SynergyDRL Debugging in PyCharm

11/11/2020

1) Intro: Why this PDF?

SynergyDRL is based on Softlearning repositories.

This means that experiment are run with Ray library, a library for distributed computing. For example, it allows a training to be executed across a few different computers.

One issue is that when running experiment using Ray, it seems that we lost the ability to debug normally using PyCharm.

In this PDF, I will explain how to debug a program that uses Ray, for example, a training in SynergyDRL.

2) What is the trick to debug in Pycharm when Ray is used?

Tips

1) We need to run the process locally in our machine/PC for debugging.

2) Even if we are already running locally, we still need to set a parameter explicitly to run the process in a single thread.

3) Concretely, when initializing ray, we need to set local_mode=True : ray.init(local_mode=True)

4) Reference: https://groups.google.com/g/ray-dev/c/7euaVHZNhEw

3) How do we make this update in SynergyDRL?

Option 1: Pull from Github (if it is more convenient)

Go to SynergyDRL github page and pull the latest version.

Install the updated requirements, especially the ray version.

If installing all requirements seem troublesome, you need to at least pip install the following:

pip install ray[rllib,debug,tune]==0.8.6

Once you have updated the codes by pulling and install the above ray version, you are ready to debug SynergyDRL in Pycharm.

Please refer to the later slides on how to debug in PyCharm.

Option 2: Update a few codes in your SynergyDRL

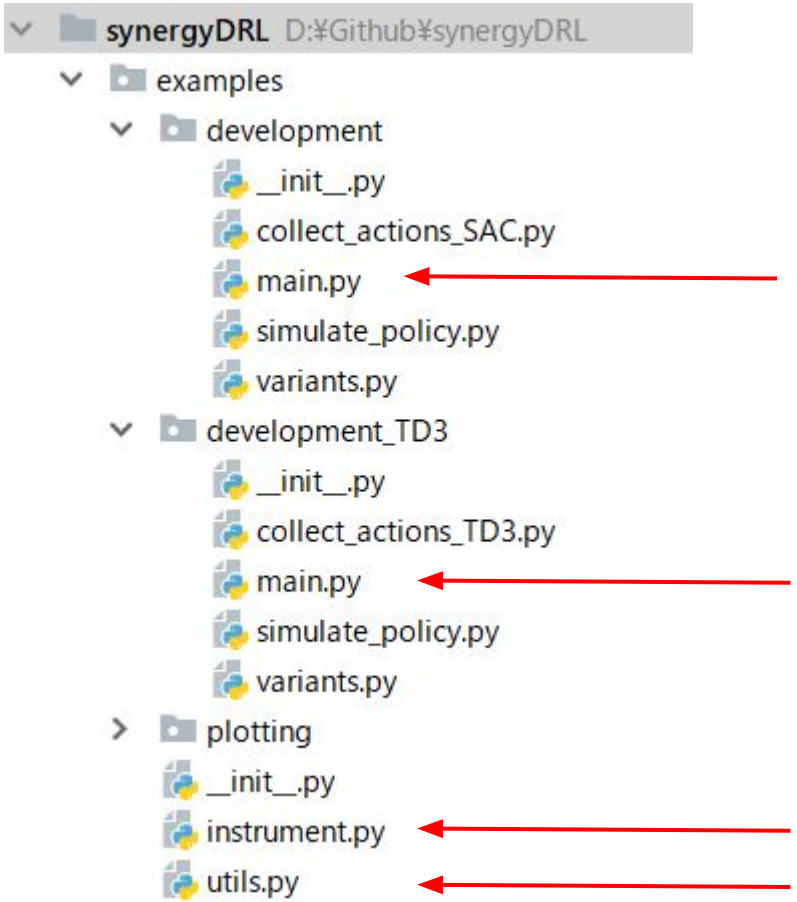
I know that pulling from Github might be troublesome as you may have modified the codes locally in your PC.

You can skip pulling from Github, but instead you can update a few lines of codes by following my instructions in the next few slides.

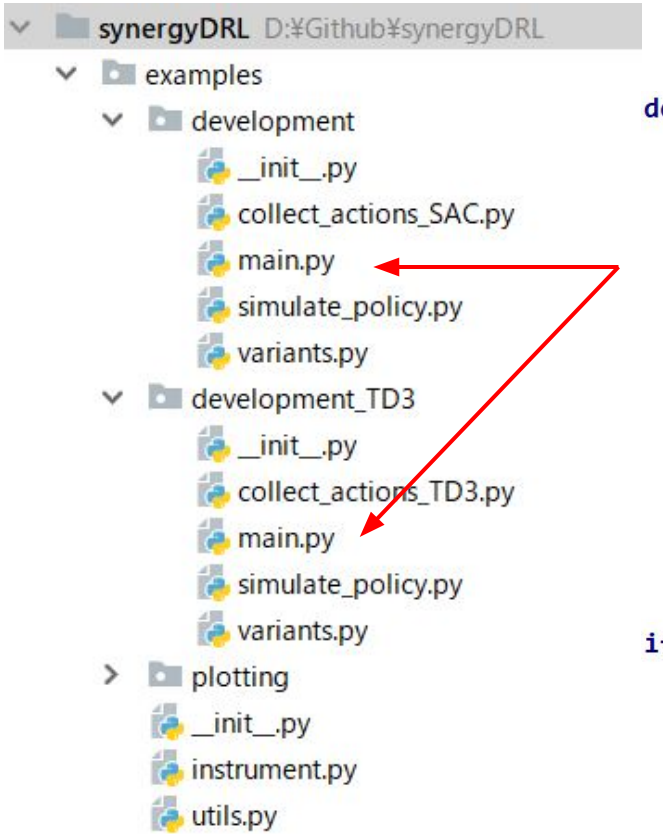
First, update ray:

```
pip install ray[rllib,debug,tune]==0.8.6
```

Option 2: Files involved :



Option 2: Step 1: Modify both main.py

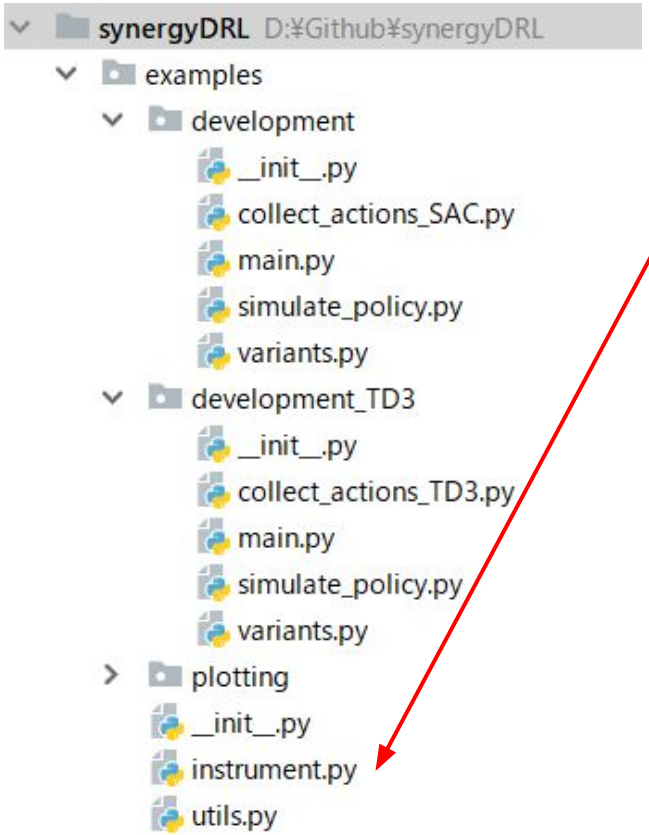



```
def main(argv=None):
    """Run ExperimentRunner locally on ray.

    To run this example on cloud (e.g. gce/ec
    'softlearning launch_example_{gce,ec2} ex

    Run 'softlearning launch_example_{gce,ec2
    instructions.
    """
    # __package__ should be 'examples.develop
    # run_example_local(__package__, argv)
    #
    run_example_local(argv[0], argv[1::])


if __name__ == '__main__':
    main(argv=sys.argv[1:])
```

Comment this

Add this

Option 2: Step 2: Modify instrument.py

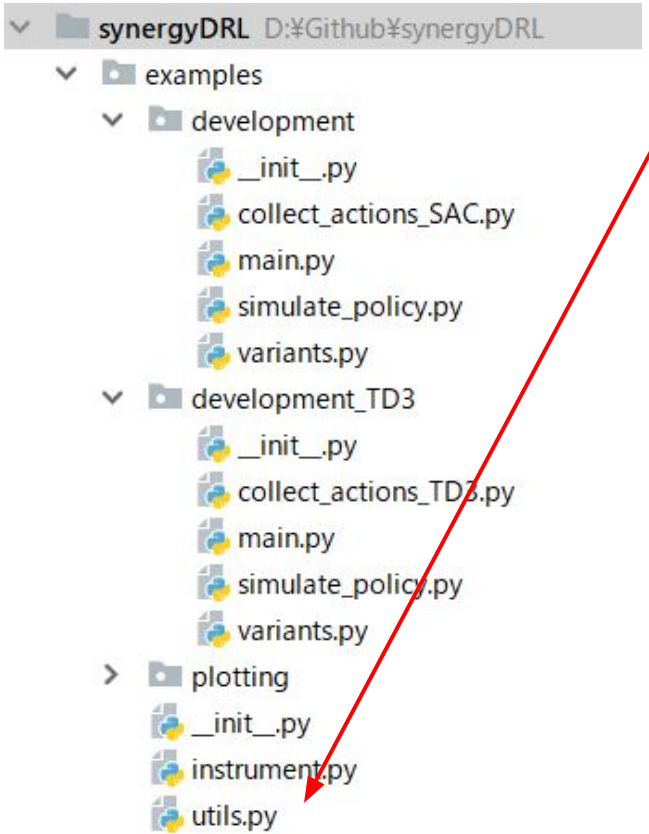



```
def run_example_local(example_module_name,
                      example_argv):



ray.init(
    num_cpus=example_args.cpus,
    num_gpus=example_args.gpus,
    resources=example_args.resources or {},
    # Tune doesn't currently support local mode
    local_mode=example_args.debug,#False
    include_webui=example_args.include_webui,
    temp_dir=example_args.temp_dir)
```

Under this function

Change this line

Option 2: Step 3: Modify utils.py




```
def get_parser(allow_policy_list=False):
```

Under this function

```
    parser.add_argument(
        '--debug', action='store_true')
    parser.add_argument(
        '--gpu_choice', type=int, default=None)

    parser.add_argument(
        '--actor_size', type=int, default=256)
```

Add these lines

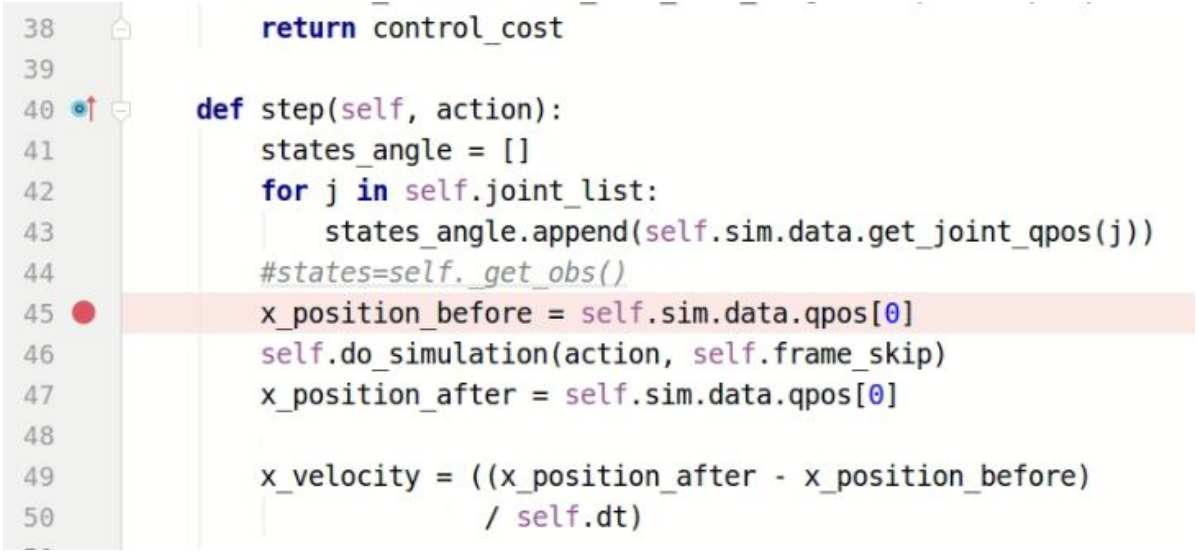
Update completed

Either you updated using Option 1 or 2, now the update is complete and we are ready to debug in PyCharm.

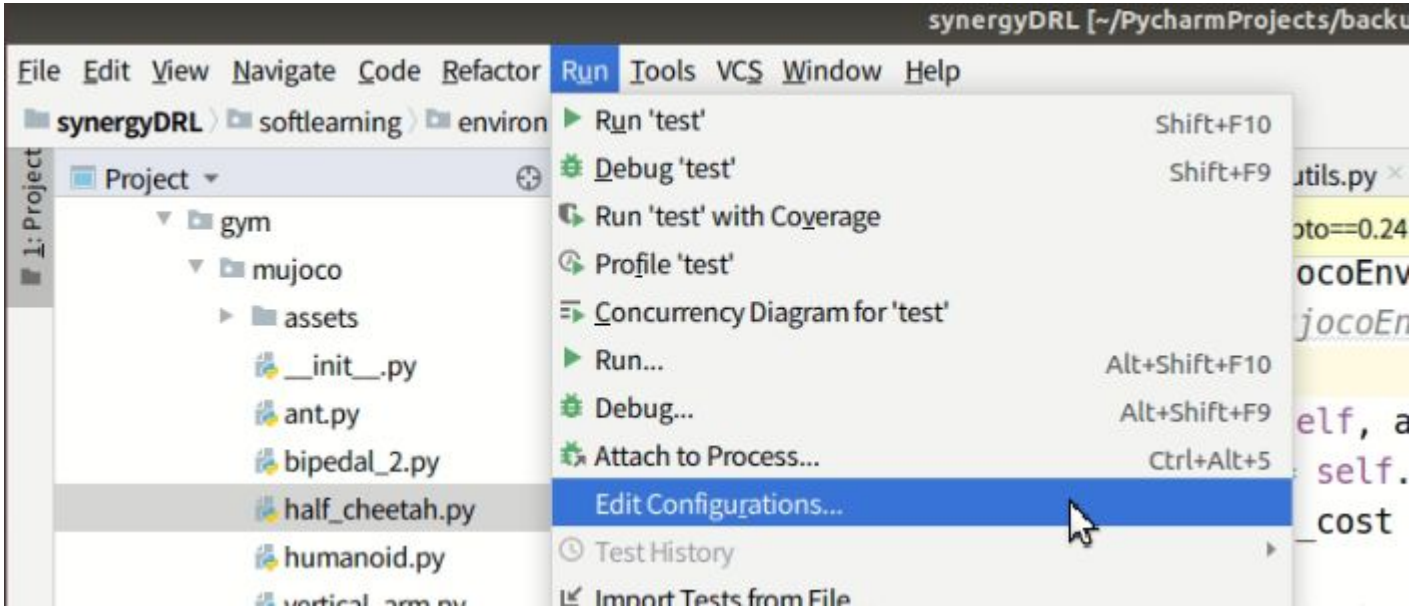
4) Debug in PyCharm

We can now set breakpoint in the program and debug normally.

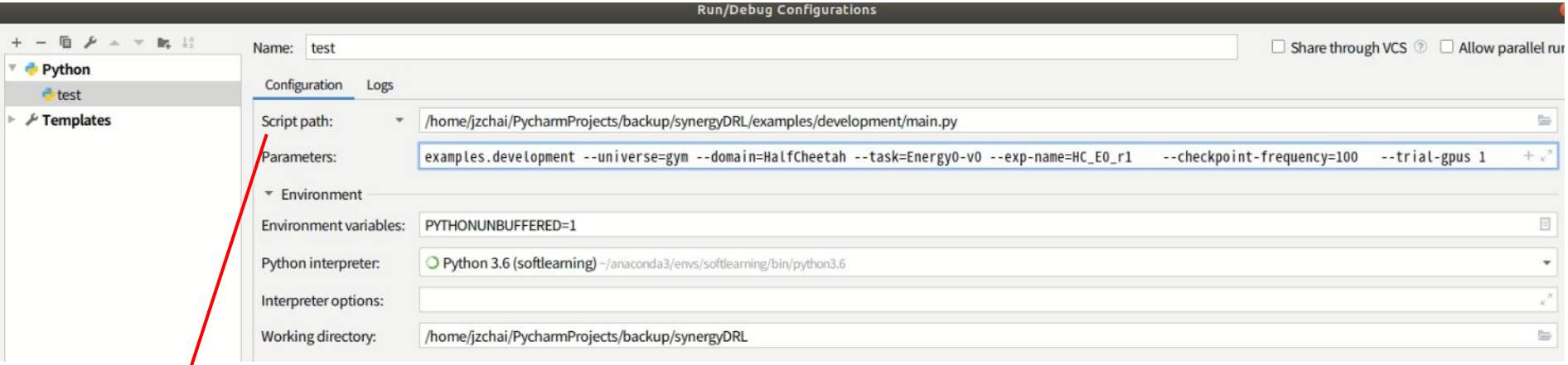
1) Set breakpoints in your programs. For example, in synergyDRL/softlearning/environments/gym/mujoco/HalfCheetah.py



2) Edit configurations in Pycharm.

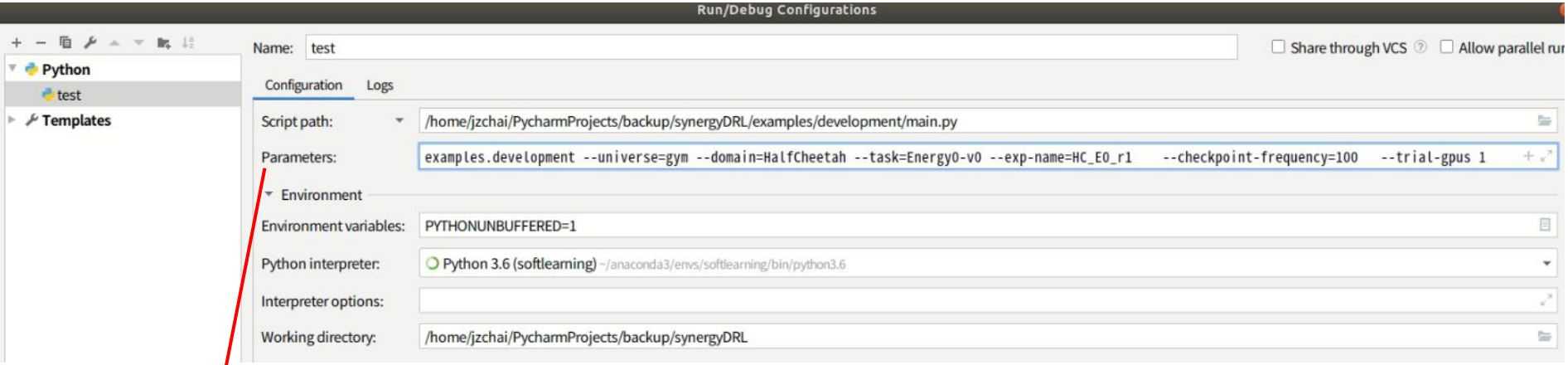


2.1) Edit configurations in Pycharm.



Either development/main.py or development_TD3/main.py

2.2) Edit configurations in Pycharm.

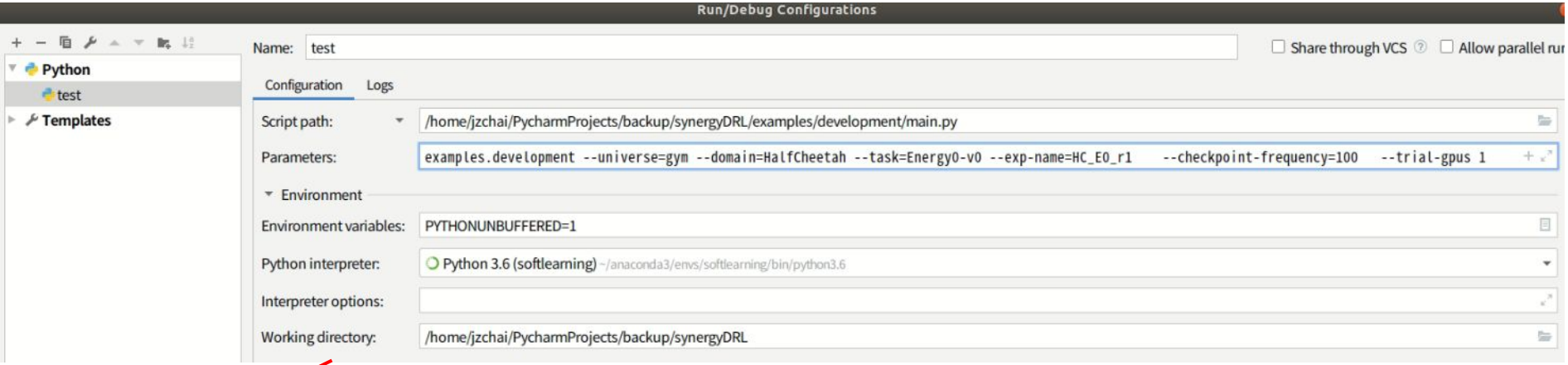


examples.development --universe=gym --domain=HalfCheetah
--task=Energy0-v0 --exp-name=HC_E0_r1 --checkpoint-frequency=100
--trial-gpus 1 --algorithm SAC --debug

This is either examples.development or examples.development_TD3

Write your experiment configurations normally and add this --debug for debugging function.

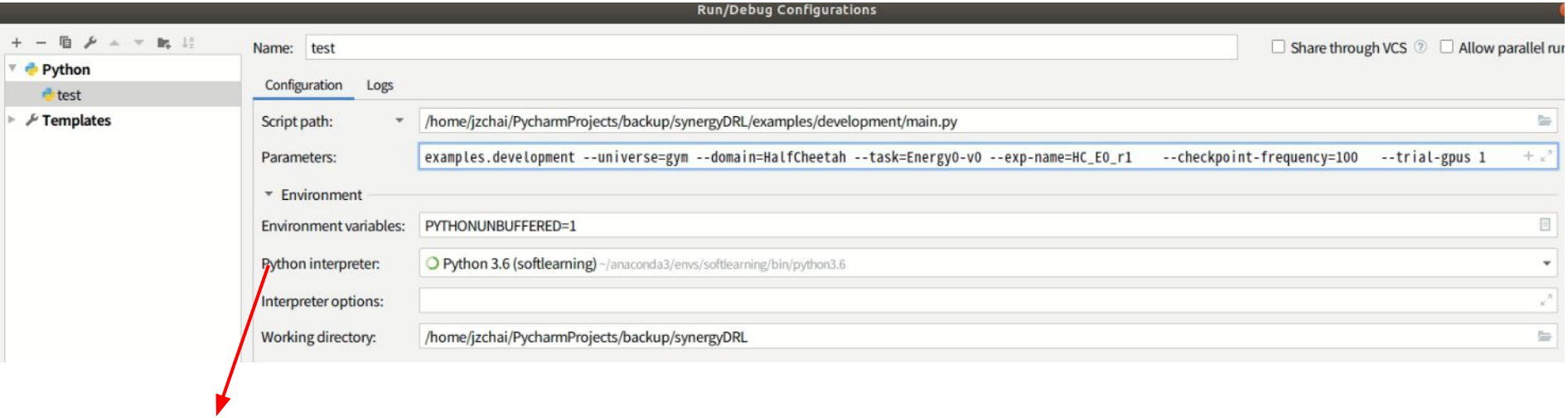
2.3) Edit configurations in Pycharm.



For working directory, I would suggest you to put synergyDRL as the base directory (as shown in the figure).

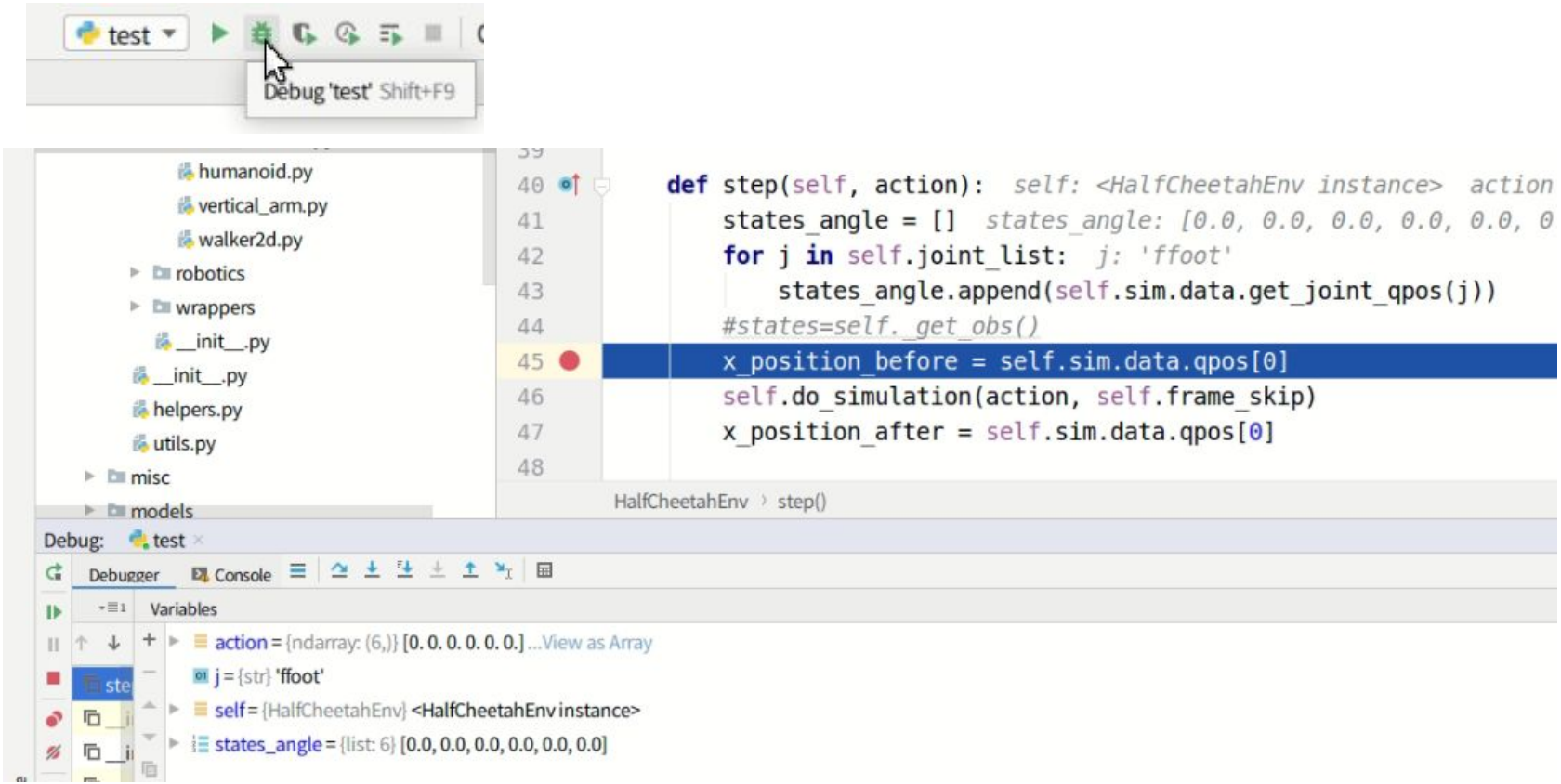
It is also possible to run elsewhere, but maybe your saved files/ outputs will be elsewhere as well. You just need to look for it.

2.4) Edit configurations in Pycharm.



Make sure you choose your correct Python interpreter.

3) Finally, debug:



5) Important Notes

1) In case of errors, make sure you updated the correct ray version.
2) When --debug is written in the configurations, this will make your experiments run locally without GPU. So, don't write this when you are not debugging and when you wish to use GPU.
3) This PDF may contain some imprecision as this is the first version. It will be updated as needed.