Project Overview

This project provides a framework for performing random search, simulated annealing and improved simulated annealing on configurable systems and visualizing the results. The tool allows users to evaluate configurations across different datasets, identify optimal configurations, and generate visualizations to aid in performance analysis.

Features

- Random search, standard simulated annealing, improved simulated annealing for exploring configurations in datasets.
- Automatic handling of maximization and minimization problems.
- Visualization of search results with performance trends and optimal points.
- Clear organization of datasets, search results, and visualization outputs.

Installation

1. Clone the Repository

```
git clone https://github.com/Co11eague/SAForConfigurationTuning.git
```

2. Install Dependencies

Ensure `pip` is up-to-date and install required packages:

```
pip install --upgrade pip
pip install -r requirements.txt
```

Dataset Description

The `datasets` folder contains CSV files representing different configurable systems. Each CSV file has the following structure:

- **Columns (1 to n-1):** Configuration parameters (discrete or continuous values).
- **Column n:** Performance objective (a numeric value).

Included Systems

System	Optimization Type
7z	Minimization
Apache	Minimization
Brotli	Minimization

System	Optimization Type
LLVM	Minimization
PostgreSQL	Minimization
Spear	Minimization
Storm	Minimization
x264	Minimization

- **Minimization Problems:** Lower performance values are better.
- **Maximization Problems:** Higher performance values are better.

Usage

1. Perform Search

Random Search

Run the RS.py script to perform random search on all datasets:

```
python RS.py
```

Simulated Annealing Search

Run the SA.py script to perform simulated annealing search on all datasets:

```
python SA.py
```

Improved Simulated Annealing Search

Run the ISA.py script to perform improved simulated annealing search on all datasets:

```
python ISA.py
```

The search results for each dataset will be stored in the `search_results` folder as CSV files.

2. Visualize Search Results

Run the visualization script to generate performance plots for all datasets:

```
python visualize_search_results.py
```

The visualizations are automatically generated and stored in the `visualization_results` folder after running script. Each dataset's visualization shows:

- Performance trends across search iterations.
- The optimal performance point highlighted in red *.

3. Execute via IDE (Optional)

If you prefer using an IDE like PyCharm, you can:

- Run `RS.py` to execute the random search process.
- Run `SA.py` to execute the simulated annealing search process.
- Run `ISA.py` to execute the improved simulated annealing search process.
- Run `visualize_search_results.py` to generate visualizations of the results.

4. Customize Search Budget

You can adjust the search budget by modifying the `budget` variable in all of the scripts.

Project Structure

```
project-folder/
  datasets/               # Contains input datasets (CSV files).
  search_results/         # Stores search results after running the script.
  visualization_results/  # Stores visualizations of search results.
  ISA.py                  # Script to run improved simulated annealing search and generate
  SA.py                   # Script to run simulated annealing search and generate results.
  RS.py                   # Script to run random search and generate results.
  requirements.pdf        # Project dependencies requirements.
  manual.pdf              # Project manual.
  replication.pdf         # Project manual for replicating results.
  visualize_search_results.py  # Script for generating visualizations.
  requirements.txt        # Python dependencies.
```

Notes

- For non-existing configurations during the search, the tool assigns:
  - **Minimization Problems:** Twice the maximum performance value in the dataset.
  - **Maximization Problems:** Half the minimum performance value in the dataset.
- Ensure datasets are formatted correctly with valid configuration and performance columns.