Generic Constraint-based Block Modeling using Constraint Programming*

Alex Mattenet[1], Ian Davidson[2], Siegfried Nijssen[1], and Pierre Schaus[1]

[1] UCLouvain, ICTEAM, Belgium
{alex.mattenet,siegfried.nijssen,pierre.schaus}@uclouvain.be
[2] Computer Science Department, University of California at Davis, USA
davidson@cs.ucdavis.edu

This is an extended abstract of an earlier publication at CP2019 [3].

1 Introduction

Block modeling is a problem that originates from the analysis of social networks. The core problem is to take a graph and divide its vertices into k clusters, in such a way that vertices in the same cluster have the same pattern of ties to other vertices. These clusters and the interactions between them summarize the graph and give insight into its large-scale structure.

More formally, in its simplest formulation, the core problem is: given a graph $G(V, E)$ whose $n \times n$ adjacency matrix is X, simplify X into a symmetric trifactorization FMF^t. Here F is an $n \times k$ block allocation matrix with the blocks/clusters stacked column wise. Here $F_{i,j} \in \{0, 1\}$ and M is a $k \times k$ image matrix showing the interaction between blocks. The objective function is to minimize the reconstruction error $||X - FMF^t||$.

As this formulation is very generic, it is useful to incorporate additional constraints such as bounds on the cluster size, constraints on the structure of the image graph M (forcing it to be a tree, a ring graph,...), constraints on the composition of the clusters, and more. This allows combining strong semantic knowledge (the constraints) along with empirical evidence (the graph).

Existing approaches for solving this problem with additional constraints focus on one or two constraint at a time and are not composable. It is impossible to use all of multiple constraints at the same time, and they are either not usable or not scalable without the predefined constraints.

In this paper, we propose a Constraint Programming approach for the block modeling problem, with an efficient dedicated global constraint. By using a CP framework, our approach is inherently composable with any of the constraints implemented in the CP solver. Furthermore, the CP solver can be used both to find exact solutions, and to find approximations using Large Neighborhood Search. We compare our CP approach with existing exact methods and local search methods, and show that it outperforms the state of the art.

* Alex Mattenet is supported by a FRIA grant.

2 Constraint Programming Approach

A CP model is composed of variables and constraints on those variables. There are four groups of variables in our model: (1) n cluster variables C, where $\mathsf{C}_i = c$ if vertex i is in cluster c; (2) k^2 image matrix variables M, where $\mathsf{M}_{cd} = 0$ if the image graph has no arc from cluster c to cluster d, and 1 if it is has such interactions. (3) k^2 variables cost, where cost_{cd} is the number of errors in block $c, d : X_{ij} \neq \mathsf{M}_{cd}$ with $\mathsf{C}_i = c$ and $\mathsf{C}_j = d$ (4) $\mathsf{totalCost}$, which is $||X - FMF^t||$.

The variables are subject to the following constraints: (1) $\mathtt{sum}(\mathsf{cost}, \mathsf{totalCost})$, which ensures that the total cost of the solution and the individual cost of every block stays consistent. (2) $\mathtt{atLeast}(1, \mathsf{C}, c), \forall c \in \{1..k\}$, which ensures that there are no empty clusters. (3) $\mathtt{blockModelCost}(X, \mathsf{M}, \mathsf{C}, \mathsf{cost}, \mathsf{totalCost})$. This is the global constraint that we add to the solver, which filters the values of the different variables along the search. It ensures $\sum_{i=1}^{n} \sum_{j=1}^{n} |X_{ij} - M_{C_i C_j}| \leq \mathsf{totalCost}$ and $\sum_{i=1}^{n} \sum_{j=1}^{n} (C_i = c) \cdot (C_j = d) \cdot |X_{ij} - M_{cd}| \leq \mathsf{cost}_{cd}\ \forall c, d$.

The pseudocode of `blockModelCost`'s filtering algorithm is given in the full paper, along with a discussion of its complexity. We present search ordering heuristics and symmetry-breaking schemes. The model is then implemented in the OscaR CP solver. With this framework, is easy to add additional constraints.

3 Evaluation and Results

We compare the performance of our CP approach to the current state of the art for exact solving of the Block Modeling problem, which uses a MIP model [2]. We measure the run time to completion on well-studied datasets from the literature, and show that our approach is order of magnitudes better.

We also compare the performance of a popular local search algorithm for Block Modeling bundled in the Pajek software [1] with a Large Neighborhood Search using our CP model. We measure the evolution of the objective function with respect to time on synthetic datasets with up to 200 vertices (Pajek's limit). For all instances over 50 vertices, the LNS method outperformed Pajek's search.

Finally, we explore the scalability of our method on graphs of up to 7000 vertices and we illustrate possible uses of additional constraints with applications to the analysis of global human migration.

References

1. Batagelj, V., Mrvar, A., Ferligoj, A., Doreian, P.: Generalized blockmodeling with pajek. Metodoloski zvezki **1**(2), 455 (2004)
2. Dabkowski, M., Fan, N., Breiger, R.: Exploratory blockmodeling for one-mode, unsigned, deterministic networks using integer programming and structural equivalence. Social Networks **47**, 93–106 (Oct 2016)
3. Mattenet, A., Davidson, I., Nijssen, S., Schaus, P.: Generic constraint-based block modeling using constraint programming. In: Principles and Practice of Constraint Programming - 25th International Conference (CP) (2019)