Data-driven Structure Detection in Optimization: Decomposition, Hub Location, and Brain Connectivity

by

Taghi Khaniyev

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Management Sciences

Waterloo, Ontario, Canada, 2018

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Oleg Prokopyev Professor, Dept. of Industrial Engineering University of Pittsburgh
Supervisor(s):	Samir Elhedhli Professor, Dept. of Management Sciences,
	Fatih Safa Erenay Associate Professor, Dept. of Management Sciences
Internal Member:	Houra Mahmoudzadeh Assistant Professor, Dept. of Management Sciences
Internal Member:	Lukasz Golab Associate Professor, Dept. of Management Sciences
Internal-External Member:	Fakhri Karray Professor, Dept. of Electrical & Computer Engineering

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Employing data-driven methods to efficiently solve practical and large optimization problems is a recent trend that focuses on identifying patterns and structures in the problem data to help with its solution. In this thesis, we investigate this approach as an alternative to tackle real life large scale optimization problems which are hard to solve via traditional optimization techniques. We look into three different levels on which data-driven approaches can be used for optimization problems.

The first level is the highest level, namely, *model structure*. Certain classes of mixed-integer programs are known to be efficiently solvable by exploiting special structures embedded in their constraint matrices. One such structure is the bordered block diagonal (BBD) structure that lends itself to Dantzig-Wolfe reformulation (DWR) and branch-and-price. Given a BBD structure for the constraint matrix of a general MIP, several platforms (such as COIN/DIP, SCIP/GCG and SAS/ DECOMP) exist that can perform automatic DWR of the problem and solve the MIP using branch-and-price. The challenge of using branch-and-price as a general-purpose solver, however, lies in the requirement of the knowledge of a structure *a priori*. We propose a new algorithm to automatically detect BBD structures inherent in a matrix. We start by introducing a new measure of goodness to capture desired features in BBD structures such as minimal border size, block cohesion and granularity of the structure. The main building block of the proposed approach is the modularity-based community detection in lieu of traditional graph/hypergraph partitioning methods to alleviate one major drawback of the existing approaches in the literature: predefining the number of blocks. When tested on MIPLIB instances using the SAS/DECOMP framework, the proposed algorithm was found to identify structures that, on average, lead to significant improvements both in computation time and optimality gap compared to those detected by the state-of-the-art BBD detection techniques in the literature.

The second level is *problem type* where problem-specific patterns/characteristics are to be detected and exploited. We investigate hub location problem (HLP) as an example. HLP models the problem of selecting a subset of nodes within a given network as hubs, which enjoy economies of scale, and allocating the remaining nodes to the selected hubs. The main challenge of using HLP in certain promising domains is the inability of

current solution approaches to handle large instances (e.g., networks with more than 1000 nodes). In this work, we explore an important pattern in the optimal hub networks: *spatial separability*. We show that at the optimal solutions, nodes are typically partitioned into allocation clusters in such a way that convex hulls of these clusters are disjoint. We exploit this pattern and propose a new data-driven approach that uses the insights generated from the solution of a smaller problem - low resolution representation - to find high quality solutions for the large HLPs.

The third and the lowest level is the *instance* level where the instance-specific data is explored for patterns that would help solution of large problem instances. To this end, we open up a new application of HLPs originating from human brain connectivity networks (BCN) by introducing the largest (with 998 nodes) and the first three-dimensional dataset in the literature so far. Experiments reveal that the HLP models can successfully reproduce similar results to those in the medical literature related to hub organisation of the brain. We conclude that with certain customizations and methods that allow tackling very large instances, HLP models can potentially become an important tool to further investigate the intricate nature of hub organisations in human brain.

Acknowledgements

First of all I would like to express my deepest gratitude to Samir Elhedhli and Fatih Safa Erenay, my supervisors, for their invaluable guidance and advisory throughout my Ph.D studies. From the very beginning, they have always pointed me to the right direction at every crossroad I came across. Topics they advised me to pursue have not only been fruitful, but also very inspirational for me. They gave me space for being creative and pursuing my intuition when in doubt, but they also managed to get me back on track before feeling lost whenever I diverged beyond reason. I feel blessed to have been supervised with such outstanding role models for my academic career.

I thank Olaf Sporns, Professor of Pyschological and Brain Sciences at Indiana University Bloomington for sharing the brain connectivity dataset used in their original study, which significantly enriched our computational experiments.

Throughout my Ph.D, I received the scholarship of Ministry of Education in Azerbaijan. I would like to thank them for their valuable support. I am also very grateful to Ozden Onur Dalgic, Gizem Sultan Nemutlu, Alpay Pasha, Dana Lynn Wagner and a long list of others for wholeheartedly believing in my aspirations and supporting me in whichever path I chose to take.

Finally, I wish to express my deepest gratitude to my parents, Tahir Khaniyev and Zülfiye Hanalioglu, for always being the biggest inspirations in my life. They have taught me to question, to be curious and to distinguish right from wrong. Without them, I would not have been who I am, now. I would also like to thank to my brother, Şahin Hanalioglu, and my sister, Ayten Hanalioglu, who have always been wonderful friends and whose successes in their own fields have always been a great inspiration to me.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AP	Australian Post
BBD	Bordered block diagonal
BCN	Brain connectivity network
CAB	Civil Aviation Board
CHLP	Cluster hub location problem
DWR	Dantzig-Wolfe reformulation
GPVS	Graph partitioning with vertex separators
HCP	Human Connectome Project
HLP	Hub location problem
IEC	Intersection elimination constraint
KEE	Khaniyev-Elhedhli-Erenay
LP	Linear program
LRP	Low resolution problem
MIP	Mixed integer program
OR	Operations research
pHAP	p hub allocation problem
ROI	Region of interest
USApHMP	Uncapacitated single allocation p-hub median problem

Chapter 1

Introduction

The advent of Big Data in recent years has drawn attention of the scientific community to developing new techniques that enable efficient processing of unprecedented amounts of data pouring in from real life applications. One of the disciplines in which such data processing has become of utmost importance is operations research (OR) which deals with the mathematical modelling of real life systems (e.g., manufacturing, transportation, finance, healthcare, etc.) and application of analytical methods to help make better decisions. Data Analytics enables organizations to get insights from data that go beyond simple reporting. This involves the use of algorithms and mathematical models to explore the data and discover patterns which are the essence of valuable knowledge (Dhaenens and Jourdan, 2016).

Typically, a mathematical model of a system is built by using the information available about the system together with a set of reasonable assumptions for the aspects of the system that are unknown. The more data recorded from a system the more information the modeller has about it, and the less assumptions s/he needs to make. Hence, intuitively, it can be said that by recording more data, one can build a more realistic model of a system which is likely to lead to better informed decisions on the problems of interest. This may be one of the reasons for the growing interest for approaches that combine Big Data and optimization (Nicosia et al., 2017).

With more data, on the other hand, mathematical models become more complicated

and the optimization problems resulting from these models become harder to solve. This is one of the main challenges of Big Data from the OR perspective. Employing data-driven methods to efficiently solve practical and large optimization problems has become a recent trend that focuses on identifying patterns and structures in the problem data to help with its solution. In this thesis, we investigate this approach as an alternative to tackle real life large scale optimization problems which are hard to solve via traditional optimization techniques. We look into three different levels on which data-driven approaches can be used for optimization problems: *model structure, problem type*, and *instance* levels. Each level is dealt with in a different chapter.

Throughout this thesis, we will be working on a subclass of optimization problems called mixed integer program (MIP). For a given set of integer and/or real-valued variables ($\boldsymbol{x}$) an MIP is an optimization problem that either minimizes or maximizes a linear objective function, subject to some linear equations or inequalities (Gamrath and Lübbecke, 2010). Any MIP can be transformed into the following form:

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad \mathbf{c}^T\boldsymbol{x} \tag{1.1a}$$

$$\text{s.t.} \quad \mathbf{A}\boldsymbol{x} = \mathbf{b} \tag{1.1b}$$

$$\boldsymbol{x} \in \mathbb{R}^n_+ \tag{1.1c}$$

$$\boldsymbol{x}_i \in \mathbb{Z} \quad \forall i \in I \tag{1.1d}$$

where $n, m \in \mathbb{N}$, $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{b} \in \mathbb{R}^m$, and $I \in \{1, ..., n\}$. In the above definition, $\mathbf{c}$, $\mathbf{A}$, and $\mathbf{b}$ are often referred to as the objective function coefficient vector, constraint matrix, and right-hand-side vector, respectively.

Certain classes of MIPs (e.g., bin packing, hub location etc.) are known to be efficiently solvable by exploiting special structures embedded in their constraint matrices. One such structure is the BBD structure that lends itself to DWR and branch-and-price.

1.1 Bordered Block Diagonal Structures

Let $\mathcal{R} = \{1, 2, ..., m\}$ and $\mathcal{C} = \{1, 2, ..., n\}$ be the index set of rows and columns of an $m \times n$ constraint matrix $\mathbf{A} = \{a_{ij} \mid i \in \mathcal{R}, j \in \mathcal{C}\}$ of an MIP. A permutation ($\mathbf{A}^\pi$) of $\mathbf{A}$ into a K-way singly BBD form is defined in Aykanat et al. (2004):

$$\mathbf{A}^\pi = \mathbf{A}[R, C] = \begin{bmatrix} A^\pi_{11} & A^\pi_{12} & ... & A^\pi_{1K} \\ A^\pi_{21} & A^\pi_{22} & ... & A^\pi_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ A^\pi_{K1} & A^\pi_{K2} & ... & A^\pi_{KK} \\ A^\pi_{S1} & A^\pi_{S2} & ... & A^\pi_{SK} \end{bmatrix} = \begin{bmatrix} D_1 & & & \\ & D_2 & & \\ & & \ddots & \\ & & & D_K \\ B_1 & B_2 & ... & B_K \end{bmatrix} \tag{1.2}$$

where R and C denote, respectively, the row and column permutations. In (1.2), the $b \times n$ submatrix $\mathbf{B} = [B_1, ..., B_K]$ is called the **border component** and the $d \times n$ submatrix $\mathbf{D} = \begin{bmatrix} D_1 & & \\ & \ddots & \\ & & D_K \end{bmatrix}$ is called the **block-diagonal component**.

The most general BBD form - which is often referred to as doubly BBD or arrowhead form - is defined by two border components, a row and a column, and a block-diagonal component. In this study, we only consider singly BBD form with row border and hereinafter refer to it as BBD for simplicity.

Permuting sparse matrices into bordered block-diagonal structures has recently drawn the attention of many researchers due to their suitability for parallel processing and decomposition. Potential applications of BBD structures such as LU and QR factorization and decomposition of linear program (LP) are discussed in Aykanat et al. (2004). We focus on one specific application of BBD structures; namely, decomposition of mixed integer programs using Lagrangian relaxation. The merit of Lagrangian relaxation lies in its ability to provide potentially tight bounds in a reasonable computational time by decomposing the original problem into smaller subproblems. It requires, however, a valid BBD structure to be known and supplied by the user. Consequently, the detection of an underlying BBD structure is a crucial step in using decomposition methods, such as Lagrangian relaxation,

to tackle large scale MIPs.

In Chapter 2, we propose an efficient data-driven algorithm to automatically detect the underlying singly BBD structures in constraint matrices of MIPs. Our computational experiments reveal that most optimization problems have an underlying BBD structure that can be exploited by decomposition. This may be described as a high level application of data-driven methods in the optimization context since the proposed approach could be used in any MIP irrespective of the nature of the underlying real life problem.

Devising a methodology on such generic level has its drawbacks. For example, certain characteristics specific to the underlying problem may be overlooked. These characteristics sometimes could prove to be very valuable in efficiently solving such problems. One such problem type which is known to be very hard to solve using generic methods is hub location problem (HLP).

1.2 Hub Location Problem

HLPs tackle the problem of selecting a subset of nodes within a given network as hubs and allocating the remaining nodes to the selected hubs. All flow between nodes is, then, routed via hubs (as the inter-hub links typically enjoy economies of scale) in such a way to minimize an overall transportation cost.

We study a specific subclass of HLPs: USApHMP in which all the hubs are assumed to be fully interconnected, each node in the network is allocated to a single hub, the hubs are uncapacitated, and the total number of hubs, p is specified a priori. There exist several MIP formulations for the USApHMP. Throughout this thesis, we use the formulation by Ernst and Krishnamoorthy (1996) since it has been shown to be the most computationally efficient model when solved by general-purpose commercial solvers (e.g., CPLEX, Gurobi, etc.). Table 1.1 provides the list of important notation used throughout the chapter and Definition 1 formally states the USApHMP and provides the MIP formulation proposed by Ernst and Krishnamoorthy (1996).

Definition 1 *Let $\mathcal{N} = \{1, 2, ..., n\}$ be a set of nodes in a given coordinate system. For each node pair (i, j), let w_{ij} represent the volume of flow and d_{ij} represent the distance between*

Table 1.1: Descriptions of parameters and decision variables in USApHMP formulation

Notation	**Description**
α	Inter-hub discount factor,
w_{ij}	The amount of flow from node i to node j,
d_{ij}	The distance from node i to node j,
$O_i = \sum_j w_{ij}$	Total amount of outflow from node i,
$D_i = \sum_j w_{ji}$	Total amount of inflow to node i,
x_{ikm}	Total amount of flow emanating from node i that is routed between hubs k and m
y_{ik}	Binary variable indicating whether node i is allocated to hub k

the two nodes. The traffic (flow) between nodes is to be routed via a set of nodes designated as hubs. Each node must be allocated to exactly one hub. A collection (distribution) cost is incurred to route traffic from (to) a origin (destination) node to (from) a hub. Moreover, a transfer cost is incurred to send flow between hubs. Given p, the number of hubs to locate on the network, the USApHMP is the problem of finding the optimal hubs locations, as well as the allocation of the remaining nodes to hubs such that the overall cost of satisfying the flow demand is minimized and can be formulated mathematically as follows:

$$
\begin{aligned}
\textit{[USApHMP]} \quad \textit{min} \quad & \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{N}}\sum_{m\in\mathcal{N}} \alpha d_{km} x_{ikm} + \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{N}} (O_i + D_i) d_{ik} y_{ik} && && \text{(1.3a)}\\
\textit{s.t.} \quad & \sum_{k\in\mathcal{N}} y_{kk} = p && && \text{(1.3b)}\\
& \sum_{k\in\mathcal{N}} y_{ik} = 1 && \forall i \in \mathcal{N} && \text{(1.3c)}\\
& y_{ik} \le y_{kk} && \forall i,k \in \mathcal{N} && \text{(1.3d)}\\
& \sum_{m\in\mathcal{N}} x_{ikm} - \sum_{m\in\mathcal{N}} x_{imk} = O_i y_{ik} - \sum_{j\in\mathcal{N}} w_{ij} y_{jk} && \forall i,k \in \mathcal{N} && \text{(1.3e)}\\
& x_{ikm} \ge 0 && \forall i,k,m \in \mathcal{N} && \text{(1.3f)}\\
& y_{ik} \in \{0,1\} && \forall i,k \in \mathcal{N} && \text{(1.3g)}
\end{aligned}
$$

In Chapter 3, we uncover an important pattern in the optimal hub networks: *spatial separability*. We show that at the optimal solutions, nodes are typically partitioned into

allocation clusters in such a way that convex hulls of these clusters are disjoint. We exploit this pattern and propose a new data-driven approach that uses the insights generated from the solution of a smaller problem - low resolution representation - to find high quality solutions for the large HLPs.

HLPs have been introduced in the context of airline transportation (O'Kelly, 1987a), postal delivery (Ernst and Krishnamoorthy, 1996), and telecommunication (Klincewicz, 1998). The main challenge of using HLP in certain promising domains is the inability of current solution approaches to handle large instances (e.g., networks with more than 1000 nodes). One such application area which has not been explored before originates from BCN.

1.3 Brain Connectivity Networks

Recent interest of the medical and neuroscience community in building a "network map" (connectome) of the human brain lead to the rapid development of medical imaging technologies that can probe the neural circuitry of the human brain with high spatio-temporal resolution. Using high-resolution diffusion tensor/spectrum imaging (DTI/DSI) data, researchers have shown that some regions in the brain cortex play a hub role and act as an intermediary to transmit signals between other regions (van den Heuvel and Sporns, 2013). This is believed to provide an evolutionary advantage in reducing the size of the brain (Bullmore and Sporns, 2012). As damage to these regions causes serious problems (van den Heuvel et al., 2010; Zalesky et al., 2011), it is important to identify these hub regions as precisely as possible. Researchers have hitherto used graph-theoretical measures (Crossley et al., 2013; Bassett and Sporns, 2017) to study the connectivity networks of human brain and identify hub regions. To the best of our knowledge, identifying hub regions in BCN has never been studied within an optimization context.

In Chapter 4, we explore the potential of USApHMP in identifying the precise locations of hub regions in BCN provided that the resulting large instances can be solved efficiently. Currently, the state-of-the-art methods are capable of solving instances with only a few hundred nodes within a reasonable amount of time. Considering the fact that there are

approximately 86 billion neurons in an adult human brain (Azevedo et al., 2009), one can argue that being able to optimally solve a problem with a few hundred nodes is not appealing enough by itself from the practical point of view as it would yield very low precision.

It is not uncommon that methodologies that work well on moderate size instances become intractable as the problem size increases. Hence, it is essential to make use of instance-specific patterns to be able to solve very large problem instances. The largest problem instances in the literature have only 200 nodes (Ernst and Krishnamoorthy, 1996). Scarcity of scalable solution methodologies may be attributed to the absence of large practical benchmark instances. To bridge this gap, a BCN dataset originally generated by Hagmann et al. (2008) is re-introduced in Chapter 4 in a way amenable to the existing HLP models. This dataset is the largest (with $n = 998$ nodes) and the first three-dimensional dataset in the literature so far. Computational experiments on this dataset reveal that the HLP models can successfully reproduce similar results to those in the medical literature related to hub organisation of the human brain. Our results in Chapter 4 may be seen as a proof of concept that with certain customizations and scalable solution methods that allow tackling very large instances, HLP models can potentially become important tools to further investigate the intricate nature of hub organisations in BCNs.

1.4 Outline

The rest of the thesis is organized in a way to explore the potential uses of data-driven methods in optimization from the highest to the lowest level. In Chapter 2, we look into the problem of detecting BBD structures in the constraint matrices of generic MIPs and exploiting such structures with decomposition techniques. In Chapter 3, we investigate a specific problem type (HLP), which is well-known to be a challenging MIP, and uncover an important characteristic of the problem that leads to a new computationally efficient data-driven solution approach. In Chapter 4, we introduce BCNs and study the problem of locating hub regions in human brain as a new potential application of HLP. Conclusions and a brief discussion on future research avenues are provided in Chapter 4.

Chapter 2

Structure Detection in Mixed Integer Programs

2.1 Introduction and Literature

Permuting a matrix into a BBD form (i.e., detecting an inherent BBD structure) is particularly valuable for the decomposition of MIPs. Let us consider the following generic MIP problem whose constraint matrix $\mathbf{A} = \begin{bmatrix} \mathbf{D} \\ \mathbf{B} \end{bmatrix}$ is given in its BBD form as in (1.2):

$$\begin{aligned}
\underset{\boldsymbol{x}}{\text{minimize}} \quad & \mathbf{c}^T \boldsymbol{x} && \text{(2.1a)} \\
\text{s.t.} \quad & \mathbf{D}\boldsymbol{x} = \mathbf{e} && \text{(2.1b)} \\
& \mathbf{B}\boldsymbol{x} = \mathbf{f} && \text{(2.1c)} \\
& \boldsymbol{x} \in \mathbb{R}^n_+ && \text{(2.1d)} \\
& \boldsymbol{x}_i \in \mathbb{Z} \quad \forall i \in I && \text{(2.1e)}
\end{aligned}$$

where $n, m, d, b \in \mathbb{N}$, $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{D} \in \mathbb{R}^{d \times n}$, $\mathbf{B} \in \mathbb{R}^{b \times n}$, $\mathbf{e} \in \mathbb{R}^d$, $\mathbf{f} \in \mathbb{R}^b$, and $I \in \{1, ..., n\}$. In the above definition, $\mathbf{D}$ and $\mathbf{B}$ are called the block-diagonal component and the border component, respectively.

Notice that if the constraints in (2.1c) are relaxed, the remaining problem is separable to K independent MIPs. **Lagrangian relaxation** relaxing constraints (2.1c) using **Lagrangian multipliers** $\boldsymbol{\mu} \in \mathbb{R}^b$ results in the subproblem $z_{SP}(\boldsymbol{\mu}) = \min\{(\mathbf{c}^T - \boldsymbol{\mu}^T\mathbf{B})\boldsymbol{x} : (2.1\text{b}), (2.1\text{d}), (2.1\text{e})\}$, where $z_{SP}(\boldsymbol{\mu})$ is the optimum objective value of the subproblem for a given $\boldsymbol{\mu}$. In practice, this relaxed problem, which is decomposable into K independent **subproblems**, can often be solved more easily than the original problem. The best **Lagrangian bound** corresponding to the set of relaxed constraints can be found by solving the **Lagrangian dual** problem $\max\{\mathbf{f}^T\boldsymbol{\mu} + z_{SP}(\boldsymbol{\mu})\}$ which can be solved using subgradient or cutting plane methods. Another decomposition technique, which is closely related to Lagrangian relaxation, is **Dantzig-Wolfe reformulation (DWR)**. For the generic MIP problem given in (2.1), DWR that convexifies constraints (2.1b), (2.1d), and (2.1e) yields an extended formulation with an LP bound that corresponds to the Lagrangian bound mentioned earlier. The **LP relaxation of the DWR** is equivalent to Lagrangian relaxation and the two are used interchangeably hereinafter.

Relaxing a minimization (maximization) problem provides a lower (upper) bound on the value of the objective function. In theory, there are numerous different relaxations available for even moderate size MIPs. The challenge is to identify a relaxation that results in the tightest bound in a reasonable amount of time. The **tightness** of a bound or the **strength** of a relaxation is typically defined as the relative deviation from the optimal objective value; i.e.,

$$Gap = \frac{z^* - \bar{z}}{z^*}\%$$

where z^* and $\bar{z}$ are the optimal objective value and the bound, respectively. When z^* is unknown, the best known objective value $\tilde{z}$ is used instead.

Besides the tightness of the bound it produces, another key aspect of a relaxation is its computational efficiency; namely, the time required to calculate the bound. As mentioned earlier, a decomposable problem is often easier to solve (i.e., requires less computational time); therefore, detecting an *a priori* decomposable structure in the constraint matrix of an MIP is often desirable. One of the key properties of the constraint matrix $\mathbf{A}$ of an MIP is that it is not affected by the order of rows and columns. In other words, changing the order of variables and/or constraints does not affect the optimal solution. This significantly

reduces the number of possible structures to search through compared to when the order mattered. To make mathematical use of this property, the constraint matrix $\mathbf{A}$ can be represented as a bipartite graph $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{R} \cup \mathcal{C}$ and $\mathcal{E} = \{(i,j) \mid a_{ij} \neq 0\}$ (Aykanat et al., 2004). The majority of BBD structure detection methods in the literature relies on this graph representation.

A subset $\mathcal{S} \subset \mathcal{V}$ is called a **K-way vertex separator** if the subgraph $\boldsymbol{\mathcal{G}}[\mathcal{V} \setminus \mathcal{S}]$ induced by removing $\mathcal{S}$ from $\boldsymbol{\mathcal{G}}$ has at least K connected components. Similarly, $\Pi = \{\mathcal{V}_1, \mathcal{V}_2, ..., \mathcal{V}_K; \mathcal{S}\}$ is said to be a **K-way vertex partition** (of $\boldsymbol{\mathcal{G}}$) **by vertex separator**, if the following conditions hold:

$$
\begin{aligned}
&\mathcal{V}_k \subset \mathcal{V}, \ \mathcal{V}_k \neq \varnothing, \ \text{for } 1 \leq k \leq K \\
&\mathcal{V}_k \cap \mathcal{V}_l = \varnothing, \ \text{for } 1 \leq k < l \leq K \\
&\mathcal{V}_k \cap \mathcal{S} = \varnothing, \ \text{for } 1 \leq k \leq K \\
&\bigcup_{k=1}^{K} \mathcal{V}_k \cup \mathcal{S} = \mathcal{V} \\
&Adj(\mathcal{V}_k) \subseteq \mathcal{S}, \ \text{for } 1 \leq k \leq K
\end{aligned}
$$

where $Adj(\mathcal{V}_k)$ is an extension of the notation $Adj(v_i)$ which denotes the set of vertices adjacent to vertex v_i in graph $\boldsymbol{\mathcal{G}}$. We use $Adj(\mathcal{V}_k)$ to denote the adjacency set of a vertex subset $\mathcal{V}_k \subseteq \mathcal{V}$, i.e., $Adj(\mathcal{V}_k) = \{v_j \in \mathcal{V} \setminus \mathcal{V}_k : v_j \in Adj(v_i) \text{ for some } v_i \in \mathcal{V}_k\}$. The **graph partitioning with vertex separators (GPVS)** is defined as follows (Aykanat et al., 2004): *Given a graph $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$, an integer K, a cost function for vertex separators, and a balance criterion for subgraphs, the GPVS problem seeks to find a K-way vertex partition $\Pi = \{\mathcal{V}_1, \mathcal{V}_2, ..., \mathcal{V}_K; \mathcal{S}\}$ that satisfies the balance criterion with minimum cost.*

Aykanat et al. (2004) uses a simple cost function $cost(\Pi) = |\mathcal{S}|$ to minimize the size of the border component (i.e., the number of rows in the border component). In principle, the BBD structure detection problem can be formulated as a GPVS problem. For a given K and balance criterion, the resulting K-way vertex partition Π can be used to reconstruct $\mathbf{A}^{\pi}$. Specifically, each connected component in the subgraph $\boldsymbol{\mathcal{G}}[\mathcal{V} \setminus \mathcal{S}]$ will correspond to a block in $\mathbf{D}$ and $\mathcal{S}$ will correspond to $\mathbf{B}$. However, the formulation of GPVS requires the input of the number of blocks K and other necessary parameters to define a balance

criterion.

The pursuit of a fully automatic BBD structure detection algorithm renders GPVS obsolete as a base method due to the aforementioned requirements. One needs a base method which requires no user input. **Community detection** offers such an advantage. A graph is said to have **a community structure** if the vertices of the graph can be partitioned into groups, referred to as **communities**, such that the density of edges within communities is higher than in between communities (Girvan and Newman, 2002). To quantify the **quality** of a community structure, several objective measures are studied in the literature. The most widely used quality measure, introduced by Newman and Girvan (2004), is called **modularity**. According to Newman and Girvan (2004): *"modularity measures the fraction of the edges in the graph that connect vertices of the same type (i.e., within-community edges) minus the expected value of the same quantity in a graph with the same degree sequence and same community divisions, but random connections between vertices"*. Mathematically, the modularity is given by:

$$Q = \sum_{k=1}^{K} \left[\frac{e_k}{e} - \left(\frac{d_k}{2e} \right)^2 \right] \tag{2.2}$$

where K is the number of communities, e_k is the number of edges within community k, d_k is the total degree of nodes in community k, and e is the total number of edges in the graph. If the number of within-community edges is no better than random, then $Q = 0$. Values approaching to $Q = 1$ indicate a strong (high quality) community structure. Consequently, the community detection problem can be formally defined as follows:

Definition 2 *Given a graph $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$, the community detection problem seeks to find a vertex partition $\Pi = \{\mathcal{V}_1, \mathcal{V}_2, ...\}$ that maximizes modularity Q.*

Notice that the number of partitions K is implicitly optimized in the above definition. Moreover, the balance criterion required for GPVS is not required for community detection. This makes the latter more suitable for automatic detection of BBD structures.

In the context of community detection, the edges between different communities can be considered as **noise** which prevents the complete separation of communities from each

other. In other words, removing the "noisy" edges will yield at least K connected components. Recall that the graph representation of a matrix is bipartite. Hence, an edge always connects a pair of row and column vertices, and a noisy edge can be removed by removing one of the vertices it connects. Specifically, since we consider the detection of singly BBD structures with row borders, one may only remove the vertices from the set of row vertices $\mathcal{R}$. Therefore, in a similar fashion, we call a vertex with at least one **out-of-community** connection **a noisy (row) vertex**. The following definition establishes the connection between community detection and BBD structure detection.

Definition 3 *Given the bipartite graph representation $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$ of $\boldsymbol{A}$, the BBD structure detection problem seeks to find the community structure $\Pi = \{\mathcal{V}_1, \mathcal{V}_2, ..., \mathcal{V}_K\}$ and the smallest set of noisy vertices $\mathcal{S} = \{s_1, s_2, ..., s_K\} \subseteq \mathcal{R}$ that maximize a given objective function $\Gamma(\Pi, \mathcal{S})$ and satisfy*

$$\boldsymbol{\mathcal{G}}[\mathcal{V} \setminus \mathcal{S}] \equiv \bigcup_{k=1}^{K} \boldsymbol{\mathcal{G}}[\mathcal{V}_k \setminus s_k]$$

$$\bigcup_{k=1}^{K} \mathcal{V}_k = \mathcal{V}$$

where $\mathcal{V} = \mathcal{R} \cup \mathcal{C}$, $\boldsymbol{\mathcal{G}}[\mathcal{V} \setminus \mathcal{S}]$ is the subgraph induced by removing the set of vertices $\mathcal{S}$, and $\boldsymbol{\mathcal{G}}[\mathcal{V}_k \setminus s_k]$ is the subgraph induced by the set of vertices $\mathcal{V}_k \setminus s_k$.

Notice that in Definition 2, the set $\mathcal{S}$ is uniquely defined (as the smallest set of noisy vertices) for a given Π and can be found, in principle, by solving the underlying optimization problem.

Literature on the detection of underlying BBD structures in sparse matrices is very limited (Weil and Kettler, 1971; Borndörfer et al., 1998; Ferris and Horn, 1998; Aykanat et al., 2004; Wang and Ralphs, 2013; Bergner et al., 2015). Weil and Kettler (1971) propose a method based on iterative permutation of rows and columns in the matrix according to a *north-west corner rule*. Borndorfer et al. (1998) discuss the relation of the BBD structure detection problem to well-known combinatorial optimization problems and propose a branch-and-cut algorithm for the exact solution of the detection problem.

To the best of our knowledge, Ferris and Horn (1998) are the first to investigate the BBD structure detection problem explicitly in the context of decomposition of LP. They represent the constraint matrices of linear programs as bipartite graphs and propose a two-phase approach for partitioning these graphs. The initial partitions obtained by spectral partitioning methods are then locally refined heuristically (Kernighan and Lin, 1970). Based on the refined partitions, they greedily place rows/columns to the border area by examining the "out-of-partition" degrees of each vertex. This approach is similar to the one we adopt in this work. Another important contribution of their study is the introduction of a goodness measure for the detected BBD structures, which accounts for both the homogeneity of the block sizes and the size of the border area. Their experiments on a large number of LP instances from NETLIB library indicate that BBD structure detection is a promising direction for the solution of large linear programs.

Aykanat et al. (2004) propose two mathematical models based on bipartite graph and hypergraph representation of matrices. They reduce the problem of permuting sparse matrices to BBD forms to the well-known GPVS and hypergraph partitioning (HP). Both GPVS and HP are known to be NP-hard problems (Bui and Jones, 1992; Lyaudet, 2010). They compare the performance of three graph/hypergraph partitioning methods: PaToH (Catalyurek and Aykanat, 1999), MeTiS (Karypis and Kumar, 1998b), and FH (Ferris and Horn, 1998). They conclude that in terms of partition quality (i.e., the percentage of border area), FH performs significantly worse than PaToH and MeTiS. They further report that, as expected, the processing times of the hypergraph model (PaToH) are much higher than those of graph models (MeTiS and FH) for the majority of the instances tested.

Bergner et al. (2015) propose a method for BBD structure detection that is well-suited for DWR which has been shown to produce high-quality solutions for well-structured mixed integer programs (Vanderbeck and Wolsey, 2010). Automatically applying DWR to a given problem with BBD structure has been implemented in several studies (Gamrath and Lübbecke, 2010; Puchinger et al., 2011; Ralphs and Galati, 2009; Vanderbeck, 2005). The hinderance DWR faces to function as a general-purpose tool is the requirement to input a BBD structure inherent to the constraint matrix. Bergner et al. (2015) pave the way to address this issue. They attempt to detect BBD structures that provide tight dual bounds at the root node of DWR (i.e., Lagrangian bounds). They propose a structure detection

algorithm based on hypergraph partitioning hMETIS (Karypis and Kumar, 1998a). They conduct a thorough analysis of potential proxy measures (e.g., number of blocks, percentage of border area, average density of blocks, average integrality gap of subproblems) for the goodness of a detected BBD structure in the context of DWR and conclude that "percentage of border area" is the best proxy measure among those investigated. It is unfortunate that they discard all other measures in favor of one which is essentially very intuitive, as a smaller border area is expected to provide a better bound. Nevertheless, their experiments with instances from MIPLIB2003/2010 (Achterberg et al., 2006; Koch et al., 2011) serve as a benchmark for the studies in this line of research, including ours. Their results show that in many instances, the bounds obtained by the commercial mixed integer programming solver CPLEX at the root node can be significantly improved by such decomposition approaches.

In their technical report, Witt and Lübbecke (2015) investigate the strength of DWRs of the classical edge formulation for the maximum weighted stable set problem. They argue that, since every constraint in this model corresponds to an edge of the underlying graph, formulating a DWR can be described as choosing a subgraph and convexifying all constraints corresponding to edges of this subgraph. They characterize DWRs not yielding a stronger LP relaxation as reformulations where this subgraph is bipartite and present a characterization of DWRs with the strongest possible LP relaxation as reformulations where the chosen subgraph contains all odd holes (and 3-cliques). This work is one of the first attempts to theoretically characterize the strength of all possible DWRs of a well-known optimization problem; namely, the Stable Set Problem.

The problem of detecting BBD structure may be related to several problems in the data-mining literature. For example, a well-known problem called *biclustering* tries to simultaneously partition the set of samples and the set of their attributes (features) into subsets (classes) in such a way that samples and features classified together are highly relevant to each other. In a matrix representation, this corresponds to a re-ordering of the matrix in such a way that the majority of nonzeros are covered within diagonal blocks with very high density. Busygin et al. (2008) review the most used and successful biclustering techniques and their related applications. The survey summarizes mathematical concepts that can be met in existing biclustering techniques from a theoretical viewpoint.

The main disadvantage of graph partitioning algorithms used in Ferris and Horn (1998), Aykanat et al. (2004) and Bergner et al. (2015) is the need to specify the number of partitions beforehand. In the absence of information on the underlying structure of a matrix, this requirement may lead to detection of inferior structures. As mentioned earlier, community detection implicitly optimizes the number of partitions which has made it a very popular tool in several real life applications such as biological networks, social networks, and citation networks (Girvan and Newman, 2002; Rosvall and Bergstrom, 2007). For a general background on community detection, the reader is referred to Fortunato (2010) and Porter et al. (2009). To our knowledge, community detection has not been investigated in the context of BBD structure detection.

Assessing the quality of detected community structures is one of the debated topics in the related literature. Porter et al. (2009) discuss the effectiveness of various quality functions proposed in the literature. Fortunato (2010) states that the most popular quality function is the modularity of Newman and Girvan (2004). Good et al. (2010) investigate the performance of modularity maximization in practical contexts.

Finding the optimal community structure in a complex network is shown to be NP-hard (Brandes et al., 2008). There are, however, several algorithms such as *Walktrap* (Pons and Latapy, 2006), *Edge-betweenness* (Girvan and Newman, 2002), *Infomap* (Rosvall and Bergstrom, 2007), and *Fast-greedy* (Clauset et al., 2004) to find sub-optimal community structures with polynomial complexity. Lancichinetti et al. (2008) provide a set of benchmark graphs on which the performances of different algorithms can be tested. Lancichinetti and Fortunato (2009) and Leskovec et al. (2010) compare the performances of various community detection algorithms on selected benchmark graphs. Lancichinetti and Fortunato (2009) conclude that based on the benchmark graphs, three algorithms introduced by Rosvall and Bergstrom (2007); Blondel et al. (2008) and Ronhovde and Nussinov (2009) appear to have the best performance in terms of the quality of community structures, with the additional advantage of low computational complexity.

The main contribution of this work is a new BBD structure detection approach based on community detection that does not require the input of the number of blocks. Second, we propose a new measure of goodness for BBD structures, that successfully accounts for the trade-off between the quality of the block-diagonal and the border components.

Finally, we conduct a comparative computational study that provides valuable insights on the untapped research potential within the realm of structure detection in optimization.

The rest of the chapter is organized as follows: In Section 2, we propose a new measure of goodness for BBD structures. In Section 3, we introduce our approach for structure detection. In Section 4, we report results from the proposed approach on test instances from the MIPLIB2003/2010 (Achterberg et al., 2006; Koch et al., 2011). In Section 5, we discuss several observations and potential research avenues. Conclusions are provided in Section 6.

2.2 Goodness of BBD Structures

In principle, any sparse matrix can be permuted into a BBD form by placing a sufficiently large number of rows into the border component. In fact, Bergner et al. (2015) experimentally show that often more than one such permutation is possible. This necessitates the use of an objective measure to evaluate the goodness of alternative BBD structures inherent in sparse matrices. It is evident from the nature of BBD structures that there are two components, border and block-diagonal, that should be accounted for when comparing alternative structures.

Consider a hypothetical 1825×1500 matrix $\mathbf{A}$ and its different permutations shown in Figure 2.1, where white (black) cells correspond to zero (nonzero) entries in the matrix. As seen, Figure 2.1.(a) has a BBD form with two diagonal blocks of sizes 1025×900 and 700×600 and a border area of 100 rows. Figure 2.1.(b) is obtained from Figure 2.1.(a) by moving 100 more rows from the block-diagonal component into the border component and has five **non-identical** blocks of equal sizes 325×325. Finally, Figure 2.1.(c) is obtained from Figure 2.1.(b) by moving 125 more rows from the block-diagonal component into the border component and has fifteen **identical** blocks of sizes 100×100. Figure 2.1 provides a simple example of how the size of the border component and the decomposition of the block-diagonal component are correlated with each other. Particularly, by increasing the size of the border component (i.e., placing more rows into the border component), the (remaining) block-diagonal component can be decomposed into a larger number of blocks.

More generally, a change in the composition of the border component inevitably leads to a change in the composition of the block-diagonal component. Hence, the qualities of the two components should also be correlated. In what follows, we first separately discuss the characteristics that are related to the qualities of the border and block-diagonal components and then attempt to combine them into a single goodness measure for BBD structures.

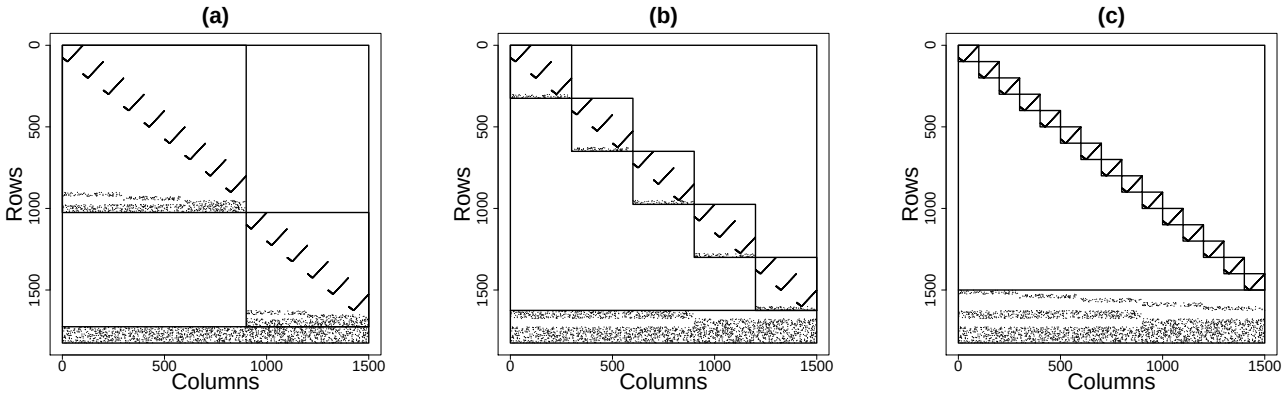

Figure 2.1: Permutations of a hypothetical matrix into different BBD forms

2.2.1 Quality of the block-diagonal component

In the context of Lagrangian relaxation, each block in the block-diagonal component corresponds to a Lagrangian subproblem that is to be solved at every iteration with a different set of Lagrangian multipliers until the best bound is found. Therefore, from a computational efficiency point of view, it is desirable to have subproblems with small sizes. One way to achieve this is to have as many subproblems as possible. We refer to this desirable characteristic as **granularity** and express it as the number of blocks in the block-diagonal component.

It is worth noting that granularity alone does not guarantee computational efficiency. Consider an extreme case where the block-diagonal component has 10 blocks, 9 of which are negligibly small compared to the 10^{th}. In this case, the subproblem corresponding to

the 10^{th} block will be of size nearly equal to the original problem. Having to solve this subproblem at every iteration will render this Lagrangian relaxation computationally inefficient. To avoid such issues, another desirable characteristic can be defined as having (nearly) equal-sized blocks, which we refer to as **homogeneity**. Homogeneity is especially desirable for parallel processing; namely, solving subproblems in parallel on different processors rather than in sequence on a single processor. The overall computation time of such a parallel processing mostly depends on the size of the largest subproblem. Hence, having high granularity and homogeneity together ensures computational efficiency of the Lagrangian relaxation.

There is yet another desirable characteristic which further contributes to the computational efficiency. Consider a block-diagonal component with not only equal-sized but **identical** blocks. This implies that the Lagrangian subproblems are identical (assuming the objective function coefficients and the right hand side values are also identical). In this case, solving only one of the subproblems at every iteration is sufficient for Lagrangian relaxation. We refer to this feature as **isomorphism**. Unfortunately, it is not as straightforward to express the degree of isomorphism as it is to express granularity and homogeneity.

To illustrate the aforementioned characteristics, consider the block-diagonal components of the matrix permutations in Figure 2.1. As mentioned earlier, Figure 2.1.(a) has two blocks of unequal sizes which indicates low granularity and homogeneity. Figure 2.1.(b), on the other hand, has higher granularity (with 5 blocks) and homogeneity (with equal sizes) compared to Figure 2.1.(a). Finally, Figure 2.1.(c) has the highest granularity and homogeneity as well as an additional advantage of isomorphism among its blocks. Therefore, one would expect a good quality measure to provide a ranking of $(a) < (b) < (c)$ of the block-diagonal components. In pursuit of such an objective quality measure, we investigated the modularity function Q, given in (2.2), as a candidate. Note that Q takes values in the range $[0, 1)$ where the minimum score corresponds to a matrix with a single block and the maximum score corresponds to **the equal distribution of non-zeros among as many separable blocks as possible**. Although equal distribution of non-zeros does not guarantee homogeneity or isomorphism, it can be used as a proxy for both. Our initial experiments on the test instances showed that if a BBD structure with identical blocks

is indeed inherent in the constraint matrix, a significantly higher modularity score (compared to other alternatives) is achieved by its block-diagonal component. To illustrate the effectiveness of modularity, we calculated the modularity scores of the three block-diagonal components in Figure 2.1 and the scores ($Q_{(a)} = 0.42 < Q_{(b)} = 0.78 < Q_{(c)} = 0.93$) concurred with our expectation to discriminate in favor of high granularity, homogeneity, and isomorphism. Therefore, we adopted modularity as the quality measure of the block-diagonal component in subsequent computational experiments.

2.2.2 Quality of the border component

Unlike block-diagonal components, quality of border components has been expressed with a simple proxy measure in the literature; namely, the percentage of border area which is calculated as the ratio of the number of rows in the border area to the total number of rows. The subset of constraints included in the border component dictates the tightness of the Lagrangian bound, or the optimality gap closed. Consequently the fewer the relaxed constraints, the better is the expected Lagrangian bound. Although there is no causal relation between the percentage of border area and the optimality gap closed, Bergner et al. (2015) experimentally show that there is a strong negative correlation between the two.

Interestingly, despite the clear nonlinearity of the correlation, Bergner et al. (2015) opt for using percentage of border area as a measure for goodness as is. To remedy this, we propose a simple exponential decay function of percentage of border area x, $P(x) = e^{-\lambda x}$, to measure the quality of the border component, where λ is a parameter to tune the effect of adding (removing) a single row to (from) the border component. Notice that the quality score P takes values in the range (0,1] where the minimum score corresponds to the border component being equal to $\mathbf{A}$ and the maximum score corresponds to the border component being empty. Notice also that the marginal effect of changing the percentage of border area decreases as the size of the border component increases and becomes negligible beyond a certain threshold. In the context of Lagrangian relaxation, this threshold point ideally corresponds to the point where the Lagrangian bound equals the LP bound. However, it is not easy to identify this point without solving the resulting decomposition problems at

every step. Practically, the parameter λ can be used to tune the threshold point.

2.2.3 Goodness score

In studying the quality of noisy block-diagonal structures, Suresh Kumar and Chandrasekharan (1990) suggest that a quality measure should have the following properties:

- *Non-negativity and scalability:* Scores are within a fixed range of non-negative values irrespective of size and sparsity of matrices.
- *Physical meaning of extrema:* The minimum and maximum scores correspond to special cases.
- *High discrimination power:* Good/bad structures are consistently scored as high/low.

We already established that both quality measures Q and P hold the aforementioned properties. As mentioned earlier, improving the quality of the block-diagonal component inevitably deteriorates the quality of the border component. Intuitively, then, the goodness of a BBD structure should depend on finding the best compromise between the two conflicting objectives. To this end, we combine the quality functions defined for the block-diagonal and the border components into a single expression to measure the goodness of BBD structures.

Definition 4 *Given a permutation* $\boldsymbol{A}^{\pi} = \begin{bmatrix} \boldsymbol{D} \\ \boldsymbol{B} \end{bmatrix}$ *of an* $m \times n$ *matrix* $\boldsymbol{A}$ *into a BBD structure where the* $d \times n$ *matrix* $\boldsymbol{D}$ *is the block-diagonal component and the* $b \times n$ *matrix* $\boldsymbol{B}$ *is the border component, the following functions measure the quality of* $\boldsymbol{D}$*,* $\boldsymbol{B}$ *and the goodness of* $\boldsymbol{A}^{\pi}$*, respectively:*

$$Q(\boldsymbol{A}^{\pi}) = \sum_{i=1}^{K} \frac{e_i}{e} \left[1 - \frac{e_i}{e}\right] \tag{2.3}$$

$$P(\boldsymbol{A}^{\pi}, \lambda) = e^{-\lambda \frac{b}{m}} \tag{2.4}$$

$$\Gamma(\boldsymbol{A}^{\pi}, \lambda) = Q(\boldsymbol{A}^{\pi}) P(\boldsymbol{A}^{\pi}, \lambda) \tag{2.5}$$

where K is the number of diagonal blocks, e_i is the number of non-zero entries in block i, and e is the total number of non-zero entries in matrix $\boldsymbol{D}$.

Recall that both Q and P can take values between 0 and 1. Hence, $\Gamma \in [0, 1)$, with Γ approaching 0 corresponds to a BBD structure with a single block (or empty block-diagonal component) and Γ approaching 1 corresponds to a BBD structure with empty border component and $K = min(m, n)$ isomorphic blocks in the block-diagonal component. In the proposed goodness measure, λ can be interpreted as a trade-off parameter between the quality of the block-diagonal and the border components. In particular, a large λ value would prioritize the quality of the border component over the quality of the block-diagonal component. In Section 2.4, we illustrate the quality and goodness functions with an example and investigate the effect of λ on the output.

Next, we introduce a BBD structure detection approach based on community detection and maximization of goodness score.

2.3 A Community Detection Based Approach

To motivate the proposed approach, we begin by discussing the effect of specifying the number of blocks K on the structures detected. As noted earlier, this is one of the drawbacks of the existing approaches. Figure 2.2 shows the original constraint matrix of the *10teams* instance from MIPLIB2003 (Achterberg et al., 2006) and two permutations with different detected BBD forms.

Notice that the border components in Figure 2.2-(b) and Figure 2.2-(c) are identical, whereas the same block-diagonal component is permuted differently and decomposed into different number of blocks, namely 2 and 9, respectively. This is a typical consequence of requiring user input for the number of blocks. Particularly, when K is set to 2 for any of the BBD structure detection algorithms existing in the literature, the structure in Figure 2.2-(b) is mostly likely to be detected despite being clearly inferior to the structure in Figure 2.2-(c). With that in mind, it can be trivially shown that the most granular (efficient) decomposition of a matrix $\mathbf{D}$ into diagonal blocks is uniquely defined and corresponds to the

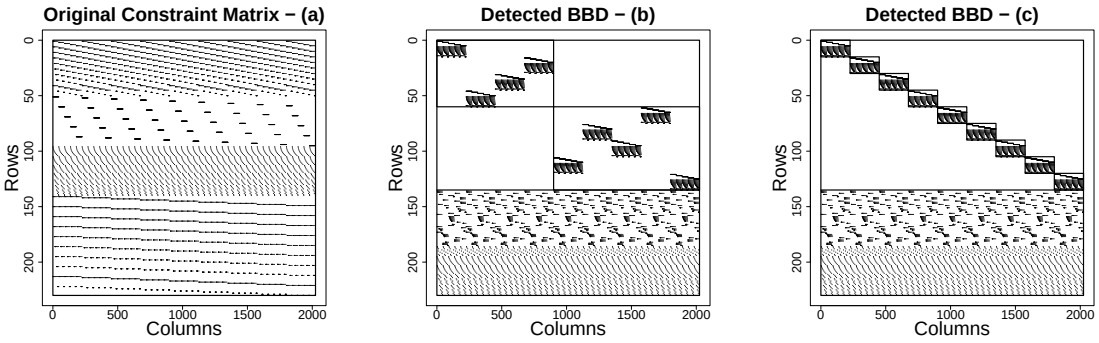

Figure 2.2: Constraint matrix of original 10teams instance and detected BBD structures

decomposition of its bipartite graph representation into connected components (Hopcroft and Tarjan, 1973). This provides a key insight that constitutes the main idea behind our approach: For a given set of rows in the border component (which corresponds to $\mathcal{S}$ in bipartite graph representation), the most granular partitioning of the remaining vertices is uniquely defined.

Recall that we argued the uniqueness of $\mathcal{S}$ for a given community structure Π in Definition 2. This, combined with the aforementioned observation, implies that there is a bilateral relationship between Π and $\mathcal{S}$ which can be exploited.

Notice that finding a candidate Π or $\mathcal{S}$ does not require defining the number of blocks *a priori*. For matrices with unknown inherent structures, this may prove very valuable since enumerating the number of partitions via trial and error within a specified range has several drawbacks. First, it requires multiple execution of structure detection algorithm as in Ferris and Horn (1998); Aykanat et al. (2004) and Bergner et al. (2015) which may be computationally burdensome, especially for large matrices. Second, the "best" inherent structure may have the number of blocks outside the range of predefined number of partitions, in which case such heuristics would detect inferior structures. Finally, by forcing a predefined number of partitions, one may end up having blocks that are further decomposable. In Section 4, we provide examples of such issues on the instances selected from MIPLIB2003/2010 (Achterberg et al., 2006; Koch et al., 2011).

Our proposed algorithm is based on iterative execution of two phases called Remove

and MERGE. At the REMOVE phase, the bipartite graph $\boldsymbol{\mathcal{G}}[\mathcal{V} \setminus \mathcal{S}]$ corresponding to the block-diagonal component (initially, $\boldsymbol{\mathcal{G}}$) is decomposed into communities and row vertices are greedily removed from $\boldsymbol{\mathcal{G}}[\mathcal{V} \setminus \mathcal{S}]$ (i.e., placed into the border component $\mathcal{S}$) to improve the quality of the block-diagonal component. Specifically, the row vertices with the highest "out-of-community" degree (i.e., number of edges a vertex has with other communities) are removed at each sub-iteration. Removal of rows continues until a noiseless (i.e., separable) block-diagonal component is attained.

At the MERGE phase, for each row in the border component $\mathcal{S}$, the "minimal" merge is identified as the list of blocks it couples. For example, if a row in $\mathcal{S}$ has three non-zeros at columns $\{j_1, j_2, j_3\}$ and the blocks corresponding to those columns are $\{k_1, k_2, k_3\}$, then its minimal merge is $\mathcal{M} = \{k_1, k_2, k_3\}$. Typically, the minimal merges for some rows turn out to be identical, i.e., some merges result in the relocation of multiple rows back to the block-diagonal component. The best merge is greedily identified at each sub-iteration as the one that provides the minimum marginal reduction in the quality score of the block-diagonal component. Merging of blocks continues until there are only two blocks left in the block-diagonal component. Each sub-iteration in REMOVE or MERGE generates an alternative BBD structure uniquely defined by $\boldsymbol{\mathcal{G}}$ and $\mathcal{S}$. Since $\boldsymbol{\mathcal{G}}$ is the same for all alternative structures, it suffices to record only the alternative vertex separators generated at each sub-iteration in a list $\mathbf{L}$.

The two-phase routine is executed iteratively until an identical set of vertex separators in $\mathbf{L}$ is regenerated or a predetermined number of iterations (t_{max}) is reached. The list of all alternative structures are outputted by the algorithm and evaluated based on the goodness score defined in (6) to retain the one with the highest score. Notice that the generation of the alternative BBD structures is independent of the λ tuning parameter and that λ is only used to identify the best structure among all alternatives.

Tables 2.1 and 2.2 provide the notation and the list of functions used in the definition of the algorithm, respectively, whereas the pseudo-code is given in Algorithm 1. The algorithm makes use of three data structures: *graph*, *set*, and ordered *list*. Set representation and operators (i.e., " $\cup$ ", " $\setminus$ ", " $\in$ ", " $\equiv$ ") are used when the order of the elements does not matter and an element cannot be repeated more than once. For example, $\{1, 2, 3\} \cup \{2, 4\} \equiv \{1, 2, 3, 4\}$. To make the distinction of different data structures

clear, in Algorithm 1 we used bold calligraphic letters (e.g., $\boldsymbol{\mathcal{G}}$) for graphs, calligraphic letters for sets (e.g., $\mathcal{S}$, $\mathcal{A}$, $\mathcal{M}$), bold lowercase letters for lists of scalars (e.g., $\mathbf{r}$, $\mathbf{n}$), and bold uppercase letters for lists of sets (e.g., $\mathbf{L}$, $\mathbf{\Pi}$, $\mathbf{U}$). Note that brackets "[ ]" are used to denote both induced subgraphs (e.g., $\boldsymbol{\mathcal{G}}[\mathcal{V}]$) and a specific element of an ordered list (e.g., $\mathbf{n}[i]$).

Table 2.1: Descriptions of the symbols used in Algorithm 1

Notation	Description
$\mathbf{A}$	An $M \times N$ matrix
$\boldsymbol{\mathcal{G}} = (\mathcal{R} \cup \mathcal{C}, \mathcal{E})$	Bipartite **graph** representation of $\mathbf{A}$
$\mathcal{V}, \mathcal{R}, \mathcal{C}$	**Set** of all, row, and column vertices in a bipartite graph $\boldsymbol{\mathcal{G}}$, respectively
$\boldsymbol{\mathcal{G}}[\mathcal{V}]$	Sub**graph** of $\boldsymbol{\mathcal{G}}$ induced by vertex set $\mathcal{V}$
$\mathcal{A}$	**Set** of vertices adjacent to a given vertex
$\mathbf{\Pi}$	A **list** whose entries are the sets of vertices in each community of a given graph
$\mathbf{B}$	A **list** whose entries are the sets of vertices in each connected component of a given graph
$\mathbf{n}$	**List (scalar)** of out-of-community degrees (noises) of each row vertex
$\mathcal{S}$	**Set** of vertex separators
$\mathbf{s}$	**List (scalar)** representation of vertex separators in increasing index order
$\mathbf{r}$	**List (scalar)** representation of remaining row vertices in increasing index order
$\Delta\mathcal{S}$	**Set** of vertices added to or removed from $\mathcal{S}$
$\mathcal{M}$	**Set** of blocks in a minimal merge
$\mathbf{M}$	**List** of minimal merges for each row in $\mathcal{S}$
$\mathbf{L}$ ($*\mathbf{L}$)	The **list** (pointer) of vertex separators obtained at each iteration
$\boldsymbol{\gamma}$	**List (scalar)** of goodness scores for each alternative BBD structure stored in $\mathbf{L}$
t_{max}	Maximum number of iterations

As seen in the pseudo-code, the community detection is one of the key subroutines in the algorithm and is executed at every iteration. This calls for a fast community detection method, possibly at the expense of reduced quality. In order to choose the most suitable method, we tested several computationally efficient methods in the literature and observed that *fast-greedy* algorithm of Clauset et al. (2004) performs significantly faster than the other methods, yields community structures with comparable quality and does not require any parameter-tuning. Hence, we opted for using *fast-greedy* algorithm as the base community detection method in this study. The algorithm tries to optimize the modularity function in a greedy manner. Initially, every vertex belongs to a separate community, and communities are merged iteratively such that each merge is locally optimal (i.e., yields the largest increase in the current value of modularity). The algorithm stops when it is

Algorithm 1 BBD Structure Detection Algorithm: KEE

function KEE($\boldsymbol{\mathcal{G}}, t_{max}$)
  $\mathbf{L}[\,] \leftarrow \varnothing$, $\mathcal{S} \leftarrow \varnothing$, $t \leftarrow 0$, *check* $\leftarrow$ **true**
  while *check* **do**
    check $\leftarrow$ REMOVE($\boldsymbol{\mathcal{G}}, \mathcal{S}, \mathbf{L}$, *check*)
    $\mathbf{B} \leftarrow$ decompose($\boldsymbol{\mathcal{G}}[\mathcal{V} \setminus \mathcal{S}]$)
    if $|\mathbf{B}| \leq 2$ **then**
      check $\leftarrow$ **false**
    check $\leftarrow$ MERGE($\boldsymbol{\mathcal{G}}, \mathcal{S}, \mathbf{L}$, *check*)
    $t \leftarrow t + 1$
    if $t \geq t_{max}$ **then**
      check $\leftarrow$ **false**
  return $\mathbf{L}$

function REMOVE($\boldsymbol{\mathcal{G}}, *\mathcal{S}, *\mathbf{L}$, *check*)
  while *check* **do**
    $\mathbf{r} \leftarrow$ list($\mathcal{R} \setminus \mathcal{S}$)
    $\boldsymbol{\Pi} \leftarrow$ community($\boldsymbol{\mathcal{G}}[\mathcal{V} \setminus \mathcal{S}]$)
    $\mathbf{n} \leftarrow$ noises($\boldsymbol{\mathcal{G}}[\mathcal{V} \setminus \mathcal{S}], \boldsymbol{\Pi}$)
    if max($\mathbf{n}$) > 0 **then**
      $\Delta\mathcal{S} \leftarrow \{\mathbf{r}[i] \mid \mathbf{n}[i] = \max(\mathbf{n})\}$
      if is.in($\mathcal{S} \cup \Delta\mathcal{S}, \mathbf{L}$) **or** $\{\mathcal{S} \cup \Delta\mathcal{S}\} \equiv \mathcal{R}$ **then**
        check $\leftarrow$ **false**
      else
        $\mathcal{S} \leftarrow \mathcal{S} \cup \Delta\mathcal{S}$
        $\mathbf{L} \leftarrow$ append($\mathcal{S}, \mathbf{L}$)
    else
      check $\leftarrow$ **false**
  return *check*

function MERGE($\boldsymbol{\mathcal{G}}, *\mathcal{S}, *\mathbf{L}$, *check*)
  $\mathbf{B} \leftarrow$ decompose($\boldsymbol{\mathcal{G}}[\mathcal{V} \setminus \mathcal{S}]$)
  while $|\mathbf{B}| > 2$ **and** *check* **do**
    $\mathbf{s} \leftarrow$ list($\mathcal{S}$)
    $\mathbf{M}[\,] \leftarrow \varnothing$
    for every $v \in \mathcal{S}$ **do**
      $\mathcal{A} \leftarrow$ neighbor($v, \boldsymbol{\mathcal{G}}$)
      $\mathcal{M} \leftarrow \varnothing$
      for every $u \in \mathcal{A}$ **do**
        $k \leftarrow$ block.of($u, \mathbf{B}$)
        $\mathcal{M} \leftarrow \mathcal{M} \cup \{k\}$
      $\mathbf{M} \leftarrow$ append($\mathcal{M}, \mathbf{M}$)
    $\mathbf{U} \leftarrow$ unique($\mathbf{M}$)
    $\mathbf{q}[\,] \leftarrow \varnothing$
    for $j = 1$ **to** $|\mathbf{U}|$ **do**
      $\Delta\mathcal{S} \leftarrow \{\mathbf{s}[i] \mid \mathbf{M}[i] \equiv \mathbf{U}[j]\}$
      $\Delta q \leftarrow$ quality.D($\boldsymbol{\mathcal{G}}, \mathcal{S} \setminus \Delta\mathcal{S}$) $-$ quality.D($\boldsymbol{\mathcal{G}}, \mathcal{S}$)
      $f \leftarrow$ frequency($\mathbf{U}[j], \mathbf{M}$)
      $\mathbf{q} \leftarrow$ append($\Delta q / f, \mathbf{q}$)
    $\Delta\mathcal{S} \leftarrow \{\mathbf{s}[i] \mid \mathbf{M}[i] \equiv \mathbf{U}[j], \mathbf{q}[j] = \max(\mathbf{q})\}$
    if is.in($\mathcal{S} \setminus \Delta\mathcal{S}, \mathbf{L}$) **or** $\{\mathcal{S} \setminus \Delta\mathcal{S}\} \equiv \varnothing$ **then**
      check $\leftarrow$ **false**
    else
      $\mathcal{S} \leftarrow \mathcal{S} \setminus \Delta\mathcal{S}$
      $\mathbf{L} \leftarrow$ append($\mathcal{S}, \mathbf{L}$)
  return *check*

Table 2.2: Descriptions of the functions used in Algorithm 1

Function	Output	Description
graph($\mathbf{A}$)	$\boldsymbol{\mathcal{G}}$	Constructs the bipartite representation of the matrix $\mathbf{A}$
noises($\boldsymbol{\mathcal{G}}, \boldsymbol{\Pi}$)	$\mathbf{n}$	Calculates the noise caused by each row vertex
neighbor($v, \boldsymbol{\mathcal{G}}$)	$\mathcal{A}$	Returns the vertices adjacent to the vertex v in the graph $\boldsymbol{\mathcal{G}}$
decompose($\boldsymbol{\mathcal{G}}$)	$\mathbf{B}$	Decomposes the graph $\boldsymbol{\mathcal{G}}$ into its connected components (blocks)
community($\boldsymbol{\mathcal{G}}$)	$\boldsymbol{\Pi}$	Decomposes the graph $\boldsymbol{\mathcal{G}}$ into communities which maximize modularity
block.of($v, \mathbf{B}$)	k	Returns the index of the block which vertex v belongs to in $\mathbf{B}$
list($\mathcal{S}$)	$\mathbf{s}$	Returns the list representation of $\mathcal{S}$ in increasing order of its elements
max($\mathbf{n}$)	n	Returns the entry with the largest value in the (scalar) list $\mathbf{n}$
is.empty($\mathbf{L}$)	$\mathbf{T/F}$	Checks whether the list $\mathbf{L}$ is empty
unique($\mathbf{M}$)	$\mathbf{U}$	Returns the list of distinct sets in the list $\mathbf{M}$
frequency($\mathcal{S}, \mathbf{L}$)	f	Returns the number of times an entry (the set $\mathcal{S}$) is repeated in the list $\mathbf{L}$
append($\mathcal{S}, \mathbf{L}$)	$\mathbf{L}$	Appends a new entry (the set $\mathcal{S}$) to the end of the list $\mathbf{L}$
is.in($\mathcal{S}, \mathbf{L}$)	$\mathbf{T/F}$	Checks whether the set $\mathcal{S}$ is an entry of the list $\mathbf{L}$
quality.D($\boldsymbol{\mathcal{G}}, \mathcal{S}$)	q	Calculates the quality of the block-diagonal component
goodness($\boldsymbol{\mathcal{G}}, \mathcal{S}, \lambda$)	g	Calculates the goodness of the BBD structure
REMOVE($\boldsymbol{\mathcal{G}}, *\mathcal{S}, *\mathbf{L}, check$)	$check$	Iteratively removes the set of rows causing maximum noise
MERGE($\boldsymbol{\mathcal{G}}, *\mathcal{S}, *\mathbf{L}, check$)	$check$	Iteratively merges the set of blocks providing minimum reduction in Q

not possible to increase the modularity anymore. The computational complexity of *fast-greedy* algorithm is $O(|\mathcal{V}|log^2|\mathcal{V}|)$, where $|\mathcal{V}|$ is the number of vertices in the given graph (Lancichinetti and Fortunato, 2009).

Before presenting the computational results, we first discuss the validity and convergence of the algorithm. As argued before, a valid BBD structure can be uniquely defined by the bipartite graph representation $\boldsymbol{\mathcal{G}}$ of matrix $\mathbf{A}$ and a vertex separator $\mathcal{S} \subseteq \mathcal{R}$. It can be seen from Algorithm 1 that at each sub-iteration of REMOVE and MERGE, the output ($\mathbf{L}$) of KEE is either updated by adding a new vertex separator or kept the same in which case the algorithm is terminated. The extreme cases where KEE returns an empty $\mathbf{L}$ or all vertex separators in $\mathbf{L}$ are 1-way separators correspond to BBD structures with a single block. In Section 2.3., we discussed that such (trivial) structures attain lowest possible goodness score ($\Gamma = 0$). They are, nevertheless, considered valid BBD structures which ensures the validity of the algorithm.

Next, it becomes evident from Algorithm 1 that KEE terminates when one of the following two conditions is satisfied: (i) a previously generated structure is regenerated or (ii) a predefined number of iterations (t_{max}) is reached. There are $2^{|\mathcal{R}|}$ distinct vertex separa-

tors, each corresponding to a different BBD structure. Hence, theoretically, even without enforcing the second termination condition, the algorithm is guaranteed to converge (i.e., terminate after finitely many iterations). However, the theoretical limit on the number of maximum iterations is astronomically high even for moderate size problems. Therefore, to ensure a timely termination, we used $t_{max} = 100$ in our computational experiments. An alternative way could be to set a limit on the total processing time or the number of distinct BBD structures generated (i.e., number of sub-iterations). It is worth noting that for all of the instances included in our computational experiments, the first condition was sufficient for a timely termination of the algorithm. However, in general, enforcing the first condition may lead to premature termination of the algorithm before finding the best structure, due to cycling.

2.4 Computational Experiments

In this section, we present a comprehensive computational analysis and comparison with the state-of-the-art, namely Bergner et al. (2015). We start by describing the experimental framework, we then analyze the structures detected by our proposed approach and compare against Bergner et al. (2015).

2.4.1 Experimental Framework

The testing is done on the benchmark instances of Bergner et al. (2015) that are extracted from MIPLIB2003/2010 (Achterberg et al., 2006; Koch et al., 2011). Two criteria are used for comparison, namely CPU time required to obtain the best Lagrangian bound and the tightness of the Lagrangian bound.

Bergner et al. (2015) introduce an algorithm, *DWR*, to detect alternative BBD structures and two variations of the algorithm, *DWR auto* and *DWR best*, to identify the best structure among the alternatives. According to their description, *DWR* executes hMETIS (Karypis and Kumar, 1998a) with different input parameters such as number of partitions and number of dummy variables to obtain alternative structures. *DWR auto* identifies

the best structure as the one with smallest border area. *DWR best*, on the other hand, selects the structure that yields the best Lagrangian bound. They state that "*DWR best* is extremely time consuming and only meant as an assessment of the method's potential". In a similar fashion, we call our proposed algorithm *KEE*, which detects alternative BBD structures in a given matrix. The main difference from *DWR* is that *KEE* requires no user input apart from the matrix itself. To be able to make a fair comparison, we also provide two variations of our algorithm. The first variation selects the structure with the best goodness score that accounts for the trade-off between the quality of the block-diagonal and the border components; namely, when the tuning parameter $\lambda = 5$; whereas, the second variation considers only the quality of the border component ($\lambda = 20$). We refer to them as *KEE auto* and *KEE best*. Algorithm 2 provides the pseudo-codes for the variations.

Algorithm 2 Variations of KEE Algorithm

function KEE.VARIATION($\mathbf{A}, var, t_{max}$)
  $\boldsymbol{\mathcal{G}} \leftarrow$ graph($\mathbf{A}$)
  $\mathbf{L} \leftarrow$ KEE($\boldsymbol{\mathcal{G}}, t_{max}$)
  if is.empty($\mathbf{L}$) **then**
    $\mathcal{S}_{best} \leftarrow \varnothing$
  else
    $\boldsymbol{\gamma}[\,] \leftarrow \varnothing$
    for $i = 1$ **to** $|\mathbf{L}|$ **do**
      if $var =$ "*auto*" **then**
        $g \leftarrow$ goodness($\boldsymbol{\mathcal{G}}, \mathbf{L}[i], \lambda = 5$)
        $\boldsymbol{\gamma} \leftarrow$ append(g, $\boldsymbol{\gamma}$)
      if $var =$ "*best*" **then**
        $g \leftarrow$ goodness($\boldsymbol{\mathcal{G}}, \mathbf{L}[i], \lambda = 20$)
        $\boldsymbol{\gamma} \leftarrow$ append(g, $\boldsymbol{\gamma}$)
    if max($\boldsymbol{\gamma}$) $= 0$ **then**
      $\mathcal{S}_{best} \leftarrow \varnothing$
    else
      $\mathcal{S}_{best} \leftarrow \{\mathbf{L}[i] \mid \boldsymbol{\gamma}[i] = \max(\boldsymbol{\Gamma})\}$
  return $(\boldsymbol{\mathcal{G}}, \mathcal{S}_{best})$

We used R programming environment for the implementation of the proposed algorithm and Gurobi 6.0.2, with default parameters and a single thread, to solve the Lagrangian subproblems and master problems. All the experiments were carried out on an Intel(R) Core(TM) i7-4510U PC (2.60GHz, 8GB memory) running Windows 8.1. We limited the processing time of all experiments to 1800 seconds and the best bound obtained until

termination was reported as the Lagrangian bound and was used to calculate the optimality gap for the cases where convergence was not achieved before termination. Note that the time limit was set to 3600 seconds in Bergner et al. (2015).

Finally, since the algorithm presented in Section 3 detects singly BBD structures, we only tested on the instances for which the best structure detected by Bergner et al. (2015) was reported to be a singly BBD structure. Although the extension of the algorithm to detect arrowhead (doubly BBD) structures which are potentially amenable to Lagrangian decomposition is possible, the quality of such a decomposition depends on several other factors such as types of linking variables (integer or continuous) and number of blocks linked by each variable. Accounting for these factors requires an extension to the goodness measure defined in Section 2. In this study, however, our focus is to illustrate the potential of the proposed fully-automated algorithm.

2.4.2 Structure Detection

We begin by analyzing the change of the quality scores, Q and P, and the goodness score Γ for a complete run of *KEE* algorithm. Figure 2.3 summarizes a typical run of *KEE* algorithm on the *reblock67* instance from MIPLIB2010 (Koch et al., 2011). As described in Section 2.3, our algorithm is based on an iterative execution of a two-phase routine: *REMOVE* and *MERGE*. The regions of the plots with white background correspond to the *REMOVE* phase and the shaded regions correspond to the *MERGE* phase. The algorithm is terminated when a previously generated structure is regenerated (to prevent infinite loops). Each plot illustrates the change of quality and goodness scores for different values of the tuning parameter ($\lambda \in \{1, 5, 20\}$). The dashed (thin straight) lines represent the change of the quality score of the block-diagonal (border) component, Q (P), and the thick straight lines represent the change of overall goodness score, Γ, throughout the complete execution of the algorithm. The marked points on the Γ lines indicate the highest goodness score attained throughout the execution; i.e., the best BBD structure detected.

Notice that the Q lines are the same for all three plots since the quality of the block-diagonal component does not depend on the tuning parameter λ. However, the P lines become more stretched as λ increases, aptly reflecting the increased marginal effect of

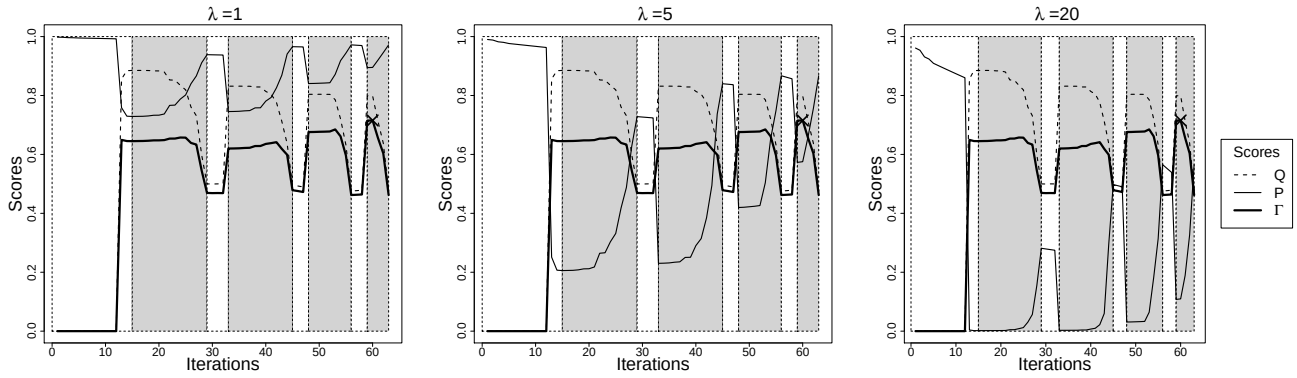

Figure 2.3: Evaluation of quality and goodness scores in the *KEE* algorithm for different values of λ for *reblock67*

changes in the border component. Moreover, we observe that Γ mimics the behavior of Q for very small values of λ and that of P for very large values of λ. Since the purpose of defining a new goodness measure is to account for the trade-off between the quality of the block diagonal and the border components, one needs to avoid the complete dominance of one quality score over the other. The plot for $\lambda = 5$ illustrates how an effective trade-off between the two quality scores is possible. Notice that the behavior of Γ alternatingly mimics that of Q and P between certain breakpoints which roughly correspond to the points where $P = Q$. Similar analysis of quality and goodness scores with a wide range of λ values for all test instances revealed that any $\lambda \in [3, 5]$ is a good choice for the tuning parameter to effectively reflect the trade-off between the quality of block-diagonal and border components. Interestingly, as briefly discussed in Section 2.2, a visual inspection of the plot presented in Bergner et al. (2015) (Figure 8a) illustrating the negative correlation between the *integrality gap closed* and the *percentage of linking constraints* also leads to a similar conclusion on the suitable range of λ values. Consequently, we use $\lambda = 5$ for *KEE auto*, whereas we use a sufficiently high $\lambda = 20$ for which the goodness score is dominated by the quality of the border component for *KEE best*.

Next, we investigate whether the proposed approach is able to detect the best BBD structure in a given matrix, if a good structure is inherent in the matrix. Figure 2.4 shows the detected BBD structures for 15 test instances from MIPLIB2003/2010 (Achterberg

et al., 2006; Koch et al., 2011). It is clear that our proposed algorithm detects BBD structures with well-balanced block sizes for 16 out of 26 test instances (see Figure 2.2 for *10teams* instance). This observation is particularly important since previous approaches such as Ferris and Horn (1998), Aykanat et al. (2004), and Bergner et al. (2015) explicitly impose a balance criterion among the diagonal blocks by using dummy variables; whereas *KEE* manages that implicitly.

Table 2.3 provides a comparison of the structures detected by *KEE auto*, *KEE best*, and *DWR best*. Unfortunately, the decomposition properties of the *DWR auto* algorithm is not available in Bergner et al. (2015). The number of diagonal blocks, the number of rows in the border area, and the maximum number of columns (rows) in a block are indicated by **K**, **b**, and **Col (Row)**, respectively. Moreover, **P** and **Q** show the quality scores of the block-diagonal and border components whereas $\mathbf{\Gamma}$ shows the overall goodness scores of the structures detected by *KEE auto*. $\hat{\mathbf{\Gamma}}$ shows the estimated goodness scores of the structures detected by *DWR best*. The estimates are calculated based on the decomposition properties provided in Bergner et al. (2015) assuming that the block-diagonal component attains the highest possible quality score for the given number of blocks. The instances for which *KEE auto* and *DWR best* detected the identical BBD structures are in **bold**, whereas instances where *KEE auto* (or in case of *fiber*, *KEE best*) and *DWR best* seem to have identical borders are in italic.

A quick inspection of the quality and goodness scores provides valuable insights. First of all, we observe that for the two instances (*10teams, harp2*) the goodness scores are very low (0.11 and 0.17, respectively). Although we concluded from Figure 2.4 that the detected structures have very granular block-diagonal components with well-balanced blocks, the overall goodness of the structures are significantly reduced by the relatively large border area. One implication of such discrepancy between the quality of the block-diagonal and the border components is the direction of structural improvement; namely, the appropriate action to be taken to improve the goodness score. In these cases, the appropriate action would be to relocate some of the rows in the border component to the block-diagonal component at the expense of reduced quality (by merging some of the blocks) in the block-diagonal component. However, our algorithm, designed to take such action, did not find a better structure simply because there is no better structure inherent in the constraint

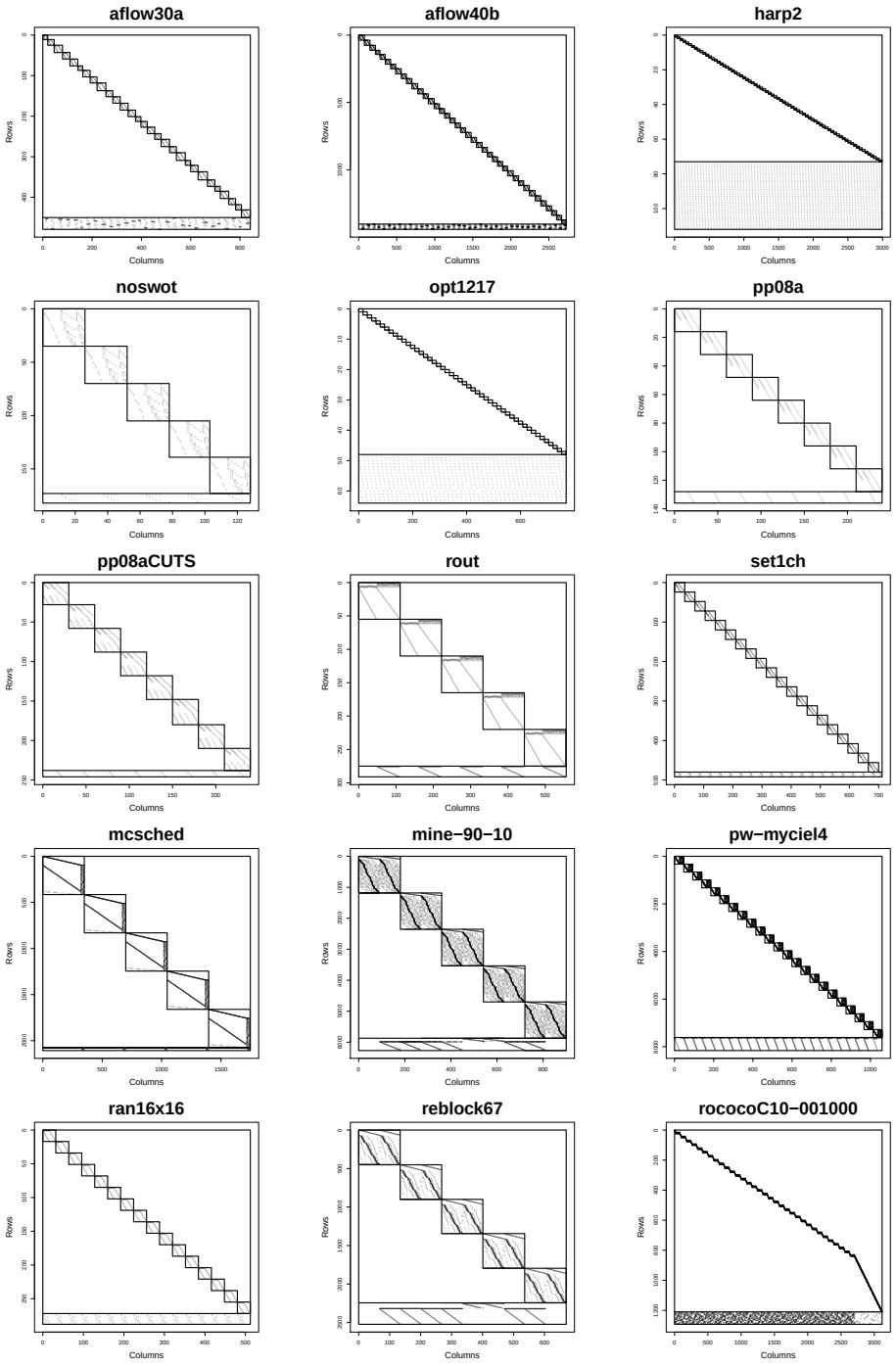

Figure 2.4: Detected BBD structures with well-balanced block sizes

Table 2.3: Comparison of BBD structures detected by KEE auto/best, and DWR best.

Instance	KEE auto							KEE best				DWR best				
	K	b	Col	Row	Q	P	Γ	K	b	Col	Row	K	b	Col	Row	$\hat{\Gamma}$
10teams	9	95	225	15	0.89	0.12	0.11	9	95	225	15	9	95	225	15	0.11
aflow30a	29	29	38	20	0.96	0.74	0.71	29	29	38	20	2	28	430	230	0.37
aflow40b	39	39	76	39	0.97	0.87	0.85	39	39	76	39	10	39	348	179	0.78
fiber	6	33	564	151	0.64	0.63	0.40	3	22	713	178	2	22	713	178	0.37
fixnet6	132	17	54	28	0.98	0.83	0.82	108	15	239	128	2	14	436	235	0.43
gesa2-o	26	94	215	219	0.88	0.68	0.60	2	26	803	819	2	26	611	611	0.45
harp2	73	39	41	1	0.98	0.17	0.17	73	39	41	1	10	39	369	9	0.16
mkc	946	107	426	240	0.99	0.85	0.84	260	29	2409	1449	2	31	2911	1815	0.48
modglob	22	23	62	41	0.92	0.67	0.62	4	8	276	186	2	8	211	144	0.44
noswot	5	9	26	35	0.80	0.78	0.62	4	8	51	70	5	9	26	35	0.62
opt1217	48	16	16	1	0.98	0.29	0.28	48	16	16	1	4	16	192	12	0.21
p2756	161	19	259	77	0.98	0.88	0.86	115	16	553	145	3	16	918	257	0.60
pp08a	8	8	30	16	0.87	0.74	0.65	8	8	30	16	8	8	30	16	0.65
pp08aCUTS	8	8	30	32	0.87	0.85	0.74	8	8	30	32	8	8	30	32	0.74
rout	5	16	111	55	0.80	0.76	0.60	5	16	111	55	5	16	111	55	0.60
set1ch	20	12	35	24	0.95	0.88	0.84	20	12	35	24	20	12	35	24	0.84
beasleyC3*	89	48	168	110	0.96	0.87	0.84	36	26	1676	1154	2	26	1252	862	0.46
gmu-35-40	9	9	357	194	0.70	0.90	0.63	9	9	357	194	2	9	432	217	0.45
mcsched	5	32	349	415	0.80	0.92	0.74	5	32	349	415	5	32	349	415	0.74
mine-166-5*	3	373	333	3273	0.63	0.80	0.50	2	183	497	4973	2	596	416	3090	0.35
mine-90-10*	5	398	181	1183	0.80	0.73	0.58	2	108	541	3727	2	92	450	3928	0.46
pw-myciel4	23	551	47	331	0.96	0.71	0.68	4	110	322	2455	2	165	530	4009	0.45
ran16x16	16	16	32	17	0.94	0.75	0.71	16	16	32	17	16	16	32	17	0.71
reblock67*	5	279	134	451	0.80	0.57	0.46	2	72	401	1483	2	69	335	1228	0.43
rmine6	3	451	430	2756	0.65	0.73	0.48	2	168	830	5461	2	354	557	3439	0.39
rococoC10	626	83	69	10	0.98	0.72	0.71	626	83	69	10	8	82	447	407	0.63

matrix. In fact, for these two instances, relocation of any row into the block-diagonal component would simply merge all the blocks, resulting in a single block. Hence, we can conclude that the highest goodness score a structure can attain is bounded by the composition of the original matrix. A normalization procedure by finding the theoretical maximum goodness score in a given matrix would provide more meaningful interpretation of the Γ values. Secondly, we observe that the goodness scores of the structures detected by *KEE auto* are consistently better than those of *DWR best*. To be fair, this is an expected result, since *DWR best* does not maximize the goodness score, but it maximizes the quality of the border component. Nevertheless, the estimated goodness score as well as the estimated quality scores of the block-diagonal and border components gives us a

similar idea for the direction of structural improvement, should one be looking for a fair trade-off between the two components.

Instances in *italic* (*aflow40b, fiber, harp2, opt1217, gmu-35-40*) point to an important observation that was briefly discussed in Section 2.3. *DWR* does not always decompose a block diagonal matrix into its most granular form. This is most likely caused by the fact that *DWR* is based on hypergraph partitioning which requires the input of the number of blocks. Notice that for *aflow40b, harp2, opt1217*, the number of blocks in the structures detected by *KEE auto* is outside the range $\{2, 3, ..., 20\}$ of **K** values enumerated in *DWR*. On the other hand, for *fiber* and *gmu-35-40*, the reason for *DWR*'s inferior decomposition is probably the compliance with the balance criterion among the block sizes required by hMeTiS algorithm. Nevertheless, the most efficient decomposition of a block diagonal matrix is unique as discussed in Section 2.4, and apparently *DWR* sometimes misses it due to input requirements.

For the instances in **bold**, *KEE auto* and *DWR best* seem to have detected identical structures. We observe that the detected structures are highly granular and have well-balanced blocks. Finally, it is worth noting that decomposition properties of *KEE best* are given to just provide evidence that when only the quality of the border component is of concern, *KEE best* is also able to detect comparable structures to those detected by *DWR best*. It is evident from the columns **b, Col**, and **Row** of *KEE best* and *DWR best* that both algorithms detect BBD structures with similar border size and maximum column/row sizes.

2.4.3 Lagrangian Bound

The Lagrangian bound resulting from the relaxation of the border area is the dual bound in Bergner et al. (2015) and obviously depends on the BBD structure detected. In Table 2.4, we compare the bounds found by *KEE auto* and *DWR auto* in terms of their optimality gaps and CPU times. Bold entries in the table indicate that the respective algorithm performs better than (or as good as) the other for that instance. Since the optimal solutions of all test instances are available, the gaps are calculated based on the optimal objective values. The best of the two bounds and CPU times are displayed in bold.

According to Table 2.4, the two algorithms perform quite similar on all instances, with the exception of four instances (starred in the table, namely, *beasleyC3, mine-166-5, mine-90-10*, *reblock67*) where *DWR auto* obtains significantly tighter bounds. For these four instances, according to Bergner et al. (2015), *DWR auto* and *DWR best* detect BBD structures that have 2 blocks and significantly smaller border areas compared to those of *KEE auto*. Hence, it is expected that their bounds would be tighter than those of *KEE auto*. This comes, however, at the expense of inferior structures as the Lagrangian bound depends only on the size of the border area. To provide a fair comparison, we compare the bounds obtained by *KEE best* against those obtained by *DWR best* in Table 2.5. As evident from Table 2.5, Lagrangian bounds and CPU times of *KEE best* algorithm are comparable but slightly inferior to those of *DWR best*.

Notice in Table 2.4 that for instances *aflow30a, aflow40b, fiber, fixnet6, gesa2-o, mod-glob, p2756*, two bounds are quite close to each other, but *KEE auto* is significantly faster than *DWR auto*. These seem to be the instances where *DWR auto* fails to find the most efficient and granular structure. Small differences in CPU times do not necessarily imply the superiority of one algorithm over another, since many factors may be of influence. A significant difference, however, is often an indicator of a superior/inferior structure detected by the compared algorithms. If we set 30 seconds as a significant difference, then *KEE auto* does better in 9 instances versus 3 for *DWR auto*.

As seen in Table 2.4, for six instances (in *italic*), the Lagrangian bound obtained by *KEE auto* is exactly equal to the LP bound, i.e., the worst possible bound. Surprisingly, for the majority of these instances, the BBD structures detected by *KEE auto* algorithm are very high quality structures. *aflow30a* and *aflow40b*, in particular, have BBD structures with a highly granular block diagonal matrix of almost equally sized blocks and a very small border area. It is in direct contradiction with the belief that a nice inherent structure leads to a good bound. This indicates that the quality of the bound may not be the best indication of a good structure.

In summary, we conclude from Tables 2.3, 2.4, and 2.5 that *KEE* is capable of detecting BBD structures that provide comparable bounds to *DWR* requiring less CPU time, in general. Considering the fact that the state-of-the-art technique *DWR* requires user input for the number of blocks and a balance criterion (number of dummy variables) to generate

Table 2.4: Comparison of Lagrangian bounds obtained by KEE auto and DWR auto.

Instance		KEE auto		DWR auto	
Name	**LP Gap (%)**	**Gap (%)**	**CPU (sec)**	**Gap (%)**	**CPU (sec)**
10teams	0.76	**0.00**	10.11	0.05	**2.22**
aflow30a	15.10	15.10	**5.82**	**14.71**	93.51
aflow40b	**13.90**	**13.90**	**8.66**	**13.90**	70.52
fiber	61.55	2.27	**24.46**	**1.07**	540.53
fixnet6	69.85	18.91	**8.27**	**18.89**	2838.48
gesa2-o	1.18	**0.70**	**34.33**	0.78	113.14
harp2	**0.61**	**0.61**	23.45	**0.61**	**1.59**
mkc	8.51	**0.16**	**96.42**	**0.16**	106.25
modglob	1.49	**0.58**	**8.57**	0.79	3600.00
noswot	4.88	**0.49**	5.36	**0.49**	**1.26**
opt1217	**25.13**	**25.13**	8.13	**25.13**	**0.07**
p2756	13.93	**0.27**	**7.17**	**0.27**	234.98
pp08a	62.61	**2.50**	2.06	**2.50**	**0.47**
pp08aCUTS	25.43	**2.50**	1.84	2.58	**0.58**
rout	8.88	**0.68**	25.39	**0.68**	**5.90**
set1ch	41.31	**0.04**	1.40	0.10	**0.61**
beasleyC3*	94.64	31.47	**58.23**	**15.38**	3600.00
gmu-35-40	**0.01**	**0.01**	150.54	**0.01**	**0.12**
mcsched	8.56	**0.06**	**67.82**	**0.06**	**84.30**
mine-166-5*	45.09	11.66	1800.00	**7.33**	2983.32
mine-90-10*	13.12	5.96	**541.79**	**0.49**	1426.12
pw-myciel4	100.00	**60.00**	28.42	**60.00**	**27.50**
ran16x16	18.48	**10.77**	5.13	**10.77**	**0.93**
reblock67*	13.61	3.85	656.65	**0.13**	**529.73**
rmine6	1.12	0.99	1800.00	**0.42**	**836.89**
rococoC10	**34.42**	**34.42**	217.10	**34.42**	**13.75**

Table 2.5: Lagrangian bounds obtained by KEE best and DWR best for the exceptional instances.

Instance	KEE best		DWR best	
Name	**Gap (%)**	**CPU (sec)**	**Gap (%)**	**CPU (sec)**
beasleyC3	18.64	1130.03	15.38	3600.00
mine-166-5	9.18	1800.00	7.33	2983.32
mine-90-10	0.88	1800.00	0.49	1426.12
reblock67	0.14	708.12	0.13	529.73

alternative structures, the results presented above accentuates the merit of *KEE*, which does not require any user input at all.

2.5 Conclusions

A community detection based approach for identifying BBD structures in mixed integer programs is introduced, studied, and tested. Unlike current approaches that are based on graph/hypergraph partitioning methods, the proposed approach relies on community detection which has the advantage of not specifying the number of partitions *a priori*. We also define a new measure of goodness to compare the alternative BBD structures detected.

The main motivation behind BBD structure detection is the possibility of using decomposition methods to solve intractable large scale integer programs. To this end, the proposed approach is designed to account for the trade-off between computational time and solution quality. Experiments on test instances from MIPLIB2003/2010 (Achterberg et al., 2006; Koch et al., 2011) reveal the effectiveness of the proposed approach in detecting good underlying BBD structures that yield tight Lagrangian bounds in short computational times.

Chapter 3

Spatial Separability in Hub Location Problems

In this chapter we look into a specific type of problem, HLP, and explore the potential of data-driven approaches in tackling such problems. HLP is one of the well-established areas of research in literature with over thirty years of contributions. HLP models the problem of selecting a subset of nodes within a given network as hubs and allocating the remaining nodes to the selected hubs. All flow between nodes is, then, routed via hubs (as the inter-hub links typically enjoy economies of scale) in such a way to minimize an overall transportation cost.

3.1 Introduction and Literature

Following the seminal work by O'Kelly (1986), the field enjoyed a proliferation of new models, solution methodologies, and applications. Current models are capable of incorporating practical features such as capacity constraints (Campbell, 1994a; Ebery et al., 2000; Boland et al., 2004; Labbé et al., 2005; Correia et al., 2010; Contreras et al., 2011c), congestion (Elhedhli and Hu, 2005; Elhedhli and Wu, 2010; de Camargo et al., 2009b), incomplete hub network topologies (Campbell et al., 2005a,b; Alumur et al., 2009; Calık

et al., 2009; Contreras et al., 2013), and uncertainty (Marianov and Serra, 2003; Contreras et al., 2011b; Alumur et al., 2012).

Solution methodologies ranging from decomposition-based methods such as Lagrangian relaxation (Pirkul and Schilling, 1998; Aykin, 1994; Elhedhli and Hu, 2005), Branch-and-Price (Contreras et al., 2011c), and Benders decomposition (de Camargo et al., 2009a; Contreras et al., 2011a) to metaheuristics such as genetic algorithms (Topcuoglu et al., 2005), tabu search and GRASP (Klincewicz, 1992; Mayer and Wagner, 2002) have been proposed. On the application side, HLPs have been introduced in the context of airline transportation (O'Kelly, 1987a; Martín and Román, 2004), postal delivery (Ernst and Krishnamoorthy, 1996), and telecommunication (Klincewicz, 1998). The reviews by Alumur and Kara (2008), Farahani et al. (2013) and Contreras (2015) provide a thorough survey of the state-of-the-art in the field and Campbell and O'Kelly (2012) discuss the past and the future of hub location research and possible fertile research avenues.

Different objective functions can be used to model the way in which hubs are determined. For example, *p-hub median* problems assume that the number of hubs to locate is known *a priori* and try to minimize the total flow/transportation (O'Kelly, 1987b; Campbell, 1994b; Ernst and Krishnamoorthy, 1996). On the other hand, *fixed cost hub location* problems assume that the number of hubs is not known *a priori*, but a fixed set-up cost for each hub site is known and try to minimize the sum of total set up and flow costs (O'Kelly, 1992b). *p-hub center* problems try to minimize the maximum of a given cost measure such as flow cost or travel time of all origin-destination pairs (Yaman and Elloumi, 2012; Campbell et al., 2007; Kara and Tansel, 2000). *Hub covering* problems assume a demand is covered if both origin and destination are within a specified distance of a hub node and try to minimize the total set-up cost of the hub network (Kara and Tansel, 2003; Hamacher and Meyer, 2006; Wagner, 2008).

HLP models are also differentiated based on the nature of the allocations of non-hub nodes to selected hubs. *Single allocation* problems assume that each node is allocated to a single hub. In other words, all the incoming and outgoing traffic for every node is routed through a single hub (O'Kelly, 1987b; Campbell, 1994b; Ernst and Krishnamoorthy, 1996). *Multiple allocation* problems, on the other hand, assume that a node can be allocated to multiple hubs. Therefore, multiple allocation problems are relaxed versions of the single

allocation problems (Hamacher et al., 2004; Marín, 2005; Contreras et al., 2011a).

Another important characteristic of HLPs is the incorporation of capacity constraints. Most HLP variants are characterized based on whether they incorporate capacity constraints or not. Constraints may be imposed on the incoming and outgoing flow at the hubs nodes as well as on the hub arcs. Uncapacitated problems do not incorporate capacity constraints. In other words, they assume infinite capacity for all hubs and hub arcs. Majority of the early work on HLPs fall under this category (O'Kelly, 1987b; Klincewicz, 1992; Mayer and Wagner, 2002). Capacitated problems may incorporate capacity constraints on the incoming/outgoing flows at the hubs nodes or on the hub arcs. Campbell (1994a); Ebery et al. (2000); Boland et al. (2004) study capacitated multiple allocation problems with constraints on hub node capacities whereas (Labbé et al., 2005; Correia et al., 2010; Contreras et al., 2011c) study capacitated single allocation problems with constraints on hub node capacities. Bryan (1998); Aykin (1994) consider problems with capacity constraints on hubs arcs and direct O/D links, respectively. Finally, Elhedhli and Wu (2010) introduce a new model where hub capacity is also a decision variable.

Breadth of recent advances in hub location research is astonishing. For example, *hub-arc* problems (Campbell et al., 2005a,b) are different than other HLP variants in that they relax the assumption of fully-interconnectedness of hub network and consider different network topologies. This relaxation is particularly meaningful for the applications where set up costs associated with hub arcs are prohibitive. *HLPs with uncertainty* consider the effect of uncertainty in costs, demands, and distances. Marianov and Serra (2003) are the first dealing with uncertainty at the hub nodes. They represent hubs as queues with limited lengths. Contreras et al. (2011b) model the classical uncapacitated multiple allocation problem as a two-stage integer stochastic program with recourse when demands and flow costs are uncertain. Alumur et al. (2012) consider HLPs with uncertainty in set-up costs and demands for both single and multiple allocation problems. Demand uncertainty typically cause congestion at the hub networks. Elhedhli and Hu (2005) introduce a new model with additional costs for congestion at the hubs. *HLPs with competition* consider problems where several firms (decision makers) exist in the market and compete to provide service to customers. They introduce models where demand flow is captured from competitor's hub network if a reduction in travel time is offered by a new hub network. Marianov et al.

(1999) introduce competitive hub location models in which follower tries to maximize the captured demand flow. Sasaki and Fukushima (2001) uses a game theoretic framework to introduce Stackelberg hub location model where competition takes place between a large company and several mid-size companies.

We study a specific subclass of HLPs: USApHMP which was described in Definition 1. One of the first linear (mixed integer) programming formulations for USApHMP was introduced by Skorin-Kapov et al. (1996) which had $O(n^4)$ variables and $O(n^3)$ constraints. Although the formulation has been shown to provide very tight LP bounds, it does not scale well with the number of nodes. Even for the problem instances of moderate size (between 50 and 100 nodes), commercial solvers reach the memory limit. The formulation proposed by Ernst and Krishnamoorthy (1996) requires smaller number of variables and constraints ($O(n^3)$ and $O(n^2)$, respectively). Although it has looser LP bound compared to that of Skorin-Kapov et al. (1996)'s formulation, commercial solvers can solve problem instances up to 100 nodes using this formulation. Ebery (2001) presented another formulation which requires $O(n^2)$ variables and $O(n^2)$ constraints. This formulation uses fewer variables than all of the other models previously presented in the literature. However, in practice, the computational time required to solve this new formulation was shown to be greater than that required to solve Ernst and Krishnamoorthy (1996)'s formulation. Therefore, in this work, we used the formulation by Ernst and Krishnamoorthy (1996) as the mathematical model for USApHMP.

Despite abundance of various mathematical models and efficient solution methodologies for moderate size problem instances, the main challenge of using HLP in certain promising domains is the inability of current solution approaches to handle large instances. (e.g., networks with more than 1000 nodes). In this chapter, we study an important characteristic of optimal hub networks: spatial separability. We discover that the optimal hub-and-spoke topology is typically composed of disjoint clusters with non-overlapping convex hulls. Although not a general property, extensive testing on randomly generated instances reveals that it holds in 99.5% of the time.

Motivated by the need for solving large practical problem instances, our main contributions in this study are to investigate the spatial separability feature observed in the optimal solutions of USApHMP instances, to elucidate the connection between USApHMP

and several other problems, and to propose a new approach exploiting this feature to tackle the largest problem instances in the literature, efficiently.

3.2 Spatial Separability

Campbell and O'Kelly (2012) argue that *"the purpose of hub location research is insight, not numbers and better insight is likely to result from the combination of improved understanding of the properties of hub location problems, better algorithms, new solutions, and a clearer understanding of the practical implications of the solutions"*. Several studies in the literature attempt to identify structures and patterns that would help mitigate the inherent difficulty of HLPs. O'Kelly (1986), for example, observes that in most cases non-hub nodes are allocated to the closest hub. However, on a later paper O'Kelly (1992a), which exploits the parallelism between clustering problems and planar hub location problems, they illustrate that the *closest-hub allocation* assumption is too strong an assumption that could potentially lead to high optimality gaps. Campbell (1990) studies the spatial conditions under which single allocation (to the nearest hub) or multiple allocation (to the hub minimizing the distance between origin and destination) is optimal for a non-hub node. They find analytical expressions for the spatial regions around hub nodes where nearest hub allocation is optimal. Peker et al. (2016) propose several spatial measures to evaluate node importance as a proxy for the likelihood to be a hub in the optimal solution and develop a heuristic which is based on restricting the solution space to *sets most likely to contain hubs*. One of the main contributions of this study is to reveal another spatial pattern of optimal USApHMP solutions which has not been studied in the literature. The observed pattern constitutes the basis of the solution methodology which is developed with the goal of tackling the newly introduced BCN instances with $n = 998$ nodes.

One rather straightforward solution approach for USApHMP is to first identify the optimal hub nodes, and then decide the optimal allocations of each node to hubs. This approach is very intuitive as the restricted problem of allocating non-hub nodes to the given hubs - p hub allocation problem (pHAP) - is much easier to solve than the original USApHMP and enumerating all possible hub combinations is not too formidable for

moderate size instances. Ebery (2001) provides a very efficient formulation for pHAP and experimentally show that USApHMP require several orders of magnitude more CPU time than pHAP and conclude that *"we should try to concentrate on techniques which may help to determine possible hub locations, since once the hub locations are known, the problem is relatively easy to solve"*.

Conversely, one could first attempt to find the optimal partition of nodes such that the ones in the same cluster would be allocated to the same hub and then find optimal hubs for each cluster. Wagner (2007) proposed a very efficient formulation for the problem of finding optimum hub locations for a given partitioning of nodes into clusters such that exactly one hub per cluster must be opened. They call this restricted problem cluster hub location problem (CHLP) and solve instances with up to 500 nodes within a second. Nevertheless, this approach has not been as popular as the former, since in the absence of an efficient way to identify optimal clusters, one has to enumerate exponentially many partitions.

In this section we attempt to provide analytical and experimental justification for the latter approach. We start our analyses with the re-examination of a statement from O'Kelly (1992a) *"The property that all observations which are closer to centroid A than to centroid B are assigned to the same group does not hold for HLPs. Therefore, we cannot use the strategy of examining only the non-overlapping partitions"*. Specifically, we are interested in redefining *non-overlapping partitions* concept. To this end, we begin with visual inspection of the optimal solutions of known problem instances and look for spatial patterns in these solutions. Figure 3.1 depicts the optimal solutions of six USApHMP instances with ($n = 25$) numbers of nodes from CAB dataset provided by O'Kelly (1986). Figures 1.a-c show the solutions for $p \in \{2, 3, 4\}$ hubs with interhub discount factor $\alpha = 0.1$, whereas Figures 1.d-f show the solutions for $p \in \{2, 3, 4\}$ hubs with interhub discount factor $\alpha = 1$. In each map, hub nodes are marked with filled circle points whereas non-hub nodes are marked with empty circle points. Moreover, nodes that are allocated to the same hub are shown with the same color. We observe that the nodes allocated to the same hub are geographically (spatially) clustered. To formalize this observation, we proceed with the following definition:

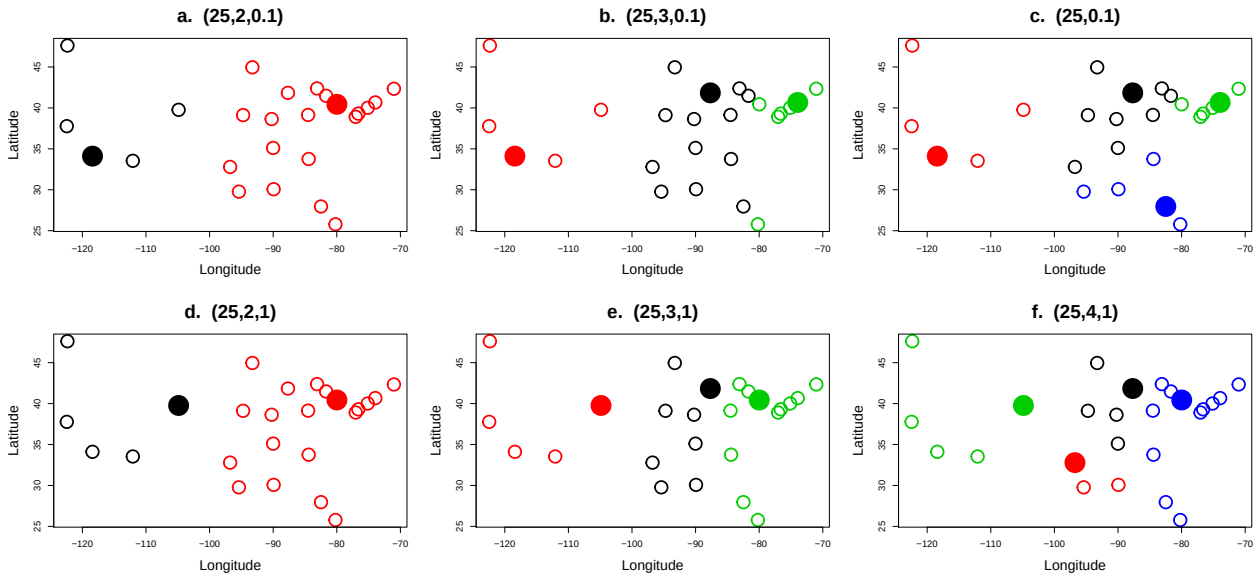

Figure 3.1: Optimal solutions of various CAB instances for different (n, p, α) parameter sets

Definition 5 *The set of nodes that are allocated to the same hub in a USApHMP solution is called an allocation cluster.*

Now, the observation from Figure 3.1 can be rephrased as follows: Allocation clusters are non-overlapping, i.e., they are separated from each other with clear boundaries.

The aforementioned statement in O'Kelly (1992a) appears to be in contradiction with observation above. However, the subtle difference lies in how the clusters are defined. O'Kelly (1992a)'s definition of a partition involves clusters with a centroid and the subset of nodes for which that centroid is the closest among all centroids in the given partition. They experimentally show that such partitions are often non-overlapping. By contrast, partitioning into allocation clusters does not specify a centroid for clusters. A consistent pattern that the regions encapsulating the nodes in each allocation cluster appear to be non-overlapping has already been illustrated in Figure 3.1 for CAB instances. In an attempt to extend its validity for problem instances other than CAB dataset, and to provide a concise mathematical expression for the aforementioned pattern, the optimal solutions of USApHMP instances from AP dataset with different numbers of nodes (n) and hubs (p) are obtained and depicted in Figure 3.2 (with $\alpha = 0.75$, $\chi = 3$, $\delta = 2$ as given by Ernst and Krishnamoorthy (1996)). Figures 3.2.a-c show the solutions for 3-hub problem with $n \in \{25, 50, 100\}$ nodes, whereas Figures 3.2.d-f show the solutions for $p \in \{5, 7, 10\}$ hubs with $n = 100$ nodes.

Each polygon in Figure 3.2 represents an allocation cluster. Observe that irrespective of the n and p values, the allocation clusters are always non-overlapping. More specifically, the convex hulls of allocation clusters always appear to be disjoint. It is worth mentioning that we tested the validity of this observation with different (n, p, α) parameters in both CAB and AP datasets and observed no exception. We call this property *spatial separability* and formally define it as follows.

Definition 6 *Let $A^i = (a_{i1}, a_{i2}, ..., a_{ik}), \forall i \in \mathcal{N} \equiv \{1, 2, ..., n\}$ be the coordinate vector of node i in k-dimensional space. A partition of nodes $\mathcal{N}$ into p (allocation) clusters $\mathcal{P} = \{P_1, P_2, ..., P_p | \bigcup_{i=1}^{p} P_i \equiv \mathcal{N}$ and $P_i \cap P_j \equiv \varnothing, \ \forall i, j \neq i\}$ is said to be spatially separable if the convex hulls of the clusters are (pairwise) disjoint, i.e., $Conv(P_i) \cap Conv(P_j) \equiv \varnothing, \ \forall i, j \neq i$.*

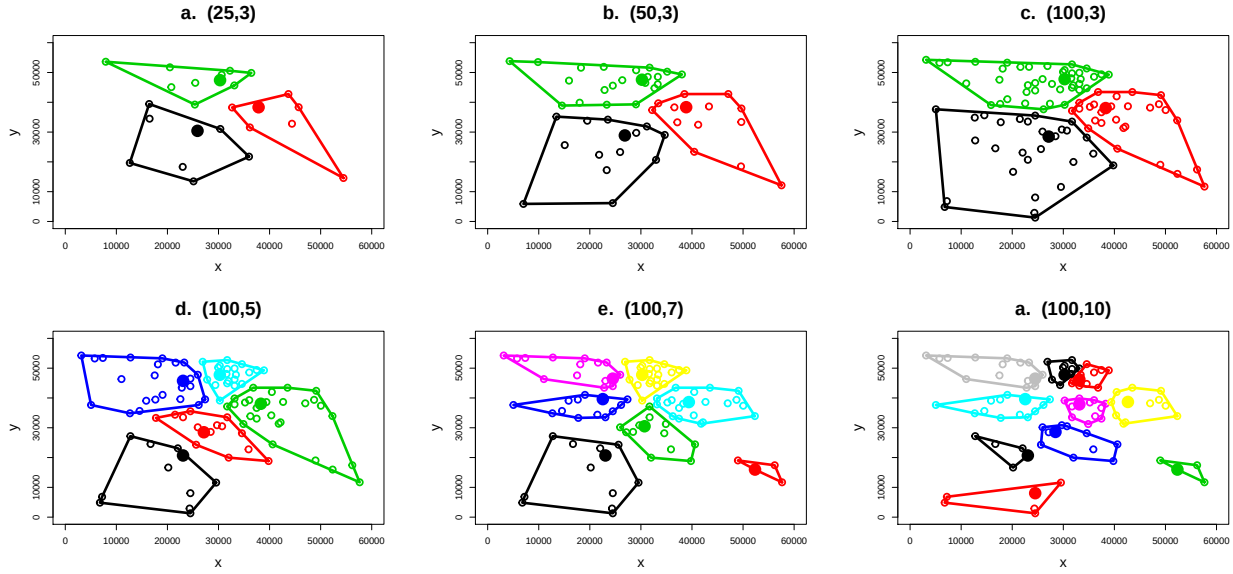

Figure 3.2: Optimal solutions of various AP instances for different (n, p) parameter sets

Definition 6 helps us remove the ambiguity in describing the observed spatial pattern by providing a mathematical expression for the spatial separability property. So far, we have consistently observed spatial separability in the optimal solutions of the known USApHMP instances. Figure 3.3, on the other hand, shows that there exist some extreme cases under which spatial separability is violated in the optimal solutions. Figure 3.3-(a) is the simplest counter example in 2D space for which the optimal solution can be made to violate spatial separability by selecting appropriate values for the flows, e.g., $w_{14} = 100, w_{13} = 1, w_{24} = 1$, and $w_{ij} = 0$ for all other pairs of nodes (where nodes are numbered from left to right). An interesting research question would be how likely these violating cases are to occur in real life settings. To this end, in Section 5.1, we conduct an extensive computational analysis on randomly generated USApHMP instances and show that these extreme cases occur so infrequently that overlooking them would not be completely unreasonable as long as this provides a computational advantage. Figure 3.3-(b) is a typical example of non-separable instances where spatial separability is violated only by a handful of nodes and relocation of those nodes so that spatial separability is restored can be done with a negligible increase

in the objective value.

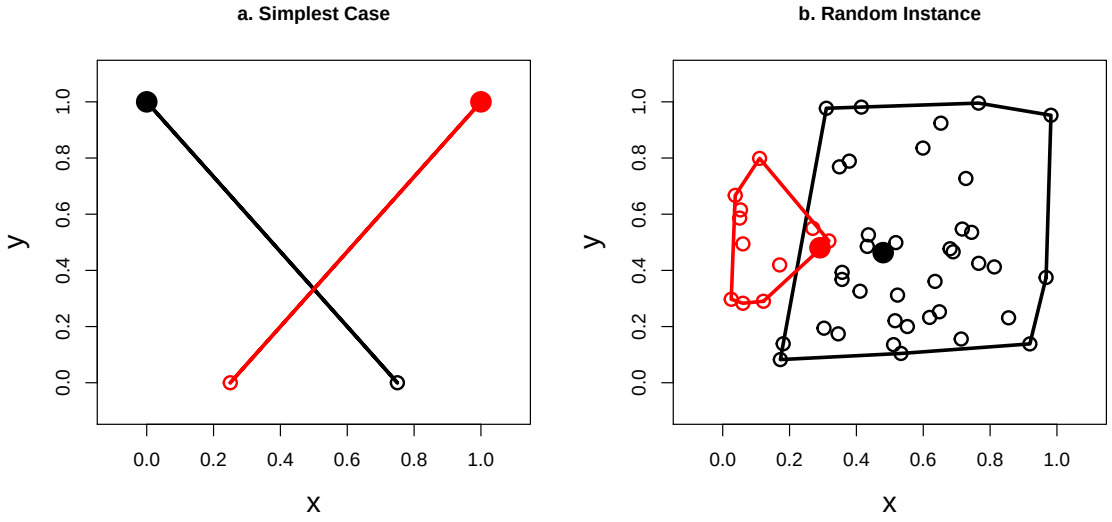

Figure 3.3: Counter examples violating spatial separability

Partitioning into clusters with disjoint convex hulls has been studied in the literature. Chakravarty et al. (1985) study a related problem: partition a given set of indexed items $(1, 2, ..., n)$ into p subsets $(P_1, P_2, ..., P_p)$ to minimize an objective function $g(P_1, P_2, ..., P_p)$. They provide several sufficient conditions on $g(.)$ that would guarantee that there is an optimum partition in which each subset consists of consecutively indexed items and show that this partitioning problem can be solved in polynomial time. Barnes et al. (1992) use results from Chakravarty et al. (1985) on consecutive optimizers and derive necessary and sufficient conditions on $g(.)$ such that there exists an optimum partition which is spatially separable.

It is worthwhile to comment on an important implication of the aforementioned results for our study. The results of Chakravarty et al. (1985) can be applied to a restricted version of USApHMP in which an additional set of constraints is introduced to impose spatial separability. This would imply that the size of the feasible region (i.e., the search space) becomes polynomial in n. In other words, the computational effort to completely enumeration all feasible partitions would be reduced from exponential complexity $O(p^n)$ to

polynomial complexity $O(n^p)$. Considering that many efficient heuristics for HLPs such as tabu search and simulated annealing depend on some kind of clever enumeration of feasible solutions, such a reduction in search space would have substantial impact on computational performance of these heuristic.

We are, however, mainly interested in making practical use of spatial separability property. To this end, we, next, introduce the set of constraints that would impose spatial separability to HLPs. Figures 3.4 and 3.5 show a spatially separable and non-separable partition, respectively, of the same set of nodes. An observation from Figures 3.4 and 3.5 allows us to impose spatial separability as a set of linear constraints to the MIP formulation of USApHMP: In a spatially separable partition no line segment contained in one cluster intersects with a line segment contained in another cluster. Lemma 1 (given without proof) can be used to check whether two given line segments intersect.

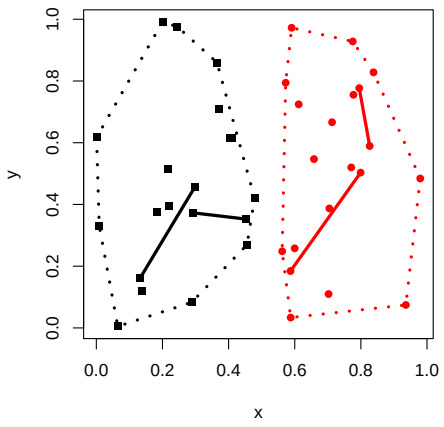

Figure 3.4: A spatially separable partition

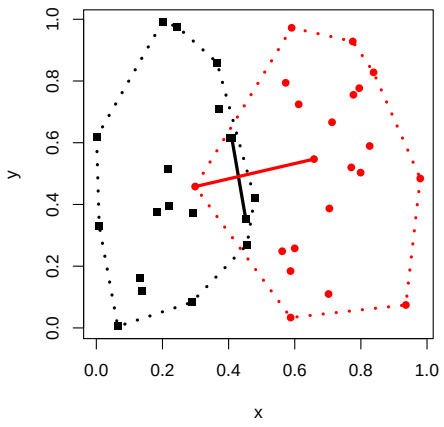

Figure 3.5: A non-separable partition

Lemma 1 *Let* (x_1, y_1) *and* (x_2, y_2) *be the two end points of the line segment* LS_1 *and* (x_3, y_3), (x_4, y_4) *be the end points of the line segment* LS_2. LS_1 *and* LS_2 *are said to*

intersect if the following conditions hold:

$$\sqrt{(x-x_1)^2+(y-y_1)^2}+\sqrt{(x-x_2)^2+(y-y_2)^2}=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2} \quad (3.1a)$$
$$\sqrt{(x-x_3)^2+(y-y_3)^2}+\sqrt{(x-x_4)^2+(y-y_4)^2}=\sqrt{(x_3-x_4)^2+(y_3-y_4)^2} \quad (3.1b)$$

where (x, y) *is the intersection point of the lines defined by the end of points of* LS_1 *and* LS_2 *and can be written as follows:*

$$x = \frac{(x_1y_2 - y_1x_2)(x_3 - x_4) - (x_1 - x_2)(x_3y_4 - y_3x_4)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)}$$
$$y = \frac{(x_1y_2 - y_1x_2)(y_3 - y_4) - (y_1 - y_2)(x_3y_4 - y_3x_4)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)}$$

Note that Lemma 1 excludes the degenerate cases where line segments are parallel to each other. Figure 3.6 illustrates different possible ways the lines defined by the end points of LS_1 and LS_2 can intersect only one of which implies the intersection of LS_1 and LS_2.

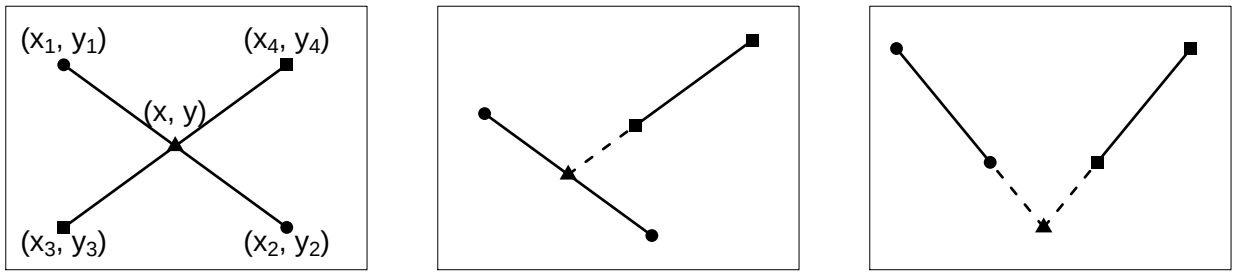

Figure 3.6: Different possibilities of intersections

To impose spatial separability, one needs to identify all pairs of intersecting line segments (i.e., quadruplets of nodes) and add constraints that prevents these line segments from being in different clusters. This would require $O(n^6)$ additional constraints: $O(n^4)$ for all possible intersecting line segments, and $O(n^2)$ for all possible cluster pairs. This is a formidable number of additional constraints which would unnecessarily complicate

the MIP formulation. An alternative approach would be similar to adding valid "comb inequalities" (Naddef and Thienel, 2002) for Travelling Salesman Problem (TSP) (Applegate, 2006). For the purposes of this study, we only need to note that comb inequalities are not contained in the original TSP formulation, but can be used as valid cuts to tighten the LP bounds at each branching node. Similar to that, instead of adding all constraints pertaining to spatial separability at once, one may opt to add only the cuts necessary to prevent the violating intersections in a given solution. For example, assume that the two line segments LS_1 and LS_2 are intersecting. Let nodes i, j and s, t be the end points of the LS_1 and LS_2 line segments, respectively. Assume also that in the given solution, the nodes i, j are included in cluster k and nodes s, t are included in a different cluster m. This is a case where spatial separability is violated and we need to remove this solution from the feasible set by ensuring at least one of the four cases listed below is satisfied:

- (a) Nodes i and j are not both in cluster k
- (b) Nodes s and t are not both in cluster m
- (c) At least three of the nodes i, j, s, t are in cluster k
- (d) At least three of the nodes i, j, s, t are in cluster m

Introducing the following set of constraints - hereinafter called *intersection elimination constraint (IEC)* - would eliminate this violation:

$$y_{ik} + y_{jk} - Mu_1 \leq 1 \tag{3.2a}$$
$$y_{tm} + y_{sm} - Mu_2 \leq 1 \tag{3.2b}$$
$$y_{ik} + y_{jk} + y_{tk} + y_{sk} + Mu_3 \geq 3 \tag{3.2c}$$
$$y_{im} + y_{jm} + y_{tm} + y_{sm} + Mu_4 \geq 3 \tag{3.2d}$$
$$u_1 + u_2 + u_3 + u_4 \geq 1 \tag{3.2e}$$
$$u_1, u_2, u_3, u_4 \in \{0, 1\} \tag{3.2f}$$

In the above set of constraints, M is a very large number and u_1, u_2, u_3, u_4 are auxiliary decision variables to ensure that at least one of the four cases is satisfied. As one may notice,

to eliminate each non-separable solution, we need to add 5 constraints and 4 variables. Hence, ideally, we would like to use as few IEC as possible. In Section 5.1, we use IECs to impose spatial separability on the randomly generated non-separable USApHMP instances to be able to calculate optimality gaps. A major drawback of using IECs is that it requires the ability to solve LP relaxation of an instance. This hinders the use of IECs for very large problems instances.

Despite its consistent occurrence and implication to significantly reduce computational effort for solving USApHMPs, *spatial separability* alone is not sufficient to devise an approach to tackle very large problem instances. It does, however, provide an important insight which we exploit in Section 3.3 to propose a new approach which can efficiently tackle very large problem instances.

3.3 Approach

Despite abundance of efficient algorithms to find feasible solutions and bounds for small-to-moderate size USApHMP instances, one important component appears to be missing in the literature: drawing insights on the optimal solution of a large instance from an (approximate) smaller representation of the same instance. Ernst and Krishnamoorthy (1996) briefly touch this issue while describing how the smaller instances are generated from the original 200-node AP dataset. They generate smaller instances by diving the coordinate system into n disjoint regions with equal number of nodes ($200/n$ nodes each) and merge the nodes within the same region into a single centroid node. This procedure is often referred to as parcellation or segmentation in different fields. The resulting maps are indeed good approximations of the original map and the optimal solutions of these smaller instances can be used as approximation to the original problem. However, these smaller instances fail to provide either an upper or a lower bound on the original problem. They do not provide an upper bound because their solution is not feasible to the original problem. On the other hand, they do not provide a lower bound because representing a set of nodes with a single centroid node results in overestimation of at least one inter-region transportation cost. The lack of interest in the literature to draw insight from

smaller instances may be attributed to their failure to provide a lower/upper bound. To bridge this gap, we propose a new approach which efficiently makes use of the solutions of smaller instances to find high quality feasible solutions - i.e., upper bounds - for the original USApHMP problem.

The main implication of the previous section can be summarized as follows: The nodes that are in spatial proximity to each other tend to behave similarly in their "choice" of allocation cluster and this tendency becomes more pronounced as the distance between the two nodes decreases. In other words, on micro level, the effect of distance (coordinates) become the dominant factor in determining co-allocation decision of two nodes. One way to interpret and make use of this insight is to lump together several spatially neighbouring nodes and assume they will be allocated to the same hub. Doing this for all the nodes in a given map will create spatially disjoint parcels containing lumped nodes for which the hub location and allocation decisions are yet to be made. This procedure is, in fact, very common in various fields and can be phrased as reducing the problem resolution. Solving this *low resolution* problem may not give us the optimal solution to the original problem, but by construction it is expected to get us very close to it. One may argue that for very large problem instances where MIP formulations become astronomically large to process, an approximate solution is the best one can hope for. In fact, this is exactly how most HLP instances in the literature are generated, e.g., dividing the coordinate system into regions consisting equal number of nodes as in Ernst and Krishnamoorthy (1996).

In this section, we propose a new modular approach driven by the aforementioned insight with an ultimate goal to tackle 1000-node USApHMP instances and find near-optimal solutions with negligible optimality gaps. We proceed with describing the building blocks (modules) of the proposed approach.

3.3.1 Parcellation

The first module, which is called *parcellation*, is used to identify an efficient partition of the nodes into a predetermined number of parcels. A parcellation is considered efficient if all parcels consist only of nodes that are in the same allocation cluster at the optimal solution. As evident from the description, measuring the exact efficiency of a parcellation requires

the knowledge of optimal solution to the original problem. Hence such parcellations will have to be obtained using heuristics. As argued earlier, distance becomes the dominant force in determining the co-allocation of two nodes on micro level. Therefore, if we require a large enough number of parcels (e.g., $r \geq 50$), the original map will be divided into small enough parcels that will justify making node allocations to parcels solely based on distance. This, in turn, relates the parcellation problem to the well-known p-median facility location problem.

Next, we provide the formal definition and the mathematical formulation for a variant of p-median problem we call r-Parcellation Problem (rPP). Given r, the number of parcels to divide a spatial network of nodes into, the rPP is the problem of finding the optimal parcel medoids, as well as the allocation of the remaining nodes to parcels such that each node is allocated to exactly one parcel and the sum of node-to-medoid distances is minimized:

$$
\begin{aligned}
[r\text{PP}] \qquad & \min \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} d_{ij} y_{ij} && && \text{(3.3a)} \\
& \text{s.t.} \sum_{j \in \mathcal{N}} x_j = r && && \text{(3.3b)} \\
& \sum_{j \in \mathcal{N}} y_{ij} = 1 && \forall i \in \mathcal{N} && \text{(3.3c)} \\
& y_{ij} \leq x_j && \forall i, j \in \mathcal{N} && \text{(3.3d)} \\
& x_j, y_{ij} \in \{0, 1\} && \forall i, j \in \mathcal{N} && \text{(3.3e)}
\end{aligned}
$$

where the decision variables are defined as follows:

$$
x_j = \begin{cases} 1 & \textit{if } \text{a medoid is located at node } j \\ 0 & \textit{otherwise} \end{cases}
$$

$$
y_{ij} = \begin{cases} 1 & \textit{if } \text{node } i \text{ is allocated to parcel } j \\ 0 & \textit{otherwise} \end{cases}
$$

Figure 3.7-b shows an example of parcellation. Efficient parcellation of a map is not unique, i.e., there are multiple different ways an optimum allocation cluster can be divided

into parcels. Therefore, rPP is not the only method yielding parcels that will lead to a quality solution. One may use a different method to find an efficient parcellation. In fact, to speed up this step, we use a heuristic by Kaufman and Rousseeuw (2009) to solve the rPP rather than solving it optimally. Once an efficient parcellation is found, the next step is to find the hub-parcel locations and parcel-to-hub allocations which is handled by the *low resolution* module.

3.3.2 Low Resolution

Given an efficient parcellation $\mathcal{P} = \{P_1, P_2, ..., P_r\}$ of nodes N such that

$$P_i \cap P_j = \varnothing, \quad \forall P_i \neq P_j \in \mathcal{P}$$
$$\bigcup_{i=1}^{r} P_i = \mathcal{N}$$

let $P(i)$ represent the parcel node i belongs to in $\mathcal{P}$. Assuming that the distances between two nodes in the same parcel are negligible, i.e., $d_{uv} \leq \epsilon, \forall u, v \in \mathcal{N} : P(u) = P(v)$, we can approximate the original problem by treating each parcel as a single supernode and duplicating the decisions made for a parcel to all the nodes contained in it.

An approximation of the original USApHMP can be done in several different ways depending on how the resulting output will be used. For example, by assuming the distances between nodes within the same parcel is equal to zero, one may construct an approximate problem to obtain a lower bound to the original problem. Alternatively, inter and intra parcel distances may be set in a way to approximate the objective value of the original USApHMP as accurately as possible. Since the main goal of this study is to produce high quality feasible solutions to the original problem, we opt to use the latter approximate

problem for which the distances and flows between the parcels are determined as follows:

$$\widetilde{d}_{ij} = \frac{\sum_{u\in P_i}\sum_{v\in P_j} w_{uv}d_{uv}}{\sum_{i\in P_i}\sum_{j\in P_j} w_{uv}} \qquad \forall i,j \in \{1,2,..,r\} \tag{3.4a}$$

$$\widetilde{w}_{ij} = \sum_{i\in P_i}\sum_{j\in P_j} w_{uv} \qquad \forall i,j \in \{1,2,..,r\} \tag{3.4b}$$

Note that the distance between parcels i and j is calculated as the weighted average of the distances between every node pair $\{(u,v) : u \in P_i,\ v \in P_j\}$ and the flow between the parcels is simply the sum of flows between all node pairs. Consequently, the unit cost of sending flow within a parcel is not zero unlike the original USApHMP. We call this approximate problem the *low resolution problem (LRP)* and formally define it as follows:

$$[\text{LRP}] \quad \min \sum_{i=1}^{r}\sum_{k=1}^{r}\sum_{m=1}^{r} \alpha\widetilde{d}_{km}\widetilde{x}_{ikm} + \sum_{i=1}^{r}\sum_{k=1}^{r}(\widetilde{O}_i + \widetilde{D}_i)\widetilde{d}_{ik}\widetilde{y}_{ik} \tag{3.5a}$$

$$\text{s.t.} \sum_{k=1}^{r}\widetilde{y}_{kk} = p \tag{3.5b}$$

$$\sum_{k=1}^{r}\widetilde{y}_{ik} = 1 \qquad \forall i \in \{1,2,...,r\} \tag{3.5c}$$

$$\widetilde{y}_{ik} \le \widetilde{y}_{kk} \qquad \forall i,k \in \{1,2,...,r\} \tag{3.5d}$$

$$\sum_{m=1}^{r}\widetilde{x}_{ikm} - \sum_{m=1}^{r}\widetilde{x}_{imk} = \widetilde{O}_i\widetilde{y}_{ik} - \sum_{j=1}^{r}\widetilde{w}_{ij}\widetilde{y}_{jk} \qquad \forall i,k \in \{1,2,...,r\} \tag{3.5e}$$

$$\widetilde{x}_{ikm} \ge 0 \qquad \forall i,k,m \in \{1,2,...,r\} \tag{3.5f}$$

$$\widetilde{y}_{ik} \in \{0,1\} \qquad \forall i,k \in \{1,2,...,r\} \tag{3.5g}$$

where $\widetilde{O}_i$ ($\widetilde{D}_i$) is the total amount of outflow (inflow) from (to) parcel P_i, $\widetilde{w}_{ij}$ is the total

amount of flow from parcel P_i to parcel P_j, and the decision variables are defined as follows:

$\widetilde{x}_{ikm}$ = Total flow emanating from parcel P_i that is routed between hub parcels P_k and P_m

$$\widetilde{y}_{ik} = \begin{cases} 1 & \textit{if parcel } P_i \textit{ is allocated to hub parcel } P_k \\ 0 & \textit{otherwise} \end{cases}$$

The term low resolution is adopted from the image processing field where a similar procedure, namely, assigning the same color to predetermined sets of neighbouring pixels is a very common practice to achieve computational efficiency at the expense of image quality. Figure 3.7-c illustrates the optimal solution to LRP for a given parcellatio. Hub parcels are marked with filled symbols and non-hub parcels are marked with hallow symbols of the hubs they are allocated to.

LRP is an essential module in the proposed approach as it approximates the original problem with fewer supernodes and thus allowing us to make use of our ability to solve smaller USApHMP instances in solving larger problem instances. Once LRP is solved for a given parcellation, what remains is to obtain a feasible solution to the original problem. We, next, describe the third module which constitutes the link between low resolution problem and the original problem.

3.3.3 Reconstruction

Notice that the solution for the low resolution problem would give us a partitioning of the parcels (and consequently, of the nodes) into p clusters. However, instead of p hub nodes, we have p hub parcels which typically add up to more than p nodes. In order to reconstruct a feasible solution to the original problem, one needs to identify exact locations of p hub nodes. Recall that finding the locations of the hub nodes for a given p-partition of nodes $\mathcal{N}$ is also an existing problem in the literature introduced by Sung and Jin (2001). Wagner (2007) calls the problem CHLP and provides the most efficient formulation for the problem. We next provide the formulation provided by Wagner (2007) which constitutes our third module *Reconstruction*:

Let $\mathcal{C} = \{C_1, C_2, ...C_p\}$ be a partition of $\mathcal{N}$ into allocation clusters such that

$$\begin{aligned} C_i \cap C_j &= \varnothing, \quad \forall C_i \neq C_j \in \mathcal{C} \\ \bigcup_{i=1}^{p} C_i &= \mathcal{N} \end{aligned} \tag{3.6}$$

$$\text{[CHLP]} \quad \min \sum_{c=1}^{p} \sum_{k \in C_c} \sum_{m \in \mathcal{N} \setminus C_c} \alpha d_{km} F_{km} x_{km} + \sum_{c=1}^{p} \sum_{k \in C_c} \sum_{i \in C_c} (O_i + D_i) d_{ik} y_k$$

$$\text{s.t.} \quad \sum_{k \in C_c} y_k = 1 \qquad \forall c = \{1, ..., p\} \tag{3.7a}$$

$$\sum_{m \in C_c} x_{km} = y_k \qquad \forall k \in \mathcal{N} \text{ and } \forall c = \{1, ..., p\} \text{ with } k \notin C_c \tag{3.7b}$$

$$\sum_{k \in C_c} x_{km} = y_m \qquad \forall m \in \mathcal{N} \text{ and } \forall c = \{1, ..., p\} \text{ with } m \notin C_c \tag{3.7c}$$

$$0 \leq x_{km} \leq 1 \qquad \forall k, m \in \mathcal{N} \tag{3.7d}$$

$$y_k \in \{0, 1\} \qquad \forall k \in \mathcal{N} \tag{3.7e}$$

where F_{km} is the total amount of flow from the cluster of node k to the cluster of node m and the decision variables are defined as follows:

$$x_{km} = \begin{cases} 1 & \textit{if } \text{both nodes } k \text{ and } m \text{ are hubs} \\ 0 & \textit{otherwise} \end{cases}$$

$$y_k = \begin{cases} 1 & \textit{if } \text{node } k \text{ is a hub} \\ 0 & \textit{otherwise} \end{cases}$$

Solution of CHLP gives us the optimal hub locations for the given partitions. However, considering the possible information loss while shrinking the original map into a low resolution one, the input partition may not be the optimal one. To avoid suboptimal solutions resulting from resolution errors, we devise a final module to recursively refine the feasible

solutions.

3.3.4 Refining

One way to improve a given feasible solution is to enumerate all its neighborhood solutions which is what heuristics such as tabu search do. Another approach would be to make use of a property of optimum solutions to USApHMP: that the solution of the CHLP with the optimum allocation clusters will give the optimum hubs and similarly, the solution of another related problem known as pHAP with the optimum hubs will give the optimum allocation clusters. Hence, by recursively solving CHLP and pHAP one may reach a local optima. Considering the fact that the feasible solution generated by the first three modules is not any random solution, but it is found by taking the problem data into account, our hope is that the local optima hit at the end of *Refining* will be very close to global optimum. Next, we provide the mathematical formulation of pHAP for a given subset $\mathcal{H} \subseteq \mathcal{N}$ of p hubs:

$$[p\text{HAP}] \quad \min \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{H}}\sum_{m\in\mathcal{H}} \alpha d_{km} x_{ikm} + \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{H}} d_{ik}(O_i + D_i) y_{ik} \tag{3.8a}$$

$$\text{s.t.} \quad y_{kk} = 1 \qquad \forall k \in \mathcal{H} \tag{3.8b}$$

$$\sum_{k\in\mathcal{H}} y_{ik} = 1 \qquad \forall i \in \mathcal{N} \tag{3.8c}$$

$$y_{ik} = 0 \qquad \forall i \in \mathcal{H} \text{ and } k \in \mathcal{H} \setminus i \tag{3.8d}$$

$$\sum_{m\in\mathcal{H}} x_{ikm} - \sum_{m\in\mathcal{H}} x_{imk} = O_i y_{ik} - \sum_{j\in\mathcal{N}} W_{ij} y_{jk} \qquad \forall i \in \mathcal{N} \text{ and } \forall k \in \mathcal{H} \tag{3.8e}$$

$$0 \le x_{ikm} \le 1 \qquad \forall i \in \mathcal{N} \text{ and } \forall k, m \in \mathcal{H} \tag{3.8f}$$

$$y_{ik} \in \{0, 1\} \qquad \forall i \in \mathcal{N} \text{ and } \forall k \in \mathcal{H} \tag{3.8g}$$

Notice that pHAP is the same formulation as USApHMP, only reduced in size. The constraint sets (1b) and (1d) become redundant in pHAP. For the sake of consistency, (9b) and (9d) are added to the formulation of pHAP, however the model can be further reduced by dropping the variables whose values are fixed by constraints (9b) and (9d).

This concludes the description of modules in the proposed approach. Hereinafter, we refer to this approach as SPATIAL and outline its flow in Algorithm 3. Table 3.2 provides brief descriptions of the functions used in Algorithm 3 and Figure 3.7 illustrates a sample flow of the approach at each module. In Sections 5.2 and 5.3, we first illustrate the efficiency of the approach on known USApHMP instances from AP dataset and then use it to find quality solution for the very large BCN instances.

Table 3.1: Descriptions of functions.

Function	**Output**	**Description**
GET.FEASIBLE($\tilde{\boldsymbol{y}}$)	$\boldsymbol{y}$	Generates a feasible solution from the optimal solution of USApHRP
GET.CLUSTERS($\boldsymbol{y}$)	$\mathcal{C}$	Extracts the allocation clusters from a given feasible solution
GET.HUBS($\boldsymbol{y}$)	$\mathcal{H}$	Extracts the hub nodes from a given feasible solution
SOLVE.RPP($\mathcal{I}, r$)	$\mathcal{P}$	Divides the map into r spatial parcels and allocates each node into one parcel
SOLVE.LRP($\mathcal{I}, \mathcal{P}$)	$\tilde{\boldsymbol{y}}$	Finds the optimal solution to low resolution problem for a given parcellation
SOLVE.CHLP($\mathcal{I}, \mathcal{C}$)	$\boldsymbol{y}$, z	Finds the optimal solution to cluster hub problem for given clusters
SOLVE.PHAP($\mathcal{I}, \mathcal{H}$)	$\boldsymbol{y}$, z	Finds the optimal solution to node allocation problem for given hubs

Algorithm 3 SPATIAL Algorithm

function SPATIAL($\mathcal{I}, r$)
  $\mathcal{P} \leftarrow$ SOLVE.RPP($\mathcal{I}, r$)
  $\tilde{\boldsymbol{y}} \leftarrow$ SOLVE.LRP($\mathcal{I}, \mathcal{P}$)
  $\boldsymbol{y} \leftarrow$ GET.FEASIBLE($\tilde{\boldsymbol{y}}$)
  $\mathcal{C} \leftarrow$ GET.CLUSTERS($\boldsymbol{y}$)
  $z_C \leftarrow \infty$
  $z_A \leftarrow 0$
  while $z_C \neq z_A$ **do**
    $[\boldsymbol{y}, z_C] \leftarrow$ SOLVE.CHLP($\mathcal{I}, \mathcal{C}$)
    $\mathcal{H} \leftarrow$ GET.HUBS($\boldsymbol{y}$)
    $[\boldsymbol{y}, z_A] \leftarrow$ SOLVE.PHAP($\mathcal{I}, \mathcal{H}$)
    $\mathcal{C} \leftarrow$ GET.CLUSTERS($\boldsymbol{y}$)
  return $\boldsymbol{y}$, z_A

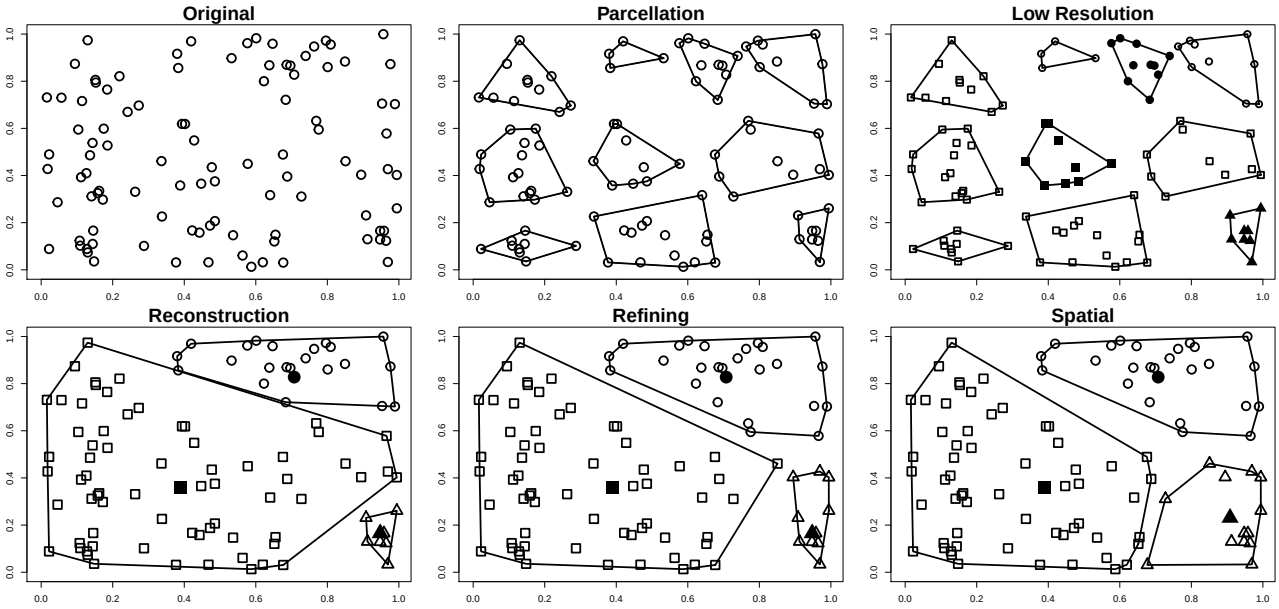

Figure 3.7: Step-by-step flow of SPATIAL Algorithm for a random instance with $n = 100$ nodes and $p = 3$ hubs

3.4 Computational Experiments

The main purpose of the computational experiments is to illustrate the non-coincidental existence of the *spatial separability* and the remarkable computational gain resulting from its incorporation in a novel approach to solve USApHMP instances. Following the brief description of the experimental framework, we first show the validity of the spatial separability property in two dimensional space. We, then, test the performance of the proposed approach on the problem instances from well-known datasets. Finally, we conduct computational experiments to illustrate the effectiveness of the hub location problems in modelling the brain hub organisation, and to provide further evidence on the scalability of the proposed approach.

The testing is done on benchmark instances (AP dataset) provided in Ernst and Krishnamoorthy (1996), BCN instances introduced in Section 4, an randomly generated instances. For a given set of nodes $\mathcal{N}$, a *map* is defined by the coordinates of the nodes in two

or three dimensional space and the flows W_{ij} between each pair of nodes. Random maps are generated by randomly assigning a values to each coordinate $\forall i \in \mathcal{N}$ and w_{ij} $\forall i, j \in \mathcal{N}$ from a uniform distribution between 0 and 1. For a given map, a USApHMP *instance* is defined by setting the values for 2 parameters: namely, p (the number of hubs to open), and α (interhub discount factor).

We used R programming environment for the implementation of the proposed algorithm and Gurobi 6.0.2, with default parameters and a single thread, to solve the MILP instances. All experiments were carried out on an Intel(R) Core(TM) i7-4510U PC (2.60GHz, 8GB memory) running Windows 10. We limited the processing time of all experiments to 3600 seconds.

3.4.1 Spatial Separability

In Figure 2, we showed that the optimal solutions of a few instances from AP dataset possess what we call spatial separability property. In this section we try to generalize this observation. To this end, we first checked 100 instances generated using AP dataset (with different parameters) and validated that the said property holds for all of them. Next, we wanted to validate that this property is not specific to AP instances. To do so, we resorted to random maps and instances. Generation of a random map was described at the beginning of this section. We generated 100 random maps for each $n \in \{25, 50\}$. Then, for each map, we generated 12 instances with different values of $p \in \{2, 3, 4, 5\}$ and $\alpha \in \{0.1, 0.5, 0.9\}$. Figure 3.8 shows an example of the spatial separability of 12 instances generated from a random map with different parameters. As clearly seen from the figure, the convex hulls of the allocation clusters in optimal solution are disjoint in all 12 instances. Similar plots are generated for all random maps. The observations mostly conferred with Figure 3.8. However, we also observed several instances for which the spatial separability was violated. Table 3.2 shows the fraction of instances for which the spatial separability holds.

Several insights are due from Table 3.2. First, for 99.5% (2387/2400) of the randomly generated instances spatial separability holds. Second, $p = 2$ is the only p value for which spatial separability is violated. Moreover, although stronger evidence is needed for such

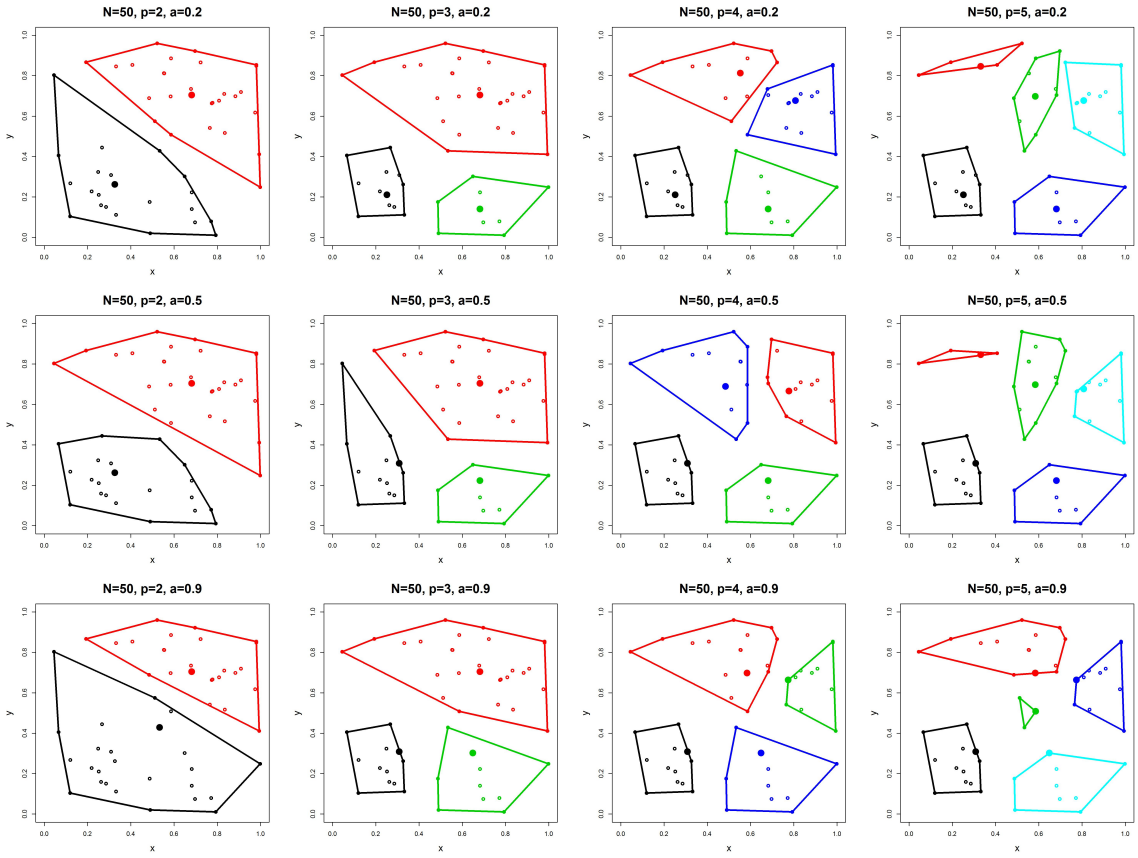

Figure 3.8: An example of spatial separability of optimal solutions on a random map

generalizations, we observe that the number of nonseparable instances increase with n and α. Unfortunately, such mass computational experiments are not possible for larger n values due to exponentially increasing processing time to solve USApHMP instances optimally. Nevertheless, the existence of violating instances suffices to refute the claim that spatial separability in 2D space is a property of USApHMP.

Although there are few instances for which spatial separability is violated, overwhelming majority of the instances are spatially separable. This brings forward a question: How much would we lose from optimality if we looked only for the spatially separable solutions. More specifically, is there an upper bound on the optimality gap of feasible solutions restricted to hold spatial separability property or can it be made arbitrarily large? Since

	$n = 25$			$n = 50$		
p	$\alpha = 0.1$	$\alpha = 0.5$	$\alpha = 0.9$	$\alpha = 0.1$	$\alpha = 0.5$	$\alpha = 0.9$
2	100/100	100/100	**99/100**	100/100	**98/100**	**90/100**
3	100/100	100/100	100/100	100/100	100/100	100/100
4	100/100	100/100	100/100	100/100	100/100	100/100
5	100/100	100/100	100/100	100/100	100/100	100/100

Table 3.2: Fraction of separable instances for different parameter values

for the majority of the violating instances in our experiments we could establish spatial separability by adding only a few intersection elimination constraints (3a-3f), we were able to check their optimality gaps manually. Not surprisingly, the optimality gaps were very small ranging between 0.1-0.5% which indicates that even if we imposed spatial separability *a priori*, we would not lose much from optimality. Whether such small optimality gaps are consistent throughout different parameter sets and maps is subject to a more detailed computational experimentation which we refrain from in this study.

3.4.2 Performance on AP Dataset

We implemented the proposed approach on instances selected from the AP dataset with 100 nodes and $p \in \{2, 3, ..., 10\}$ hubs. Table 4 shows the optimal solutions (OPT), the LP relaxation (LP) using the MILP formulation given in Ernst and Krishnamoorthy (1996), and the results from the proposed approach with different number of parcels ($r \in \{25, 50\}$). Since the optimal solutions to the instances listed above could be found by Gurobi, both LP gap (lower bound) and SPATIAL gap (upper bound) are calculated with respect to the optimal solution.

It is evident that the proposed approach generates high quality solutions in a reasonable amount of time. In terms of CPU time, SPATIAL takes significantly less time compared to both OPT and LP, which hints about the scalability of the proposed approach. As expected, SPATIAL($r = 25$) requires much less time than SPATIAL($r = 50$). It is, however, interesting to observe that the gaps for SPATIAL($r = 25$) are very similar to those for SPATIAL($r = 50$), which implies that the resolution ratio of 1:4 is almost as efficient as the

Table 3.3: Computational results on selected problem instances from AP dataset

	OPT		LP		SPATIAL($r = 25$)		SPATIAL($r = 50$)	
p	CPU (sec)	z	CPU (sec)	Gap (%)	CPU (sec)	Gap (%)	CPU (sec)	Gap (%)
2	158.26	180223.80	40.54	0.46	0.70	0.00	3.12	0.00
3	406.41	160847.00	93.30	0.76	0.78	0.00	3.34	0.00
4	647.69	145896.58	77.00	0.96	0.75	0.00	6.67	0.00
5	2636.83	136929.44	132.97	1.77	1.72	0.00	19.38	0.58
6	7250.79	129214.65	126.18	1.88	1.79	0.63	24.01	0.63
7	3210.05	122577.13	130.08	2.16	1.42	1.04	20.79	1.04
8	2260.60	116410.90	116.00	1.76	1.67	1.33	19.05	2.01
9	2335.53	111108.79	102.20	1.37	1.17	2.24	29.65	2.37
10	1620.84	106469.57	85.84	1.32	1.71	2.83	25.46	1.38

ratio of 1:2. Finding the resolution ratio which provides the optimum trade-off between the computation time and the solution quality would be an interesting research question, though it is out of the scope of this study.

Next, we conduct experiments to compare the performance of SPATIAL($r = 50$) with Gurobi in terms of optimality gap and CPU time. Table 3.4 shows the results from Gurobi and SPATIAL($r = 50$) for different parameter values. For each (n, p, α) parameter set considered, the following experiment is conducted: First, SPATIAL($r = 50$) is run and the CPU time and objective value z_S are recorded. Then, Gurobi is run for the same amount of time and the lower bound (LB) as well as the upper bound z_G (i.e., the best feasible solution found within the same amount of time) is recorded. Since the optimal solutions are not known for these instances, the gaps are calculated as $\delta_S = 100\ (z_S - LB)/z_S$ and $\delta_G = 100\ (z_G - LB)/z_G$, respectively and the best of the two gaps is written in bold. Typically for instances with $n = 200$, Gurobi was not able to find a feasible solution within the time required for running SPATIAL. For those cases, we let Gurobi run until it found a second feasible solution. This was done to avoid any unfairness in comparison, since it is typical for commercial solvers like Gurobi to find a drastically improved second feasible solution within a few seconds after the first one. Nevertheless, it becomes clear in Table 3.4 that SPATIAL significantly outperforms Gurobi across all parameters sets, which points

Table 3.4: Comparing the performance of SPATIAL and GUROBI on AP instances

		$\alpha = 0.1$				$\alpha = 0.5$				$\alpha = 0.9$			
n	p	CPU	LB	δ_S (%)	δ_G (%)	CPU	LB	δ_S (%)	δ_G (%)	CPU	LB	δ_S (%)	δ_G (%)
	2	3.4	68471.5	**0.06**	0.16	7.3	77766.5	**1.16**	13.45	20.4	85382.7	**1.98**	3.17
	3	3.5	59000.6	0.35	**0.32**	16.9	70239.7	**1.91**	**1.91**	58.8	79090.9	**4.22**	7.54
	4	4.4	51956.3	**0.42**	1.28	17.9	65045.8	**1.73**	2.79	96.5	75478	**3.95**	15.48
	5	9.4	47373.3	1.27	**0.32**	45.9	61476.5	**3.26**	11.96	182.6	73054.7	**4.98**	8.00
100	6	7.6	43543.4	0.78	**0.42**	44.3	58765.7	**3.71**	14.78	752.3	71334.1	**5.38**	22.81
	7	11.8	40640.8	**1.75**	4.05	36.2	56448.4	**3.85**	14.7	260.0	69969.4	**5.11**	24.82
	8	8.0	38041	**0.88**	1.01	50.0	54511.1	**3.42**	17.14	584.0	68792.7	**5.55**	23.25
	9	11.5	35616.3	**1.4**	2.74	59.5	52907.6	**3.34**	17.4	1312.4	67836.9	**4.42**	18.59
	10	11.4	33484.4	**1.34**	3.96	90.4	51480	**4.06**	12.54	1618.7	67017.4	**4.33**	24.31
	2	4.3	69188.4	**0.05**	**0.05**	9.9	78526.6	**1.31**	12.36	24.7	85741.3	**1.85**	12.05
	3	7.4	59618.1	0.25	**0.2**	30.6	70918.7	**2.16**	2.38	67.4	79584.9	**4.76**	9.16
	4	4.6	52605.4	0.49	**0.11**	26.3	65594.9	**2.03**	3.42	116.6	75867.1	**5.06**	17.65
	5	15.1	48546.3	**0.67**	1.47	42.0	62208.8	**3.64**	16.12	380.9	73476.3	**5.97**	21.86
200	6	4.9	44880.8	**0.85**	5.32	44.8	59584.9	**4.46**	31.16	1076.2	71760.6	**5.94**	12.87
	7	16.3	41793.2	**0.82**	7.86	63.7	57301.3	**4.46**	20.09	1096.9	70418.0	**6.57**	21.34
	8	22.0	39103.9	**1.32**	4.47	113.6	55359.9	**3.92**	31.37	1004.4	69312.2	**5.94**	24.03
	9	17.6	36800.9	**1.62**	18	141.2	53750.5	**5.06**	19.62	1359.2	68391.2	**6.21**	20.91
	10	15.4	34679.0	**2.06**	24.83	154.5	52401.5	**5.18**	23.7	1460.4	71141.8	**4.99**	22.06

to the global efficiency of the proposed approach.

A final remark on the results from Table 3.4 is that the performance of SPATIAL deteriorates with increasing α both in terms of CPU time and in terms of optimality gap. This observation concurs with the results presented in Table 3.2. Namely, since the spatial separability is more likely to be violated for larger values of α, it is expected that the performance of the proposed approach, which is based on spatial separability, deteriorates with increasing α.

3.5 Conclusions

We investigated HLP as an example of the use of data-driven techniques on the *problem*. The main challenge of using HLP in certain promising domains is the inability of current solution approaches to handle large instances (e.g., networks with more than 1000 nodes). In this work, we uncovered an important pattern in the optimal hub networks: *spatial*

separability. We showed that at the optimal solutions, nodes are typically partitioned into allocation clusters in such a way that convex hulls of these clusters are disjoint. We, then, exploited this pattern and proposed a new data-driven approach that uses the insights generated from the solution of a smaller problem - low resolution representation - to find high quality solutions for the large HLPs. Computational experiments conducted on the well-known benchmark instances corroborates the effectiveness of the proposed methodology.

Chapter 4

Identifying Hub Regions in Brain Connectivity Networks

In this chapter, we introduce a new application area for the HLP originating from human BCN that we hope will benefit from the recent advances in hub location. We conduct comprehensive computational experiments on a dataset introduced in this chapter with the largest (998 nodes) and first three-dimensional problem instances in the literature. We implement the data-driven method proposed in Chapter 3 to provide evidence for the effectiveness of the proposed approach in producing similar results to the ones established in medical literature.

4.1 Introduction and Literature

Recent interest of the medical and neuroscience community in building a "network map" (connectome) of the human brain lead to the rapid development of medical imaging technologies that can probe the neural circuitry of human brains with high spatio-temporal resolution. Using high-resolution diffusion tensor/spectrum imaging (DTI/DSI) data, researchers have shown that some regions in the brain cortex play a hub role and act as an intermediary to transmit signals between other regions (van den Heuvel and Sporns,

2013). This is believed to provide an evolutionary advantage in reducing the size of the brain (Bullmore and Sporns, 2012). As damage to these regions causes serious problems (van den Heuvel et al., 2010; Zalesky et al., 2011), it is important to identify these hub regions as precisely as possible. Researchers have hitherto used graph-theoretical measures (Crossley et al., 2013; Bassett and Sporns, 2017) to study the connectivity networks of the human brain and identify hub regions. To the best of our knowledge, identifying hub regions in BCN has never been studied within an optimization context.

We believe that HLP can be useful in identifying the precise locations of hub regions in BCN provided that the resulting large instances can be solved efficiently. Currently, the state-of-the-art methods are capable of solving instances with only a few hundred nodes within a reasonable amount of time. Considering the fact that there are approximately 86 billion neurons in an adult human brain (Azevedo et al., 2009), one can argue that being able to optimally solve a problem with a few hundred nodes is not appealing enough by itself from the practical point of view as it would yield very low precision. To make the mathematical HLP models more relevant for practical purposes, general-purpose methods that can efficiently tackle very large problem instances should be available to end users. We propose one such method in Section 3.

It is not uncommon that methodologies that work well on moderate size instances become intractable as the problem size increases. Hence, it is essential to test the performance of proposed methodologies not only on small-to-moderate size problems, but also on very large problem instances. The largest problem instances in the literature, however, have only 200 nodes (Ernst and Krishnamoorthy, 1996). Scarcity of scalable solution methodologies may be attributed to the absence of large practical benchmark instances. To bridge this gap, a BCN dataset originally generated by Hagmann et al. (2008) is re-introduced in Section 4 in a way amenable to existing HLP models. This dataset is the largest (with $n = 998$ nodes) and the first three-dimensional dataset in the literature so far. In Section 5, we generate benchmark instances using BCN dataset to evaluate the performance of the proposed approach.

The human brain is one of the most complex systems known. It has approximately 86 billion neurons in a human brain and each neuron, in turn, is connected to a number of other neurons. Figure 4.1 shows the anatomy of a typical neuron and the way it is

connected to another neuron through synapses. The cell bodies of neurons are located at the outer shell (cortex, or gray matter) of the brain whereas the axons (the fibers that connect cell bodies) are located inside the cortex (white matter). It is now understood that every single activity we do, every thought that comes to our minds, every emotion we experience, and every memory we have is coded in our brains. Evidence suggests that cognitive processes ranging from visual recognition (Behrmann and Plaut, 2013) and language (Friederici and Gierhan, 2013) to emotion (Pessoa, 2012) are results of dynamic interactions, via instantaneous electrical impulses known as action potential, between different regions in our brains. Therefore, a complete understanding of how our brains work will only be possible once we are able to map the entire brain network.

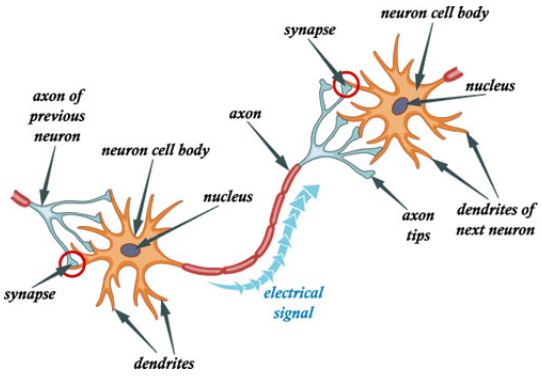

Figure 4.1: Representation of a single neuron cell

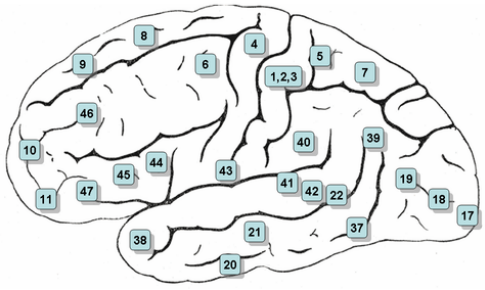

Figure 4.2: Brodmann Areas in the human brain

The Human Connectome Project (HCP) initiated in 2013 and aiming to create a complete map of the anatomic and functional connections in the human brain is probably one of the biggest collective projects involving scientists from various disciplines. The two biggest challenges of this project are: (1) to probe the resolution levels that allow distinguishing every single neuron, (2) to store and process the data emanating from such high-resolution imaging. One of the methods that proved very useful in abating the enormity of the task is to cluster neurons based on their locations and the observed functions. This allows us to work on the resolution level of the brain areas instead of single neurons. Figure 4.2 shows an example of how the brain is divided into different areas commonly referred to as Brodmann areas. This parcellation of the brain cortex into 52 areas based on histological

properties of neurons was proposed by Brodmann (1908) and has since been widely accepted by neuroscientists. The recent advances in imaging technologies allowed scientists to improve the parcellation proposed by Brodmann (1908). Glasser et al. (2016) proposed a new parcellation with 180 areas per hemisphere bounded by sharp changes in cortical architecture, function, connectivity, and/or topography using multi-model magnetic resonance images from the HCP database. Based on the current trends, it is safe to say that as our understanding of the brain improves, our need for higher resolution data acquisition and processing will increase proportionally, too. Hence, tools that allow processing high resolution data will be in high demand.

One of the revelations of the recent advances in brain studies is the realization that the functional specialization of the brain regions alone cannot fully account for most aspects of the brain function and that dynamic interactions across multiple regions are required to accomplish cognitive processes (van den Heuvel and Sporns, 2013). One aspect of the brain organisation that appear particularly important in enabling integration of distributed neural information is that important integrative functions are performed by a specific set of brain regions and their anatomical connections. In recent years, anatomical and functional organization of the brain has been approached from the perspective of complex networks and numerous studies described neural systems in terms of graphs or networks comprising nodes (neurons and/or the brain regions) and edges (synaptic connections, interregional pathways) (Fox et al., 2006; Vincent et al., 2007; Honey et al., 2007). The comprehensive network map of the nervous system of a given organism is called its *connectome* and represents a structural basis for dynamic interactions to emerge between its neural elements (Le Bihan et al., 2001; Hagmann et al., 2003). Several studies have independently concluded that human connectome combines attributes pertaining to specialization with attributes that ensure efficient communication (integration). At the center of the latter lies network elements that are often referred to as hub nodes/regions which are generally characterized by their dense connectivity to other regions and their central placement in the network.

4.2 The Human Brain as a Complex Network

Identifying the locations of hub regions as precisely as possible is very important from clinical perspective since any damage to these regions is likely to cause more serious problems, especially in cognitive functions requiring high integrative processes. This knowledge could have immediate implications for neurosurgery such as deciding the location of surgical opening, trade-off between removing tumour region and keeping hub region, applying/avoiding radiation to hub regions.

Current literature on identifying the hub regions in the human brain rely highly on network representations where the brain is parcelled into contiguous regions each corresponding to a node, and the density of the white matter between any pair of these regions corresponding to an edge. Several graph theoretical concepts such as strength, betweenness, and coreness are then used as proxy to measure connectivity and centrality of nodes. The advantage of using these proxy measures is the computational efficiency in that it takes only seconds to provide a hubness rank for all the network nodes. However, it is very common that a measure pertaining to connectivity provide completely different hubness ranks than those obtained by a measure of centrality. Researchers, then, typically average the ranks obtained by various measures to calculate an overall hubness rank.

Reconciling the discrepancy between connectivity-based and centrality-based ranks by a simple average carries the risk to be an oversimplification of the underlying relationship. In fact, it is known in the hub location literature that a high discrepancy between centrality and connectivity in a network is typically an indicator of an underlying multi-centre hub topology, i.e., not a single contiguous hub region but multiple hub regions across the network and a linear combination of the two measures often performs very poorly in identifying the optimal hub structure Klincewicz (1992). Therefore, one may argue that the mathematical models for the HLPs could potentially be a more proper way to account for the aforementioned discrepancy. HLP models provide not only a list of hub locations, but also a complete list of flow pathways between pairs of nodes that elucidates the physical reason why a certain node is more likely to be a hub.

One of the main contributions of this study is to provide a comparative computational experiments in Section 4.3.1 to explore the potential of HLP models as an alternative tool

in explaining the underlying hub organisation of the human brain. We attempt to do that by using an existing dataset from the literature by Hagmann et al. (2008). The dataset is generated by dividing the brain cortex into 998 ROIs with an average size of $1.5cm^2$ and measuring the density of white matter between ROIs using diffusion spectrum imaging (DSI) technology. It includes the following information:

- *FiberDensity*: A 998×998 connection matrix of 998 ROIs measured as the thickness/density of white fiber between every pair of ROIs. This data roughly corresponds to the flow matrix ($\mathbf{W}$) used in hub location literation.
- *EdgeLength*: A 998×998 distance matrix of 998 ROIs measured as the length of fiber between every pair of ROIs with nonzero fiber density. To account for the distances between ROI pairs with no fiber connection, another matrix *ShortestLength* is generated by calculating the shortest distances between every pair.
- *ROICoordinates*: A 3×998 matrix which consists of the three dimensional coordinates of ROIs.
- *ROILabels*: A vector of size 998 which shows the Brodmann area each ROI belongs to.

There are several issues to address before conducting computational experiments. Firstly, the spatial separability property -which the SPATIAL approach is based upon- assumes euclidean distances whereas the distances between two brain ROIs are recorded as fiber length which does not necessarily satisfy triangular inequality. Moreover, when we compared the recorded fiber lengths between two ROIs against the euclidean distances calculated based on the ROI coordinates, we noticed for some ROI pairs, the fiber length was smaller than the euclidean distance which indicates that there are measurement errors in the recorded data as euclidean distance is the smallest possible distance between the two points in space. By excluding these pairs, we calculated the correlation coefficient between the euclidean distances and the fiber lengths as 0.904. This high correlation implies that the fiber connections between two ROIs roughly follow a straight line which justifies using euclidean distances as a proxy for the fiber lengths.

Methodology adopted by Hagmann et al. (2008) to determine hub ROIs in the brain network calculates the network measures such as degree, strength, coreness, efficiency and centrality for each node and then ranks the nodes based on their overall score. They suggest that this overall ranking could be interpreted as the "hubness" of a node. In the paper, they report results on the low resolution brain map with 66 Brodmann areas. Following same the steps, we first regenerated similar results for the high resolution brain map with 998 ROIs. Figure 4.3 shows the 20 highest ranking ROIs with respect to each network measure mentioned above and the overall hubness rank.

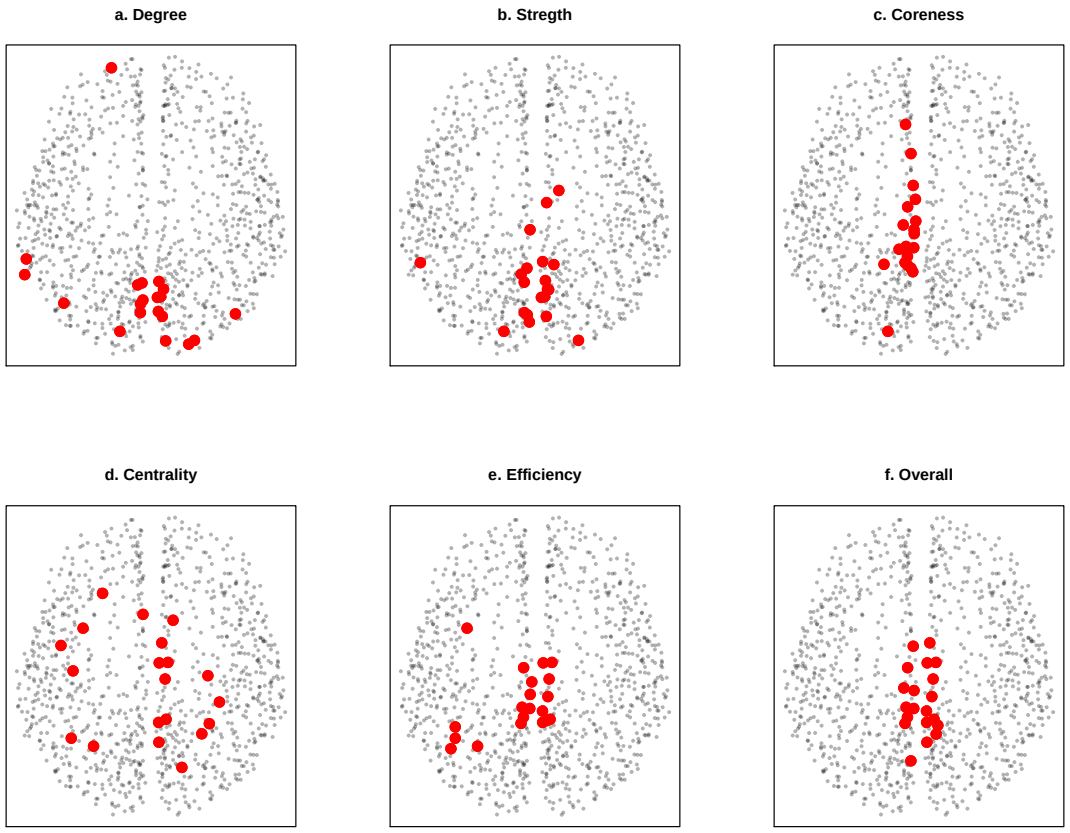

Figure 4.3: Hub ROIs found by each network measure

Several observations are due from this Figure 4.3. It is argued by Hagmann et al. (2008)

that a hub ROI is expected to rank among the highest for all the attributes represented by the aforementioned proxy network measures. It becomes evident from Figure 4.3, however, that the rankings with respect to these network measures do not necessarily concur. Although there seems to be some overlap (along the area between two hemispheres), discrepancies outweigh the similarities. For example, the *centrality* measure seems to suggest a more distributed hub organisation across the brain map, whereas *coreness* suggests a very concentrated hub region along the area between two hemispheres. On the other hand, *degree* and *strength* provide very similar results in terms of the distribution of hub ROIs across the brain map. This brings up the question: aren't we biasing the *overall* hubness rank towards the distribution provided by *degree* and *strength* as we essentially account for the "same" distribution twice?

When combining multiple conflicting measures into a single overall measure, taking a simple average is usually considered a naive approach since it ignores possibly different weight factors associated with each measure. Although, using different weights for each proxy measure may alleviate this issue to some extend, there are many other issues with using proxy measures that are hard to address. Are these five proxy measures all that one would expect a hub ROI to rank good at? An important measure that seems to be missing, for instance, is the measure of interaction between hubs. Since hubs, by definition, are interacting entities, one would expect the "hubness" of a node in a given network to depend on the set of other hub nodes. This, however, is not a trivial measure to account for by a simple proxy measure due to its inherent non-linearity. USApHMP model is known to account for this non-linearity which is, in large part, the reason why it is very challenging to solve very big problem instances.

4.3 Computational Experiments

Finding a mathematical model that captures the underlying mechanisms of hubs is an essential step towards understanding the true nature of the brain hub organisation. This is what we attempt to accomplish in this section by computationally testing the validity of USApHMP in modelling the brain hub organisation and the effectiveness of Spatial

algorithm in providing near-optimal solutions to large USApHMP instances. We conduct experiments mainly to compare the results obtained by the state-of-the-art techniques in the medical literature (Hagmann et al., 2008) and those obtained by the SPATIAL algorithm. Throughout this section, we use the real BCN data provided in Hagmann et al. (2008) which constitute the largest hub location problem instances (with $n = 998$ nodes) in the literature to the best of our knowledge.

4.3.1 Comparison of SPATIAL and Hagmann

We, next, provide a comparison between the main results of Hagmann et al. (2008) and the solutions found by the USApHMP model. It is worth mentioning that unless otherwise stated, throughout the chapter the results pertaining to the USApHMP models are obtained by using $\alpha = 0.5$ since they appear to be in more agreement with the results of Hagmann et al. (2008). In the original paper, they identify 16 Brodmann areas (*posterior cingulate cortex, the precuneus, the cuneus, the paracentral lobule, the isthmus of the cingulate, the banks of the superior temporal sulcus, and the inferior and superior parietal cortex*, in both hemispheres) as hub regions, which corresponds to 236 ROIs in total. This implies that almost one quarter of the whole brain cortex assumes a hub role. Moreover, this also assumes that the ROIs within the same Brodmann area will all either be hub or non-hub, which does not help much in isolating the specific hub regions. Finally, using USApHMP to find $p = 236$ optimal hub ROIs in a network of $n = 998$ nodes would be computationally infeasible. Hence, we resort to some simplifications.

First, we select only the top 10 hub ROIs identified by Hagmann et al. (2008) (shown in Figure 4.4-a) since with $p = 10$ the problem becomes feasible to work on. Notice that hub ROIs are spatially adjacent to each other. This implies that there is a single contiguous hub region which is made up of 10 ROIs. However, it is known that USApHMP model typically distributes the hubs all over the given map. Hence, it is very unlikely that USApHMP with $p = 10$ hubs will reproduce similar results in terms of hub locations. On the other hand, the spatial contiguity of hub region signals a connection with one of the problems introduced in Section 3.3: the LRP. Specifically, if the original brain map with 998 ROIs is reduced to a lower resolution map with approximately 100 contiguous regions, then the 1-hub LRP

is expected to produce similar results to the one reported in Hagmann et al. (2008).

Figure 4.4-b shows the optimal solution to 1-hub LRP. The 10 ROIs identified as hubs by LRP belong to the following brain areas: *paracentral lobule, posterior cingulate cortex, and precuneus* in both right and left hemispheres. As expected, the locations of the hub ROIs are roughly in agreement with those found by Hagmann et al. (2008). Slight differences may be attributed to approximate nature of both approaches, and in principle, they can be calibrated in a way to produce more similar results. For example, a different parcellation technique for the LRP could be used so that one of the resulting parcels contains exactly the same ROIs as the hub ROIs of Hagmann et al. (2008). In fact, just to confirm, we manually placed Hagmann's hub ROIs in a single parcel (P_1) and divided the rest of the ROIs into 99 parcels ($P_2, P_3, ...P_{100}$). As argued, the resulting LRP finds P_1 as the hub region. When we calculated the objective value of the solution with hubs identified by Hagmann et al. (2008) ($z_a = 29883.75$ for Figure 4.4-a), it turned out to be better than that of the original 1-LRP ($z_b = 30906.03$ for Figure 4.4-b) which indicates that the objective function of USApHMP is suitable to assess the *quality* of a given subset of hubs and also implies that there may be an even better subset with better objective value.

To this end, we implemented another variant of USApHMP as follows: First, we find a single hub ROI that is selected as the optimal hub for USA1HMP (i.e., single hub problem). Then, we iteratively add new ROIs to the subset of hubs until we have $p = 10$ hub ROIs. In doing so, we restrict the candidate ROI set only to those that are spatially contiguous to the current hubs and greedily select the candidate addition of which to the current hub set improves the objective function value the most. Figure 4.4-c shows this solution which resembles the Hagmann's more both in terms of the locations of hub ROIs (i.e., stretched along the vertical axis) and in terms of the objective function value ($z_c = 28481.40$).

These preliminary results are important as they imply that a variant (LRP or Greedy-contiguous) of USApHMP model can be used to reproduce the results similar to those reported in medical literature which are based solely on network topological measures. The significance of this implication lies in what more USApHMP model can provide. A mathematical model can tell us not only which ROIs are likely to behave as hubs but also what makes them more favourable candidates for "hubness" compared to the others. To

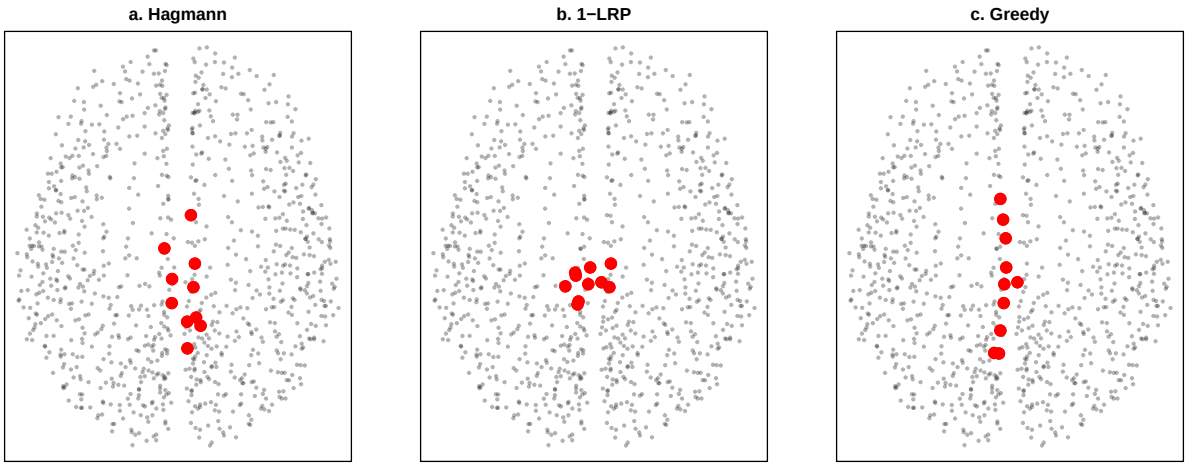

Figure 4.4: Comparison of hub ROIs found by Hagmann et al. (2008) and USA*p*HMP

elaborate, there are two main factors that determine which nodes are selected to be hubs in a given network: spatial centrality and flow-based centrality. The former simply indicates how centrally a node is located in the network whereas the latter pertains to the amount of flow that goes through close proximity of a node. The two are not always in perfect agreement and often need to be reconciled. In fact, of the six network measures used in Hagmann et al. (2008), some such as *efficiency* pertain to spatial/topological centrality and some such as *strength* to flow-based centrality. We observed that there is a significant discrepancy between the node ranks calculated according to different network measures. This makes the assumption that *average rank of a node is a good proxy for its hubness* an oversimplification. On the other hand, USA*p*HMP provides an alternative framework under which the two factors can be reconciled in a way to minimize the "transportation" cost of the overall system.

4.3.2 Multi-center Hub Organisation

Moreoever, hub ROIs identified by network measures will likely always correspond to a monocentric hub structure because typically two neighbouring nodes have similar scores

on these network measures. However, monocentricity of the brain hub structures has not yet been established. In fact, in the same study Hagmann et al. (2008) consider another model where ROIs are clustered into different modules and hub ROIs are identified for each module. This model is more similar to USApHMP in the general sense, yet it, too, is just a descriptive model. Namely, it does not provide information on what is the advantage of having a certain set of hub ROIs as opposed to a different set. Conversely, by providing the pathways all pairwise flows are routed through for a given set of hubs, USApHMP can provide deeper insight on the physical/economical advantage of one set over another. Hence, it is worthwhile to illustrate the parallelism between the result of module-based approach of Hagmann et al. (2008) and the results of SPATIAL($p = 6, \alpha = 0.5, r = 66$) algorithm. Figures 4.5-a, b show the 6 allocation clusters identified by SPATIAL algorithm, and by Hagmann et al. (2008), respectively on the superior view of the brain. To have a better understanding of how the clusters look in 3D, we also included the lateral views of left and right hemispheres for both SPATIAL and Hagmann in Figures 4.5-c-f. ROIs in each cluster (module) are marked with the same color in both figures. Moreover, a straight line is drawn between each ROI and the hub it is allocated to to illustrate the pathways.

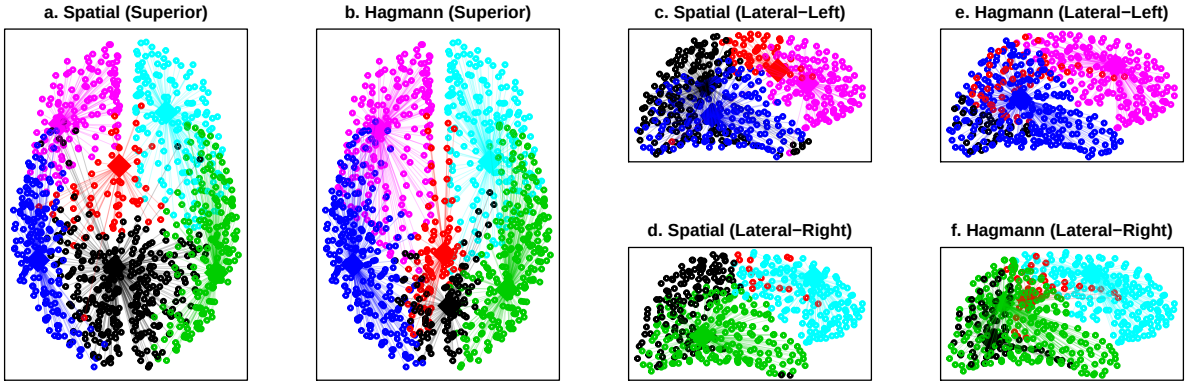

Figure 4.5: Comparison of the six brain modules identified by SPATIAL and Hagmann et al. (2008)

We observe a clear resemblance in the distribution of ROIs to clusters and modules.

First of all, both figures reflect the highly symmetrical nature of the brain. Specifically, both figures show that there are two clusters (marked with black and red colors) that span across left and right hemispheres, whereas the remaining four clusters consist of ROIs only from a single hemisphere (blue and pink from left hemisphere, green and turquoise from right hemisphere, respectively). A symmetry is also apparent among these four clusters; specifically, blue-green and pink-turquoise clusters are approximately pairwise symmetrical. The only visible difference between Hagmann's modules and SPATIAL's allocation clusters is the span of interhemispheric clusters (black and red). In particular, SPATIAL's interhemispheric clusters appears to be more widely spread than Hagmann's which is mostly constricted along the axis between the two hemispheres. To quantify the difference between the two clustering, we solved a CHLP with Hagmann's modules and compared its objective value ($z_H = 29936.71$) against the objective value from the SPATIAL ($z_S = 29248.25$). 2.35% relative gap between the two objective values can be interpreted as a quantitative indicator of the similarity between the two solutions.

4.3.3 Benchmark Instances

Whether the ideal segmentation of the human brain indeed consists of 6 modules as suggested by Hagmann et al. (2008) is an interesting research question. Unfortunately, it requires a more elaborate computational analysis ideally with a bigger dataset collected from multiple individuals to answer this question. Based on the results presented so far, however, it seems like USApHMP model in general and the solution algorithm SPATIAL in particular, could serve as a useful tool in such analyses. To this end, we finally provide the objective values and the CPU times of SPATIAL($r = 66$) for several BCN instances with different p and α values.

These results may not serve to provide any important clinical insight, nevertheless, they serve as a benchmark for future studies which aim to tackle very large instances such as BCN. Similar to the observation from Table 3.4, the performance of the SPATIAL algorithm deteriorates with increasing α in terms of CPU time. Another observation is that increasing the number of hubs from p to $p + 1$ improves the objective value significantly for the small values of p, however past a certain threshold - which seems to be around

	$\alpha = 0.1$		$\alpha = 0.5$		$\alpha = 0.9$	
p	z_S	CPU (s)	z_S	CPU (s)	z_S	CPU (s)
2	38220.1	186.3	38658.2	486.2	39303.8	384.3
3	33002.6	261.0	33933.6	183.4	34767.9	196.0
4	28554.5	325.2	29556.1	345.7	30460.6	275.6
5	25822.7	236.8	27143.7	177.9	28061.6	278.2
6	24426.2	232.7	25724.9	267.8	26804.2	209.3
7	23326.3	389.5	24568.1	294.0	25645.0	433.0
8	22561.1	413.0	23760.8	658.4	24664.8	1769.9
9	21443.7	497.4	22728.3	1271.9	24154.4	4198.9
10	20386.4	519.9	22769.8	2608.7	24088.3	3933.4

Table 4.1: Objective values and CPU times of SPATIAL algorithm for different BCN instances

$p_t = 8$ -, the improvement is quite small. Potentially, this threshold point may be used to figure out the optimum number of modules the human brain should be segmented to. The justification for such an approach would be that there is no "economical" advantage of further segmentation beyond p_t where the objective value seem to converge to a certain value. Obviously, to make such an assertion, one needs to first establish the validity of the HLP model to effectively represent the underlying economical model of the human brain. The validity of USApHMP model to represent the underlying economic nature of the brain hub organisation needs to be tested with larger datasets and compared against the models already proposed in the literature (Bullmore and Sporns, 2012; Avena-Koenigsberger et al., 2018) which is beyond the scope of this study.

4.4 Conclusions

We opened up the application of hub location in the BCN with the largest and first three-dimensional real dataset in the hub location literature. We hope this study is going to motivate further research in developing optimization-based models that can help explain the underlying economical/evolutionary advantage of hub organisation in the human brain

networks, which currently seems to be a missing line of research in neuroscience literature.

Simple mathematical models that can approximate the physical dynamics in the human brain with a reasonable accuracy could become a very valuable tool for neuroscience in uncovering the complicated nature of the human brain. From the neuroscience perspective, these generative models would be superior to descriptive network measures that are currently in use (such as centrality, coreness, etc. mentioned in Section 4.2) as they not only provide an answer to the question of "what is it?", but also an (alternative) explanation for the question "how is it?" On the other hand, from the OR perspective, a possible application of mathematical models (such as hub location) in neuroscience is very valuable since there are already some established knowledge about the nature of the human brain and this knowledge should be accurately predicted by any model employed, which offers a great opportunity for the purposes of model validation.

Chapter 5

Conclusions and Future Work

It is unavoidable that with the exponentially increasing amount of data being collected from all different sources, mathematical models very elegant but without the ability to solve practically meaningful size problems will eventually become obsolete. In this thesis work, we showcased how data-driven methods can be adopted to alleviate one of the major challenges in large scale optimization - the computational efficiency (scalability) - on three different levels. We hope the ideas proposed and conceptually proven to work efficiently in this thesis will pave the way to the proliferation of similar studies in the literature.

5.1 Structure Detection

In this section we provide a discussion based on the insights we gained from the results in Section 2.4 and other computational analyses we did not explicitly include. We present several important observations in the structure detection and Lagrangian relaxation contexts and discuss possible extensions of the proposed approach as well as potential future research directions.

A rigorous investigation of Figure 2.4 reveals that, for the 16 instances where detected BBD structures consisted of well-balanced blocks, there is more to the balance of blocks than just size; i.e., the blocks have nearly (if not completely) identical structures. In terms

of graph representation, the subgraphs corresponding to each block are (nearly) isomorphic. The implications of this observation are far-reaching. First, this implies the possibility of using the solution of one subproblem to obtain solutions to other subproblems, hence significantly reducing the computational costs. Second, the idea of detecting isomorphic subgraphs can be incorporated into the structure detection algorithms to enhance the quality of structures.

The fact that the *KEE* algorithm detects such isomorphic subgraphs by coincidence (i.e., without explicitly looking for them) already shows great promise for the potential of such approaches. Cordella et al. (2001) provide an efficient algorithm for matching large graphs; i.e., determining whether two input graphs are isomorphic. It is, therefore, possible to measure the degree of isomorphism in a detected structure. A very straightforward way would be to do a pairwise comparison on the isomorphism of the subgraphs corresponding to diagonal blocks and calculate the ratio of the number of pairwise isomorphic subgraphs to the total number of pairs. However, this would be useful only at the stage of "selecting among alternative structures detected", rather than looking for isomorphism at the detection stage. The latter requires solving an extended problem, known as graph subisomorphism, where the input graphs are of different sizes, and the goal is to check whether any subgraph of the larger graph is isomorphic to the smaller graph, and if so, to find the mapping between the vertices of the two graphs.

We have experimented the algorithm proposed by Cordella et al. (2001) and observed that it is capable of detecting isomorphism within under a second for moderately large graphs. However, when determining the subisomorphism, it becomes inefficient as the difference between the sizes of two graphs increases, which hinders its direct usage in a computationally efficient BBD structure detection algorithm. Finally, despite the implication of weakness, the expression "nearly identical" also points to another potential research avenue: identification and reincorporation of "missing information" in the problem description.

Our test results show that the KEE algorithm performs very effectively in detecting the best inherent BBD structures. However, it is evident from several examples among the test instances that the best BBD structure is not always the best structure in terms of the Lagrangian bound (see, for example *aflow30a* and *aflow40b*). This observation has

two implications: (1) while relaxing constraints/rows, one has to consider putting more weight on the rows with more relevant information. Bergner et al. (2015) introduce a few potential measures to this end, but only on the variables/columns. Identification of information-rich rows from the composition of constraint matrix may prove worthy of further investigation. (2) Lagrangian relaxation (root node solution of DWR) may not be the most efficient way of exploiting the advantages of a good BBD structure. A complete DWRDWR implementation (with intelligent branch-and-price policies) using the BBD structures detected may potentially be a more efficient way of solving large-scale MIP instances than commercial solvers. As mentioned in Section 2.2, several frameworks with generic branch-and-price implementations are already available in the literature.

An interesting observation revealed itself with an earlier version of our proposed algorithm. The naive version of our algorithm was not able to detect a nice structure for the *pp08a* instance which we knew, by visual inspection, was inherent in the constraint matrix. However, it was able to detect such a nice structure for the *pp08aCUTS* instance which appears to be the same problem, only with some additional cuts (probably by a commercial solver) in the constraint matrix. A similar case was observed with the *gesa2-o* and *gesa2* instances. This observation implies that it is possible to overlook a good inherent structure due to the sparse nature of a matrix and that the addition of certain cuts, thereby altering the original composition of the matrix, may prove useful in detecting superior BBD structures. This implication could lead to a promising variation of our algorithm since commercial solvers usually obtain tight bounds at the root node by adding valid cuts to the original problem.

To explore the potential of this approach, we modified the constraint matrices of the test instances by adding the cuts generated by CPLEX at the root node to the original matrices and executed our algorithm on the modified constraint matrices. Unfortunately, our limited analysis on a few test instances was to no avail. For some instances, the detected structures were inferior to those detected on the original matrices which meant little, if any, improvement in the bound would be achieved by Lagrangian relaxation. For the instances where good structures were detected on the modified matrices, the improvement of the Lagrangian bound which can be attributed to detected structure, i.e., not to the added cuts, was also negligible. The former can be attributed to the aggressive addition of cuts

by Cplex at the root node which inevitably corrupts the inherent BBD structure by adding more noise to the constraint matrix. The latter, on the other hand, seems to be another example of the observation we discussed earlier, i.e., a good structure does not always lead to a good bound. One possible way of exploring the potential of this approach would be to control the cutting plane heuristics of commercial solvers and check whether there exist certain types of cuts which provide strong bounds without distorting the inherent block diagonal structures much.

5.2 Spatial separability in hub location problems

We uncovered and investigated an important feature of optimal hub-and-spoke networks that we used to devise a solution methodology to solve large HLPs. The effectiveness of the proposed methodology was tested on well-known benchmark instances from the literture as well as a new dataset originating from brain connectivity networks which is introduced as a new application of the HLPs. Computational experiments showed that the proposed algorithm - Spatial - is very effective both in terms of computation time and optimality gap and is capable of handling very large instances (with up to 1000 nodes) that has never been tackled in the literature so far, to the best of our knowledge.

There is a plethora of future research directions in this line of research. For example, spatial separability seems to prevail in the vast majority of the hub-and-spoke networks. This raises the question whether the few cases which violate this feature can be singled out by some mild assumptions so that the objective function of the USApHMP satisfies the necessary conditions of Barnes et al. (1992). The two-dimensional space in which we have sought spatial separability consists only of the information on the location of the nodes and completely overlooks the flow information. Is it possible to introduce some extra dimensions which encapsulate the flow information such that incorporation of these dimensions eliminates the few non-separable cases and restores the spatial separability (in higher dimensions) as a property of the optimal solutions of USApHMP? If so, what would be the physical meaning of these dimensions and what is the best way to express these extra dimensions mathematically? The simplest idea that comes to mind is to calculate

for each node i the fraction of total flow that is between the nodes falling to the left (bottom) of node i and the nodes falling to the right (top) of node i. When we re-checked the spatial separability with these additional dimensions, all the non-separable instances we had among our random instances hold the separability property. This indicates that there could be a way to impose spatial separability as a rule (rather than just a consistent pattern) by introducing proper flow-based dimensions.

5.3 Hub organisations in human brain connectivity networks

A new application area (BCN) for hub location was introduced with the largest and first three-dimensional real dataset in the hub location literature. Computational experiments showed that the results established in the medical literature can be reproduced to a certain degree using the proposed approach. This supports the claim that HLP models in the OR literature could potentially become very valuable tools in further investigation of the complex hub organisation of the human brain.

The similarity of the results obtained by 1-hub LRP and those of Hagmann et al. (2008) signals that there may be a need for a mathematical model of a related problem. The problem could be phrased as follows: "Given p the number of distinct hub regions and a limit r on the total number of ROIs which can be hubs, what is the best way to distribute r ROIs among p spatially contiguous regions". Such a model would combine the strengths of both approaches, i.e., the tendency of USApHMP to distribute hubs over the whole map and the ability of network measures to result in similar hubness for spatially adjacent nodes. Similar studies where imposing connectivity constraints on existing models can be found in literature (Carvajal et al., 2013; King et al., 2012).

This study aimed just to provide an introduction to the brain connectivity networks as a new application of hub location problems. Although some of the results obtained in this study coincided with the findings in the literature, it also became evident that a more specialized model of the hub location problems may be required to best represent the dynamics of the brain networks. For example, the sparsity in the actual fiber connections

should be taken into account. Moreoever, the fiber densities could be considered as capacities of the arcs between ROIs and the actual flows could be represented by the functional connectivity datasets which were not included in this study. Relating the architecture of functional and structural connectivity networks is one of the most challenging topics within human connectome project for which such a specialized model could provide a fresh perspective.

References

Achterberg, T., Koch, T., and Martin, A. (2006). Miplib 2003. *Operations Research Letters*, 34(4):361–372.

Alumur, S., Kara, B., and Karasan, O. (2009). The design of incomplete single allocation hub networks. *Transp Res B Methodol*, 43(10):936–951.

Alumur, S. and Kara, B. Y. (2008). Network hub location problems: The state of the art. *European Journal of Operational Research*, 190(1):1–21.

Alumur, S. A., Nickel, S., and Saldanha-da Gama, F. (2012). Hub location under uncertainty. *Transportation Research Part B: Methodological*, 46(4):529–543.

Applegate, D. L. (2006). *The traveling salesman problem: a computational study*. Princeton University Press, Princeton, NJ.

Avena-Koenigsberger, A., Misic, B., and Sporns, O. (2018). Communication dynamics in complex brain networks. *Nature Reviews Neuroscience*, 19(1):17.

Aykanat, C., Pinar, A., and Çatalyürek, Ü. V. (2004). Permuting sparse rectangular matrices into block-diagonal form. *SIAM Journal on Scientific Computing*, 25(6):1860–1879.

Aykin, T. (1994). Lagrangian relaxation based approaches to capacitated hub-and-spoke network design problem. *European Journal of Operational Research*, 79(3):501–523.

Azevedo, F. A., Carvalho, L. R., Grinberg, L. T., Farfel, J. M., Ferretti, R. E., Leite, R. E., Lent, R., Herculano-Houzel, S., et al. (2009). Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. *Journal of Comparative Neurology*, 513(5):532–541.

Barnes, E. R., Hoffman, A. J., and Rothblum, U. G. (1992). Optimal partitions having disjoint convex and conic hulls. *Mathematical Programming*, 54(1-3):69–86.

Bassett, D. S. and Sporns, O. (2017). Network neuroscience. *Nature Neuroscience*, 20(3):353.

Behrmann, M. and Plaut, D. C. (2013). Distributed circuits, not circumscribed centers, mediate visual recognition. *Trends in Cognitive Sciences*, 17(5):210–219.

Bergner, M., Caprara, A., Ceselli, A., Furini, F., Lübbecke, M. E., Malaguti, E., and Traversi, E. (2015). Automatic Dantzig-Wolfe reformulation of mixed integer programs. *Mathematical Programming*, 149(1-2):391–424.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

Boland, N., Krishnamoorthy, M., Ernst, A. T., and Ebery, J. (2004). Preprocessing and cutting for multiple allocation hub location problems. *European Journal of Operational Research*, 155(3):638–653.

Borndörfer, R., Ferreira, C. E., and Martin, A. (1998). Decomposing matrices into blocks. *SIAM Journal on Optimization*, 9(1):236–269.

Brandes, U., Delling, D., Gaertler, M., Gorke, R., Hoefer, M., Nikoloski, Z., and Wagner, D. (2008). On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2):172–188.

Brodmann, K. (1908). Beiträge zur histologischen lokalisation der grosshirnrinde. *J.Psychol. Neurol.*, 10:231–246.

Bryan, D. (1998). Extensions to the hub location problem: Formulations and numerical examples. *Geographical Analysis*, 30(4):315–330.

Bui, T. N. and Jones, C. (1992). Finding good approximate vertex and edge partitions is NP-hard. *Information Processing Letters*, 42(3):153–159.

Bullmore, E. and Sporns, O. (2012). The economy of brain network organization. *Nature Reviews Neuroscience*, 13(5):336–349.

Busygin, S., Prokopyev, O., and Pardalos, P. M. (2008). Biclustering in data mining. *Computers & Operations Research*, 35(9):2964–2987.

Calık, H., Alumur, S. A., Kara, B. Y., and Karasan, O. E. (2009). A tabu-search based heuristic for the hub covering problem over incomplete hub networks. *Computers & Operations Research*, 36(12):3088–3096.

Campbell, A. M., Lowe, T. J., and Zhang, L. (2007). The p-hub center allocation problem. *European Journal of Operational Research*, 176(2):819–835.

Campbell, J. F. (1990). Freight consolidation and routing with transportation economies of scale. *Transportation Research Part B: Methodological*, 24(5):345–361.

Campbell, J. F. (1994a). Integer programming formulations of discrete hub location problems. *European Journal of Operational Research*, 72(2):387–405.

Campbell, J. F. (1994b). Integer programming formulations of discrete hub location problems. *European Journal of Operational Research*, 72(2):387–405.

Campbell, J. F., Ernst, A. T., and Krishnamoorthy, M. (2005a). Hub arc location problems: part iiformulations and optimal algorithms. *Management Science*, 51(10):1556–1571.

Campbell, J. F., Ernst, A. T., and Krishnamoorthy, M. (2005b). Hub arc location problems: part iintroduction and results. *Management Science*, 51(10):1540–1555.

Campbell, J. F. and O'Kelly, M. E. (2012). Twenty-five years of hub location research. *Transportation Science*, 46(2):153–169.

Carvajal, R., Constantino, M., Goycoolea, M., Vielma, J. P., and Weintraub, A. (2013). Imposing connectivity constraints in forest planning models. *Operations Research*, 61(4):824–836.

Catalyurek, U. V. and Aykanat, C. (1999). Hypergraph-partitioning-based decomposition for parallel sparse-matrix vector multiplication. *IEEE Transactions on Parallel and Distributed Systems*, 10(7):673–693.

Chakravarty, A. K., Orlin, J. B., and Rothblum, U. G. (1985). Consecutive optimizers for a partitioning problem with applications to optimal inventory groupings for joint replenishment. *Operations Research*, 33(4):820–834.

Clauset, A., Newman, M. E., and Moore, C. (2004). Finding community structure in very large networks. *Physical Review E*, 70(6):066111.

Contreras, I. (2015). Hub location problems. In *Location Science*, pages 311–344. Springer.

Contreras, I., Cordeau, J.-F., and Laporte, G. (2011a). Benders decomposition for large-scale uncapacitated hub location. *Operations Research*, 59(6):1477–1490.

Contreras, I., Cordeau, J.-F., and Laporte, G. (2011b). Stochastic uncapacitated hub location. *European Journal of Operational Research*, 212(3):518–528.

Contreras, I., Díaz, J. A., and Fernández, E. (2011c). Branch and price for large-scale capacitated hub location problems with single assignment. *INFORMS Journal on Computing*, 23(1):41–55.

Contreras, I., Tanash, M., and Vidyarthi, N. (2013). *The Cycle Hub Location Problem*. CIRRELT, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation.

Cordella, L. P., Foggia, P., Sansone, C., and Vento, M. (2001). An improved algorithm for matching large graphs. In *3rd IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition*, pages 149–159.

Correia, I., Nickel, S., and Saldanha-da Gama, F. (2010). The capacitated single-allocation hub location problem revisited: A note on a classical formulation. *European Journal of Operational Research*, 207(1):92–96.

Crossley, N. A., Mechelli, A., Vértes, P. E., Winton-Brown, T. T., Patel, A. X., Ginestet, C. E., McGuire, P., and Bullmore, E. T. (2013). Cognitive relevance of the community structure of the human brain functional coactivation network. *Proceedings of the National Academy of Sciences*, 110(28):11583–11588.

de Camargo, R. S., de Miranda Jr, G., and Luna, H. P. L. (2009a). Benders decomposition for hub location problems with economies of scale. *Transportation Science*, 43(1):86–97.

de Camargo, R. S., Miranda Jr, G., Ferreira, R. P. M., and Luna, H. (2009b). Multiple allocation hub-and-spoke network design under hub congestion. *Computers & Operations Research*, 36(12):3097–3106.

Dhaenens, C. and Jourdan, L. (2016). Optimization and big data. In *Metaheuristics for Big Data*, pages 1–21.

Ebery, J. (2001). Solving large single allocation p-hub problems with two or three hubs. *European Journal of Operational Research*, 128(2):447–458.

Ebery, J., Krishnamoorthy, M., Ernst, A., and Boland, N. (2000). The capacitated multiple allocation hub location problem: Formulations and algorithms. *European Journal of Operational Research*, 120(3):614–631.

Elhedhli, S. and Hu, F. X. (2005). Hub-and-spoke network design with congestion. *Computers & Operations Research*, 32(6):1615–1632.

Elhedhli, S. and Wu, H. (2010). A lagrangean heuristic for hub-and-spoke system design with capacity selection and congestion. *INFORMS Journal on Computing*, 22(2):282–296.

Ernst, A. T. and Krishnamoorthy, M. (1996). Efficient algorithms for the uncapacitated single allocation p-hub median problem. *Location Science*, 4(3):139–154.

Farahani, R. Z., Hekmatfar, M., Arabani, A. B., and Nikbakhsh, E. (2013). Hub location problems: A review of models, classification, solution techniques, and applications. *Computers & Industrial Engineering*, 64(4):1096–1109.

Ferris, M. C. and Horn, J. D. (1998). Partitioning mathematical programs for parallel solution. *Mathematical Programming*, 80(1):35–61.

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 486(3):75–174.

Fox, M. D., Corbetta, M., Snyder, A. Z., Vincent, J. L., and Raichle, M. E. (2006). Spontaneous neuronal activity distinguishes human dorsal and ventral attention systems. *Proceedings of the National Academy of Sciences*, 103(26):10046–10051.

Friederici, A. D. and Gierhan, S. M. (2013). The language network. *Current Opinion in Neurobiology*, 23(2):250–254.

Gamrath, G. and Lübbecke, M. E. (2010). Experiments with a generic Dantzig-Wolfe decomposition for integer programs. In *Experimental Algorithms*, pages 239–252. Springer.

Girvan, M. and Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826.

Glasser, M. F., Coalson, T. S., Robinson, E. C., Hacker, C. D., Harwell, J., Yacoub, E., Ugurbil, K., Andersson, J., Beckmann, C. F., Jenkinson, M., et al. (2016). A multi-modal parcellation of human cerebral cortex. *Nature*, 536(7615):171–178.

Good, B. H., de Montjoye, Y.-A., and Clauset, A. (2010). Performance of modularity maximization in practical contexts. *Physical Review E*, 81(4):046106.

Hagmann, P., Cammoun, L., Gigandet, X., Meuli, R., Honey, C. J., Wedeen, V. J., and Sporns, O. (2008). Mapping the structural core of human cerebral cortex. *PLoS Biology*, 6(7):e159.

Hagmann, P., Thiran, J.-P., Jonasson, L., Vandergheynst, P., Clarke, S., Maeder, P., and Meuli, R. (2003). Dti mapping of human brain connectivity: statistical fibre tracking and virtual dissection. *Neuroimage*, 19(3):545–554.

Hamacher, H. W., Labbé, M., Nickel, S., and Sonneborn, T. (2004). Adapting polyhedral properties from facility to hub location problems. *Discrete Applied Mathematics*, 145(1):104–116.

Hamacher, H. W. and Meyer, T. (2006). *Hub cover and hub center problems*. University of Kaiserslautern, Germany.

Honey, C. J., Kötter, R., Breakspear, M., and Sporns, O. (2007). Network structure of cerebral cortex shapes functional connectivity on multiple time scales. *Proceedings of the National Academy of Sciences*, 104(24):10240–10245.

Hopcroft, J. and Tarjan, R. (1973). Algorithm 447: efficient algorithms for graph manipulation. *Communications of the ACM*, 16(6):372–378.

Kara, B. and Tansel, B. (2003). The single-assignment hub covering problem: Models and linearizations. *Journal of the Operational Research Society*, 54(1):59–64.

Kara, B. Y. and Tansel, B. C. (2000). On the single-assignment p-hub center problem. *European Journal of Operational Research*, 125(3):648–655.

Karypis, G. and Kumar, V. (1998a). hMETIS 1.5: A hypergraph partitioning package.

Karypis, G. and Kumar, V. (1998b). A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices. *University of Minnesota, Department of Computer Science and Engineering, Army HPC Research Center, Minneapolis, MN*.

Kaufman, L. and Rousseeuw, P. J. (2009). *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons.

Kernighan, B. W. and Lin, S. (1970). An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal*, 49(2):291–307.

King, D. M., Jacobson, S. H., Sewell, E. C., and Cho, W. K. T. (2012). Geo-graphs: an efficient model for enforcing contiguity and hole constraints in planar graph partitioning. *Operations Research*, 60(5):1213–1228.

Klincewicz, J. G. (1992). Avoiding local optima in thep-hub location problem using tabu search and grasp. *Annals of Operations Research*, 40(1):283–302.

Klincewicz, J. G. (1998). Hub location in backbone/tributary network design: a review. *Location Science*, 6(1-4):307–335.

Koch, T., Achterberg, T., Andersen, E., Bastert, O., Berthold, T., Bixby, R. E., Danna, E., Gamrath, G., Gleixner, A. M., Heinz, S., et al. (2011). Miplib 2010. *Mathematical Programming Computation*, 3(2):103–163.

Labbé, M., Yaman, H., and Gourdin, E. (2005). A branch and cut algorithm for hub location problems with single assignment. *Mathematical Programming*, 102(2):371–405.

Lancichinetti, A. and Fortunato, S. (2009). Community detection algorithms: a comparative analysis. *Physical Review E*, 80(5):056117.

Lancichinetti, A., Fortunato, S., and Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(4):046110.

Le Bihan, D., Mangin, J.-F., Poupon, C., Clark, C. A., Pappata, S., Molko, N., and Chabriat, H. (2001). Diffusion tensor imaging: concepts and applications. *Journal of Magnetic Resonance Imaging*, 13(4):534–546.

Leskovec, J., Lang, K. J., and Mahoney, M. (2010). Empirical comparison of algorithms for network community detection. In *Proceedings of the 19th International Conference on World Wide Web*, pages 631–640. ACM.

Lyaudet, L. (2010). NP-hard and linear variants of hypergraph partitioning. *Theoretical Computer Science*, 411(1):10–21.

Marianov, V. and Serra, D. (2003). Location models for airline hubs behaving as m/d/c queues. *Computers & Operations Research*, 30(7):983–1003.

Marianov, V., Serra, D., and ReVelle, C. (1999). Location of hubs in a competitive environment. *European Journal of Operational Research*, 114(2):363–371.

Marín, A. (2005). Uncapacitated euclidean hub location: Strengthened formulation, new facets and a relax-and-cut algorithm. *Journal of Global Optimization*, 33(3):393–422.

Martín, J. C. and Román, C. (2004). Analyzing competition for hub location in intercontinental aviation markets. *Transportation Research Part E: Logistics and Transportation Review*, 40(2):135–150.

Mayer, G. and Wagner, B. (2002). Hublocator: an exact solution method for the multiple allocation hub location problem. *Computers & Operations Research*, 29(6):715–739.

Naddef, D. and Thienel, S. (2002). Efficient separation routines for the symmetric traveling salesman problem i: general tools and comb separation. *Mathematical Programming*, 92(2):237–255.

Newman, M. E. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113.

Nicosia, G., Pardalos, P., Giuffrida, G., and Umeton, R. (2017). *Machine Learning, Optimization, and Big Data: Third International Conference, MOD 2017, Volterra, Italy, September 14–17, 2017, Revised Selected Papers*, volume 10710. Springer.

O'Kelly, M. E. (1986). The location of interacting hub facilities. *Transportation Science*, 20(2):92–106.

O'Kelly, M. E. (1987a). A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research*, 32(3):393–404.

O'Kelly, M. E. (1987b). A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research*, 32(3):393–404.

O'Kelly, M. E. (1992a). A clustering approach to the planar hub location problem. *Annals of Operations Research*, 40(1):339–353.

O'Kelly, M. E. (1992b). Hub facility location with fixed costs. *Papers in Regional Science*, 71(3):293–306.

Peker, M., Kara, B. Y., Campbell, J. F., and Alumur, S. A. (2016). Spatial analysis of single allocation hub location problems. *Networks and Spatial Economics*, 16(4):1075–1101.

Pessoa, L. (2012). Beyond brain regions: Network perspective of cognition–emotion interactions. *Behavioral and Brain Sciences*, 35(3):158–159.

Pirkul, H. and Schilling, D. A. (1998). An efficient procedure for designing single allocation hub and spoke systems. *Management Science*, 44(12-part-2):S235–S242.

Pons, P. and Latapy, M. (2006). Computing communities in large networks using random walks. *J. Graph Algorithms Appl.*, 10(2):191–218.

Porter, M. A., Onnela, J.-P., and Mucha, P. J. (2009). Communities in networks. *Notices of the AMS*, 56(9):1082–1097.

Puchinger, J., Stuckey, P. J., Wallace, M. G., and Brand, S. (2011). Dantzig-Wolfe decomposition and branch-and-price solving in G12. *Constraints*, 16(1):77–99.

Ralphs, T. and Galati, M. (2009). DIP–decomposition for integer programming.

Ronhovde, P. and Nussinov, Z. (2009). Multiresolution community detection for megascale networks by information-based replica correlations. *Physical Review E*, 80(1):016109.

Rosvall, M. and Bergstrom, C. T. (2007). An information-theoretic framework for resolving community structure in complex networks. *Proceedings of the National Academy of Sciences*, 104(18):7327–7331.

Sasaki, M. and Fukushima, M. (2001). Stackelberg hub location problem. *Journal of the Operations Research Society of Japan*, 44(4):390–402.

Skorin-Kapov, D., Skorin-Kapov, J., and O'Kelly, M. (1996). Tight linear programming relaxations of uncapacitated p-hub median problems. *European Journal of Operational Research*, 94(3):582–593.

Sung, C. and Jin, H. (2001). Dual-based approach for a hub network design problem under non-restrictive policy. *European Journal of Operational Research*, 132(1):88–105.

Suresh Kumar, C. and Chandrasekharan, M. (1990). Grouping efficacy: a quantitative criterion for goodness of block diagonal forms of binary matrices in group technology. *International Journal of Production Research*, 28(2):233–243.

Topcuoglu, H., Corut, F., Ermis, M., and Yilmaz, G. (2005). Solving the uncapacitated hub location problem using genetic algorithms. *Computers & Operations Research*, 32(4):967–984.

van den Heuvel, M. P., Mandl, R. C., Stam, C. J., Kahn, R. S., and Pol, H. E. H. (2010). Aberrant frontal and temporal complex network structure in schizophrenia: a graph theoretical analysis. *Journal of Neuroscience*, 30(47):15915–15926.

van den Heuvel, M. P. and Sporns, O. (2013). Network hubs in the human brain. *Trends in Cognitive Sciences*, 17(12):683–696.

Vanderbeck, F. (2005). BaPCod-a generic branch-and-price code. *http://wiki.bordeaux.inria.fr/realopt*.

Vanderbeck, F. and Wolsey, L. A. (2010). Reformulation and decomposition of integer programs. In *50 Years of Integer Programming 1958-2008*, pages 431–502. Springer.

Vincent, J. L., Patel, G. H., Fox, M. D., Snyder, A. Z., Baker, J. T., Van Essen, D. C., Zempel, J. M., Snyder, L. H., Corbetta, M., and Raichle, M. E. (2007). Intrinsic functional architecture in the anaesthetized monkey brain. *Nature*, 447(7140):83.

Wagner, B. (2007). An exact solution procedure for a cluster hub location problem. *European Journal of Operational Research*, 178(2):391–401.

Wagner, B. (2008). Model formulations for hub covering problems. *Journal of the Operational Research Society*, 59(7):932–938.

Wang, J. and Ralphs, T. K. (2013). Computational experience with hypergraph-based methods for automatic decomposition in discrete optimization. In *CPAIOR*, pages 394–402. Springer.

Weil, R. L. and Kettler, P. C. (1971). Rearranging matrices to block-angular form for decomposition (and other) algorithms. *Management Science*, 18(1):98–108.

Witt, J. and Lübbecke, M. (2015). Dantzig-Wolfe reformulations for the stable set problem. Technical Report 2015–029, Operations Research, RWTH Aachen University.

Yaman, H. and Elloumi, S. (2012). Star p-hub center problem and star p-hub median problem with bounded path lengths. *Computers & Operations Research*, 39(11):2725–2732.

Zalesky, A., Fornito, A., Seal, M. L., Cocchi, L., Westin, C.-F., Bullmore, E. T., Egan, G. F., and Pantelis, C. (2011). Disrupted axonal fiber connectivity in schizophrenia. *Biological Psychiatry*, 69(1):80–89.