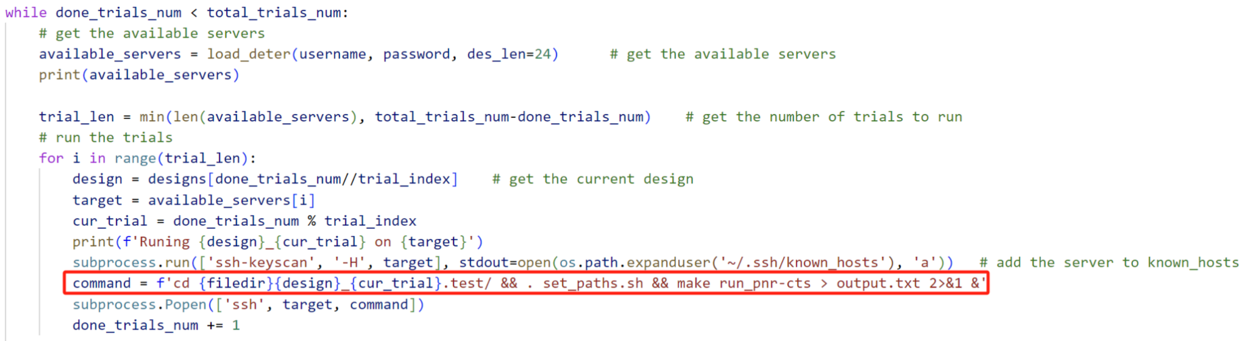
E6693 Design Assignment 3 Report

Zhe Mo, UNI: zm2404

Code Interpretation:

This time, I wrote a Python file to run 24 trials automatically. In the previous assignment, I also attempted to do that, but I failed because I implemented “**. set_paths.sh**” and “**make run_pnr-cts**” in different shell environments, so the tools couldn’t correctly work with tcl files. I implemented them in one call, “**.set_paths.sh && make run_pnr-cts**” this time.



```
while done_trials_num < total_trials_num:
    # get the available servers
    available_servers = load_deter(username, password, des_len=24)      # get the available servers
    print(available_servers)

    trial_len = min(len(available_servers), total_trials_num-done_trials_num)    # get the number of trials to run
    # run the trials
    for i in range(trial_len):
        design = designs[done_trials_num//trial_index]    # get the current design
        target = available_servers[i]
        cur_trial = done_trials_num % trial_index
        print(f'Runing {design}_{cur_trial} on {target}')
        subprocess.run(['ssh-keyscan', '-H', target], stdout=open(os.path.expanduser('~/.ssh/known_hosts'), 'a'))   # add the server to known_hosts
        command = f'cd {filedir}{design}_{cur_trial}.test/ && . set_paths.sh && make run_pnr-cts > output.txt 2>&1 &'
        subprocess.Popen(['ssh', target, command])
        done_trials_num += 1
```

Before distributing the trials to different machines, I collected each machine’s load. I got 24 machines in terms of idleness, or I could get fewer machines and finish trials in many rounds. Since this assignment involved a lot of machine usage, I only assigned one trial to each machine.

The shortcoming of my code is that it doesn’t notify when all trials are done and all metrics are ready to be collected. And also, it doesn’t handle bugs. I think although adding the notification functions and bug handling makes the code functions more complete, it requires a lot of additional work. Thus, I left this part to manual operations.

After all trials were done, I ran the code “**collect_all_trials.py**” to collect all generated metrics and added the cost functions at this step. This step generated an Excel with trials in each group (8 groups in total, 2 weights, 2 designs, 2 cost functions) ranked in order of cost functions. All results are shown in the Excel spreadsheet “**all_trials_metrics.xlsx**”.

And finally, I wrote a Python script, “**copy_file_to_submission.py**”, to copy all necessary files and folders to the folder “DA3_submission”, just for convenience.