Run Tuning via Command Line or Python

You can run the full tuning process (excluding Bayesian optimization) from the command line:

```
py -3.10 main.py -na
```

This will apply the selected tuning algorithm to all configurable systems. Below is an example of the interactive prompts:

```
global_seed:  42
please set your dataset folder:
- datasets
please set your output folder:
- results\my_search_results
please set your budget:
- 50
please set your search method:
you can choose from: 1.bestconfig_fast 2.fourier 3.random 4.bestconfig 5.flash 6.flash_linear 7.bayesian
- 3
```

Once the search completes, the best configurations and performance results for each system will be shown in both the terminal and saved `.sav` files, for example:

```
System: 7z.csv (Seed: 6677)   Best Config:   [2, 0, 1, 0, 1, 70, 512, 2]   Best Performance: 4751.6
System: Apache.csv (Seed: 368)   Best Config:   [1, 1, 2, 1, 0, 1, 512, 0]   Best Performance: 32.099
...
```

It is recommended to use `lab3_bestconfig_linear` or `lab3_flash_linear`, which often yield better performance under the same budget.

Call Tuning via Python Code

You can also call the tuning process programmatically in your Python code:

```
import main

bestconfig_search(
    datasets_folder="datasets",
    budget=50,
    output_folder="results/my_search_results",
    search_method="bestconfig_linear",  # or "flash_linear", etc.
    global_seed=42
)
```

This provides a flexible way to integrate tuning directly into your own workflow or scripts.