Neurocomputing 237 (2017) 414–417

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Original Software Publication

LibCoopt: A library for combinatorial optimization on partial permutation matrices

Zhi-Yong Liu[a,c,d,*], Jing-Jing Wang[a], Xu Yang[a], Shi-Hao Feng[b]

[a] State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[b] School of Electronic and Information Engineering, Shanghai Jiaotong University, Shanghai 200240, China
[c] CAS Centre for Excellence in Brain Science and Intelligence Technology (CEBSIT), Shanghai 200031, China
[d] School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100049, China

ARTICLE INFO

Communicated by Zhu Jianke

Keywords:
Combinatorial optimization
Partial permutation matrices
Graduated projection
Deterministic annealing

ABSTRACT

LibCoopt is an open-source matlab code library which provides a general and convenient tool to approximately solve the combinatorial optimization problems on the set of partial permutation matrices, which are frequently encountered in computer vision, bioinformatics, social analysis, etc. To use the library, the user needs only to give the objective function and its gradient function associated with the problem. Two typical problems, the subgraph matching problem and the quadratic assignment problem, are employed to illustrate how to use the library and also its flexibility on different types of problems.

Code metadata

Code metadata description	
Current code version	v0.1.1
Permanent link to code/repository used of this code version	https://github.com/Neurocomputing/NEUCOM-D-16-02113
Legal Code License	MIT license
Code versioning system used	git
Software code languages, tools, and services used	Matlab/Mex
Compilation requirements, operating environments & dependencies	Windows.
If available Link to developer documentation/manual	https://github.com/RowenaWong/libcoopt
Support email for questions	wangjingjing2014@ia.ac.cn

1. Introduction

Combinatorial optimization on partial permutation matrices plays a key role in many computer science problems, such as the subgraph matching problem (*SGM*) and the quadratic assignment problem (*QAP*). These problems are usually NP-hard, and therefore some approximations are necessary for efficiency reasons [1,2]. In literature, these problems were usually solved by different specifically designed methods. In this paper we try to handle these problems from a unified viewpoint. Specifically, the graduated nonconvexity and concavity porcedure (*GNCCP*) [3], proposed by us previously, is adopted as the combinatorial optimization algorithmic framework. An important advantage of *GNCCP* is that only the objective function and its gradient function are involved when it is applied to combinatorial optimization problems. Based on *GNCCP*, in this paper we introduce an open-source matlab code library, LibCoopt, which provides a general and convenient tool for combinatorial optimization on partial permutation matrices. The software is applied to two typical combinatorial optimization problems, *SGM* and *QAP*, to show how to use it, as well as to illustrate its flexibility.

[*] Corresponding author.
E-mail address: zhiyong.liu@ia.ac.cn (Z.-Y. Liu).

http://dx.doi.org/10.1016/j.neucom.2016.11.048
Received 27 June 2016; Received in revised form 14 November 2016; Accepted 24 November 2016
Available online 08 December 2016

2. Problems and background

2.1. Formulation

We consider the combinatorial optimization problem defined by

$$\min_X F(X),\ \text{s. t. } X \in \Pi,\ \Pi:=\left\{X|X_{ij} = \{0, 1\},\ \sum_{j=1}^{N} X_{ij} = 1,\ \sum_{i=1}^{M} X_{ij} \leq 1,\ \forall\ i, j\right\},\ M \leq N, \tag{1}$$

where Π is the set of partial permutation matrices of the size $(M \times N)$. The objective function $F(X)$ is assumed to be differentiable. Such a formulation covers a wide range of important problems, such as *SGM* and *QAP*.

2.2. Background and related works

The LibCoopt has its root in *GNCCP*. The *GNCCP* generalizes the convex-concave relaxation procedure (*CCRP*) [1,4], which involves a linear combination of a convex relaxation and a concave relaxation of the original objective function, and exhibited superior performance on the equal-sized graph matching problem. However, it is not trivial to generalize the *CCRP* to other combinatorial optimization problems because of the difficulties in finding the convex or concave relaxation. It was shown that the *GNCCP* equivalently realizes a type of *CCRP* on partial permutation matrices, but in a much simpler way without explicitly involving the convex or concave relaxation.

To use *GNCCP*, Π is firstly relaxed to its convex hull Ω:=$\{\mathbf{X}|\mathbf{X}_{ij} \geq 0, \sum_{j=1}^{N} \mathbf{X}_{ij} = 1, \sum_{i=1}^{M} \mathbf{X}_{ij} \leq 1, \forall\ i, j\}$. Then the *GNCCP* takes the following form

$$F_\zeta(\mathbf{X}) = \begin{cases} (1-\zeta)F(\mathbf{X}) + \zeta \mathrm{tr}\mathbf{X}^T\mathbf{X} & \text{if } 1 \geq \zeta \geq 0, \\ (1+\zeta)F(\mathbf{X}) + \zeta \mathrm{tr}\mathbf{X}^T\mathbf{X} & \text{if } 0 > \zeta \geq -1, \end{cases},\ \mathbf{X} \in \Omega. \tag{2}$$

The algorithmic framework for *GNCCP* is given by Algorithm 1. In the algorithm, the gradient $\nabla F_\zeta(\mathbf{X})$ takes the following form

$$\nabla F_\zeta(\mathbf{X}) = \begin{cases} (1-\zeta)\nabla F(\mathbf{X}) + 2\zeta\mathbf{X} & \text{if } 1 \geq \zeta \geq 0, \\ (1+\zeta)\nabla F(\mathbf{X}) + 2\zeta\mathbf{X} & \text{if } 0 > \zeta \geq -1. \end{cases} \tag{3}$$

Algorithm 1. Algorthrithmic framework of *GNCCP*.

```
$\zeta \leftarrow 1, X \leftarrow X^0$
while $\zeta > -1 X \in \Pi$ do
    while $X$ not convergedo
        $Y = \arg\min_Y tr\nabla F_\zeta(X)^T Y$, s. t. $Y \in \Omega$.
        $\alpha = \arg\min_\alpha F_\zeta(X + \alpha(Y - X))$, s. t. $\alpha \in [0, 1]$
        $x \leftarrow X + \alpha(Y - X)$
    end while
    $\zeta \leftarrow \zeta - d\zeta$
end while
```

As shown in Algorithm 1, the *GNCCP* involves only the objective function ($F_\zeta(X)$) and its gradient ($\nabla F_\zeta(X)$). Therefore it provides a unified framework for quite a lot of combinatorial optimization problems on partial permutation matrices as long as the objective function is differentiable. Based on *GNCCP*, the LibCoopt is introduced below.

3. Software architecture and implementation

The architecture of LibCoopt is shown in Fig. 1. For different combinatorial optimization problems on partial permutation matrices, LibCoopt provides an interface to input both the objective function and its gradient function. The two functions take the problem related data as input, while LibCoopt itself does not directly face the data. It is in this sense that LibCoopt is claimed to be a general and convenient tool for combinatorial optimization on partial permutation matrices.

LibCoopt is mainly implemented by Matlab script, with some computationally intensive parts implemented by Mex files. Currently only the Windows operation system based version is provided.

The core Matlab function is

```
Solution = Coopt(@F, @nF, Data, Para)
```

where `Solution` is the final combinatorial optimization solution including the minimal point, objective value, and running time. The first two inputs `@F` and `@nF` are the function handles of the customized objective function and its gradient function. The third input `Data` is the problem related data. And `Para` is the parameter structure.

To show how to use LibCoopt in a specific problem, the objective functions and their gradient functions of two typical combinatorial optimization problems, i.e. *SGM* and *QAP*, are provided in the library. Before introducing the corresponding Matlab functions, some brief preliminaries are given below. For the adjacency matrix based *SGM* model (denoted by *GMAD*) [3], the objective function is

$$\text{GMAD: } \min F(X) = \|A_M - XA_D XT\|_F^2 \quad \text{s. t.}\quad X \in \Pi, \tag{4}$$

where A_M and A_D denote the adjacency matrices associated with the two input graphs. For the affinity matrix based *SGM* model (denoted by *GMAF*) [5], the objective function is

$$\text{GMAF: } \max F(X) = vec(X)^T A vec(X) \quad \text{or} \quad \min F(X) = vec(X)^T K vec(X) \\ \text{s. t. } X \in \Pi, \tag{5}$$

where A is a $MN \times MN$ affinity matrix encoding the edge similarities between graphs, and similarly K denotes the dissimilarity matrix which can be directly obtained by $K = -A$. We use the minimization problem based on K in this paper. For *QAP* [3], the objective function is

$$\text{QAP: } F(X) = tr(AXB^T X^T) \quad \text{s. t. } X \in \Pi(M = N) \tag{6}$$

where A and B are two equal-sized matrices.

In LibCoopt, for the objective function and gradient function of *GMAD*, the corresponding Matlab functions are `F_GMAD(X,Data)` and `nF_GMAD(X,Data)`. For *GMAF*, they are `F_GMAF(X,Data)` and `nF_GMAF(X,Data)`, and for *QAP*, they are `F_QAP(X,Data)` and `nF_QAP(X,Data)`.

The users may directly try `run_Coopt_GMAD(DataPath)`, `run_Coopt_GMAF(DataPath)`, and `run_Coopt_QAP(DataPath)` to test LibCoopt on these problems, where `DataPath` is the path of a sample data. These functions call `Coopt` in similar ways, and provide data preprocessing and other specific processing for different problems. Users can use data in folder `ToyData` for testing. More demos and description can be found at https://github.com/RowenaWong/libcoopt.

Moreover, by specifying the files in the folder `Other`, LibCoopt can be also applied to other combinatorial optimization problem as long as it can be formulated into a differentiable objective function on partial permutation matrices. Taking the traveling salesman problem (*TSP*) for example, it can be formulated by

$$F(X) = tr(FXD^T X^T) \text{ s. t. } X \in \Pi\ (M = N), \tag{7}$$

where D is the distance matrix and F is a constant matrix defined by

$$F_{ij} = \begin{cases} 1 & \text{if } j = i + 1 \text{ or } i = N, j = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

This formulation is similar to *QAP* and the derivation of its gradient is straightforward. Thus by formulating *TSP* in this way, LibCoopt is applicable to it.