

Iterative Algorithm for Network Flow

This repository contains the code, data and plots per the paper ***From Minimal Paths to Maximal Cuts: A Dualization-Inspired Multi-State Method***.

The algorithm is implemented in the Python programming language:

- `algorithm.py` - implementation of the algorithm;
- `main.py` - a simple script for running the algorithm;
- `main.ipynb` - a Jupyter Notebook for running the algorithm.

`algorithm.py`

`algorithm.py` implements the algorithm through the class `max_vectors`. The constructor of this class receives only the minimal vectors and the maximal capacities of the edges.

The class implements all the necessary functions internally. After instantiating an object of this class, the maximal vectors are already computed.

To obtain the maximal vectors, call the `get_max_vectors()` method of the object.

For example, if the maximal capacities (\mathcal{C}) and the set of minimal vectors (\mathcal{V}) are:

$$\begin{aligned}\mathcal{C} &= \{2, 2, 2, 2, 1\} \\ \mathcal{V} &= \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}, \\ v_1 &= \langle 2, 2, 0, 0, 0 \rangle, \\ v_2 &= \langle 0, 0, 2, 2, 0 \rangle, \\ v_3 &= \langle 1, 1, 1, 1, 0 \rangle, \\ v_4 &= \langle 2, 1, 0, 1, 1 \rangle, \\ v_5 &= \langle 0, 2, 1, 1, 1 \rangle, \\ v_6 &= \langle 0, 1, 2, 1, 1 \rangle, \\ v_7 &= \langle 1, 0, 1, 2, 1 \rangle\end{aligned}$$

then we pass the following lists to the constructor:

```
capacities = [2, 2, 2, 2, 1]
min_vectors = [
    [2, 2, 0, 0, 0],
    [0, 0, 2, 2, 0],
    [1, 1, 1, 1, 0],
    [2, 1, 0, 1, 1],
    [0, 2, 1, 1, 1],
    [0, 1, 2, 1, 1],
    [1, 0, 1, 2, 1]
]
```

Example execution of the algorithm:

```
from algorithm import maxVectors

if __name__ == '__main__':
    capacities = [2, 2, 2, 2, 1]
    min_vectors = [
        [2, 2, 0, 0, 0],
        [0, 0, 2, 2, 0],
        [1, 1, 1, 1, 0],
        [2, 1, 0, 1, 1],
        [0, 2, 1, 1, 1],
        [0, 1, 2, 1, 1],
        [1, 0, 1, 2, 1]
    ]

    MV = maxVectors(capacities, min_vectors)
    max_vectors = MV.get_max_vectors()

    print('MAX Vectors:')
    for vec in max_vectors:
        print(vec)
```

Output:

```
MAX Vectors:
[0, 1, 1, 2, 1]
[0, 2, 1, 2, 0]
[1, 2, 0, 2, 1]
[0, 2, 2, 1, 0]
[1, 2, 2, 0, 1]
[2, 0, 0, 2, 1]
[2, 0, 1, 2, 0]
[2, 1, 0, 2, 0]
[2, 0, 2, 1, 1]
[2, 1, 2, 0, 1]
```