Metaheuristically Optimized Multicriteria Clustering for Medium-scale Networks

David Chalupa, Jiří Pospíchal

Abstract We present a highly scalable metaheuristic approach to complex network clustering. Our method uses a multicriteria construction procedure (MCP), controlled by adaptable constraints of local density and local connectivity. The input of the MCP - the permutation of vertices, is evolved using a metaheuristic based on local search. Our approach provides a favorable computational complexity of the MCP for sparse graphs and an adaptability of the constraints, since the criteria of a ”good clustering” are still not generally agreed upon in the literature. Experimental verification, regarding the quality and running time, is performed on several well-known network clustering instances, as well as on real-world social network data.

1 Introduction

Analysis of complex networks is a very lively area of inter-disciplinary research. In physics, complex networks are studied as dynamic systems, with respect to the processes of their growth, evolution and their structure [12, 22]. In computer science, they are related to a large spectrum of practical applications, including data mining [18], web engineering [19] or bioinformatics [1, 6]. Additionally, many popular complex networks emerge from the analysis of current trends on the Internet, e.g. social networks [3] and research citation networks [18].

These networks generally tend to have clustered structures. A *cluster* can be informally described as a group of similar entities in the network, which generally tend to create densely connected areas. A problem of searching for a decomposition of the network into clusters is often modeled using graph theory and therefore, it is referred to as *graph clustering* [17].

David Chalupa, Jiří Pospíchal
Institute of Applied Informatics, Faculty of Informatics and Information Technologies, Slovak University of Technology, Ilkovičova 3, 842 16 Bratislava, Slovakia, e-mail: chalupa@fiit.stuba.sk

There are numerous applications of graph clustering. In this paper, we focus mostly on the applications in *social networks*, in which it is often referred to as *community detection* [12]. These applications include marketing, recommendation, personalization, media analysis and human resource management [13, 17]. Quite a large attention is drawn also by the structure of the networks of terrorist organizations [13]. Clustering of social networks have also been studied in the context of epidemiology of sexually transmitted diseases [14]. Other applications include grouping of gene expressions in bioinformatics, most notably in protein interaction [6] and gene-activation dependencies [1].

Unfortunately, due to the less formal nature of graph clustering, there exist many different approaches, how to define, compute and evaluate clusters and clustering. It is difficult to find out, how suitable a formulation is, since there is no metric of quality, which is generally agreed to be the most reliable, while being computationally tractable [17]. Therefore, graph clustering is an interesting application for soft computing, where one can use the emergence of augmented clustering not only to optimize but also to observe the nature of the problem formulation. In this work, we are dealing with these issues and come up with the concept of multicriteria constrution procedures (MCPs), which encapsulate the selected criteria, while the optimization of the input to the MCP is performed with a general metaheuristic.

In the terms of graph theory, a clustering of an undirected graph $G = [V, E]$ can be formalized as a set $S = \{V_1, V_2, ..., V_k\}$ of disjoint subsets of V often called *classes*, i.e. $\forall i = 1..k\ V_i \subset V$. Let $d = \frac{|E|}{|V|(|V|-1)/2}$ be the *density* of the graph to cluster. Then, the subgraphs $G(V_i)$ induced by the classes V_i ($\forall i = 1..k$) should be more dense than the graph itself, i.e. $\forall i = 1..k\ d(G(V_i)) > d(G)$. The values $d(G(V_i))$ will be referred to as the *intra-cluster densities*. An important fact is that this condition is very similar to the formalization of graph coloring and the clique covering problems [10].

The paper is organized as follows. Section 2 provides an overview of the topic and the related work. In Section 3, we propose the concept of MCPs and the metaheuristically optimized multicriteria clustering based on MCPs. In Section 4, we provide the experimental results of our approach. Finally, in Section 5, we summarize the work.

2 Background and Related Work

Let $\overline{G}$ be the complementary graph to G. In the graph coloring problem, the objective is to minimize the number of colors k for classes $S = \{V_1, V_2, ..., V_k\}$, where $\forall i = 1..k\ d(G(V_i)) = 0$. The minimal value of k, for which this is possible, is called chromatic number and denoted as $\chi(G)$. Clearly, $\forall\ V_i \in S\ d(\overline{G}(V_i)) = 1$. The problem to obtain this partitioning will be referred to as the clique covering problem, which is NP-hard [10]. It can be seen quite easily that clique covering is a special case of graph clustering, where each cluster is a clique. On the other hand, the most trivial constraint that $\forall i = 1..k\ d(G(V_i)) > d(G)$ often leads to trivial solutions, which should be avoided. Therefore, the formulation of the criteria for graph clustering is understood as an issue of searching for a balance between this formulation on one hand and the clique covering on the other [17]. Regarding the complexity of the problems, it is worth mentioning that not only the clique covering but also many of the meaningful graph clustering quality measures are known to be NP-hard or NP-complete [5, 21]. Nevertheless, all these problems are related to the structure of the graph. Although this is difficult to formalize, we can assume that the more asymmetrical the graph is, the more information is hidden in the structure to guide an optimization algorithm.

Regarding the relevant algorithms, *hierarchical clustering* uses a selected similarity measure and either repeatedly divides the graph or merges some partial clusters. In this sense, we will refer to either divisive or agglomerative clustering. In hierarchical clustering, a tree of candidate clusters, called dendrogram, is used [17]. Another approach is represented by the *spectral methods*, using the eigenvalues of the graph's adjacency matrix, which is diagonalized and the vertices are reordered so that the vertices in the same clusters are next to each other [17]. *Local search methods* are used for the problem as well, however, they use the procedure to find a single cluster for a vertex, which is then used as a seed [16]. A representative of the *visual and geometric methods* is the spring force algorithm, which is often used to visualize clustered graphs and thus also solves the problem in graphical representation [17].

In the clique covering problem, chromatic number of the complementary graph can be estimated in polynomial time by *sequential greedy heuristics*, e.g. the well-known Brélaz's heuristic [2]. Another approach is the *local search*, representing a very large class of algorithms based on iterative improving of current solutions. The basic local search algorithm, hill climbing, uses elementary changes of solutions to improve them [15]. Popular stochastic extensions of hill climbing include the simulated annealing [20, 11] and tabu search [8].

3 The Proposed Approach

In this section, we describe the concept of MCPs and propose an MCP based on local densities of clusters and local connectivities of their vertices as the criteria for the solution construction. Then, we describe the metaheuristic, which is used to optimize the input of MCPs, i.e. the permutation of vertices.

3.1 The Criteria for Graph Clustering

The objective of our approach to graph clustering is to minimize the number of clusters in a clustering S of a graph G: min $k = |S|$; $S = \{V_1, V_2, ..., V_k\}$, subject to the following global constraints:

1. Each vertex is clustered and the clusters are non-overlapping: $[V_1 \cup V_2 \cup \ldots \cup V_k] = V \wedge [V_1 \cap V_2 \cap \ldots \cap V_k] = \emptyset$.
2. The clusters are more dense than the whole graph: $\forall i = 1..k\ d(G(V_i)) > d(G)$.

Furthermore, we consider the following local constraints, which are dynamic, i.e. they are influenced by the current state during the construction of the clustering, rather than remaining static for the whole procedure:

3. The relative connectivity of a vertex to be newly added to the cluster must be higher than its relative connectivity to the residual, currently non-clustered subgraph: $\frac{w_c}{|V_{c,i}|} > \frac{\delta_r}{|V_r| - 1}$, where $V_{c,i}$ is the set of vertices in cluster c at the iteration i of the MCP, w_c is the number edges, brought into the cluster by the vertex to be newly added and $|V_r|$ and δ_r are the number of vertices and the degree of the newly added vertex in the subgraph containing only the currently non-clustered vertices.
4. If there are more candidate clusters, the one with highest connectivity is taken: $c = \arg\max_c \frac{w_c}{|V_{c,i}|}$, where for the cluster c, $\frac{w_c}{|V_{c,i}|}$ must be a feasible value, according to the previous rule.
5. The vertex to be newly added must bring at least as many edges, as is the current average intra-cluster degree in the particular cluster, while a small tolerance τ may be sometimes allowed: $w_c + \tau \geq \frac{2|E_{c,i}|}{|V_{c,i}|}$, where $E_{c,i}$ is the number of edges in $G(V_{c,i})$.

Criteria 1 and 2 are implied from the basic properties a good clustering should fulfill. Criterion 3 is used to verify, whether it is favorable to add v_j to a cluster or to rather create a new cluster and let some of the currently unclustered vertices join it. Criterion 4 is used to solve the situations, when more clusters fulfil criterion 3. Finally, criterion 5 is used to ensure a relative uniformity of intra-cluster vertex degrees, which is important in order to avoid situations, when several small clusters are unnecessarily joined. Parameter τ plays an important role here, since $\tau = 0$ leads to clusters with very uniform intra-cluster degrees, while $\tau > 0$ is favorable for clusters with stronger centrality.

3.2 The Multicriteria Construction Procedures (MCPs)

The general framework for an MCP is described in the pseudocode of Algorithm 1. As the input, we have a permutation of vertices. In the step 2, we take the current vertex v_j from the permutation. In the step 3, we apply the criteria to choose a label c, thus, joining v_j to the corresponding cluster in the step 4. The step 5 is used to update the auxiliary data specified in Section 3.1, which are needed to implement the multicriteria cluster choice efficiently. This procedure is repeated, until all vertices are clustered.

Algorithm 1: A General Framework for an MCP

```
A General Framework for an MCP
  Input: graph $G = [V, E]$
  permutation $P = [P_1, P_2, ..., P_{|V|}]$ of vertices
  Output: a clustering $S$ of $G$
1 for $i = 1..|V|$
2     $j = P_i$
3     $c = find\_cluster(v_j)$
4     $V_c = V_c \cup \{v_j\}$
5     $update\_auxiliary\_data(V_c)$
6 return $S = \{V_1, V_2, ..., V_k\}$
```

It is important that all the criteria are formulated in the way that during the implementation, one can verify each of the criteria only by scanning the neighbors of the currently chosen vertex. This restriction leads to an $\mathcal{O}(\delta)$ average complexity per iteration of an MCP, where δ is the average degree of a vertex in the graph. Let v_j be the chosen vertex, c the chosen cluster and let $V_{c,i}$ and $E_{c,i}$ be the vertex and edge set of cluster c at the i-th iteration of an MCP. Then, w_c will be the number of edges brought into the cluster by v_j, counted by scanning of the neighbors of v_j.

Using this notation, we will describe MCP-DC as our construction algorithm for graph clustering. It uses the 5 criteria, as they have been described in the previous section.

The implementation of the $find\_cluster$ procedure is as follows. One can easily derive that the local density needed in criterion 2 is fulfilled if:

$$d(G)|V_{c,i}|(|V_{c,i}|+1) - 2|E_{c,i}| - 2w_c < 0. \qquad (1)$$

The local connectivity in criterion 3 is fulfilled if the following holds:

$$|V_{c,i}| - w_c \frac{V_r - 1}{\delta_r} < 0. \qquad (2)$$

The maximization of the connectivity in criterion 4, i.e. the ratio $\frac{w_c}{|V_{c,i}|}$, can be implemented simultaneously with criterion 3, since the necessary values are calculated in the verification of criterion 3. Finally, the criterion 5 yields the following condition, where $\tau \geq 0$ is a parameter of tolerance for the intra-cluster degree of the newly added vertex:

$$\frac{2|E_{c,i}|}{|V_{c,i}|} - \tau - w_c \leq 0. \qquad (3)$$

These observations lead to a construction algorithm, in which the output depends on the permutation of vertices and, according to the following theorem, the number of iterations is proportional to the number of edges.

Theorem 1. *MCP-DC can be implemented to run in $\mathcal{O}(\delta|V|) = \mathcal{O}(|E|)$ time.*

Proof. $|V_{c,i}|$ and $|E_{c,i}|$ can be trivially recalculated in $\mathscr{O}(1)$ time per iteration. The previous formulations of the MCP-DC criteria can be implemented by iterative subtracting of a constant (in the cases of criteria 2 and 5) or the ratio $\frac{V_r - 1}{\delta_r}$ (in the case of criterion 3) from the respective values. Explicit storage of values w_c yields the same for criterion 4. Restoration of the former values after subtraction can be done by simulating the inverse process. All these operations need $\mathscr{O}(\delta)$ average time per iteration, thus, they lead to an $\mathscr{O}(\delta|V|) = \mathscr{O}(|E|)$ running time of MCP-DC. □

3.3 The Metaheuristic for Optimization of the Permutation of Vertices in MCPs

The proposed criteria indicate that we are facing a highly constrained problem. On the other hand, encapsulation of the constrained part in the MCP leads to two major advantages in the optimization. First, for each permutation, there exists a clustering, which will be constructed by an MCP. Secondly, it was confirmed in our experiments that this formulation does not tend to create hard multimodal functions. In fact, on real-world data, we were able to optimize the permutation using a simple local search metaheuristic.

The metaheuristic we used, begins with a random permutation, which can be generated in place in $\mathscr{O}(|V|)$ time [4]. The initial clustering is constructed using an MCP. Then, at each iteration of local search, we try a single random vertex exchange in the permutation and evaluate the new number of clusters using the MCP. The new permutation is accepted if and only if, for the new permutation P' leading to k' clusters, it holds that $k' \leq k$, where k is the current number of clusters. The local search is stopped when the number of iterations without improvement exceeds certain threshold. We denote this as s_{max}.

The choice of this simple metaheuristic was influenced by three factors. The first reason is the nature of the landscape, which we indicated above. The second factor is the $\mathscr{O}(|E|)$ complexity of the objective function - MCP-DC, where each redundant run would significantly increase the global running time. Finally, the third factor is that the general stochastic extensions, such as tabu search [8], do not tend to improve the performance on this type of landscapes. It could perhaps be useful to find a heuristic to guide the algorithm to choose the right. However, we have tried several extensions, e.g. a roulette wheel selection of vertices according to their degrees but the results did not show any improvement.

4 Experimental Evaluation

In this section, we present the experimental evaluation of our approach. First, we visually illustrate the emergence of good clustering using our approach on a real-world sample from a social network. Then, we provide computational results on several instances obtained by MCP-DC and MCP-DC with the metaheuristic. Last but not least, we measure the running times for different network sizes.

4.1 The Emergence of Good Clustering

Fig. 1 illustrates the process of optimization using the local search algorithm and MCP-DC on a small instance of real-world social network data, where in each picture, the vertices in the same cluster are grouped together. This network will be referred to as Social network I. In this case, with MCP-DC and the metaheuristic, we achieved 5 highly relevant communities. The drawings visually indicate this emergence of clustered structure, where the evolution is driven only by random exchanges of vertices.

4.2 Computational Results

To evaluate our algorithm, we used it to solve the problem in two well-known benchmarks - Zachary karate club [23] and the American college football network [7]. We also used instances from two social networks, where the data from Social network II was obtained using a web crawler. We also used a graph generated by an artificial model [3]. Table 1 summarizes the computational results obtained in 10 independent runs, s_{max} is the maximal allowed number of iterations without improvement and τ is the intra-cluster degree tolerance factor. The primary criterion was k - the number of clusters. We also measured the average number of iterations and the time needed to obtain the clustering.

Zachary karate club is known to consist of two partitions, we show the two partitions found by our algorithm in Fig. 2. Fig. 2 also shows the result obtained for the American college football network, which is known to consist of 10 conferences. In both cases, our algorithm found the clusters reliably. Fig. 3 shows results on the

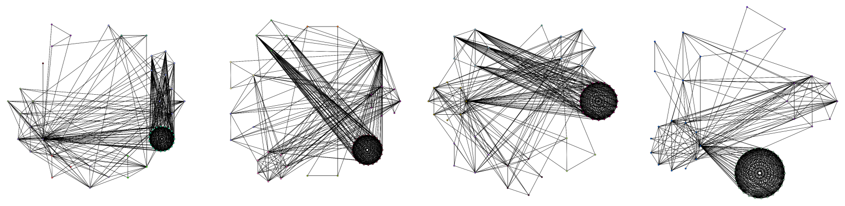

Fig. 1 An illustration of the emergence of gradually better clustering using the metaheuristic optimization and MCP-DC. The drawings are done after 0, 100, 1000 and 10000 iterations (from left to right). The number of clusters is optimized from 12 to 5, where the 5 clusters are highly relevant for the data.

Table 1 Comparison of the results obtained by only MCP-DC and MCP-DC with the metaheuristic (MCP-DC+MH). The metrics are k - the number of clusters, the number of local search iterations and the running time.

source	$\|V\|, \|E\|$	s_{max}	τ	MCP-DC	MCP-DC+MH		
				k	k	iter.	time
Zachary karate club [23]	$34, 78$	5×10^3	1	7 - 15	2	7035	< 1 s
American college football [7]	$115, 615$	10^6	0	18 - 23	10 - 12	1237965	252 s
Social network I	$52, 830$	5×10^4	0	12 - 16	5 - 6	76194	9 s
Social network II	$500, 924$	5×10^4	1	161 - 197	12 - 15	154964	71 s
Artificial model [3]	$500, 3536$	5×10^4	0	68 - 79	55 - 60	163449	188 s

social networks, where the clustering of the smaller network was verified manually, while the relevance of the clustering of the larger network is indicated by the sparseness in between. What is perhaps less visible in this drawing, is that the presence of hubs is very pronounced in these clusters. We note that we did not provide a numerical metric of quality (e.g. the Adjusted Rand Index [9]) due to exorbitant space, which results obtained using such metrics, together with precise analysis would require.

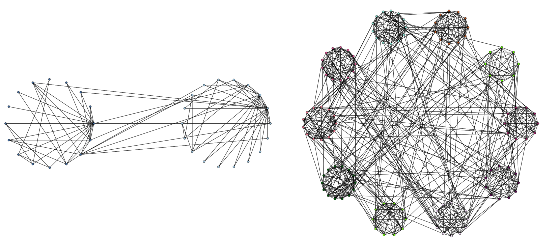
Fig. 2 Visualizations of results, obtained by our approach for the benchmark data: a clustering of the Zachary karate club into 2 communities (left) and a clustering of American college football league, which is known to consist of 10 conferences (right).

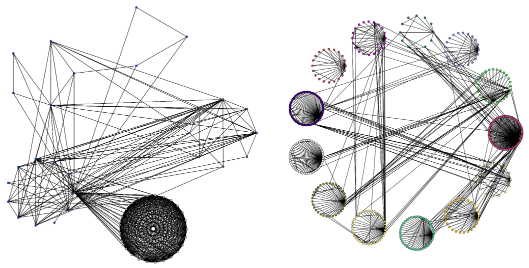
Fig. 3 Visualizations of results, obtained by our approach for the social network data: locally extracted sample from Social network I (left) and data obtained by crawling Social network II, which contains more pronounced hubs (right).

4.3 The Running Time

We have shown that MCP-DC has an $\mathscr{O}(|E|)$ complexity, which is favorable for sparse graphs. In these experiments, we measure the time practically needed by the MCP-DC with the metaheuristic. We used the crawler of Social network II to obtain networks of different sizes, up to 10000 vertices. In our algorithm, we set $s_{max} = 2 \times 10^4$ and $\tau = 1$.

Table 2 The times needed to obtain the clustering on samples from Social network II with different sizes.

$\|V\|$	500	1000	2000	10000
$\|E\|$	924	1876	4247	28675
k	14 - 19	26 - 38	68 - 82	449 - 453
iter.	75363	97621	177622	484703
time	36 s	1 m 34 s	6 m	102 m 49 s

Table 2 contains the results we obtained. First, we can see that the constant value of s_{max} causes that the number of iterations grows only moderately. This factor, and the slow growth of $|E|$, which is typical for most complex networks, implies that the computational time does not grow exponentially. To be fair, although solid suboptimal results can be achieved even with smaller values of s_{max}, the current form of our approach is suitable mostly for medium-scale instances (around 10^3 vertices). However, we believe that the adaptability our approach maintains, is a key to solid scalability also for very large graphs.

5 Conclusion

We presented the concept of multicriteria construction procedures (MCPs) for network clustering. In this context, we designed MCP-DC - an MCP using the criteria of local density and local connectivity. In our approach, the input of MCP-DC - the permutation of vertices, is optimized using a general metaheuristic. This makes it useful not only as a graph clustering algorithm but also as a tool to discover pros and cons of the criteria, which are used to construct the clustering, since these are still not generally agreed upon in the literature.

Our approach was verified on well-known benchmarks for graph clustering, as well as on samples obtained from real-world social networks. These experiments showed much promise both in relevance of results and scalability of the approach.

Acknowledgements This contribution was supported by Grant Agency VEGA SR under the grant 1/0553/12.

References

1. F. Boyer, A. Morgat, L. Labarre, J. Pothier, and A. Viari. Syntons, metabolons and interactons: an exact graph-theoretical approach for exploring neighbourhood between genomic and functional data. *Bioinformatics*, 21(23):4209–4215, 2005.
2. D. Brélaz. New methods to color vertices of a graph. *Communications of the ACM*, 22:251–256, 1979.
3. D. Chalupa. On the Ability of Graph Coloring Heuristics to Find Substructures in Social Networks. *Information Sciences and Technologies, Bulletin of ACM Slovakia*, 3(2):51–54, 2011.
4. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms (3rd ed.).* MIT Press, 2009.
5. P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering in large graphs and matrices. *Machine Learning*, 56:9–33, 2004.
6. L. Gao, P. Sun, and J. Song. Clustering algorithms for detecting functional modules in protein interaction networks. *J. Bioinformatics and Computational Biology*, 7(1):217–242, 2009.
7. M. Girvan and M. E. J. Newman. Community structure in social and biological networks. In *Proceedings of the National Academy of Sciences, USA 99*, page 82718276, 2002.
8. F. Glover. Tabu search - part i. *INFORMS Journal on Computing*, 1(3):190–206, 1989.
9. L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.
10. R. M. Karp. Reducibility among combinatorial problems. In *Proceedings of a Symposium on Complexity of Computer Computations*, pages 85–103, 1972.
11. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.
12. M. E. J. Newman. Detecting community structure in networks. *The European Physical Journal B*, 38:321–330, 2004.
13. J. Pattillo, N. Youssef, and S. Butenko. Clique Relaxation Models in Social Network Analysis. *Handbook of Optimization in Complex Networks*, pages 143–162, 2012.
14. R. B. Rothenberg, J. J. Potterat, and D. E. Woodhouse. Personal Risk Taking and the Spread of Disease: Beyond Core Groups. *The Journal of Infectious Diseases, Supplement 2*, 174:S144–S149, 1996.
15. S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach (2nd ed.).* Upper Saddle River, New Jersey: Prentice Hall, 2003.
16. S. E. Schaeffer. Stochastic local clustering for massive graphs. In *PAKDD*, pages 354–360, 2005.
17. S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.
18. J. Sun, Y. Xie, H. Zhang, and C. Faloutsos. Less is more: Sparse graph mining with compact matrix decomposition. *Stat. Anal. Data Min.*, 1:6–22, February 2008.
19. J. Tang, T. Wang, J. Wang, Q. Lu, and W. Li. Using complex network features for fast clustering in the web. In *Proceedings of the 20th international conference companion on World wide web*, WWW '11, pages 133–134, New York, NY, USA, 2011. ACM.
20. V. Černý. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45:41–51, 1985.
21. J. Šíma and S. E. Schaeffer. On the np-completeness of some graph cluster measures. In *Proceedings of the 32nd International Conference on Current Trends in Theory and Practice of Computer Science, LNCS 3831*, Berlin, Heidelberg, Germany, 2006. Springer Verlag GmbH.
22. D. J. Watts. *Small Worlds.* Princeton University Press, 1999.
23. W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33:452–473, 1977.