User guide

Gerardium Rush

Fluorite team

1. Introduction

Thank you very much for using our tool to determine the most optimal distribution of circuit units for a given number of separators.

This will be a brief explanation of how to use the code, it is a very simple tool, so not a lot of instruction will be required.

2. Installation

In the README of the GitHub repository, an explanation has been provided on how to set up the proper environment and the list of requirements. A series of CMake files have also been included to ensure that the building process works as intended. We have run tests and there should be no issue when compiling or running the code on any platform, even when considering the use of OpenMP. If the proper instructions are followed, the user should not run into any trouble.

3. Usage

Now, for the explanation of the actual code.

First, after the user runs the code executable, the next screen will appear:

```
PS C:\Users\pablo\source\repos\Final_Group_project\out\build\x64-Release\bin> ./Circuit_Optimizer
Flourite Germaninum Rush
Please enter the number of units for the circuit
To use the default value(5 units), please enter 0
```

As can be seen, the user is able to choose the desired number of units to run the simulation on. In this case, we will input 10 units, and then the following screen will appear.

```
PS C:\Users\pablo\source\repos\Final_Group_project\out\build\x64-Release\bin> ./Circuit_Optimizer
Flourite Germaninum Rush
Please enter the number of units for the circuit
To use the default value(5 units), please enter 0
10
Please enter the mode you want to use :
1. HyperPsearch
2. Default
```

The user will now have two choices:

1. A Hyperparameter Search mode, where for the given number of units, the best hyperparameters will be searched.
2. The default mode, where the user can input a personal choice of hyperparameters or use the default ones.

If the user chooses the default mode, the hyperparameters will have to be manually inputted, or the default ones can be used. This can be seen in the following image:

```
PS C:\Users\pablo\source\repos\Final_Group_project\out\build\x64-Release\bin> ./Circuit_Optimizer
Flourite Germaninum Rush
Please enter the number of units for the circuit
To use the default value(5 units), please enter 0
10
Please enter the mode you want to use :
1. HyperPsearch
2. Default
2
Please enter the population size
To use the default value(100), please enter 0
0
Please enter the mutation rate
To use the default value(0.05), please enter 0
0
Please enter the crossover rate
To use the default value(0.90), please enter 0
0
Please enter the generation size
To use the default value(1000), please enter 0
0
Please enter the early stopping generations
To use the default value, please enter 0
0
```

In this case, only the default parameters have been chosen. After this step, the program will start the optimization, and the progress will be shown on screen.

After the optimization has been concluded, the results will be shown on screen:

```
generation: 240
generation: 260
generation: 280
generation: 300
generation: 320
Early stopping in 334 iterations
Best solution found:
9 3 5 10 3 3 9 1 0 0 7 3 2 0 8 0 6 0 11 0 4
Best fitness value is: 375.497
375.497

File opened successfully.
To continue using the program enter any number
To exit, please enter -1
```

As can be seen, the optimized result has been obtained. The user can then input –1 to exit the program, any other number will return the user to the main screen.

If the user had chosen the first option, Hyperparameter Search, the next screen will be shown:

```
PS C:\Users\pablo\source\repos\Final_Group_project\out\build\x64-Release\bin> ./Circuit_Optimizer
Flourite Germaninum Rush
Please enter the number of units for the circuit
To use the default value(5 units), please enter 0
10
Please enter the mode you want to use :
1. HyperPsearch
2. Default
1
Please enter the number of hyperparameter sets you want to use :
5
```

The user must choose how many random sets of hyperparameter parameters will be created, to be compared against each other. In this case, it was selected to generate 5 randoms sets.

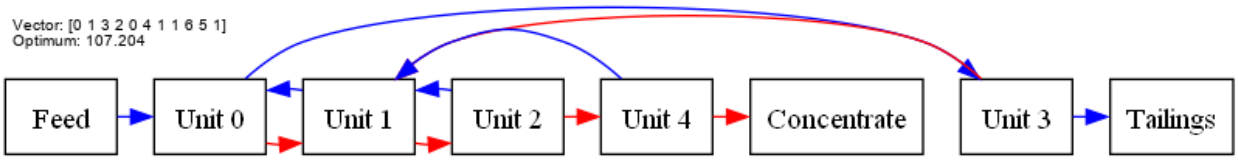
After introducing this input, the optimization will be carried out, and after it has finished the user will receive the results.

```
THE RESULT OF SEARCHING IS:
Best score: 375.497
Best vector: 6 4 11 10 5 4 7 5 8 5 3 1 4 4 2 4 9 5 6 4 0
Best parameters are:
Population size: 108
Mutation rate: 0.0187975
Crossover rate: 0.865488
Mutation_type_rates are Substitutions:  0.867774 Rearrangements: 0.132226
Crossover_type_rates are One-point: 0.376537 Multiple-points: 0.376537 Uniform: 0.246925
File opened successfully.
To continue using the program enter any number
To exit, please enter -1
```

These will be the best hyperparameters found in this case, the user can then exit the program or continue.

4. Postprocessing

When running the main program and finishing the optimization, two files will be generated, **Monetary_Value.txt** and **Circuit_Vector.txt.** The main repository also contains two postprocessing files, one a .py file and one a .ipynb, the user can choose to use a Jupyter Notebook or a normal Python script. Using either of these options, and the two files generated by the optimizer, a graphical representation of the found optimal circuit can be shown. An example is shown next:



It must be considered that it might be necessary for the user to slightly change the .py / .ipynb files, updating the paths to the .txt files loaded in the programs, depending on the operating system the user is working on, but these should be trivial changes.

Also, the user might need to adjust some parameters in the graphical options, as some configuration options might look better when working with a low number of units (5 for example), but not look great when working with 10 or more units (as the image gets more crowded). The way the files are set up, it is trivial for the user to make small graphical changes according to the desired output.