Springer Proceedings in Mathematics & Statistics

Valery A. Kalyagin
Petr A. Koldanov
Panos M. Pardalos *Editors*

Models, Algorithms and Technologies for Network Analysis

NET 2014, Nizhny Novgorod, Russia, May 2014

Springer

Springer Proceedings in Mathematics & Statistics

Volume 156

More information about this series at http://www.springer.com/series/10533

Springer Proceedings in Mathematics & Statistics

This book series features volumes composed of select contributions from workshops and conferences in all areas of current research in mathematics and statistics, including OR and optimization. In addition to an overall evaluation of the interest, scientific quality, and timeliness of each proposal at the hands of the publisher, individual contributions are all refereed to the high quality standards of leading journals in the field. Thus, this series provides the research community with well-edited, authoritative reports on developments in the most exciting areas of mathematical and statistical research today.

Valery A. Kalyagin • Petr A. Koldanov
Panos M. Pardalos
Editors

Models, Algorithms and Technologies for Network Analysis

NET 2014, Nizhny Novgorod, Russia, May 2014

Springer

Editors
Valery A. Kalyagin
Laboratory of Algorithms and Technologies for Network Analysis (LATNA)
National Research University
Higher School of Economics
Nizhny Novgorod, Russia

Petr A. Koldanov
Laboratory of Algorithms and Technologies for Network Analysis (LATNA)
National Research University
Higher School of Economics
Nizhny Novgorod, Russia

Panos M. Pardalos
Department of Industrial and System Engineering
Center for Applied Optimization
University of Florida
Gainesville, FL, USA

ISSN 2194-1009 ISSN 2194-1017 (electronic)
Springer Proceedings in Mathematics & Statistics
ISBN 978-3-319-29606-7 ISBN 978-3-319-29608-1 (eBook)
DOI 10.1007/978-3-319-29608-1

Library of Congress Control Number: 2016937007

Mathematics Subject Classification (2010): 90-02, 90C31, 90C27, 90C09, 90C10, 90C11, 90C35, 90B06, 90B10, 90B15, 90B18, 90B40, 90B80, 68R01

© Springer International Publishing Switzerland 2016
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made.

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG Switzerland

Preface

This volume is a collective monograph with different contributions. Many of them are related with the 4th International Conference on Network Analysis held in Nizhny Novgorod, Russia, May 11–13, 2014. The main focus of these contributions is the development of modern approaches for network analysis with different applications. The previous books which cover the similar topics are: [1–4].

According to the covered topics, the volume can be divided into three parts: optimization in networks, network data mining, and economics and other applications.

The first part deals with optimization problems in networks. In the chapter "Maximally Diverse Grouping and Clique Partitioning Problem with Skewed General Variable Neighborhood Search," a new variant of variable neighborhood search referred to as skewed general variable neighborhood search is used to solve a maximally diverse grouping problem and a clique partitioning problem. Extensive computational results show that the developed heuristic significantly outperforms its competitors.

In the chapter "Test Generation for Digital Circuits Based on Continuous Approach to Circuit Simulation Using Different Continuous Extensions of Boolean Functions," the analysis of continuous extensions of Boolean functions for test generation using continuous optimization is conducted. It represents the results of the developed software for a number of ISCAS schemes.

In the chapter "Konig Graphs for 4-Paths II: Wided Cycles," characterization of the graphs, whose each induced subgraph has the property that the maximum number of induced 4-paths is equal to the minimum cardinality of the set of vertices such as every induced 4-path contains at least one of them, is given. All such graphs obtained from simple cycles by replacing some vertices into cographs are described.

In the chapter "Optimization Algorithms for Shared Groups in Multicast Routing," the multicast group routing problem is considered. The problem requires the construction of one or more routing trees such that each destination has its demand satisfied by one or more data sources. The problem can be viewed as a generalization of the multicast routing problem with a single data source. This problem has important applications in the design of collaborative communication networks, among other uses. While the problem is NP-hard, it is possible to develop algorithms

for its solution that approximate the solution in practice. Existing techniques for solving multicast group routing problems are discussed. Some fast heuristics for this problem are proposed. It is shown that computational experiments support the quality of the results achieved by these algorithms.

In the chapter "Minimizing the Fuel Consumption of a Multiobjective Vehicle Routing Problem Using the Parallel Multi-Start NSGA II Algorithm," a new multiobjective formulation of the vehicle routing problem (VRP), the multiobjective fuel consumption vehicle routing problem (MFCVRP), using two different objective functions is presented. The first objective function corresponds to the optimization of the total travel time and the second objective function is the minimization of the fuel consumption of the vehicle taking into account the travel distance, the load of the vehicle, and other route parameters. Two cases of the MFCVRP are solved.

In the chapter "Manifold Location Routing Problem with Applications in Network Theory," the problem of determining locations of facilities and the item distribution to customers from these facilities from supply chain networks is considered. Determination of the facility locations is a well-known problem in the literature, named facility location problem (FLP). The distribution of items via vehicles is known as the VRP. It is emphasized that solving VR and FL problems simultaneously can yield a robust facility location and reduced-cost distribution within the customer-supplier network.

In the chapter "A Branch and Bound Algorithm for the Cell Formation Problem," the cell formation problem (an NP-hard optimization problem) is considered for cell manufacturing systems. A branch and bound algorithm which provides exact solutions of the cell formation problem is proposed.

The second part of the book is devoted to data mining in networks. In the chapter "Hybrid Community Detection in Social Networks," several ideas to design hybrid methods for community detection are discussed.

In the chapter "Spectral Properties of Financial Correlation Matrix," the random matrix theory (RMT) is applied to investigate the cross-correlation matrix of a financial time series in four different stock markets: Russian, American, German, and Chinese. The deviations of distribution of eigenvalues of market correlation matrix from RMT global regime are investigated. Specific properties of each market are observed and discussed.

In the chapter "Statistical Uncertainty of Minimum Spanning Tree in Market Network," the procedure for the minimum spanning tree construction as a multiple decision statistical procedure is considered. The statistical uncertainty of the procedure is investigated.

In the chapter "Uncertainty of Identification of Cliques and Independent Sets in Pearson and Fechner Correlations Networks," two market network models for the analysis of NYSE stock market are used: Pearson correlation network and Fechner correlation network. The problem of estimation of statistical uncertainty of identification of maximum cliques and maximum independent sets in Pearson and Fechner correlation networks is considered. It is shown that identification of

maximal cliques and maximal independent sets in Fechner correlation network is distribution-free in a certain class of distributions which is not true for Pearson correlation network.

In the chapter "Investigation of Connections Between Pearson and Sign Correlations in Market Network," the connection between the sign correlation of Fechner and the classic Pearson correlation is investigated. Testing hypothesis of the connection by real market data is conducted.

In the chapter "Testing the Stationarity of Sign Coincidence in Market Network," a hypothesis of the stationarity of observations from financial market are considered. For testing the stationarity hypotheses, the Kolmogorov–Smirnov test and multiple comparisons Bonferroni and Holm procedures are applied. The stationarity hypothesis is tested for sign coincidence of returns by binomial proportions. It is shown that the null hypothesis of stationarity is rejected for prices and not rejected for returns and their sign coincidence on some significance level.

In the chapter "Synchronization and Network Measures in a Concussion EEG Paradigm," the neurophysiologic changes in a patient who suffered a concussion during a football practice via quantitative and graph theoretic measures and to evaluate the results with respect to the preconcussion state are characterized. The deviations in the selected quantitative and graph theoretic measures partially corroborate the usual clinical characteristics of the post-concussion state, but further investigation for additional data and evaluation of alternative quantitative measures is needed.

In the chapter "Video-Based Pedestrian Detection on Mobile Phones with the Cascade Classifiers," the problem of real-time pedestrian recognition on mobile phones is discussed. A specialized procedure of data gathering and preprocessing to train cascade classifiers is proposed. Experimental results in testing under real road conditions with several mobile phones are given. It is emphasized, that sometimes it is necessary to choose faster, but less accurate, object detection algorithm, because in this case it is possible to process more number of frames in a fixed period of time. Hence, the total object detection accuracy can be increased.

In the chapter "Clustering in Financial Markets," graph partition of a particular kind of complex networks referred to as power law graphs is considered. In particular, analysis on the market graph, constructed from time series of price return on the American stock market, is conducted. Two different methods originating from clustering analysis in social networks and image segmentation are applied to obtain graph partitions, and the results are evaluated in terms of the structure and quality of the partition. It is shown that the market graph possesses a clear clustered structure only for higher correlation thresholds. Partitions for different time series are considered to study the dynamics and stability in the partition structure.

The third part contains applications of network analysis. In the chapter "Key Borrowers Detected by the Intensities of Their Short-Range Interactions," the issue of systemic importance that an individual financial institution can disturb the whole financial system is discussed. Interconnectedness is considered as one of the key drivers of systemic importance. Several measures have been proposed in the literature in order to estimate the interconnectedness of financial institutions and

systems. However, they do not fully take into consideration an important dimension of this characteristic: intensities of agent interactions. This paper proposes a novel method that solves this issue. The approach is based on the power index and centrality analysis and is employed to find a key borrower in a loan market. The approach is applied at the European Union level.

In the chapter "A Semantic Solution for Seamless Data Exchange in Supply Networks," the semantic solution for data exchange in dynamically changing supply networks is proposed. Despite the dynamic nature of supply networks, there is a necessity to manage their efficiency. The first step in this direction is proper data exchange, which leads to transparency of networks and speeds up interactions of their participants. Despite plenty of data standards, there is a lack of approaches to data modeling that allowed seamless knowledge sharing within supply networks. This paper introduces a new ontology-based data metamodel of supply networks based on organizational ontology, consistent theory for data modeling, and the ontologized standard in logistics domain.

In the chapter "Langmuir Solitons in Plasma with Inhomogeneous Electron Temperature and Space Stimulated Scattering on Damping Ion-Sound Waves," an analytical solution for solitons in plasma is obtained in an approximate form. It is shown that analytical and numerical results have a good agreement.

In the chapter "Equilibria in Networks with Production and Knowledge Externalities," a game equilibrium is investigated in a network in each node of which an economy is described by the simple two-period model of endogenous growth with production and knowledge externalities. Each node of the network obtains an externality produced by the sum of knowledge in neighbor nodes. Uniqueness of the inner equilibrium is proved. Three ways of behavior of each agent are distinguished: active, passive, and hyperactive. Behavior of agents in dependence on received externalities is studied. It is shown that the equilibrium depends on the network structure.

We would like to take the opportunity to thank all the authors and referees for their efforts to contribute the chapters. In addition, we would like to thank Springer for giving us the opportunity for this work. This work is partly supported by Russian Federation Government grant N. 11.G34.31.0057.

Nizhny Novgorod, Russia — Valery A. Kalyagin
Nizhny Novgorod, Russia — Petr A. Koldanov
Gainesville, FL, USA — Panos M. Pardalos

References

1. Goldengorin, B.I., Kalyagin, V.A., Pardalos, P.M. (eds.): Models, Algorithms and Technologies for Network Analysis. In: Proceedings of the First International Conference on Network Analysis. Springer Proceedings in Mathematics and Statistics, vol. 32. Springer, Cham (2013)

2. Goldengorin, B.I., Kalyagin, V.A., Pardalos, P.M. (eds.): Models, Algorithms and Technologies for Network Analysis. In: Proceedings of the Second International Conference on Network Analysis. Springer Proceedings in Mathematics and Statistics, vol. 59. Springer, Cham (2013)
3. Batsyn, M.V., Kalyagin, V.A., Pardalos, P.M. (eds.): Models, Algorithms and Technologies for Network Analysis. In: Proceedings of Third International Conference on Network Analysis. Springer Proceedings in Mathematics and Statistics, vol. 104. Springer, Cham (2014)
4. Kalyagin, V.A., Pardalos, P.M., Rassias, T.M. (eds.): Network Models in Economics and Finance. Springer Optimization and Its Applications, vol. 100. Springer, Cham (2014)

Contents

Contributors

Habib Abdulrab INSA de Rouen, LITIS Lab., Rouen, France

Fuad Aleskerov DeCAn Lab, Institute of Control Sciences of Russian Academy of Sciences, National Research University (NRU) Higher School of Economics, Moscow, Russia

Elena Andreeva National Research University Higher School of Economics, Nizhny Novgorod, Russia

Irina Andrievskaya LIA Lab, National Research University (NRU) Higher School of Economics, Moscow, Russia

N.V. Aseeva National Research University Higher School of Economics, Nizhny Novgorod, Russia

Eduard Babkin National Research University Higher School of Economics, Nizhny Novgorod, Russia

Mikhail Batsyn Laboratory of Algorithms and Technologies for Networks Analysis, National Research University Higher School of Economics, Nizhny Novgorod, Russia

Jack Brimberg Royal Military College of Canada, Kingston, ON, Canada

Lei Cui Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA

Gianluca Del Rossi University of South Florida, Tampa, FL, USA

Ding-Zhu Du Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA

Hongwei Du Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China

E.M. Gromov National Research University Higher School of Economics, Nizhny Novgorod, Russia

Joseph Gutmann University of South Florida, Tampa, FL, USA

Valery A. Kalyagin National Research University Higher School of Economics, N. Novgorod, Russia

Nickolay Kascheev Faculty of Business Informatics and Applied Mathematics, National Research University Higher School of Economics, Nizhny Novgorod, Russia

Daniil Kascheev Faculty of Business Informatics and Applied Mathematics, National Research University Higher School of Economics, Nizhny Novgorod, Russia

Maxim Kazakov National Research University Higher School of Economics, N. Novgorod, Russia

Petr Koldanov National Research University Higher School of Economics, Nizhny Novgorod, Russia

Alexander P. Koldanov Lab LATNA, National Research University Higher School of Economics, Nizhny Novgorod, Russia

Anastasia Komissarova National Research University Higher School of Economics, Nizhny Novgorod, Russia

Alexei Korolev National Research University Higher School of Economics, St. Petersburg, Russia

Oleg Kremnyov National Research University Higher School of Economics, Nizhny Novgorod, Russia

Andrey Latyshev National Research University Higher School of Economics, Nizhny Novgorod, Russia

John Lloyd University of South Florida, Tampa, FL, USA

Magdalene Marinaki School of Production Engineering and Management, Technical University of Crete, Chania, Greece

Yannis Marinakis School of Production Engineering and Management, Technical University of Crete, Chania, Greece

Vladimir Matveenko National Research University Higher School of Economics, St. Petersburg, Russia

Athanasios Migdalas Department of Civil Engineering, Aristotle University of Thessalonike, Thessalonike, Greece

Industrial Logistics, Lulea University of Technology, Lulea, Sweden

Nenad Mladenović University of Valenciennes, Valenciennes, France

Mathematical Institute SANU, Belgrade, Serbia

Dmitry Mokeev Laboratory of Algorithms and Technologies for Networks Analysis, National Research University Higher School of Economics, Nizhny Novgorod, Russia

Dmitry E. Mozokhin National Research University Higher School of Economics, Nizhny Novgorod, Russia

T.V. Nasedkina National Research University Higher School of Economics, Nizhny Novgorod, Russia

Carlos A.S. Oliveira Quantitative Research Department, F-Squared Investments Inc., Ewing, NJ, USA

I.V. Onosova National Research University Higher School of Economics, Nizhny Novgorod, Russia

Ioannis Pappas University of Florida, Gainesville, FL, USA

Panos M. Pardalos Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL, USA

National Research University Higher School of Economics, Nizhny Novgorod, Russia

Elena Permjakova National Research University (NRU) Higher School of Economics, Moscow, Russia

Tatiana Poletaeva National Research University Higher School of Economics, Nizhny Novgorod, Russia

Iraklis-Dimitrios Psychas School of Production Engineering and Management, Technical University of Crete, Chania, Greece

James Sackellares MD VA Hospital, Gainesville, FL, USA

Andrey V. Savchenko Laboratory of Algorithms and Technologies for Network Analysis, National Research University Higher School of Economics, Nizhny Novgorod, Russia

Ksenia G. Shipova National Research University Higher School of Economics, Nizhny Novgorod, Russia

Kristina Sörensen KTH Royal Institute of Technology, Stockholm, Sweden

Emre Tokgoz Department of Industrial Engineering, Quinnipiac University, Hamden, CT, USA

Theodore B. Trafalis Department of Industrial and System Engineering, University of Oklahoma, Norman, OK, USA

V.V. Tyutin National Research University Higher School of Economics, Nizhny Novgorod, Russia

Dragan Urošević Mathematical Institute SANU, Belgrade, Serbia

Irina Utkina Laboratory of Algorithms and Technologies for Networks Analysis, National Research University Higher School of Economics, Nizhny Novgorod, Russia

Weili Wu Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA

Part I
Optimization in Networks

Maximally Diverse Grouping and Clique Partitioning Problems with Skewed General Variable Neighborhood Search

Jack Brimberg, Nenad Mladenović, and Dragan Urošević

Abstract The maximally diverse grouping problem (MDGP) requires finding a partition of a given set of elements into a fixed number of mutually disjoint subsets (or groups) in order to maximize the overall diversity between elements of the same group. The clique partitioning problem (CPP) has a similar form as the MDGP, but is defined as the minimization of dissimilarity of elements in an unknown number of groups. In this paper a new variant of variable neighborhood search referred to as skewed general variable neighborhood search (SGVNS) is used to solve both problems. Extensive computational results show that the developed heuristic significantly outperforms its competitors. This demonstrates the usefulness of a combined approach of diversification afforded with *skewed VNS* and intensification afforded with the local search in *general* VNS.

Keywords Diverse grouping • Clique partitioning • Variable neighborhood search

1 Introduction

The maximally diverse grouping problem (MDGP) requires finding a partition of a given set of elements into a fixed number of mutually disjoint subsets (or groups) in order to maximize the overall diversity between elements of the same group.

J. Brimberg
Royal Military College of Canada, Kingston, ON, Canada
e-mail: jack.brimberg@rmc.ca

N. Mladenović (✉)
University of Valenciennes, Valenciennes, France

Mathematical Institute SANU, Belgrade, Serbia
e-mail: nenad.mladenovic12@gmail.com

D. Urošević
Mathematical Institute SANU, Knez Mihailova 36, Belgrade, Serbia
e-mail: draganu@mi.sanu.ac.rs

© Springer International Publishing Switzerland 2016

V.A. Kalyagin et al. (eds.), *Models, Algorithms and Technologies for Network Analysis*, Springer Proceedings in Mathematics & Statistics 156,
DOI 10.1007/978-3-319-29608-1_1

The clique partitioning problem (CPP) is similar. Instead of maximizing diversity among groups, one wants to find an unknown number of groups such that the sum of edge weights over the induced subgraphs (or cliques) is minimized.

MDGP Formulation. The MDGP may be formulated as follows (see, e.g., Fan et al. [11]): $P = \{p_i : i = 1, \ldots, N\}$ denotes the set of elements of interest, and p_{ik}, $k \in \{1, \ldots, K\}$, the attribute values associated with each element p_i. The measure of diversity between any pair of elements p_i and p_j may be given by some distance function, e.g.,

$$d_{ij} = \sqrt{\sum_{k=1}^{K} (p_{ik} - p_{jk})^2}$$

which calculates the Euclidean distance between corresponding points in the attribute space. The objective is the partition of a set P into G disjoint groups so that the sum of diversities over the individual groups is maximized.

Let $x_{ig} = 1$ if p_i is assigned to the group g, and 0 otherwise, $i = 1, 2, \ldots, N$, $g = 1, 2, \ldots, G$. The model may be written as the following quadratic binary integer programming problem:

$$\max \sum_{g=1}^{G} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_{ij} x_{ig} x_{jg} \tag{1}$$

$$s.t.$$

$$\sum_{g=1}^{G} x_{ig} = 1, \ i = 1, 2, \ldots, N \tag{2}$$

$$\sum_{i=1}^{N} x_{ig} \geq a_g, \ g = 1, 2, \ldots, G \tag{3}$$

$$\sum_{i=1}^{N} x_{ig} \leq b_g, \ g = 1, 2, \ldots, G \tag{4}$$

$$x_{ig} \in \{0, 1\}, i = 1, 2, \ldots, N, g = 1, 2, \ldots, G. \tag{5}$$

The constraints (2) ensure that each element is assigned to exactly one group. Constraints (3) and (4) impose minimum and maximum group sizes, respectively. In most studies of MDGP, it is assumed that group sizes must be equal so that $N = mG$, where m is an integer, and $a_g = b_g = m$ for all groups $g = 1, 2, \ldots, G$. Then constraints (3) and (4) in the model above can be replaced with

$$\sum_{i=1}^{N} x_{ig} = m, \ g = 1, 2, \ldots, G. \tag{6}$$

The above model allows groups to differ in size, although in practical cases such as study groups, the variation in size should be kept relatively small.

CPP Formulation. The CPP can be described as follows. Given a weighted, non-oriented, and complete graph $G = (V, E, w)$, we wish to find a partition of the node set into an unknown number of nonempty, disjoint subsets $V_1, V_2, \ldots, V_k$ such that the sum of edge weights over the k induced subgraphs (or cliques) $G_1, G_2, \ldots, G_k$ is minimized. Let c_{ij} denote the weight (or cost) of edge (i, j), for all pairs of nodes (i, j), $1 \leqslant i < j \leqslant N$, where N equals the number of nodes in graph G similarly as in the MDGP. Note, however, that unlike the MDGP where diversities are specified by edge distances $d_{ij} \geq 0$, the costs c_{ij} can now take on negative values. A mathematical formulation of the problem is given by (see [6]):

$$\begin{aligned}
&\min \sum_{(i,j)\in E} c_{ij}x_{ij} \\
&\text{s.t.} \\
&x_{ij} + x_{jr} - x_{ir} \leq 1, && \forall 1 \leqslant i < j < r \leqslant N \\
&x_{ij} - x_{jr} + x_{ir} \leq 1, && \forall 1 \leqslant i < j < r \leqslant N \\
&-x_{ij} + x_{jr} + x_{ir} \leq 1, && \forall 1 \leqslant i < j < r \leqslant N \\
&x_{ij} \in \{0, 1\}, && \forall 1 \leqslant i < j \leqslant N.
\end{aligned} \tag{CPP}$$

If $x_{ij} = 1$, then nodes i and j are in the same cluster, and the cost c_{ij} of edge (i, j) is added in the objective function; otherwise, it is not. The constraint set ensures that all edges in the complete subgraph G_t, $t = 1, 2, \ldots, k$, are included in the solution.

The CPP may be extended to incomplete graphs by inserting fictitious edges with large positive weights between pairs of nodes wherever edges are missing in the original graph. The large positive weight ensures that each subgraph G_t, $t = 1, 2, \ldots, k$, is a clique that contains no fictitious edges. If all edges in a complete graph G have negative (or zero) weights, the problem becomes trivial, with the optimal solution given by $k = 1$ and $G_1 = G$. Thus, the CPP becomes interesting only when some of the edge weights have positive values.

MDGP Applications. One of the earliest applications of MDGP was in the formation of student work groups. For example, in MBA programs it is very important to divide a class into diverse study groups in order to enhance the learning environment (Weitz and Lakshminarayanan [27], Desrosiers et al. [9]). Another application concerns the formation of peer review groups to evaluate research proposals. Again, the objective is to form diverse groups in order to ensure that projects are evaluated from several different points of view (see Hettich and Pazzani [16]). For other applications cited in the literature see, e.g., Lotfi and Cerveny [18] for exam scheduling, Weitz and Lakshminarayanan [27] for very large scale integration (VLSI) design, and Kral [17] for the storage of large programs onto paged memory.

Problems similar to MDGP have been considered by Bhadury et al. [2], Baker and Benn [1], and Desrosiers et al. [9]. A simplified model is proposed in [2], where a high degree of intra-team similarity and/or inter-team dissimilarity is requested.

A network flow problem, known as the dining problem, is used to assign MBA students to different projects. The case study presented in [1] consists of assigning 235 students to eight tutor groups. The model used belongs to the mixed linear goal programming class and is equivalent to the min-sum formulation of Desrosiers et al. [9], where the ℓ_1-norm is used to measure dissimilarities between students. In the proposed local search, swaps between the two worst teams are used to improve the current solution. Desrosiers et al. [9] use a centroid to represent each group of entities, and examine two objectives: *min-sum*—minimize the sum of distances between group centroids and the general centroid of all data; *min-max*—minimize the maximum of such distances. The authors also apply their method to partition 120 MBA students in groups of 5 at HEC, Montreal. If the ℓ_1-norm is used, the model becomes linear. However, if Euclidean distance is considered, the corresponding model is more similar to MDGP, and belongs to the class of quadratic 0–1 programs. Exact and heuristic solution techniques are adapted for solving the problem.

CPP Applications. The CPP has important applications in the social sciences (e.g., see [6], and the references therein). Wang et al. [26] demonstrate that the CPP compares favorably with K-means and latent class analysis for recovery of cluster structure in real data sets. An advantage of the CPP model is that we do not need to know the number of clusters beforehand. One of the most cited applications of CPP is in the aggregation of binary equivalence relations. In this context, the c_{ij} parameters represent the number of attributes on which nodes i and j disagree minus the number on which they agree. For example, if there are 10 attributes in all being compared and given nodes i and j have the same measurement on 8 of them, $c_{ij} = 2 - 8 = -6$, and there is a strong tendency to place these two nodes in the same cluster. If on the other hand $c_{ij} = 8 - 2 = 6$, there would be a tendency to place them in separate clusters.

Recent Heuristics for Solving MDGP. The MDGP is known to be NP-hard [12], and therefore, heuristics have been developed to solve it. Fan et al. [11] propose a hybrid genetic algorithm to solve the problem. Gallego et al. [13] suggest tabu search with strategic oscillation. The artificial bee colony optimization (ABCO) method proposed by Rodriguez et al. [23] may be considered as a state-of-the-art heuristic for solving MDGP. A competitive heuristic based on general variable neighborhood search (GVNS) is found in [24]. GVNS is a variant where a deterministic local search that uses several neighborhoods (called variable neighborhood descent, or VND for short) is applied in the local search step in place of the single neighborhood used in basic VNS.

Recent Heuristics for Solving CPP. The earliest methods to solve CPP use a simple relocation procedure (analogous to the Insertion move for MDGP) that interchanges nodes across clusters until a local optimum is reached (e.g., [22]). Marcotorchino and Michaud [19] expanded the search to include merging of clusters and exchanging cluster memberships for pairs of nodes. Simulated annealing and tabu search heuristics are proposed by De Amorim et al. [8], based on the simple relocation neighborhood above. However, this also includes the possibility of relocating a node to the empty set, thereby increasing the number of clusters by 1.

Charon and Hudry [7] investigate various "noising" procedures that distort the data in order to allow uphill moves. Brusco and K ohn [6] develop a state-of-the-art "neighborhood search heuristic," following the ideas of the variable neighborhood search (VNS) metaheuristic, where the shaking (or perturbation) step is carried out in a random fashion instead of the systematic approach of VNS. The local search employs the single node relocation neighborhood.

Outline. In this paper we present two related algorithms recently developed in [4, 5] for solving the MDGP and CPP, respectively. The methodology is an extension of the GVNS based heuristic proposed in [24]. In brief, the new algorithms consider a skewed version of GVNS that allows moves to inferior points located in promising regions of the solution space. In the next section the relationship between the two problems is analyzed. The new heuristic is described in detail in Sect. 3, as applied to the MDGP including a description of the data structure used for efficient implementation of the VND local search. In Sect. 4 the implementation differences for solving the CPP are summarized. A summary of computational results on a wide range of test problems taken from the literature is given is Sects. 5 and 6 for MDGP and CPP, respectively. For almost all problem instances examined, significantly better solutions are obtained than those of the best competitors.

Concluding remarks are given in Sect. 7. This work unifies the results of two journal articles [4, 5].

2 Relation Between CPP and MDGP

In the MDGP the number of groups (or cliques) is fixed, and typically these groups must have the same size, or nearly the same size. However, in the CPP we don't know the number of groups, and there are no limitations on group size. To get around this difficulty, we simply set the number of groups initially to an upper limit of N (the number of nodes in G), and allow some of these groups to be empty. No limitations on group size are imposed, so that the number of groups is effectively treated as a variable in the model formulation.

Now let $y_{ig} = 1$, if node i belongs to group g, and 0 otherwise. We reformulate CPP as the following equivalent quadratic binary integer program:

$$\max -\sum_{g=1}^{N}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} c_{ij} y_{ig} y_{jg}$$

s.t. (CPP$_e$)

$$\sum_{g=1}^{N} y_{ig} = 1, \qquad i = 1, \ldots, N$$

$$y_{ig} \in \{0, 1\}, \qquad i = 1, \ldots, N, g = 1, \ldots, N$$

This formulation is equivalent to the general MDGP except that (a) all constraints on group size have been deleted and (b) the number of groups is no longer specified. Note that the nonlinear formulation in (CPP_e) allows a reduction in the number of constraints compared to the original linear formulation (CPP) from $O(N^3)$ to N. Meanwhile the number of binary variables remains close to the same in both formulations.

A similar 0/1 quadratic program as (CPP_e) may be found in Wang et al. [25] as an alternate formulation of the CPP. However, in [25], a maximum number of cliques ($k_{\max} < N$) are assumed to be known, which is not necessary here. Also, those authors use the quadratic formulation to solve the Group Technology Problem, whereas we are showing here that the CPP may be viewed as a relaxed form of the MDGP.

By recasting the model in the form of an MDGP, we are now able to borrow any solution method for MDGP and apply it (after some small modifications) to solve our original CPP. We will take advantage of this fact by adapting our new VNS based heuristic that was originally applied on the MDGP, and is described next.

3 Solving MDGP with Variable Neighborhood Search

VNS is a well-known metaheuristic, or framework for building heuristics, whose basic idea is a systematic change of neighborhood structures during a search for a better solution (Mladenović and Hansen [20] and Brimberg et al. [3]). The inner loop of basic VNS contains three steps: (a) shaking; (b) local search; and (c) neighborhood change. A successful VNS variant, called general VNS, uses a mechanism of changing neighborhoods not only in the diversification or shaking step, but also in the intensification or deterministic local search step (Hansen et al. [15], Mladenović et al. [21]). Skewed VNS is another VNS variant that modifies the "neighborhood change" step. A current solution is allowed to move to an inferior solution only if the latter is very far and of similar quality.

This section gives the details of the VNS implementation for solving MDGP. The main idea is to combine the concepts of general VNS (GVNS) and skewed VNS (SVNS), by allowing skewed moves within GVNS; that is, we combine the intensification of the local search in GVNS with the diversification provided by SVNS. The resulting framework is called skewed general variable neighborhood search (SGVNS), as proposed in Brimberg et al. [4].

3.1 Solution Space of MDGP

The solution space includes all possible feasible divisions of elements into groups. A division is feasible if and only if each created group g contains at least a_g and at most b_g elements. A current solution is represented by a vector x^c of length N such

that $x^c[i]$ is the label of the group containing the element i ($i = 1, 2, \ldots, N$). In order to speed up the local search, we also maintain matrix sd^c such that $sd^c[i][g]$ is the sum of diversities between the element i and all the elements assigned to the group g in the current solution:

$$sd^c[i][g] = \sum_{j=1,2,\ldots,N; x^c[j]=g} d_{ij}.$$

Note that for the current solution, matrix sd^c can be computed in $O(N^2)$.

For the solution represented by the vector x^c (and by the corresponding matrix sd^c) it is possible to calculate the objective function value given in formula (1) in the following way:

$$f^c = f(x^c) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_{ij} \chi(x^c, i, j)$$

where

$$\chi(x_c, i, j) = \begin{cases} 1 & \text{if } x^c[i] = x^c[j] \\ 0, & \text{if } x^c[i] \neq x^c[j] \end{cases}.$$

Note also that the same objective value can be calculated by using the previously calculated matrix sd^c:

$$f(x^c) = \frac{1}{2} \sum_{i=1}^{N} sd^c[i][x^c[i]]$$

3.2 VND Local Search

The local search is implemented using VND, for which the following neighborhoods are designed (see [24]): Insertion, Swap, and 3-Chain.

Insertion. Neighborhood Insertion contains solutions obtained by moving only one element from its current group to another group. By using the previously described matrix sd^c, it is possible to compute efficiently for each feasible move the change in value of the objective function f. Denote with x^n a solution obtained from solution x^c by moving the element i from its current group g_1 to a group g_2. In this case, the sum of diversities in all groups except groups g_1 and g_2 is unchanged. The element i is removed from the group g_1 and because of that, the sum of diversities in the group g_1 decreases by the sum of diversities between i and all other elements belonging to the group g_1. The element i is inserted into g_2; hence, the sum of diversities in g_2 increases by the sum of all diversities between i and the elements belonging to

the group g_2. It is easy to conclude that the difference between objective function values for solutions x^c and x^n is

$$\Delta f = f(x^n) - f(x^c) = sd^c[i][g_2] - sd^c[i][g_1].$$

If $\Delta f > 0$, the element i is moved from the group g_1 to the group g_2, ($x^c[i] = g_2$); then, groups g_1 and g_2 are modified and values $sd^c[j][g_1]$ and $sd^c[j][g_2]$ must be updated in the following way:

$$sd^c[j][g_1] = sd^c[j][g_1] - d_{ji}$$

and

$$sd^c[j][g_2] = sd^c[j][g_2] + d_{ji}.$$

Since the updating must be performed for each element j, updating of matrix sd^c after performing an Insertion move has complexity $O(N)$. The number of solutions in the Insertion neighborhood of x^c is $O(GN)$. See Algorithm 1 for details of the search. Note that the subroutine `Updatesd` is used to denote in general the updating of matrix sd^c after a move is made to a neighboring solution (Insertion, Swap, or 3-Chain).

Swap. Neighborhood Swap contains solutions obtained by swapping a single pair of elements belonging to different groups. Let element i be in group g_i and element j in group g_j of the current solution x^c. Denote with x^n the solution obtained after moving the element i into group g_j and the element j into group g_i. Since the element i is removed from the group g_i, the diversities between element i and the elements remaining in the group g_i do not contribute to the objective function value of the

```
Function LSIns(x, sd, f);
rez ← false;
for v ← 1 to N do
    for g ← 1 to G do
        if x[v] ≠ g then
            df ← sd[v][g] − sd[v][x[v]];
            if df > 0 then
                x[v] ← g;
                f ← f + df;
                Updatesd(x, sd, v, g);
                rez ← true
            end
        end
    end
end
return rez
```

Algorithm 1: VND implementation of local search in Insertion neighborhood

```
Function LSSwap(x, sd, f);
rez ← false;
for v ← 1 to N − 1 do
    for u ← v + 1 to N do
        if x[v] ≠ x[u] then
            df ← sd[v][x[u]] + sd[u][x[v]] − sd[v][x[v]] − sd[u][x[u]] − 2d_{v,u};
            if df > 0 then
                Swap(x, v, u);
                f ← f + df;
                Updatesd(x, sd, v, u);
                rez ← true;
                return rez
            end
        end
    end
end
return rez
```

Algorithm 2: VND implementation of local search in Swap neighborhood

new solution. But, because the element i is inserted in the group g_j, all diversities between i and the elements belonging to the group g_j contribute to the objective function value of the new solution. Similar facts are true for the element j. So, we can finally calculate the difference between the objective values of the current and neighboring solutions:

$$\Delta f = f(x^n) - f(x^c) = (sd^c[i][g_j] - sd^c[i][g_i]) + (sd^c[j][g_i] - sd^c[j][g_j]) - 2d_{ij}.$$

It is obvious that the change of the objective value for each solution from neighborhood Swap is done in $O(1)$, while the cardinality of Swap is $O(N^2)$. After performing a Swap move, it is necessary to update matrix sd^c, and the complexity of this update is $O(N)$. (We can consider Swap or 2-opt as two successive Insertions.)

3-Chain. A 3-Chain move is determined by three elements, i, j, and k, belonging to three different groups, g_i, g_j, and g_k, respectively, by moving the element i to the group g_j, the element j to the group g_k, and the element k to the group g_i. Neighborhood 3-Chain consists of all solutions obtained by performing a single 3-Chain move.

We can calculate the difference between objective function values as follows:

$$\begin{aligned}\Delta f = f(x^n) - f(x^c) =& (sd^c[i][g_j] - sd^c[i][g_i]) + (sd^c[j][g_k] - sd^c[j][g_j]) + \\ & (sd^c[k][g_i] - sd^c[k][g_k]) - (d_{ij} + d_{jk} + d_{ki}).\end{aligned}$$

So, the complexity of solution checking is $O(1)$, but the cardinality of the whole neighborhood is $O(N^3)$. Updating of the matrix sd^c after a 3-Chain move is made is performed in a straightforward way by combining three successive Insertion moves.

```
Function VND-2(x, sd, f);
end ← false;
while not end do
    repeat
        rez ← LSIns(x, sd, f);
    until rez = false;
    if LSSwap(x, sd, f) = false then end ← true
end
```

Algorithm 3: VND-2

Two VND Variants. We propose two variants of variable neighborhood descent:

- VND-2—use neighborhoods Insertion and Swap in this order and
- VND-3—use all three developed neighborhoods in the following order: Insertion, Swap, and 3-Chain.

If no improvement is found in a given neighborhood of the current solution, the search resumes in the next neighborhood in the sequence. If an improvement is found, the local search always resumes in the Insertion neighborhood of the new current solution, i.e., the first neighborhood in the sequence. If no improvement can be found in any of the listed neighborhoods of x^c, the local search is terminated. See Algorithms 1–3 for a standard implementation of VND-2. Our implementation is slightly different in that all pairs of elements are examined once in LSSwap, and hence, a few Swap moves may occur before exiting the loop and returning to the Insertion neighborhood. Also note that VND-2 is the only variant attempted in the computational experiments due to limitations placed on the computing time.

3.3 Initial Solution

The calculation of the initial solution is done in two phases. In the first phase, we ensure that each group g contains at least a_g elements by inserting the chosen elements. In the second phase, we distribute the remaining elements so that each group g contains at most b_g elements. At the beginning, we select G elements at random and insert them into different groups.

Denote with E_g a set of elements currently assigned to a group g. During the first phase, we maintain the set of groups that have fewer elements than the desired minimum:

$$G' = \{g : |E_g| < a_g\}.$$

Each iteration of the first phase consists of selecting at random one unassigned element, denoted with i, and assigning it to a selected group. The selected element must be inserted into one of the groups from the set G'. So, for each group $g \in G'$, we calculate the average distance between the selected element i and all the elements belonging to the group g, and select the group whose average distance value is the

```
Procedure InitialSolution(x, sd, f);
AV ← {1, 2, ..., N}; AG ← {1, 2, ..., G};
for g ← 1 to G do c_g ← 0 end
while AG ≠ ∅ do
    g ← RandomElem(AG); v ← RandomElem(AV);
    x[v] ← g; c_g ← c_g + 1;
    AV ← AV \ {v};
    if c_g = a_g then AG ← AG \ {g} end
end
AG ← ∅;
for g ← 1 to G do
    if c_g < b_g then AG ← AG ∪ {g} end
end
while AV ≠ ∅ do
    g ← RandomElem(AG); v ← RandomElem(AV);
    x[v] ← g; c_g ← c_g + 1;
    AV ← AV \ {v};
    if c_g = b_g then AG ← AG \ {g} end
end
Computesd(x, sd);
f ← Computeobj(x)
```

Algorithm 4: Initial solution

largest. The selected group g is the one for which the expression D_{ig} defined as

$$D_{ig} = \frac{\sum_{j \in E_g} d_{ij}}{|E_g|}$$

is maximized.

The first phase ends when the set G' becomes empty. During the second phase, we maintain a set of groups which has fewer elements than the desired maximum:

$$G' = \left\{ g : |E_g| < b_g \right\}.$$

All other steps of the second phase are the same. The second phase finishes when all elements are assigned.

The initial solution can also be created at random: instead of assigning a selected element to a group with maximal average distance, we assign it to a randomly selected group.

3.4 Shaking

In order to perform a perturbation of the current solution, we define the kth neighborhood as the set of solutions obtained by k consecutive Swap moves. Thus, in the shaking step, we generate a random solution from the kth neighborhood of

Procedure Shake(k, x, sd, f);
while $k > 0$ **do**
  $(u, v) \leftarrow$(RandVert, RandVert);
  if $x[u] \neq x[v]$ **then**
    Swap(x, u, v);
    $k \leftarrow k - 1$
  end
end
Computesd(x, sd);
$f \leftarrow$ Computeobj(x)

Algorithm 5: Shake procedure

Function SGVNS($k_{min}, k_{step}, k_{max}, t_{max}$);
InitialSolution(x^c, sd^c, f^c);
VND-2(x^c, sd^c, f^c);
$(x^b, sd^b, f^b) \leftarrow (x^c, sd^c, f^c)$;
$k \leftarrow k_{min}$;
while CPUTime() $\leq t_{max}$ **do**
  $(x^n, sd^n, f^n) \leftarrow (x^c, sd^c, f^c)$;
  Shake(k, x^n, sd^n, f^n);
  VND-2(x^n, sd^n, f^n);
  if $f(x^n)/f(x^c) + \alpha\rho(x^c, x^n) > 1$ and $f(x^n)/f(x^b) + \alpha\rho(x^b, x^n) > 1$ **then**
    $(x^c, sd^c, f^c) \leftarrow (x^n, sd^n, f^n)$;
    if $f(x^c) > f(x^b)$ **then** $(x^b, sd^b, f^b) \leftarrow (x^c, sd^c, f^c)$ **end**;
    $k \leftarrow k_{min}$
  else
    $k \leftarrow k + k_{step}$;
    if $k > k_{max}$ **then** $k \leftarrow k_{min}$
    end
  end
end
return x^b

Algorithm 6: SGVNS

the current solution by performing k random Swap moves. In each of the moves, we randomly select two elements that belong to two different groups.

3.5 Skewed General Variable Neighborhood Search

Unlike classical VNS, the idea is to allow skewed moves to inferior solutions in promising regions of the solution space. In our implementation this means that after shaking and local search, we move to the new solution x^n if the following conditions are satisfied:

$$\frac{f(x^n)}{f(x^c)} + \alpha\rho(x^c, x^n) > 1 \quad \text{and} \quad \frac{f(x^n)}{f(x^b)} + \alpha\rho(x^b, x^n) > 1, \tag{7}$$

where x^b is the best found solution, and x^c the current solution. We use $\rho(x^{(1)}, x^{(2)})$ to denote the distance between solutions $x^{(1)}$ and $x^{(2)}$. This distance is defined in the following way:

$$\rho(x^{(1)}, x^{(2)}) = \frac{\left|x^{(1)} \Delta x^{(2)}\right|}{N^2/G}, \tag{8}$$

where

$$\begin{aligned} x^{(1)} \Delta x^{(2)} = \big\{(i,j) : ((x^{(1)}[i] = x^{(1)}[j]) \wedge (x^{(2)}[i] \neq x^{(2)}[j])) \vee \\ ((x^{(1)}[i] \neq x^{(1)}[j]) \wedge (x^{(2)}[i] = x^{(2)}[j]))\big\}, \end{aligned} \tag{9}$$

which estimates the "fraction" of pairs belonging to the same group in one solution, but not to the same group in the other solution. Note that the average number of elements in any of the groups is approximately $\frac{N}{G}$, and thus there are $\binom{\frac{N}{G}}{2} \approx \frac{N^2}{2G^2}$ pairs of elements in each group. Because there are G groups, the total number of pairs is $\frac{N^2}{2G}$ in one solution, and $2 \times \frac{N^2}{2G} = \frac{N^2}{G}$ elements in both solutions. The parameter α is assigned a value of 0.05 (selected after detailed testing).

Combining the different parts described above leads to the implementation of SGVNS that is summarized in Algorithm 6.

4 Implementation Differences for Solving CPP with SGVNS

To avoid duplication in this section, we only summarize the differences between the SGVNS implementation for CPP and the one for MDGP. Further details on the CPP implementation are given in [5]. Computational results in Sect. 6 demonstrate that the new heuristic provides a powerful solution approach just as its counterpart for the MDGP.

The solution space is constructed in the same way as for the MDGP. Thus, the current solution is represented by vector x^c of length N ($x^c = (x_1^c, x_2^c, \ldots, x_N^c)$) such that x_i^c is the label of the group (clique) containing element i ($i = 1, 2, \ldots, N$). In order to speed up the local search, we also maintain matrix sd^c such that $sd^c[i][g]$ is the sum of edge weights between element i and all elements assigned to the group g in the current solution.

The number of groups (active columns in sd^c) is initially set to N with one element in each group. This, in fact, gives the initial solution. Unlike MDGP with a constant number of groups (G), the number of groups ($g_{\max}$) in CPP will be allowed to change during the solution process. To accomplish this, we always maintain an empty group in the list of groups (last column in sd^c).

The local search is implemented using the same VND as for MDGP, including the same nested structure of Insertion and Swap neighborhoods. If the Insertion step is performed, we change the current solution, and then it becomes necessary