) – Base name to give each job (will have a number add to end)
- **num_slots** (*int*) – Number of slots each job should use.
- **temp_dir** (*str*) – Local temporary directory for storing output for an individual job.
- **white_list** (*list of str*) – If specified, limit nodes used to only those in list.
- **queue** (*str*) – The SGE queue to use for scheduling.

- **quiet** (*bool*) – When true, do not output information about the jobs that have been submitted.
- **local** (*bool*) – Should we execute the jobs locally in separate processes instead of on the the cluster?
- **max_processes** (*int*) – The maximum number of concurrent processes to use if processing jobs locally.
- **interpreting_shell** (*str*) – The interpreting shell for the jobs.
- **copy_env** (*boolean*) – copy environment from master node to worker node?
- **add_env** (*dict*) – Environment variables to add to the environment. Overwrites variables which already exist due to `copy_env=True`.
- **gpu** (*int*) – number of GPUs to request
- **h_vmem** (*str, optional*) – hard virtual memory limit (e.g. "4G")
- **h_rt** (*str, optional*) – hard runtime limit (e.g. "00:59:00")
- **resources** (*list of str, optional*) – list of additional custom resources specifications
- **par_env** (*str*) – parallel environment to use.
- **completion_mail** (*boolean*) – whether to send an e-mail upon completion of all jobs
- **require_cluster** (*bool*) – Should we raise an exception if access to cluster is not available?

Returns List of Job results

1.4 `conf` Module

Global settings for GridMap. All of these settings can be overridden by specifying environment variables with the same name.

author Christian Widmer

author Cheng Soon Ong

author Dan Blanchard (dblanchard@ets.org)

var USE_MEM_FREE Does your cluster support specifying how much memory a job will use via mem_free? (Default: `False`)

var DEFAULT_QUEUE The default job scheduling queue to use. (Default: `all.q`)

var CREATE_PLOTS Should we plot cpu and mem usage and send via email? (Default: `True`)

var SEND_ERROR_MAIL Should we send error emails? (Default: `True`)

var SMTP_SERVER SMTP server for sending error emails. (Default: last three sections of the current machine's fully qualified domain name)

var ERROR_MAIL_SENDER Sender address to use for error emails. (Default: `error@gridmap.py`)

var ERROR_MAIL_RECIPIENT Recipient address for error emails. (Default: $USER@$HOST, where $USER is the current user's username, and $HOST is the last two sections of the current machine's fully qualified domain name, or just the hostname if it does not contain periods.)

var MAX_TIME_BETWEEN_HEARTBEATS How long should we wait (in seconds) for a heartbeat before we consider a job dead? (Default: 90)

var IDLE_THRESHOLD Percent CPU utilization (ratio of CPU time to real time * 100) that a process must drop below to be considered not running. (Default: 1.0)

var MAX_IDLE_HEARTBEATS Number of heartbeats we can receive where the process has <= IDLE_THRESHOLD CPU utilization and is sleeping before we consider the process dead. (Default: 3)

var NUM_RESUBMITS How many times can a particular job can die, before we give up. (Default: 3)

var CHECK_FREQUENCY How many seconds pass before we check on the status of a particular job in seconds. (Default: 15)

var HEARTBEAT_FREQUENCY How many seconds pass before jobs on the cluster send back heart beats to the submission host. (Default: 10)

var DEFAULT_PAR_ENV Default parallel environment to use (Default: `smp`)

1.5 `data` Module

This modules provides all of the data-related function for gridmap.

author Christian Widmer

author Cheng Soon Ong

author Dan Blanchard (dblanchard@ets.org)

`gridmap.data.`**`zdumps`**(*obj*)

dumps pickleable object into bz2 compressed string

Parameters **`obj`** (*`object or function`*) – The object/function to store.

Returns An bz2-compressed pickle of the given object.

gridmap.data.**zloads**(*pickled_data*)
loads pickleable object from bz2 compressed string

Parameters **pickled_data** (*bytes*) – BZ2 compressed byte sequence

Returns An unpickled version of the compressed byte sequence.

1.6 job Module

This module provides wrappers that simplify submission and collection of jobs, in a more 'pythonic' fashion.

We use pyZMQ to provide a heart beat feature that allows close monitoring of submitted jobs and take appropriate action in case of failure.

author Christian Widmer

author Cheng Soon Ong

author Dan Blanchard (dblanchard@ets.org)

exception gridmap.job.**DRMAANotPresentException**
Bases: exceptions.ImportError

class gridmap.job.**Job**(*f*, *args*, *kwlist=None*, *cleanup=True*, *mem_free=u'1G'*, *name=u'gridmap_job'*, *num_slots=1*, *queue=u'all.q'*, *interpreting_shell=None*, *copy_env=True*, *add_env=None*, *par_env=u'smp'*, *gpu=0*, *h_vmem=None*, *h_rt=None*, *resources=None*)
Bases: object

Central entity that wraps a function and its data. Basically, a job consists of a function, its argument list, its keyword list and a field "ret" which is filled, when the execute method gets called.

Note: This can only be used to wrap picklable functions (i.e., those that are defined at the module or class level).

---

args

cause_of_death

cleanup

copy_env

environment

execute()
Executes function f with given arguments and writes return value to field ret. If an exception is encountered during execution, ret will contain a pickled version of it. Input data is removed after execution to save space.

function

gpu

h_rt

h_vmem

heart_beat

home_address

host_name

id

interpreting_shell

kwlist

log_stderr_fn

log_stdout_fn

mem_free

name

native_specification
define python-style getter

num_resubmits

num_slots

par_env

path

queue

resources

ret

timestamp

traceback

track_cpu

track_mem

uniq_id

white_list

working_dir

exception gridmap.job.**JobException**
Bases: exceptions.Exception

New exception type for when one of the jobs crashed.

class gridmap.job.**JobMonitor**(*temp_dir=u'/scratch/'*)
Bases: object

Job monitor that communicates with other nodes via 0MQ.

all_jobs_done()
checks for all jobs if they are done

check(*session_id*, *jobs*)
serves input and output data

check_if_alive()
check if jobs are alive and determine cause of death if not

gridmap.job.**grid_map**(*f*, *args_list*, *cleanup=True*, *mem_free=u'1G'*, *name=u'gridmap_job'*, *num_slots=1*, *temp_dir=u'/scratch/'*, *white_list=None*, *queue=u'all.q'*, *quiet=True*, *local=False*, *max_processes=1*, *interpreting_shell=None*, *copy_env=True*, *add_env=None*, *gpu=0*, *h_vmem=None*, *h_rt=None*, *resources=None*, *completion_mail=False*, *require_cluster=False*, *par_env=u'smp'*)
Maps a function onto the cluster.

Note: This can only be used with picklable functions (i.e., those that are defined at the module or class level).

Parameters

- **f** (*function*) – The function to map on args_list
- **args_list** (*list*) – List of arguments to pass to f
- **cleanup** (*bool*) – Should we remove the stdout and stderr temporary files for each job when we're done? (They are left in place if there's an error.)
- **mem_free** (*str*) – Estimate of how much memory each job will need (for scheduling). (Not currently used, because our cluster does not have that setting enabled.)

- **name** (*str*) – Base name to give each job (will have a number add to end)
- **num_slots** (*int*) – Number of slots each job should use.
- **temp_dir** (*str*) – Local temporary directory for storing output for an individual job.
- **white_list** (*list of str*) – If specified, limit nodes used to only those in list.
- **queue** (*str*) – The SGE queue to use for scheduling.
- **quiet** (*bool*) – When true, do not output information about the jobs that have been submitted.
- **local** (*bool*) – Should we execute the jobs locally in separate processes instead of on the the cluster?
- **max_processes** (*int*) – The maximum number of concurrent processes to use if processing jobs locally.
- **interpreting_shell** (*str*) – The interpreting shell for the jobs.
- **copy_env** (*boolean*) – copy environment from master node to worker node?
- **add_env** (*dict*) – Environment variables to add to the environment. Overwrites variables which already exist due to `copy_env=True`.
- **gpu** (*int*) – number of GPUs to request
- **h_vmem** (*str, optional*) – hard virtual memory limit (e.g. "4G")
- **h_rt** (*str, optional*) – hard runtime limit (e.g. "00:59:00")
- **resources** (*list of str, optional*) – list of additional custom resources specifications
- **par_env** (*str*) – parallel environment to use.
- **completion_mail** (*boolean*) – whether to send an e-mail upon completion of all jobs
- **require_cluster** (*bool*) – Should we raise an exception if access to cluster is not available?

Returns List of Job results

`gridmap.job.`**handle_resubmit**(*session_id*, *job*, *temp_dir=u'/scratch/'*)

heuristic to determine if the job should be resubmitted

side-effect: job.num_resubmits incremented job.id set to new ID

gridmap.job.**process_jobs**(*jobs*, *temp_dir=u'/scratch/'*, *white_list=None*, *quiet=True*, *max_processes=1*, *local=False*, *require_cluster=False*)

Take a list of jobs and process them on the cluster.

Parameters

- **jobs** (*list of Job*) – Jobs to run.
- **temp_dir** (*str*) – Local temporary directory for storing output for an individual job.
- **white_list** (*list of str*) – If specified, limit nodes used to only those in list.
- **quiet** (*bool*) – When true, do not output information about the jobs that have been submitted.
- **max_processes** (*int*) – The maximum number of concurrent processes to use if processing jobs locally.
- **local** (*bool*) – Should we execute the jobs locally in separate processes instead of on the the cluster?
- **require_cluster** (*bool*) – Should we raise an exception if access to cluster is not available?

Returns List of Job results

gridmap.job.**send_completion_mail**(*name*)

send out success email

gridmap.job.**send_error_mail**(*job*)

send out diagnostic email

1.7 web Module

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

g

Index