AMMM-COURSE PROJECT

Description

The goal of this project is to select faculty members from the university to form a committee to update the curriculum of the computer science degree. The selection process must match department-specific requirements and ensure that the selected members are compatible, according to a compatibility matrix. Furthermore, the average compatibility among selected members should be maximized while satisfying the following constraints.

CPLEX

Files

- **Model File**:
  The model file **project.mod** contains the mathematical model for the optimization problem. It serves as the primary source for the optimization logic.

  ```
  var src = new IloOplModelSource("project.mod");
  ```

- **Data File**:
  The data file **project.1.dat** contains the input parameters for the model. If you want to change the input data for the optimization, you must edit this file accordingly.
  `var data = new IloOplDataSource("project.1.dat");` ### Execution To run the project, use **IBM ILOG CPLEX** to load the model and data files. The program will use **project.mod** as the model source and **project.1.dat** as the input data source.

HEURISTICS

Algorithm Overview

This project provides three different heuristic and meta heuristic algorithms: **Greedy**, **Local Search**, and **GRASP**. The main difference between these algorithms is the selection strategy for forming the committee and the associated parameters.

You can choose which algorithm to use by simply modifying the `algorithm` parameter in the configuration file.

Configuration:

- `algorithm`: Specifies which algorithm to use. Set this to one of the following options:
  - `greedy` — for the Greedy algorithm.

  - `local_search` — for the Local Search algorithm.
  - `grasp` — for the GRASP algorithm.

- **`input_file`**: Path to the input file containing problem instance data. Default value is `data/project.1.dat`
- **`solution_file`**: Path to the output file where the solution will be saved.Default value is `output_greedy.sol`
- **`verbose`**: Verbose mode: set to True for detailed output, False for minimal output. Default value is `True`.

For Greedy and Local Search, the `tune`,the `alpha`,the `max_iterations` and GRASP-specific parameters (like `alpha_start`, `alpha_end`, `alpha_step`) are not required.

GREEDY

Execution:

- Navigate to the **heuristics/** directory:
  `$ cd heuristics/`
- Run the following command `$ python3 heuristic.py --config_file config/greedy.dat`
- The generated instances will be saved in the solutions directory.

GREEDY + LOCAL SEARCH

Execution:

- Navigate to the **heuristics/** directory:
  `$ cd heuristics/`
- Run the following command `$ python3 heuristic.py --config_file config/local_search.dat`
- The generated instances will be saved in the solutions directory. ## GRASP

Configuration

These configurations are specific parameters for the grasp algorithm. - **`alpha_start`**: Starting value of alpha for GRASP tuning

- **`alpha_end`**: Ending value of alpha for GRASP tuning
- **`alpha_step`**: Step size for alpha in GRASP tuning
- **`alpha`**: Fixed alpha value used in GRASP if tuning is not needed
- **`max_iterations`**: Maximum number of iterations for the algorithm
- **`tune`**: Set to **True** to tune the alpha parameter, **False** to use the fixed alpha value

Two Modes of GRASP

1) Fixed Alpha Mode (tune=False):

- If `tune=False`, the algorithm will use the single fixed value for alpha that you provide in the configuration.
- The **alpha** parameter controls the trade-off between greedy (deterministic) choices and random (adaptive) choices in the construction phase. #### 2) Tuning Mode (tune=True):
- If `tune=True`, the algorithm will automatically search for the optimal **alpha** value by trying multiple values in the range from **alpha_start** to **alpha_end** with a step size of **alpha_step**.
- In this case, the **alpha** value is not used directly. Instead, the algorithm will run multiple times, each time with a different value of alpha, and will choose the one that gives the best performance.

Execution

- Navigate to the **heuristics/** directory:
  `$ cd heuristics/`
- Run the following command `$ python3 heuristic.py --config_file config/grasp.dat`
- The generated instances will be saved in the solutions directory. # INSTANCE GENERATOR ## Configuration

The **config.py** file allows you to customize the instance generation process. Below is the default configuration:

```
class Config:
    instancesDirectory = 'output'
    fileNamePrefix = 'project.'
    fileNameExtension = 'dat'
    numInstances = 10
    minDepartments = 2
    maxDepartments = 4
    minMembers = 50
    maxMembers = 70
```

Explanation of Parameters

- **`instancesDirectory`**: Directory where the generated instances will be saved.
- **`fileNamePrefix`**: Prefix for the generated instance file names.
- **`fileNameExtension`**: File extension for the generated instance files.
- **`numInstances`**: Number of instances to generate.
- **`minDepartments / maxDepartments`**: Minimum and maximum number of departments in an instance.
- **`minMembers / maxMembers`**: Minimum and maximum number of members in an instance.

Execution

To run the instance generator, follow these steps: - Navigate to the **instanceGenerator/** directory:
`$ cd instanceGenerator/` - Modify the **config.py** file located under the config directory to adjust the generation parameters.

- Run the following command to generate random instances: `$ python3 main.py` The generated instances will be saved in the output directory.