Chapter 2
The p-Median Problem

2.1 Introduction

This chapter focuses on the p-median problem (PMP) that will be used in the next chapter in order to construct an efficient approach for CFP. Those not interested in technical details may safely skip most of this chapter, except of the derivation of the pseudo-Boolean formulation that is crucial for understanding the next chapters.

The PMP is a well-known NP-hard problem which was originally defined by Hakimi [75, 76] and involves location of p facilities on a network in such a manner that the total weighted distance of serving all demands is minimized. It has been widely studied in literature and applied in cluster analysis, quantitative psychology, marketing, telecommunications industry [27], sales force territories design [111], political districting [17], optimal diversity management (ODM) [26], cell formation in group technology [160], vehicle routing [85], and topological design of computer communication networks [125].

The basic PMP model that has remained almost unchanged during recent 30 years is the so-called ReVelle and Swain integer linear programming formulation [44, 130]. Note that this formulation contains Boolean decision variables and, hence, this is a Boolean linear programming formulation. Since then, the PMP has been the subject of considerable research involving the development of some different types of adjusted model formats [43, 45, 47, 133], and recently [4, 44, 55], as well as the development of advanced solution approaches ([128] and references within) and some recent publications [11, 18, 27, 139]. For a comprehensive list of references to the PMP we address the reader to [108, 128] and a bibliographical overview from [131].

A *Boolean linear programming formulation of the PMP* can be defined on a weighted bipartite graph $G = (V,A,C)$ with the set of vertices $V = I \cup J$, the set of arcs $A \subseteq I \times J$, and a *cost matrix* of nonnegative weights $C = \{c_{ij} : c_{ij} \geq 0, (i,j) \in A\}$ as follows. For the given sets $I = \{1,2,\ldots,m\}$ of sites at which plants (cluster centers) can be located, $J = \{1,2,\ldots,n\}$ of clients (cluster points) with unit demand at each client site, a matrix $C = [c_{ij}]$ of nonnegative costs (distances, or some other

B. Goldengorin et al., *Cell Formation in Industrial Engineering*, Springer Optimization and Its Applications 79, DOI 10.1007/978-1-4614-8002-0_2,

dissimilarity measure) of supplying each $j \in J$ from each $i \in I$, the number p of plants to be opened, the PMP can be written as

$$\min \quad \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}, \tag{2.1}$$

$$\text{s.t.} \quad \sum_{i=1}^{m} x_{ij} = 1, \quad j = 1, \ldots, n, \tag{2.2}$$

$$x_{ij} \leq \overline{y}_i, \quad i = 1, \ldots, m; \quad j = 1, \ldots, n, \tag{2.3}$$

$$\sum_{i=1}^{m} \overline{y}_i = p, \tag{2.4}$$

$$\overline{y}_i \in \{0, 1\}, \quad i = 1, \ldots, m, \tag{2.5}$$

$$x_{ij} \in \{0, 1\}, \quad i = 1, \ldots, m; \quad j = 1, \ldots, n. \tag{2.6}$$

For any feasible solution $(x_{ij}, \overline{y}_i)$, $\overline{y}_i = 1$ if plant i is open, and $\overline{y}_i = 0$, otherwise; $x_{ij} = 1$ if client j is assigned to plant i, and $x_{ij} = 0$, otherwise. Constraints (2.2) assign each client to exactly one plant, and constraints (2.3) forbid the assignment of a client to a closed plant, constraint (2.4) fixes the number of opened plants to p.

A PMP instance is described by an $m \times n$ matrix $C = [c_{ij}]$ and the number $1 \leq p \leq |I|$. We assume that the entries of C are nonnegative and finite, i.e., $C \in \mathbb{R}_{+}^{mn}$. If $I = J$, we have the classic ReVelle and Swain's PMP model [130] with n^2 Boolean decision variables.

Further progress with improvements of ReVelle and Swain's PMP model was made by Rosing et al. [133], Cornuejols et al. [45], Dearing et al. [47], Church [43], Church [44] and recently by AlBdaiwi et al. [4] and Elloumi [55]. All of them have incorporated in different ways the following properties of PMP:

- Based on an ordering of the distances c_{ij} with respect to a given demand point they have either reduced the number of clients or have excluded from (2.1)–(2.6) a repetition of decision variables x_{ij} and x_{kj} corresponding to the equal costs $c_{ij} = c_{kj}$ for some $j \in J$.
- The $mn + m$ Boolean decision variables are replaced by m Boolean decision variables and mn nonnegative decision variables, i.e., (2.6) is replaced by

$$x_{ij} \geq 0, \quad i = 1, \ldots, m; \quad j = 1, \ldots, n. \tag{2.7}$$

To the best of our knowledge there is no PMP model that adjusts the numbers of non-negative decision variables and corresponding linear constraints depending on the number p of medians.

In this chapter we start our study of the classic p-median model represented as a Boolean linear programming model by posing the following questions:

- What are the optimal numbers of decision variables partitioned into Boolean and non-Boolean variables?
- What is the optimal number of constraints?

- Are the above-mentioned numbers of decision variables and constraints dependent on the PMP input data, more specifically on the number p of medians?

This chapter proposes a new model formulation for the PMP that contains all previously suggested improvements which we have incorporated in a concise and simplified notation including our adjustment of decision variables and corresponding linear constraints depending on the number p of medians. This new p-median formulation is called a *mixed Boolean pseudo-Boolean model (MBpBM)* for the PMP. We show that our model can result in a substantially smaller mixed Boolean linear programming formulation for a given application of the PMP and can be used either to find a global optimum by means of general-purpose mixed integer linear programming (MILP) solvers or to develop new exact and approximate algorithms based on the well-known methods in mixed-integer programming (see, e.g., [158]).

Some of the above-mentioned improvements were separately done for the PMP without taking into account the ongoing progress with model formulations for another common model within minisum location-allocation problems, namely the simple plant location problem (SPLP), often referred to as the uncapacitated facility location problem (UFLP) [46] or the warehouse location problem (see, e.g., [131]). The SPLP is similar to the PMP, and the methods used to solve one are often adapted to solve the other. The objective function of the SPLP is one of determining the cheapest method of meeting the demands of a set of clients $J = \{1,\ldots,n\}$ from plants that can be located at some candidate sites $I = \{1,\ldots,m\}$. The costs involved in meeting the client demands include the fixed cost of setting up a plant at a given site, and the per unit transportation cost of supplying a given client from a plant located at a given site. Both PMP and SPLP are defined on bipartite graphs and differ in the following details. First, the SPLP involves a fixed cost for locating a facility at a given vertex while the PMP does not. Second, unlike the PMP, SPLP does not have a constraint on the number of opened facilities. Typical SPLP formulations separate the set of potential facilities (sites location, cluster centers) from the set of demand points (clients). In the PMP these sets are identical, i.e., $I = J$. Such problems are well known in cluster analysis (see, e.g., [27]). Both problems form underlying models in several combinatorial problems, like set covering, set partitioning, information retrieval, simplification of logical Boolean expressions, airline crew scheduling, vehicle dispatching [41], and assortment (see, e.g., [68, 124]), and are subproblems of various location analysis problems [131].

An instance of the SPLP has an optimal solution in which each client is satisfied by exactly one plant. A similar observation is valid for the PMP. Hammer [77] (see also [47]) used this fact to derive a pseudo-Boolean representation of the SPLP. The pseudo-Boolean polynomial (pBp) developed in that work has terms that contain both a literal and its complement. At the end of [77] it is shown by means of an example that only linear monomials can have negative coefficients. Subsequently, Beresnev [20] developed a different pseudo-Boolean formulation in which each term contains only literals or only their complements. We have found this formulation easier to manipulate and hence adjusted Beresnev's formulation of the SPLP to the PMP in [4, 65].

The purpose of this chapter is twofold. First, we design a new model for the PMP and show that the number of nonnegative decision variables and corresponding constraints depend on the number of p-medians and will be adjusted in our model. Moreover, these numbers are minimal within the class of mixed integer linear programs for the PMP. Second, we show that our new model allows solving by means of a general-purpose solver on a PC some PMP benchmark instances previously intractable by both general-purpose solvers and the state-of-the-art exact algorithms, as well as handling smaller instances more efficiently.

In order to demonstrate the properties of the PMP and compare performance of the formulations we used benchmark instances from the four most popular libraries: OR, TSP, ODM, and RW. The first one, the OR library, was introduced by Beasley [16] and is available at [94]. Every node is both a potential location and a client, and the costs are the lengths of the shortest paths between the corresponding nodes.

The TSP library was originally proposed for the traveling salesman problem (TSP) and is available at [95]. TSP instances are defined as sets of points in a two-dimensional plane. Every point is considered both a potential location and a client, and the costs are simply Euclidean distances.

Instances from the next library that we studied are based on the ODM problem. For the description of this problem and instances see [26].

Finally, we considered instances proposed by Resende and Werneck [129]. These problems are defined on random distance matrices. In every case, the number of potential facilities m is equal to the number of clients n and distances are integers taken uniformly at random from the interval $[1,n]$. The library contains five instances with $n = 100, 200, 250, 500, 1000$.

The chapter is organized as follows. Section 2.2 focuses on the pseudo-Boolean formulation of the PMP and its basic properties. In Sect. 2.3 we analyze the size reduction techniques applicable to the PMP. Next, in Sect. 2.4 we present our new MBpBM formulation, discuss its minimality and provide results of numerical experiments. Sections 2.5 and 2.6 provide two applications of the pseudo-Boolean formulation: estimation of instance data complexity and characterization of equivalent instances. Finally, Sect. 2.7 concludes the chapter with a summary and future research directions.

2.2 The Pseudo-Boolean Representation

Recall that given sets $I = \{1,2,\ldots,m\}$ of sites in which plants can be located, $J = \{1,2,\ldots,n\}$ of clients, a matrix $C = [c_{ij}]$ of transportation costs (supplying costs, distances, similarities, etc.) for each $j \in J$ from each $i \in I$, the number p of plants to be opened and a unit demand at each client site, the PMP is one of finding a set $S \subseteq I$ with $|S| = p$, such that the total cost

$$f_C(S) = \sum_{j \in J} \min\{c_{ij} \,|\, i \in S\} \tag{2.8}$$

of satisfying all unit demands is minimized. Note that non-unit demands $d_j \neq 1$ can be scaled by $c'_{ij} = c_{ij}d_j$, and the number of served clients by each plant is unbounded (the so-called *uncapacitated* location problem; see, e.g., [128, 131]). An instance of the problem is described by an $m \times n$ matrix $C = [c_{ij}]$ and the number $1 \leq p \leq |I|$. We assume that entries of C are nonnegative and finite, i.e., $C \in \mathbb{R}_+^{mn}$. The *Combinatorial Formulation of PMP* is to find

$$S^* \in \arg\min\{f_C(S) : \emptyset \subset S \subseteq I, |S| = p\}. \tag{2.9}$$

It is possible to reformulate the objective function $f_C(S)$ of PMP (2.8) in terms of a pBp (see [20, 77]). It is enough to find a pseudo-Boolean representation for each addend $\min\{c_{ij} \,|\, i \in S\}$ and sum up addends for all $j \in J$. In the rest of this section we will use the following notions. Mappings $f : \{0,1\}^n \to \mathbb{R}$ are called pseudo-Boolean functions. All pseudo-Boolean functions can be uniquely represented as *multi-linear polynomials* of the form (see, e.g., [23])

$$f(\mathbf{x}) = \sum_{S \subseteq I} \alpha_S \prod_{i \in S} x_i. \tag{2.10}$$

The expressions $\alpha_S \prod_{i \in S} x_i$ and $\prod_{i \in S} x_i$ are called a *monomial* and a *term*, respectively. In this book multi-linear polynomials are called *pBps* and monomials with the same term are called *similar monomials*. For example, the following pairs of monomials $2x_1x_5$ and $5x_1x_5$; $3x_3x_4x_7$ and $5x_3x_4x_7$ are similar monomials. We say that a pBp is in the *reduced form* if it contains no similar monomials. In other words, the algebraic summation of similar monomials is called *reduction*. Representation of the cost function (2.8) in terms of a pBp needs two additional notions: a *permutation matrix* and a *difference matrix*.

An $m \times n$ *permutation matrix* $\Pi = [\pi_{ij}]$ is a matrix with each column $\Pi_j = (\pi_{1j}, \ldots, \pi_{mj})^{\mathrm{T}}$ defining a permutation of $1, \ldots, m$ that if being applied to the corresponding column of the cost matrix makes its entries sorted in a non-decreasing order. There may exist several permutation matrices for a given instance of the PMP. Given a matrix C, the set of all permutation matrices Π such that $c_{\pi_{1j}j} \leq c_{\pi_{2j}j} \leq \cdots \leq c_{\pi_{mj}j}$ for $j = 1, \ldots, n$ is denoted by $perm(C)$.

Given this notion of the permutation matrix, consider the expression $\min\{c_{ij} \,|\, i \in S\}$ for some fixed $j \in J$. Clearly, the minimum is attained if $S = I$, i.e., the smallest value is chosen among all entries c_{ij} for a fixed column j. It is clear that the unit demand of column j cannot be satisfied cheaper than this smallest value. Assume that this smallest value is attained at an entry $c_{\pi_{1j}j}$ of column j such that π_{1j} indicates the number of the row containing this smallest entry $c_{\pi_{1j}j}$ in column j. In terms of the original PMP, if the site numbered by π_{1j} is open, then the unit demand of client j will be satisfied by costs $c_{\pi_{1j}j}$; otherwise (if the site π_{1j} is closed, but all other sites in $I \setminus \{\pi_{1j}\}$ are opened) the cheapest way to satisfy the unit demand of client j is by the value of a second smallest entry $c_{\pi_{2j}j}$. The value of a second smallest entry $c_{\pi_{2j}j}$ can be represented as follows: $c_{\pi_{2j}j} = c_{\pi_{1j}j} + [c_{\pi_{2j}j} - c_{\pi_{1j}j}]$. Similarly, if both sites π_{1j}, π_{2j} are closed and all other sites are opened, then the unit demand of client j will be satisfied by the value of a third smallest entry

$c_{\pi_3 j} = c_{\pi_1 j} + [c_{\pi_2 j} - c_{\pi_1 j}] + [c_{\pi_3 j} - c_{\pi_2 j}]$, etc. In other words, depending on the set of opened and closed sites from I the corresponding smallest value of $\min\{c_{i,j} \mid i \in S\}$ can be represented by the sum of the smallest values of entries in column j and the corresponding differences of ordered entries in column j. By introducing a Boolean variable $y_{\pi_1 j} = 0$ if the site $\pi_1 j$ is opened and $y_{\pi_1 j} = 1$ if the site $\pi_1 j$ is closed, we are able to express, for example, the costs of satisfying the unit demand j depending on whether the site $\pi_1 j$ is opened or closed (if $\pi_1 j$ is closed, then we assume that $\pi_2 j$ is open, i.e., $y_{\pi_2 j} = 0$), as follows: $c_{\pi_2 j} = c_{\pi_1 j} + [c_{\pi_2 j} - c_{\pi_1 j}] y_{\pi_1 j}$.

To illustrate this idea, let us consider the first column C^1 of matrix C (2.23), namely $C^1 = (c_{11}, c_{21}, c_{31}, c_{41})^{\mathrm{T}} = (7, 10, 16, 11)^{\mathrm{T}}$. After ordering its entries in a non-decreasing order $7 < 10 < 11 < 16$ we have that the corresponding permutation is $\Pi^1 = (1, 2, 4, 3)^{\mathrm{T}}$. If the Boolean vector $(y_1, y_2, y_3, y_4)^{\mathrm{T}}$ reflects an opened (closed) plants at cite $i = 1, 2, 3, 4$, then depending on the set of opened plants $S \subseteq \{1, 2, 3, 4\}$ we have $\min\{c_{i1} \mid i \in S\} = [7 + 3y_1 + 1y_1y_2 + 5y_1y_2y_4]$. For example, if $S = \{2, 4\}$, then $\mathbf{y} = (1, 0, 1, 0)^{\mathrm{T}}$, and $\min\{c_{i1} \mid i \in \{2, 4\}\} = 7 + 3 \times 1 + 1 \times 1 \times 0 + 5 \times 1 \times 0 \times 0 = 10$.

Corresponding to a permutation matrix $\Pi = [\pi_{ij}]$, a *difference matrix* $\Delta = \delta_{ij}$ containing differences between the transportation costs for each $j \in J$ is uniquely defined as follows:

$$\begin{aligned} \delta_{1k} &= c_{\pi_{1k} k}, \\ \delta_{rk} &= c_{\pi_{rk} k} - c_{\pi_{(r-1)k} k}, \quad r = 2, \ldots, m. \end{aligned} \tag{2.11}$$

Defining

$$y_i = \begin{cases} 0 & \text{if } i \in S \\ 1 & \text{otherwise,} \end{cases} \qquad i = 1, \ldots, m \tag{2.12}$$

we can indicate any solution S by a vector $\mathbf{y} = (y_1, y_2, \ldots, y_m)^{\mathrm{T}}$. Its total cost is given by the following pBp:

$$\mathscr{B}_{C,\Pi}(\mathbf{y}) = \sum_{j=1}^{n} \left\{ \delta_{1j} + \sum_{k=2}^{m} \delta_{kj} \prod_{r=1}^{k-1} y_{\pi_{rj}} \right\}. \tag{2.13}$$

Note, this pBp is different from those used by Hammer [77] and Dearing et al. [47] containing both variables and their complements.

We call a pBp $f(\mathbf{y})$ (2.10) a *Hammer–Beresnev polynomial* if there exists a PMP instance C and $\Pi \in perm(C)$ such that $f(\mathbf{y}) = \mathscr{B}_{C,\Pi}(\mathbf{y})$ for each $\mathbf{y} \in \{0, 1\}^m$, since this representation of the total cost was first presented in the context of UFLPs independently in [20, 77]. The following theorem from [5] gives necessary and sufficient conditions for this.

Theorem 2.1. *A general pBp is a Hammer–Beresnev polynomial if and only if all its coefficients are nonnegative.*

Proof. The "if" statement is trivial. In order to prove the "only if" statement, consider a PMP instance defined by the cost matrix C, an ordering matrix $\Pi \in perm(C)$,

and a Hammer–Beresnev polynomial $\mathscr{B}_{C,\Pi}(\mathbf{y})$ in which there is a monomial of degree k with a negative coefficient. Since monomials in $\mathscr{B}_{C,\Pi}(\mathbf{y})$ are contributed by the elements of C only, a monomial with a negative coefficient implies that $\delta_{k,j}$ is negative for some $j \in 1,\ldots,n$. But this contradicts the fact that $\Pi \in perm(C)$. □

In [4] it is shown that the total cost function (2.13) is identical for all permutations in $perm(C)$. Hence, we can remove the Π in $\mathscr{B}_{C,\Pi}(\mathbf{y})$ without introducing any confusion. We denote a Hammer–Beresnev polynomial corresponding to a given PMP instance C by $\mathscr{B}_C(\mathbf{y})$ and define it as

$$\mathscr{B}_C(\mathbf{y}) = \mathscr{B}_{C,\Pi}(\mathbf{y}), \tag{2.14}$$

where $\Pi \in perm(C)$.

A solution $\mathbf{y}$ is feasible if $\sum_{i=1}^{m} y_i = m - p$. Thus, every product of more than $m-p$ variables is 0 for any feasible solution. This observation allows excluding monomials of high degree from the objective function and we call this procedure *truncation of the Hammer–Beresnev polynomial*. The polynomial subjected to truncation and summation of similar monomial is denoted by $\mathscr{B}_{C,p}(\mathbf{y})$ and has the following form:

$$\mathscr{B}_{C,p}(\mathbf{y}) = \sum_{j=1}^{n} \left\{ \delta_{1j} + \sum_{k=2}^{m-p+1} \delta_{kj} \cdot \prod_{r=1}^{k-1} y_{\pi_{rj}} \right\}. \tag{2.15}$$

It should be mentioned that the truncated Hammer–Beresnev polynomial $\mathscr{B}_C(\mathbf{y})$ usually contains less than $(m-p) \times n$ monomials as presence of equal entries in columns of the cost matrix leads to zero differences δ_{kj} and similar monomials can be subjected to algebraic summation (e.g., constants δ_{1j} can be always summed up into one value). Further, we will denote the truncated Hammer–Beresnev polynomial with reduced similar monomials by $\mathscr{B}_{C,p}(\mathbf{y})$ it can be expressed as:

$$\mathscr{B}_{C,p}(\mathbf{y}) = \sum_{r=0}^{k} \alpha_r \prod_{i \in T_r} y_i = \sum_{r=0}^{k} \alpha_r \mathscr{T}_r, \tag{2.16}$$

where T_r is a set of Boolean variables y_i included in term $\mathscr{T}_r$ and k is the number of non-constant monomials in $\mathscr{B}_{C,p}(\mathbf{y})$.

We can reformulate (2.9) in terms of Hammer–Beresnev polynomials as the *pseudo-Boolean formulation of PMP*:

$$\mathbf{y}^* \in \arg\min\{\mathscr{B}_{C,p}(\mathbf{y}) : \mathbf{y} \in \{0,1\}^m, \sum_{i=1}^{m} y_i = m - p\}. \tag{2.17}$$

Example 2.1. Consider a PMP instance from [55] with $m = 4$, $n = 5$, $p = 2$ and

$$C = \begin{bmatrix} 1 & 6 & 5 & 3 & 4 \\ 2 & 1 & 2 & 3 & 5 \\ 1 & 2 & 3 & 3 & 3 \\ 4 & 3 & 1 & 8 & 2 \end{bmatrix}. \tag{2.18}$$

A possible ordering matrix for this problem is given by

$$\Pi = \begin{bmatrix} 1 & 2 & 4 & 1 & 4 \\ 3 & 3 & 2 & 2 & 3 \\ 2 & 4 & 3 & 3 & 1 \\ 4 & 1 & 1 & 4 & 2 \end{bmatrix}, \tag{2.19}$$

and the difference matrix is

$$\Delta = \begin{bmatrix} 1 & 1 & 1 & 3 & 2 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 2 & 3 & 2 & 5 & 1 \end{bmatrix}, \tag{2.20}$$

The Hammer–Beresnev polynomial representing the total cost function for this instance in the form (2.13) is

$$\begin{aligned} \mathscr{B}_C(\mathbf{y}) = {} & [1 + 0y_1 + 1y_1y_3 + 2y_1y_2y_3] + \\ & [1 + 1y_2 + 1y_2y_3 + 3y_2y_3y_4] + \\ & [1 + 1y_4 + 1y_2y_4 + 2y_2y_3y_4] + \\ & [3 + 0y_1 + 0y_1y_2 + 5y_1y_2y_3] + \\ & [2 + 1y_4 + 1y_3y_4 + 1y_1y_3y_4]. \end{aligned} \tag{2.21}$$

Taking into account that $p = 2$, after truncation and reduction of similar monomials in (2.21) we obtain the following pseudo-Boolean representation of the instance:

$$\begin{aligned} \mathscr{B}_{C,p=2}(\mathbf{y}) = {} & 8 + 1y_2 + 2y_4 + 1y_1y_3 + 1y_2y_3 + 1y_2y_4 + 1y_3y_4 \to \min \\ & s.t. \\ & y_1 + y_2 + y_3 + y_4 = m - p = 2, \\ & \mathbf{y} \in \{0,1\}^m. \end{aligned} \tag{2.22}$$

It is easy to see that the objective function in (2.22) contains only 7 nonzero coefficients while the initial cost matrix (2.18) has 20 entries. This implies that the pseudo-Boolean representation allows reduction of the memory needed to store the PMP instance data. Of course, not only coefficients but also terms must be stored; however, these overheads in most cases will be overwhelmed by the substantial reduction of the polynomial.

2.3 Reduction Techniques

The pseudo-Boolean representation of a PMP instance has very attractive properties, which we are going to consider in the rest of this chapter. First of all, a pBp can be subjected to several quite straightforward types of reductions.

2.3.1 Reduction of the Number of Monomials in the pBp

Recall that given a variable vector $\mathbf{y} = (y_1, y_2, \ldots, y_m)^T$, the expressions $\mathscr{T} = \prod_{i \in T} y_i$ and $\alpha\mathscr{T} = \alpha \prod_{i \in T} y_i$ ($T \subseteq \{1, \ldots, m\}$, $\alpha \in \mathbb{R}$) are called a *term* and a *monomial*, respectively. We also call two monomials *similar* if their terms are identical. Finally, by *reduction of monomials* we mean algebraic summation of similar monomials.

Reduction of the number of monomials in pBp consists of three stages. First, as some locations may have equal distance to several clients, the corresponding entries in the difference matrix are zero and the number of terms in the polynomial is usually less than mn (see column $\#T$ in Table 2.1). This reduction is similar to the one introduced by many authors [20, 43, 45, 47, 55] and can be illustrated by the following small example: let $m = 4$, $n = 5$ and the cost matrix is

$$C = \begin{bmatrix} 7 & 15 & 10 & 7 & 10 \\ 10 & 17 & 4 & 11 & 22 \\ 16 & 7 & 6 & 18 & 24 \\ 11 & 7 & 6 & 12 & 8 \end{bmatrix}. \tag{2.23}$$

A possible permutation matrix and the corresponding difference matrix are

$$\Pi = \begin{bmatrix} 1 & 3 & 2 & 1 & 4 \\ 2 & 4 & 3 & 2 & 1 \\ 4 & 1 & 4 & 4 & 3 \\ 3 & 2 & 1 & 3 & 2 \end{bmatrix} \tag{2.24}$$

and

$$\Delta = \begin{bmatrix} 7 & 7 & 2 & 7 & 8 \\ 3 & 0 & 2 & 4 & 2 \\ 1 & 8 & 0 & 1 & 4 \\ 5 & 2 & 4 & 6 & 8 \end{bmatrix}. \tag{2.25}$$

Thus, the pBp is $\mathscr{B}_C = [7 + 3y_1 + 1y_1y_2 + 5y_1y_2y_4] + [7 + 0y_3 + 8y_3y_4 + 2y_1y_3y_4] + [4 + 2y_2 + 0y_2y_3 + 4y_2y_3y_4] + [7 + 4y_1 + 1y_1y_2 + 6y_1y_2y_4] + [8 + 2y_4 + 4y_1y_4 + 8y_1y_3y_4]$. As there are two zeros in the difference matrix, the initial (in contrast to reduced and truncated) pBp has $mn - 2 = 18$ nonzero terms (we will denote this characteristic by $\#T$).

Second, the pBp can be subjected to reducing similar monomials (by its essence, it corresponds to the second reduction rule from [55], p. 11). In the considered example this procedure leads to a polynomial $33 + 7y_1 + 2y_2 + 2y_4 + 2y_1y_2 + 8y_3y_4 + 4y_1y_4 + 11y_1y_2y_4 + 10y_1y_3y_4 + 4y_2y_3y_4$ with ten monomials. We denote the number of monomials in such pBp with reduced similar monomials by $\#T_r$.

Finally, as shown in [4], for any feasible solution $\mathbf{y}$, the value of truncated polynomial $\mathscr{B}_{C,p}$ obtained from $\mathscr{B}_C$ by deleting all terms of degree higher than $(m - p)$ is equal to the value of the initial pBp. For example, $\mathscr{B}_C = 33 + 7y_1 + 2y_2 + 2y_4 + 2y_1y_2 + 8y_3y_4 + 4y_1y_4 + 11y_1y_2y_4 + 10y_1y_3y_4 + 4y_2y_3y_4$ with $p = 2$, i.e., $\mathscr{B}_{C,2} = 33 + 7y_1 + 2y_2 + 2y_4 + 2y_1y_2 + 8y_3y_4 + 4y_1y_4$ has just seven monomials.

Table 2.1 Reduction of the pBp for benchmark instances

Library	Instance	m	Entries in matrix C	$\#T$	$\#T_r$	Reduction (%)
OR	pmed1	100	10,000	7,506	6,722	32.78
OR	pmed15	300	90,000	20,182	17,428	80.64
OR	pmed26	600	360,000	29,963	25,694	92.86
OR	pmed40	900	810,000	36,326	31,642	96.09
ODM	BN48	42	411	411	329	19.95
ODM	BN1284	1,284	88,542	88,447	85,416	3.53
ODM	BN3773	3,773	349,524	348,063	341,775	2.22
ODM	BN5535	5,535	666,639	665,577	654,709	1.79
TSP	rd100	100	9,900	9,394	9,243	6.63
TSP	D657	657	430,992	368,233	367,355	14.77
TSP	fl1400	1,400	1,958,600	838,110	836,557	57.29
TSP	pcb3038	3,038	9,226,406	5,763,280	5,759,404	37.58
RW	rw100	100	10,000	6,357	6,232	37.68
RW	rw200	200	40,000	25,351	25,099	37.25
RW	rw250	250	62,500	39,542	39,228	37.24
RW	rw500	500	250,000	158,007	157,362	37.06
RW	rw1000	1,000	1,000,000	631,805	630,543	36.95

This makes it possible for the particular problem with fixed number of medians to truncate the polynomial thus reducing its size to at most $(m-p)\cdot n$.

In order to determine the effect of the mentioned above techniques, a number of experiments with instances from the four libraries were carried. Results of pseudo-Boolean formulation and reduction of similar monomials for typical representatives of each library are given in Table 2.1. We computed reduction (see the rightmost column of the table) as $(mn-\#T_r)/mn \times 100\,\%$. As can be seen from the table, instances from OR library allow the highest reduction of the number of terms in the pBp. For example, for the instance pmed40 the size of the polynomial is about 4 % of the number of entries in the cost matrix. So, from the point of view of our notion of complexity, these instances are the easiest ones. Instances from TSP and RW libraries also allow compact representation of the polynomial, while ODM instances are the most complex ones and allow only minor reduction of the number of terms.

Of certain interest is a relation between instance size and the achieved reduction (rightmost column in Table 2.1). For OR and TSP libraries this factor tends to increase for larger problems implying that pseudo-Boolean representation is efficient for large instances from these classes. However, for ODM library the situation is opposite, so from this point of view ODM instances are also hard. With randomized graphs from RW library the reduction ratio is almost constant, so these instances are somewhere in between the previous two groups.

Despite the differences in performance between the above-mentioned libraries, truncation of the polynomial has similar impact on the required space for all the considered instances (resulting in at most $(m-p)\cdot n$ entries). We have observed that nonzero entries are uniformly distributed over the rows of the difference matrix

Δ (in other words, the numbers of nonzero monomials of different degrees are approximately the same). It means that with increasing p the number of monomials in the truncated polynomial $\mathscr{B}_{C,p}$ decreases in a linear fashion from $\#T_r$ to 1 (if $p = m$ the polynomial is just a constant). Moreover, if we denote by $p^* \leq m$ the (minimum) number of rows that contain all minima in columns, then the polynomial reduces to a constant for $p \geq p^*$.

2.3.2 Reduction of the Number of Clients (Columns)

In order to show why the reduction of the number of clients (columns) is possible we have to give the following definitions.

Definition 2.1. Two PMP instances defined on costs matrices C and D are called *equivalent* if C and D are of the same size (number of rows) and $\mathscr{B}_{C,p}(\mathbf{y}) = \mathscr{B}_{D,p}(\mathbf{y})$.

Definition 2.2. Having an $m \times n$ cost matrix C, by *aggregation of clients* (columns), we mean construction of such $m \times n'$ matrix D that $\mathscr{B}_{C,p}(\mathbf{y}) = \mathscr{B}_{D,p}(\mathbf{y})$ and $n' < n$.

This means that if there exist some cost matrix D that leads to the same polynomial as C and D has fewer columns, then the PMP defined on C can be substituted by the problem defined on D. So, given a cost matrix C and the number of medians p, one can try to find such a matrix D that corresponds to the same truncated polynomial as C and has the minimum possible number of columns.

The idea behind this type of processing is as follows. Each chain of embedded terms in a pBp corresponds to some permutation and a column of differences. At the same time, over the terms of the polynomial it is possible to define a relation of partial order that, in turn, can be represented by the Hasse diagram. It is clear that all the terms can be covered by n chains that correspond to n columns of the difference matrix. It means that all vertices of the Hasse diagram can be covered by n (internally) vertex-disjoint chains. However, observation that for some instances reduction of similar monomials leads to a substantial decrease in their number suggests a possibility that all terms can be covered by fewer chains. Having a chain of embedded terms it is possible to reconstruct a permutation and a row of the difference matrix. Thus, reduced number of chains covering all terms implies reduced number of clients in the aggregated matrix and the problem of finding the smallest n' is reduced to finding the minimum number of chains that cover all terms of the polynomial (or all vertices of the corresponding Hasse diagram). According to the well-known Dilworth's decomposition theorem (see, e.g., Theorem 14.2 in [136], p. 218), this minimal number of chains is equal to the maximum size of an antichain (in our case it is the maximum number of non-embedded terms).

In order to compute the minimum number of chains we used the MINLEAF algorithm described in [74] that constructs a minimum leaf outbranching.[1] Having

[1] MINLEAF is a polynomial-time algorithm and is essentially based on finding the maximum cardinality matching.

such an outbranching it is possible to reconstruct the chains such that the number of chains is equal to the number of leaves in the outbranching. After that, an equivalent matrix, each column of which is induced by one of the obtained chains, can be restored. As in the formulation of the PMP each column of the costs matrix corresponds to a client whose demand is to be satisfied and existence of the equivalent matrix with smaller number of columns implies that in the initial instance some clients can be aggregated.

Within the mentioned above small example (2.23) this procedure leads to the following. The reduced pBp $\mathscr{B}_C(\mathbf{y}) = 33 + 7y_1 + 2y_2 + 2y_4 + 2y_1y_2 + 8y_3y_4 + 4y_1y_4 + 11y_1y_2y_4 + 10y_1y_3y_4 + 4y_2y_3y_4$ corresponds to the following Hasse diagram:

$$\begin{array}{ccccccccc} & & y_2 & \to & y_1y_2 & \to & y_1y_2y_4 & & \\ & \nearrow & & \nearrow & & \nearrow & & \searrow & \\ const & \to & y_1 & \to & y_1y_4 & \to & y_1y_3y_4 & \to & y_1y_2y_3y_4 \\ & \searrow & & \nearrow & & \nearrow & & \nearrow & \\ & & y_4 & \to & y_3y_4 & \to & y_2y_3y_4 & & \end{array} \tag{2.26}$$

It is easy to check that the size of the maximum antichain is 3, so all the terms of $\mathscr{B}_C(\mathbf{y})$ can be covered by three chains and the aggregated matrix has three columns. Below are the chains (each being presented as a column), permutation and difference matrices:

$$\begin{array}{lll} y_2 & y_1 & y_4 \\ y_1y_2 & y_1y_4 & y_3y_4 \\ y_1y_2y_4 & y_1y_3y_4 & y_2y_3y_4 \\ y_1y_2y_3y_4 & y_1y_2y_3y_4 & y_1y_2y_3y_4 \end{array} \tag{2.27}$$

$$\Pi' = \begin{bmatrix} 2 & 1 & 4 \\ 1 & 4 & 3 \\ 4 & 3 & 2 \\ 3 & 2 & 1 \end{bmatrix} \quad \Delta' = \begin{bmatrix} 0 & 0 & 33 \\ 2 & 7 & 2 \\ 2 & 4 & 8 \\ 11 & 10 & 4 \end{bmatrix}. \tag{2.28}$$

Having these two matrices it is possible to restore the cost matrix D of the aggregated instance:

$$D = \begin{bmatrix} 2 & 0 & 47 \\ 0 & 21 & 43 \\ 15 & 11 & 35 \\ 4 & 7 & 33 \end{bmatrix}. \tag{2.29}$$

2.3.2.1 Experiments

As was mentioned above, the minimum number of aggregated clients (columns) does not exceed n. On the other hand, it cannot be smaller than the maximum number of terms with same degree in the reduced polynomial. In particular, for the case of instances from OR library, this leads to the following. As the cost matrix for such instances has a zero diagonal, the minimal element of ith column is located in the ith row and the first row of the permutation matrix contains no equal entries.

This means that the (reduced) pBp contains n linear terms and cannot be covered by less than n chains. So, the OR instances, if considered "as is," do not allow any aggregation of clients. This result brought us to an idea of considering the corrected instances without zeros on the diagonal (it is filled by some positive numbers during application of the Floyd's algorithm). Further we mark such instances with an asterisk (e.g., pmed1*). As all the other considered libraries are free of the mentioned "hardness," they can be directly used for experiments with aggregation of clients.

In our experiments we considered truncated polynomials and determined the minimum number of aggregated columns (n') for all values of p from 1 to $m-1$ (if $p=m$, the truncated polynomial is just a constant and it can be covered by one chain). Let us denote by p'' the smallest number of medians at which the truncated polynomial can be covered by less than n chains.

The results for typical representatives from each library are given in Figs. 2.1 and 2.2. As can be seen from the figures, for corrected OR and ODM problems $p''=0$ and even a non-truncated polynomial can be covered by $n-1$ chains, thus making it possible to aggregate one client. At the same time, for TSP and RW instances, any aggregation becomes possible only as p gets very close to m.

Thus, we can summarize that the reduction of the number of clients is negligible for all the considered benchmark libraries.

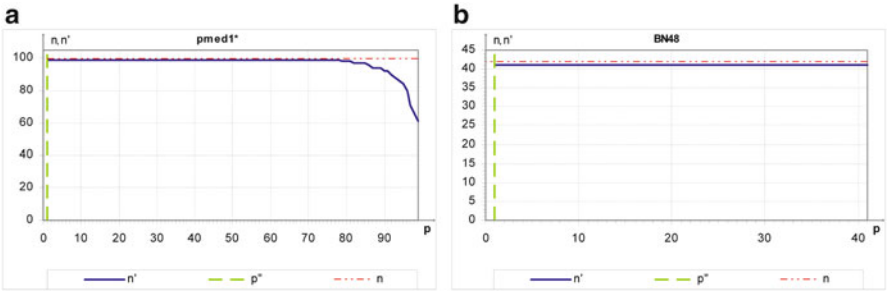

Fig. 2.1 Aggregation of clients for benchmark instances from (**a**) OR and (**b**) ODM libraries

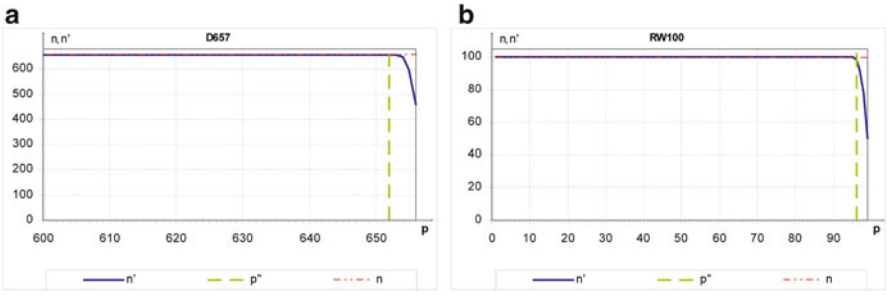

Fig. 2.2 Aggregation of clients for benchmark instances from (**a**) TSP and (**b**) RW libraries

2.3.3 Preprocessing: An Overview and Application to the PMP

The essence of preprocessing that we consider is to find such locations that can be excluded from consideration as they are not contained in some optimal solution (see also [61]). At the same time, the technique considered in this section is independent of any solution algorithm (e.g., it does not use upper and lower bounds) and is based purely on the structural properties of the input cost matrix.

Let us define the p-truncation operation applied separately to each column of the cost matrix as setting p largest entries to the value of the smallest of them. This procedure ensures that the pBp of the p-truncated matrix is equal to the truncated pBp of the initial matrix. The following theorem (see [4], Theorem 4 and [5]), provides a direct suggestion for preprocessing based on p-truncation.

Theorem 2.2. *Assume that in a given PMP instance with $p < m$ some row i in the cost matrix C contains all the columns maxima after p-truncation operations are performed on all columns of C. Then there exists an optimal solution $\mathbf{y}^*$ to the instance with $y_i^* = 1$.*

Proof. The fact that row i in a p-truncated matrix contains all columns maxima implies that location i is among the $m - p$ most expensive locations for every client. This, in turn, means that in a feasible solution each client j can be served cheaper from a different location. Thus, location i can be excluded from consideration and the corresponding y-variable can be fixed to 1. □

In other words, the theorem means that if some variable y_i is not contained in the truncated polynomial, then there exists an optimal solution $\mathbf{y}^*$ with $y_i^* = 1$. In order to illustrate this we would like to consider the following example. Let the cost matrix be defined as (the rightmost column of numbers enumerates rows of the matrix):

$$C = \begin{bmatrix} 1\,3\,9 \\ 2\,5\,3 \\ 9\,7\,8 \\ 5\,9\,7 \\ 4\,4\,5 \end{bmatrix} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix}. \tag{2.30}$$

Also, let p be $p = \lceil m/2 \rceil = 3$ (that corresponds to the hardest case from a combinatorial point of view). The p-truncated matrix $C_{p=3}$ is

$$C_{p=3} = \begin{bmatrix} 1\,3\,7 \\ 2\,5\,3 \\ 4\,5\,7 \\ 4\,5\,7 \\ 4\,4\,5 \end{bmatrix} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix}. \tag{2.31}$$

The objective function can be represented by the pBp $\mathscr{B}_C(\mathbf{y}) = 7 + 2y_1 + 2y_2 + 2y_1y_2 + 1y_1y_5 + 2y_2y_5 + 3y_1y_2y_5 + 1y_2y_4y_5 + 4y_1y_2y_4y_5 + 2y_1y_2y_3y_5 + 1y_2y_3y_4y_5$. After truncation one obtains $\mathscr{B}_{C,p=3}(\mathbf{y}) = 7 + 2y_1 + 2y_2 + 2y_1y_2 + 1y_1y_5 + 2y_2y_5$.

As can be seen, the truncated pBp does not contain two variables y_3, y_4, so they can be set to 1 as this does not affect the value of $\mathscr{B}_{C,p=3}(\mathbf{y})$. This means that the initial matrix C given by (2.30) can be reduced to matrix D with fewer rows:

$$D = \begin{bmatrix} 1\ 3\ 7 \\ 2\ 5\ 3 \\ 4\ 4\ 5 \end{bmatrix} \begin{matrix} 1 \\ 2\,. \\ 5 \end{matrix} \tag{2.32}$$

It should be noticed that if one sets all other variables y_1, y_2, y_5 to 0, this immediately gives the optimal solution. Thus, for this small example the problem can be solved just by data preprocessing.

However, with large instances this technique does not always allow solving the problem. Given a PMP cost matrix, we studied how the possibility of preprocessing depends on the value of p. As the value of p grows, the number of entries in any column whose values are revised increases. So, the higher the value of p, the greater the chance that a row of C is eliminated due to Theorem 2.2. This explains why PMP instances with $p = p_0$, $p_0 < m/2$, are more difficult to solve than instances on the same cost matrix with $p = m - p_0$, even though the number of feasible solutions for both cases are identical. Let p' be the smallest value of p for which p-truncation eliminates at least one row in C. Let us also denote by p^* the minimum number of rows that contain the minimum entry of each column of C. Then, the PMP instance defined on C with $p > p^*$ has open facilities that do not serve any client and increasing the value of p over p^* does not improve the objective value.

Table 2.2 presents a characterization of benchmark instances introduced in Table 2.1 in terms of p' and p^*. As can be seen from the table, preprocessing becomes possible only for large number of medians as $p' > m/2$ holds for all the considered benchmark instances. One may notice that the benchmark libraries are arranged in order of increasing difficulty: value of p' are getting closer to m. The values of p^* are very close to m for all the considered instances implying that all benchmark libraries contain no degenerate instances and most of the rows can be potentially included into an optimal solution.

2.3.4 Minimality of the Pseudo-Boolean Representation

In the previous sections we described a number of reductions that are based on the pseudo-Boolean formulation of the PMP and substantially reduce the amount of data that unambiguously describes the instance. However, there emerges a natural question: can one do better by using a different approach? The following lemma gives an answer to this question.

Theorem 2.3. *The pseudo-Boolean formulation (2.17) of PMP allows the most compact representation of its instance.*

Proof. The intuition is as follows. Take the reduced and truncated pBp and consider a monomial $\alpha\mathscr{T}$ with a nonzero coefficient that corresponds to an entry of the cost

Table 2.2 Values of p' and p^* for benchmark instances

Library	Instance	m	p'	p^*
OR	pmed1*	100	90	93
OR	pmed15*	300	180	285
OR	pmed26*	600	452	581
OR	pmed40*	900	644	882
ODM	BN48	42	27	35
ODM	BN1284	1,284	653	1,211
ODM	BN3773	3,773	3,385	3,742
ODM	BN5535	5,535	2,179	5,503
TSP	rd100	100	97	97
TSP	D657	657	477	653
TSP	fl1400	1,400	1,177	1,395
TSP	pcb3038	3,038	3,026	3,033
RW	rw100	100	90	95
RW	rw200	200	186	193
RW	rw250	250	241	243
RW	rw500	500	489	492
RW	rw1000	1,000	978	992

matrix c_{ij} that does not contribute to any optimal solution. This means that there exist, a client j that cannot be assigned to location i. There can be several causes for this:

1. For any subset S of p opened locations there always exists a location $i' \in S$ such that $c_{i'j} < c_{ij}$. In this case client j is never served from location i.
2. For client j location i can be replaced by location i', i.e., there exist some location i' such that $c_{ij} = c_{i'j}$. In this case client j can be served from location i' instead of i.
3. For some subset of locations S client j is equivalent to some client j'. (By equivalence of clients with regard to the set of locations S we mean that sorting locations from S by distance from j and j' gives two equal sequences.) In this case these two clients can be viewed as one with aggregate serving costs $c_{ij} + c_{ij'}$ for all $i \in S$.

The latter two cases are symmetric: case 2 means that from the point of view of client j locations i and i' are equally distant, while case 3 means that from the point of view of the set of locations S clients j and j' are equal. In case 1 the coefficient α is set to 0 during the truncation. For the second case we have zero coefficient as the difference $\Delta[.,j] = c_{ij} - c_{i'j}$ is zero. Finally, for the third case equivalent clients are eliminated by reduction of similar monomials.

Next, consider the number of coefficients in the truncated and reduced Hammer–Beresnev polynomial $\mathscr{B}_{C,p}(\mathbf{y})$. Suppose, there exist a model with one less coefficient. This implies that some monomial $\alpha_r \prod_{i \in T_r} y_i$ can be deleted from $\mathscr{B}_{C,p}(\mathbf{y})$ to obtain a new polynomial $\mathscr{B}'_{C,p}(\mathbf{y})$. However, it is always possible to select an input matrix C such that

$$f_C(S) = \mathscr{B}_{C,p}(\mathbf{y}^S)$$

and

$$\min\{\mathscr{B}'_{C,p}(\mathbf{y}), \sum_{i=1}^{m} y_i = m - p\} = \min\{\mathscr{B}_{C,p}(\mathbf{y}), \sum_{i=1}^{m} y_i = m - p\} - \alpha_r.$$

Taking into account that $\alpha_r > 0$ (by definition of $\mathscr{B}_{C,p}(\mathbf{y})$), the optimal values of the two formulations are different and we have a contradiction. □

Theorem 2.3 has important consequence for applicability of the pseudo Boolean formulation. Let us consider an arbitrary model of the PMP within the class of mixed Boolean linear programming (LP) models. The size of a mixed Boolean LP model is determined by the following four factors:

- Number of Boolean variables
- Number of continuous variables
- Number of constraints (and number of terms in each constraint)
- Number of monomials in the objective function.

We claim that the minimum mixed-Boolean LP model for PMP can be derived from its pseudo-Boolean representation, as demonstrated in the next section.

2.4 A Compact Mixed Boolean LP Model

In order to obtain a mixed Boolean LP model we have linearized all nonlinear terms in 2.16 by introducing additional variables $z_r = \prod_{i \in T_r} y_i$. Since $\alpha_r \geq 0$ and PMP is a minimization problem (2.17) one can replace each nonlinear equality $\prod_{i \in T_r} y_i = z_r$ by an equivalent nonlinear inequality $\prod_{i \in T_r} y_i \leq z_r$ which is equivalent to the following system of linear inequalities: $\sum_{i \in T_r} y_i - |T_r| + 1 \leq z_r$ and $z_r \geq 0$. In any optimal PMP solution the variable z_r is set to 0 if and only if at least one variable $y_i = 0$, and $z_r = 1$ if and only if all $y_i = 1$, i.e., $z_r \in \{0,1\}$. Now the pseudo-Boolean formulation of PMP with a nonlinear objective function (2.16) can be presented as the following mixed Boolean linear programming model:

$$\text{minimize} \quad \left\{ \alpha_0 + \sum_{r=1}^{m} \alpha_r y_r + \sum_{r=m+1}^{k} \alpha_r z_r \right\} \tag{2.33}$$

$$\text{s.t.} \quad \sum_{i=1}^{m} y_i = m - p; \tag{2.34}$$

$$\sum_{i \in T_r} y_i - |T_r| + 1 \leq z_r, \; r = m+1, \ldots, k; \tag{2.35}$$

$$z_i \geq 0, \quad i = m+1, \ldots, k. \tag{2.36}$$

$$y_i \in \{0,1\}, \; i = 1, \ldots, m. \tag{2.37}$$

The objective function (2.33) is split into three parts: the first part α_0 is the sum of all smallest entries δ_{1j} per column (client) j, the second part reflects the penalties incurred by the next to the smallest entries δ_{2j}, and the third part represents all other penalties corresponding to δ_{ij} for $1 < i \leq m - p$.

We call the formulation (2.33)–(2.37) a MBpBM for PMP. In case of the example cost matrix defined by (2.18) the MBpBM formulation can be easily derived from (2.22):

$$\min 8 + y_2 + 2y_4 + z_5 + z_6 + z_7 + z_8 \tag{2.38}$$

$$\text{s.t. } z_5 + 1 \geq y_1 + y_3, \tag{2.39}$$

$$z_6 + 1 \geq y_2 + y_3, \tag{2.40}$$

$$z_7 + 1 \geq y_2 + y_4, \tag{2.41}$$

$$z_8 + 1 \geq y_3 + y_4, \tag{2.42}$$

$$\sum_{i=1}^{4} y_i = m - p = 2, \tag{2.43}$$

$$z_i \geq 0,\ i = 5, \ldots, 8; \tag{2.44}$$

$$y_i \in \{0, 1\},\ i = 1, \ldots, 4. \tag{2.45}$$

In the following Lemma 2.1 we explain how to reduce the number of Boolean variables y_i involved in the restrictions (2.35). If for $T_{r_1} \neq \emptyset$ we have that $T_{r_1} \subset T_{r_2}$, then the number of variables corresponding to the inequality with z_{r_2} in (2.35) might be reduced as follows:

$$z_{r_1} + \sum_{i \in T_{r_2} \setminus T_{r_1}} y_i - |T_{r_2} \setminus T_{r_1}| + 1 \leq z_{r_2}. \tag{2.46}$$

Lemma 2.1. *Let $\emptyset \neq T_{r_1} \subset T_{r_2}$ be a pair of embedded sets of Boolean variables y_i. Thus, two following systems of inequalities*

$$\sum_{i \in T_{r_1}} y_i - |T_{r_1}| + 1 \leq z_{r_1} \tag{2.47}$$

$$\sum_{i \in T_{r_2}} y_i - |T_{r_2}| + 1 \leq z_{r_2} \tag{2.48}$$

$$z_{r_1} \geq 0, \quad z_{r_2} \geq 0 \tag{2.49}$$

and

$$\sum_{i \in T_{r_1}} y_i - |T_{r_1}| + 1 \leq z_{r_1} \tag{2.50}$$

$$z_{r_1} + \sum_{i \in T_{r_2} \setminus T_{r_1}} y_i - |T_{r_2} \setminus T_{r_1}| \leq z_{r_2} \tag{2.51}$$

$$z_{r_1} \geq 0, \quad z_{r_2} \geq 0 \tag{2.52}$$

are equivalent.

Proof. Our proof will be done if we show that the following inequalities $\sum_{i\in T_{r_2}} y_i - |T_{r_2}| + 1 \le z_{r_2}$ and $z_{r_1} + \sum_{i\in T_{r_2}\setminus T_{r_1}} y_i - |T_{r_2} \setminus T_{r_1}| \le z_{r_2}$ are equivalent. Note that $z_{r_2} = 1$ if and only if $y_i = 1$ for all $i \in T_{r_2}$ which implies that $z_{r_1} = 1$ since $T_{r_1} \subset T_2$, but $z_{r_2} = 0$ if and only if at least one variable $y_i = 0$ for some $i \in T_{r_2}$. If $i \in T_{r_1}$, then $z_{r_1} = 0$ implies $z_{r_2} = 0$, even if $y_i = 1$ for all $i \in T_{r_2} \setminus T_{r_1}$; otherwise $y_i = 0$ for some $i \in T_{r_2} \setminus T_{r_1}$ implies $z_{r_2} = 0$ even if $z_{r_1} = 1$. □

In the following theorem we indicate that our new problem formulation (2.33)–(2.37) is equivalent to the pseudo-Boolean formulation (2.17).

Theorem 2.4. *PMP formulations (2.33)–(2.37) and (2.17) are equivalent.*

Proof. The "if" statement is trivial and we start with "only if" part, i.e., we are going to show that any feasible solution to (2.33)–(2.37) is feasible to (2.17). Constraints (2.35) ensure that for any subset of opened sites within T_r the corresponding penalties in both objective functions will be zero. Otherwise (if all sites within T_r are closed), the same penalty value will be added to the objective functions of (2.33)–(2.37) and (2.17). □

It is clear that the restriction (2.35) for z_r can be expressed by means of embedded terms with different degrees such that $T = \left\{\cup_{t=1}^{k} T_{r_t}\right\} \subseteq T_r$.

Based on the compact representation of a PMP instance within pseudo-Boolean formulation (2.17) one may conclude that this formulation has extracted only essential information to represent the PMP from optimization point of view. In particular, for each client j only sites with p-truncated and pairwise different distances are essential for an optimal PMP solution. These distances form the objective function of our mixed Boolean linear programming formulation (MBpBM) as well as the set of linear constraints. Since each linear constraint (2.35) represents a non-linear monomial in the objective function of pseudo-Boolean formulation (2.17) we have incorporated in the MBpBM the number p of medians as follows: for larger values of p our MBpBM has less non-negative variables and corresponding constraints induced by non-linear monomials. It means that we are in a position to check whether our MBpBM is an optimal model within the class of Mixed Boolean Linear Programming models. If we will be able to show that the matrix of all linear constraints in our MBpBM induced by non-linear monomials contains the smallest number of nonzero entries, then taking into account that the objective function of our MBpBM has the smallest number of nonzero coefficients, one may conclude that our MBpBM is an optimal one. Unfortunately, in general it is not the case. It is not difficult to show that the problem of finding a constraint matrix with the smallest number of nonzero entries is at least as hard as the *classic set covering problem* (see, e.g., [60]). Let us consider a partially ordered set with subsets of cardinality at least two corresponding to all non-linear monomials in the truncated and reduced pBp. Take any set F with the largest cardinality. We say that the set F is covered by its subsets $F_i \subset F$ if $|F \setminus (\cup_{i\in L} F_i)| \le 1$ and F is covered by a single subset if $|F \setminus F_r)| = 1$. It is clear that the following linear constraint corresponding to a single covering

$$z_{F_r} + y_r - 1 \leq z_F$$

has the smallest number of nonzero entries. In general case even for a single set F to find the best covering by subsets embedded in F is an NP-hard problem (see, e.g., [60]), but our problem is more difficult since we are looking for an optimal covering not only for the set F but also for all its subsets minimizing the number of nonzero entries of all corresponding linear constraints. In other words, we have shown that to find an optimal model within the class of mixed Boolean linear programming models is an NP-hard problem, even if the number of corresponding linear constraints is a linear function on the number of all nonnegative decision variables.

Note that the number of nonzero coefficients in the objective function of MBpBM is minimal because the number of nonzero coefficients corresponding to non-linear terms is minimal (just by means of contradiction it can be easily shown that if we assume that there is an objective function with strictly less number of nonzero coefficients, then there is either a feasible or an optimal solution to the PMP for which the objective function value defined on the corresponding solution is strictly less than the objective function computed on the given PMP instance). Since the number of linear constraints (2.35) is equal to the number of nonzero coefficients α_r for $r = m+1, \ldots, k$ one may conclude that both numbers, namely the number of nonzero coefficients in (2.33) and the number of linear constraints (2.35), are minimal. These considerations are formalized in the following theorem.

Theorem 2.5. *The numbers of coefficients in the objective, variables, and constraints in MBpBM are minimal within the class of mixed Boolean LP models for PMP.*

Proof. First of all note that the number of Boolean variables cannot be smaller than m as otherwise some potential locations are not taken into account. Next, minimality of the number of coefficients in the objective immediately follows from the minimality of the pseudo-Boolean representation (Theorem 2.3). This implies minimality of the number of nonnegative variables (corresponding to nonlinear monomials) as the number of Boolean variables is fixed to m and is closely related to the number of linear monomials. This, in turn, implies minimality of the number of constraints as it is exactly the number of nonnegative variables and each nonnegative variable needs at least one constraint to be biased with Boolean variables. □

Theorem 2.5 can be illustrated by the following example using the cost matrix (2.18). The MBpBM can be easily derived from the pseudo-Boolean formulation (2.22) and looks like:

$$\begin{aligned}
& f(\mathbf{y},\mathbf{z}) = 8 + y_2 + 2y_4 + z_5 + z_6 + z_7 + z_8 \to \min \\
& \text{s.t.} \\
& y_1 + y_2 + y_3 + y_4 = 2, \\
& z_5 \geq y_1 + y_3 - 1, \\
& z_6 \geq y_2 + y_3 - 1, \\
& z_7 \geq y_2 + y_4 - 1, \\
& z_8 \geq y_3 + y_4 - 1, \\
& y_i \in \{0,1\},\ i = 1, \ldots, 4, \\
& z_i \geq 0,\ i = 5, \ldots, 8.
\end{aligned} \tag{2.53}$$

The obtained model has 4 Boolean y-variables, 4 nonnegative z-variables, 5 constraints and 6 terms in the objective function. For Elloumi's MILP model [55] these numbers are 4, 17, 23, and 12, respectively.

Properties of our model, such as decreasing number of variables and constraints, give some insight into the properties of the polytope of feasible solutions to the PMP. The fact that the total number of variables in MBpBM is always less than $(m-p)\cdot n$ implies that the p-median polytope never has a full dimension of $m\cdot n$, i.e., some dimensions either are fixed (by p-truncation) or are duplicate (these are removed by combining similar monomials in the Hammer–Beresnev polynomial). At the same time MBpBM allows measuring the actual dimension of the polytope and implies that this dimension decreases with increasing p as more and more dimensions become fixed (more and more assignments of clients to facilities become prohibited).

2.4.1 Further Reductions

Even though MBpBM is a very compact model, we can further reduce it by involving upper and lower bounds on the cost of optimal solutions (see [68]). Suppose, we know from some heuristic a (global) upper bound f^{UB} on the cost of optimal solutions. This can be even a virtual upper bound (see for more details virtual upper bounds created within the data correcting approach for solving different combinatorial optimization problems [64, 67]), i.e., without a feasible solution. Let us now consider some term $\mathscr{T}_r = \sum_{i\in T_r} y_i$ from the pBp and define a vector $\mathbf{y}^r$ in the following way: for any $i \in T_r$ set $y_i^r = 1$ and set all other elements of $\mathbf{y}^r$ to zero. It is easy to see that $\mathscr{B}_{C,p}(\mathbf{y}^r)$ is a valid lower bound for the subspace of solutions with all locations from T_r closed. If we denote this lower bound by f_r^{LB}, then the following holds:

$$f_r^{LB} = \mathscr{B}_{C,p}(\mathbf{y}^r). \tag{2.54}$$

The essence of the reduction that we consider here can be expressed by the following lemma.

Lemma 2.2. *If for some term $\mathscr{T}_r = \prod_{i\in T_r} y_i$ of a truncated and reduced Hammer–Beresnev polynomial holds $f_r^{LB} > f^{UB}$, then for every optimal solution $\mathbf{y}^*$ holds $\mathscr{T}_r(\mathbf{y}^*) = 0$.*

Proof. Taking into account that we have a truncated polynomial, any term r has a degree of at most $m-p$ and $|T_r| \le m-p$. This means that to keep $\mathscr{T}_r$ equal to 1 at least p sites are to be opened, implying that the cost of any feasible solution with sites from T_r closed is at least $\mathscr{B}_{C,p}(\mathbf{y}^r)$ (by closing additional sites we can only increase the objective value). By condition of the lemma we have that there exists a cheaper feasible solution, thus any feasible solution $\mathbf{y}$ with $y_i = 1$ for any $i \in T_r$ is not optimal. □

The following counter-example shows that the inequality in Lemma 2.2 should be strict.

Example 2.2. Consider an instance defined by the following cost matrix C:

$$C = \begin{bmatrix} 0 & 6 & 6 \\ 1 & 0 & 8 \\ 2 & 9 & 9 \\ 5 & 4 & 0 \end{bmatrix}. \tag{2.55}$$

A possible permutation and difference matrices are

$$\Pi = \begin{bmatrix} 1 & 2 & 4 \\ 2 & 4 & 1 \\ 3 & 1 & 2 \\ 4 & 3 & 3 \end{bmatrix} \quad \Delta = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 4 & 6 \\ 1 & 2 & 2 \\ 3 & 3 & 1 \end{bmatrix}. \tag{2.56}$$

In case $p = 2$ the Hammer–Beresnev polynomial is

$$\mathscr{B}_{C,p=2}(\mathbf{y}) = 1y_1 + 4y_2 + 6y_4 + 1y_1y_2 + 3y_1y_4 + 2y_2y_4. \tag{2.57}$$

Considering the first term $\mathscr{T}_1 = y_1$, we have $T_1 = \{1\}$, $\mathbf{y}^1 = (1,0,0,0)$, and $f_1^{LB} = \mathscr{B}_{C,p=2}(\mathbf{y}^1) = 1$. Suppose, the upper bound is the same $f^{UB} = 1$. If the inequality in Lemma 2.2 was non-strict, then this would imply that for any optimal solution $y_1 = 0$. However, it can be checked that for the unique optimal solution to this instance $\mathbf{y}^* = (1,0,1,0)$, this does not hold. Note that $y_3^* = 1$ due to Theorem 2.2.

Now we can check the condition of Lemma 2.2 for every term of the pBp, starting from terms with the lowest degree. Such order of checking terms is beneficial because if some term is found to be zero in any optimal solution (let us call such terms *null terms*), then all terms of higher degree containing it are also zero. Here we would like to point out two possibilities of dealing with null terms within our MBpBM model.

The first and the most straightforward approach is to set the variable that corresponds to a null term to zero throughout the formulation. This eliminates one term from the objective function, but preserves the number of constraints. At the same time the number of nonzero entries in the constraint matrix can increase as elimination of some term (or corresponding z-variable) reduces the possibility of applying Lemma 2.1. We call the model reduced according to this approach based on bounds MBpBMb. The following example shows how this reduction works.

Consider the cost matrix C (2.18), $p = 2$, and an MBpBM model (2.53). One can compute the global upper bound f^{UB}, for example, by greedy heuristics that works as follows. It starts with all locations opened (i.e., $\mathbf{y} = (0,0,0,0)$) and at each step closes such location (sets such y_i to 1) that results in the smallest increase in the value of the objective function. The procedure is repeated until $m - p$ locations are closed ($m - p$ entries of $\mathbf{y}$ are set to 1). For the cost matrix given by (2.18) this

procedure gives $f^{UB} = 9$. Then, for every term $\mathscr{T}_r$ of the objective function, we construct a vector $\mathbf{y}^r$ and compute the lower bound to the unknown optimal value $\mathscr{B}_{C,p=2}(\mathbf{y}^r)$:

$$\begin{array}{lll}
\mathscr{T}_1 = y_2 & \mathbf{y}^1 = (0,1,0,0) & \mathscr{B}_{C,p=2}(\mathbf{y}^1) = 9, \\
\mathscr{T}_2 = y_4 & \mathbf{y}^2 = (0,0,0,1) & \mathscr{B}_{C,p=2}(\mathbf{y}^2) = 10 > f^{UB}, \\
\mathscr{T}_3 = z_5 = y_1 y_3 & \mathbf{y}^3 = (1,0,1,0) & \mathscr{B}_{C,p=2}(\mathbf{y}^3) = 9, \\
\mathscr{T}_4 = z_6 = y_2 y_3 & \mathbf{y}^4 = (0,1,1,0) & \mathscr{B}_{C,p=2}(\mathbf{y}^4) = 10 > f^{UB}, \\
\mathscr{T}_5 = z_7 = y_2 y_4 & \mathbf{y}^5 = (0,1,0,1) & \mathscr{B}_{C,p=2}(\mathbf{y}^5) = 12 > f^{UB}, \\
\mathscr{T}_6 = z_8 = y_3 y_4 & \mathbf{y}^6 = (0,0,1,1) & \mathscr{B}_{C,p=2}(\mathbf{y}^6) = 11 > f^{UB}.
\end{array} \tag{2.58}$$

By comparing the obtained values with the computed upper bound we have that in any optimal solution $\mathscr{T}_2$, $\mathscr{T}_4$, $\mathscr{T}_5$, and $\mathscr{T}_6$ are zero, i.e., in our MBLP model we can fix variables y_4, z_6, z_7, and z_8 to zero:

$$8 + y_2 + z_5 + 0z_6 + 0z_7 + 0z_8 \to \min \tag{2.59}$$

$$\text{s.t. } y_1 + y_2 + y_3 = 2, \tag{2.60}$$

$$z_5 + 1 \geq y_1 + y_3, \tag{2.61}$$

$$z_6 + 1 \geq y_2 + y_3, \tag{2.62}$$

$$0 + 1 \geq y_2 + 0, \tag{2.63}$$

$$0 + 1 \geq y_3 + 0, \tag{2.64}$$

$$y_4 = 0, \tag{2.65}$$

$$z_i \geq 0,\ i = 5,\ldots,8; \tag{2.66}$$

$$y_i \in \{0,1\},\ i = 1,\ldots,4. \tag{2.67}$$

By substituting the fixed values into all constraints and the objective function and observing that some constraints (2.63) and (2.64) became redundant, we obtain the reduced model:

$$8 + y_2 + z_5 \to \min \tag{2.68}$$

$$\text{s.t. } z_5 + 1 \geq y_1 + y_3, \tag{2.69}$$

$$1 \geq y_2 + y_3, \tag{2.70}$$

$$y_1 + y_2 + y_3 = 2, \tag{2.71}$$

$$y_4 = 0, \tag{2.72}$$

$$z_5 \geq 0; \tag{2.73}$$

$$y_i \in \{0,1\},\ i = 1,\ldots,4. \tag{2.74}$$

Further we will call this variation of MBpBM with reduction based on bounds—MBpBMb.

Another approach to dealing with null terms allows to reduce not only the size of the objective function but also the number of constraints. Its essence is expressed by the following lemmas.

Lemma 2.3. *Increasing a coefficient at a null term does not affect the cost of optimal solutions.*

Proof. Straightforward from the definition of null terms. If some term is found to be equal to 0 in any optimal solution, then increasing the coefficient of the corresponding monomial will not result in new optimal solutions. □

Lemma 2.4. *If for some term r_1 in the Hammer–Beresnev polynomial there exists an embedded term r_0 with large enough coefficient α_{r_0}, then r_1 can be given a zero coefficient without affecting the cost of optimal solutions.*

Proof. The cost of any feasible solutions for which term r_0 evaluates to 1 is bounded from below by α_{r_0}. If there exists a feasible solution of cost less than α_{r_0}, all terms containing r_0 evaluate to 0 in any optimal solution. This, in turn, implies that their coefficients are irrelevant and can be set to 0. □

Thus, if a null term is found, it can be given a large enough coefficient (exceeding a cost of an arbitrary feasible solution) and all terms for which it is embedded can be eliminated from the Hammer–Beresnev polynomial without a need in additional constraints. We call the model with this reduction MBpBMb1. Even though it allows smaller reduction of the objective function, it can benefit from the reduced number of constraints. For the considered above instance with cost matrix C (2.18) and $p = 2$ the MBpBMb1 model is as follows:

$$8 + y_2 + 1{,}000y_4 + z_5 + 1{,}000z_6 \to \min \tag{2.75}$$

$$\text{s.t. } z_5 + 1 \geq y_1 + y_3, \tag{2.76}$$

$$z_6 + 1 \geq y_2 + y_3, \tag{2.77}$$

$$y_1 + y_2 + y_3 + y_4 = 2, \tag{2.78}$$

$$y_4 = 0, \tag{2.79}$$

$$z_i \geq 0,\ i = 5, \ldots, 8; \tag{2.80}$$

$$y_i \in \{0, 1\},\ i = 1, \ldots, 4. \tag{2.81}$$

Here we have set large coefficients to 1,000 for the sake of clarity, while in practice the value of $f^{UB} + 1 = 10$ suffices for this purpose. It is also clear that if we find a linear null term, then we fix the corresponding y-variable to 0 and eliminate this term from the objective function (instead of raising its coefficient to infinity). Moreover, if for some null term there are no terms into which it is embedded, then it is beneficial to eliminate it and add a corresponding constraint. If these exceptions are introduced into MBpBMb1, then for the considered example both formulations (MBpBMb and MBpBMb1) become equal, although the mechanisms by which some constraints were dropped are different. While in MBpBMb third and fourth constraints became redundant as most of the variables in them were set to 0, in MBpBMb1 these constraints did not exist at all because corresponding terms were eliminated from the pBp.

Finally, it should be mentioned that instead of $f_r^{LB} = \mathscr{B}_{C,p}(\mathbf{y}^r)$, a somewhat stronger lower bound can be used.

Lemma 2.5. *ϕ_r^{LB} defined as*

$$\phi_r^{LB} = f_C(\overline{T}_r) + \min_{k_i \in \overline{T}_r} \sum_{i=1}^{|\overline{T}_r|-p} [f_C(\overline{T}_r \setminus \{k_i\}) - f_C(\overline{T}_r)] \tag{2.82}$$

is a valid lower bound for the subspace of feasible solutions having locations from T_r closed, where $f_C(.)$—cost function of the PMP, i.e., $f_C(\overline{T}_r) = \mathscr{B}_{C,p}(\mathbf{y}^r)$, and $\overline{T}_r$ denotes the complement of T_r, i.e., $\overline{T}_r = \{1,\ldots,m\} \setminus T_r$.

Proof. As the cost function of the PMP $f_C(.)$ is a supermodular function [68], the following holds for any $S \subseteq T \subseteq \{1,\ldots,m\}$

$$f_C(S) \geq f_C(T) + \sum_{k \in T \setminus S} [f_C(T \setminus \{k\}) - f_C(T)]. \tag{2.83}$$

In our case S is unknown restricted ($S \subseteq T$ must hold) optimal solution (set of opened locations) of cardinality $|S| = p$ implying that

$$f_C(S) \geq f_C(T) + \min_{k_i \in T} \sum_{i=1}^{|T|-p} [f_C(T \setminus \{k_i\}) - f_C(T)] \tag{2.84}$$

where $\overline{T} = \{1,\ldots,m\} \setminus T$. If we now set $T = \overline{T}_r$, then the proof is completed. □

In the computational experiments reported in the following sections (and involving MBpBM and its modifications) we used lower bounds given by (2.82).

2.4.2 Computational Experiments

In order to show the applicability of our compact MBpBM formulation, a number of computational experiments were held. We used benchmark instances from two of the most widely used libraries: J. Beasley's OR library and randomly generated RW instances by Resende and Werneck (see, e.g., [55]). The common class of benchmark instances included in almost all publications devoted to the PMP itself is just the OR library instances. Since the main purpose of our experiments is to show that our model for PMP is one of the best currently known models (see [44, 55]) which could be used to solve PMP instances to optimality based on general-purpose software, we have used Xpress-MP and 15 largest instances (see Tables 2.4 and 2.3) from OR library [94]. This benchmark library contains 40 different PMP instances, each representing a graph of n vertices, each vertex being a client and a potential facility and a specific value of p. Graphs are complete and range in size from 100 (with 10,000 arcs) to 900 (with 810,000 arcs) nodes. The distance c_{ij} between two nodes is the length of a shortest path linking them.

We have conducted our experiments on a personal computer with Intel 2.33 GHz processor, 1.95 GB RAM and Xpress-MP as an MILP solver.

Table 2.3 MBpBM for different numbers of medians in pmed39/pmed40

	pmed39				pmed40			
p	$f_C(S^*)$	M_{tr}	constr	MBpBM	$f_C(S^*)$	M_{tr}	constr	MBpBM
1	14,720	29,042	706,612	7.3	17,425	31,641	744,257	12.8
5	11,069	28,839	705,976	79.7	12,305	31,268	743,056	39.3
9	9690	27,883	702,341	429.2	10,740	30,155	738,633	54.8
10	9,423	27,637	701,168	121.2	10,491	29,905	737,406	88.0
20	7,894	26,008	690,135	80.5	8,717	28,143	725,285	104.3
30	7,051	24,983	679,640	567.4	7,731	27,109	714,318	138.7
40	6,436	24,272	669,999	182.9	7,037	26,372	703,930	113.4
50	5,941	23,688	660,138	93.9	6,518	25,813	694,355	782.7
60	5,545	23,229	651,280	38.6	6,083	25,304	684,402	171.9
70	5,215	22,815	641,971	5.7	5,711	24,883	674,846	33.3
80	4,929	22,439	632,520	4.6	5,398	24,503	665,476	5.5
90	4,684	22,147	624,079	4.5	5,128	24,164	656,067	5.7
100	4,462	21,844	615,198	4.9	4,878	23,851	646,520	5.4
200	2,918	19,731	529,883	2.7	3,132	21,623	557,661	3.1
300	1,968	18,252	449,160	2.0	2,106	20,066	473,237	2.4
400	1,303	17,000	371,790	1.5	1,398	18,725	391,820	1.7
500	821	15,812	298,029	1.3	900	17,431	314,045	1.3
600	471	14,495	223,027	1.0	530	16,040	237,199	0.9
700	244	12,962	151,916	0.7	271	14,337	161,826	0.6
800	100	10,684	81,510	0.3	100	11,781	86,772	0.7

Tables 2.4 and 2.5 summarize the computational results obtained for the largest 15 OR instances and random RW instances, correspondingly. The first three columns contain the name of instance, the number of m nodes, and the number p of medians. The next three columns are related to the running times (in seconds) for the considered above variations of our model: the initial MBpBM formulation and its modifications incorporating reductions based on bounds. The last column reflects computing times for Elloumi's NF model that we implemented and tested within the same environment as our models so that to ensure consistent comparison of performance. Note that all running times reported are times spent by a MILP solver, i.e., not including the time needed to construct a pseudo-Boolean representation. Note also that the complexities of constructing MBpBM and NF are approximately the same.

Our computational experiments with OR and RW instances can be summarized as follows. Our basic MBpBM formulation outperforms Elloumi's New Formulation in most of the tested cases, especially for larger numbers of medians p. At the same time the reduction based on bounds (see column MBpBMb) has comparatively poor performance in general. This can be explained by an increased number of nonzero coefficients in a constraint matrix (for large RW instances this formulation cannot be handled by Xpress-MP due to memory limitations). However, there exist instances (e.g., pmed36 and pmed38) for which MBpBMb performs better than other variations of the formulation based on a pBp and for the instance pmed36 has three times smaller computing times comparatively to NF. Better performance

Table 2.4 Comparison of computing times for our and Elloumi's NF formulations (15 largest OR library instances)

Instance	m	p	MBpBM	MBpBMb	MBpBMb1	Elloumi
pmed26	600	5	163.84	194.08	111.81	180.31
pmed27	600	10	27.59	41.00	21.31	43.73
pmed28	600	60	2.48	8.63	2.13	3.61
pmed29	600	120	1.78	6.50	1.31	2.91
pmed30	600	200	1.50	5.56	0.78	4.81
pmed31	700	5	153.22	132.91	57.05	90.95
pmed32	700	10	33.13	53.17	43.39	37.64
pmed33	700	70	3.09	10.11	2.69	4.73
pmed34	700	140	3.72	8.03	1.97	7.11
pmed35	800	5	70.30	233.66	154.41	514.72
pmed36	800	10	2,256.83	2,014.70	4,252.13	6,737.25
pmed37	800	80	3.91	12.61	3.08	7.00
pmed38	900	5	1,328.34	368.73	2,041.28	307.00
pmed39	900	10	572.81	713.59	444.08	473.95
pmed40	900	90	5.39	15.53	4.02	8.42

of larger models can be explained by the fact that increased number of variables and coefficients provides more options for branching and thus may lead to better pivoting of the MILP solution procedure. Finally, the revised reduction based on bounds MBpBMb1 outperformed other considered models in almost all cases except pmed32, pmed36, and pmed38 (for pmed36 it is better than NF but is worse than MBpBM). We would also like to mention one instance from TSP library [95], namely fl1400, with $p = 400$ which is unsolved in [11] and has been solved to optimality by our MBpBM in 598.5 s. Note that Beltran's et al. [18] advanced semi-Lagrangian approach based on proximal-analytic center cutting plane method has not solved the instance fl1400 with $p = 400$ to optimality as well and returns an approximation within 0.11 % in 678 s.

2.5 Instance Data Complexity

2.5.1 Data Complexity and Problem Size Reduction

Problem size reduction is a very common technique in integer programming and combinatorial optimization that can be used to find a compact representation of PMP instances. It is aimed at constructing an instance of a smaller size that is assumed to be easier to solve and provides an optimal solution to the initial instance. Moreover, it is straightforward that if the procedure of size reduction is as time-consuming as the procedure for solving the initial problem, it has no sense. These considerations lead to the following definition:

Table 2.5 Comparison of computing times for our and Elloumi's NF formulations (Resende and Werneck random instances)

Instance	m	p	MBpBM	MBpBMb	MBpBMb1	Elloumi
rw100	100	10	678.91	671.28	452.52	845.30
	100	20	4.00	6.44	2.22	5.25
	100	30	0.09	0.43	0.03	0.13
	100	40	0.08	0.34	0.02	0.14
	100	50	0.06	0.28	0.02	0.13
rw250	250	10	–	–	–	–
	250	25	–	–	–	–
	250	50	340.86	1,633.58	225.83	335.86
	250	75	1.09	8.50	0.48	2.08
	250	100	0.50	5.44	0.11	0.88
	250	125	0.66	5.02	0.27	1.38
rw500	500	10	–	–	–	–
	500	25	–	–	–	–
	500	50	–	–	–	–
	500	75	–	–	–	–
	500	100	–	–	–	–
	500	150	2.97	105.63	1.22	12.27
	500	200	2.25	54.78	0.28	4.11
	500	250	1.77	44.83	0.13	4.36
rw1000	1,000	10	–	*	–	–
	1,000	25	–	*	–	–
	1,000	50	–	*	–	–
	1,000	75	–	*	–	–
	1,000	100	–	*	–	–
	1,000	200	–	*	–	–
	1,000	300	118.91	*	13.40	234.99
	1,000	400	11.49	*	1.16	21.81
	1,000	500	9.08	*	0.77	28.47

– not solved within 1 h

$*$ not enough memory for the MILP solver

Definition 2.3. We will call instance $\mathscr{D}$ a *reduced version* of instance $\mathscr{C}$ ($\mathscr{D} = red(\mathscr{C})$) if it satisfies the following conditions:

1. $\emptyset \subset opt.solutions(\mathscr{D}) \subseteq opt.solutions(\mathscr{C})$
2. $Size(\mathscr{D}) \leq size(\mathscr{C})$
3. $\mathscr{D}$ can be obtained from $\mathscr{C}$ in polynomial time.

The first requirement guarantees that by solving $\mathscr{D}$ to optimality one immediately obtains an optimal solution to $\mathscr{C}$ (here we assume the feasibility of $\mathscr{C}$), while the second one is related to the reduction itself. Finally, the last requirement is needed to make the definition useful in practice: if for some NP-hard problem computing the reduced instance $\mathscr{D}$ is as hard as solving $\mathscr{C}$, then such a reduction is senseless. Based on this definition of a reduced instance we define complexity of the instance data in the following way:

Definition 2.4. By *complexity* of the instance data $\mathscr{C}$ (relative to a particular problem) we mean the minimum capacity of the storage needed to be able to obtain an optimal solution to the initial instance:
$comp(\mathscr{C}) = \min\{size(\mathscr{D}) : \mathscr{D} = red(\mathscr{C})\}$.

It should be noticed that without a reference to a particular problem (in our case—the PMP) this definition is meaningless. However, even when the problem is fixed, it provides neither a direct way to constructing a compact representation of the data nor even for determining the minimum required space. Further we briefly describe existing approaches to reducing the problem size and thus to obtaining upper bounds of instance data complexity.

As the cost matrix of a PMP instance has $m \times n$ elements, it is clear that this value is the most trivial upper bound for $comp(\mathscr{C})$. This value is achieved by the classical MILP representation (see [130]) of the PMP with its objective function defined as

$$\sum_{j \in J} \sum_{i \in I} c_{ij} x_{ij}. \tag{2.85}$$

Here c_{ij} denote entries of the cost matrix and x_{ij} are decision variables ($x_{ij} = 1$ if jth client is served from ith location, otherwise $x_{ij} = 0$). Cornuejols et al. [45] introduced an alternative formulation of the problem. For any client j, let K_j be the number of different distances from j to any location. It follows that $K_j \leq m$. Let $D_j^1 < D_j^2 < \ldots < D_j^{K_j}$ be these distances, sorted. For each client j it is possible to define a hierarchy of neighborhoods V_j^k such that each V_j^k is a set of locations within the distance D_j^k from client j. Naturally, in an optimal solution a client j is assigned to its neighborhood with the smallest D_j^k containing the opened location. Thus, instead of x_{ij} this formulation uses variables z_j^k such that $z_j^k = 1$ if and only if there are no opened locations in V_j^k. The objective function in this case is defined as:

$$\sum_{j \in J} \left(D_j^1 + \sum_{k=1}^{K_i - 1} (D_i^{k+1} - D_i^k) z_i^k \right). \tag{2.86}$$

Informally, this representation implies that only different elements in each column of the cost matrix are meaningful and the problem size can be reduced by storing only the pairwise different elements from each column. The further reduction is proposed in [55]. It states that if for some j, k, j', k' holds $V_j^k = V_{j'}^{k'}$, then for any feasible solution $z_j^k = z_{j'}^{k'}$ and some terms in (2.86) can be merged. Several reductions are also presented in [43], but they are similar to those described above. There are also some papers aimed at reduction of the number of constraints in the MILP formulation of the problem (see, e.g., [9, 10]); however, the number of coefficients in the objective function remains the same.

It should be noticed that most of the reduction techniques described in literature are based on a MILP formulation of the PMP and apply artificial tricks exploiting some features of the instance. On the contrary, as we showed in Sect. 2.3, a pseudo-Boolean representation itself naturally leads to several reductions that allow

obtaining better estimates of the instance data complexity and include all known reductions. (Note that due to the fact that the construction of the pBp and all its reductions can be done in polynomial time, the third condition of Definition 2.3 is satisfied.)

2.5.2 Complex Benchmark Instances

In this section we consider the aspects of constructing complex benchmark instances that can be used for testing solution algorithms and introduce our library of such instances.

To have maximum possible complexity, a PMP instance defined on an $m \times n$ cost matrix should not be amenable to any of the reductions described above. Thus, first of all, the entries of the difference matrix should be nonzero, such that all monomials in the pBp have nonzero coefficients, or, equivalently:

Claim. The most complex instances have pairwise different and nonzero entries in every column of the cost matrix (assuming that sizes of the cost matrix are fixed).

However, as explained below, these two restrictions on the entries of the difference matrix are not sufficient to ensure the complexity. Suppose, for some column j the difference between the minimal and second minimal element $\Delta c[1, j]$ is comparable to the (unknown) costs of optimal solution. In this case the location (row), at which the minimum for jth client (column) is attained, will be included into any optimal solution. Such additional structure can be exploited by the solution algorithms and thus reduces the complexity of the instance. This particular case can be generalized in the following way. Suppose, the (truncated) pBp contains a monomial $\alpha\mathscr{T} = \alpha \prod_{i \in T} y_i$ with a large enough coefficient α that exceeds the costs of the optimal solution (or its somehow computed upper bound). Then, clearly, for any optimal solution $\mathbf{y}$ holds $\mathscr{T}(\mathbf{y}) = 0$, implying that at least one of the variables in $\mathscr{T}$ must be set to zero and at least one of the corresponding locations is opened. In fact, this condition can be made even stronger if one considers not only the coefficient α at $\mathscr{T}$ but also the sum of α and coefficients of all monomials with terms embedded in $\mathscr{T}$. It is also quite straightforward that this test is more likely to fail as the range of the entries of difference matrix (or, equivalently, coefficients of the pBp) becomes smaller, up to the limit case when they all are equal. These considerations lead to the following claim.

Claim. Instances that lead to the pBp with all coefficients equal (except a constant—monomial of degree zero) are the most complex ones (assuming that the number of monomials is fixed).

Once we know how to construct a "complex" pBp, we are interested in maximizing the number of monomials in it. To achieve this, there should be no similar monomials in the pBp representation of the problem. It should be mentioned that

constants obtained from pseudo-Boolean representation of all the columns can be reduced into one monomial, so every PMP instance has a complexity of at most

$$comp(\mathscr{C}) \leq mn - (n-1) = n(m-1) + 1. \tag{2.87}$$

To ensure that only constants can be aggregated, the permutation matrix Π must conform with the following requirement: the sets of first k entries of columns Π^j in Π should be pairwise different for any $k : 1 \leq k \leq m$. This requirement can be expressed in an alternative form. Let us consider a Hasse diagram defined over the subsets of $\{1,\ldots,m\}$. It is easy to see that each permutation $\Pi^j = (\pi_{1j}, \pi_{2j}, \ldots, \pi_{mj})^{\mathrm{T}}$ corresponds to a chain of embedded subsets $\{\pi_{1j}\} \subset \{\pi_{1j}, \pi_{2j}\} \subset \ldots \subset \{\pi_{1j}, \ldots, \pi_{mj}\}$ that, in turn, corresponds to a $\emptyset - \{1,\ldots,m\}$ path in the Hasse diagram. Now the requirement can be formulated as follows:

Claim. In order to prohibit reduction of similar monomials, the permutation matrix should correspond to a collection of internally vertex-disjoint $\emptyset - \{1,\ldots,m\}$ paths in the Hasse diagram defined on subsets of $\{1,\ldots,m\}$.

Taking into account that there are at most m such paths, for PMP instances with $n > m$, it is always possible to reduce at least $n - m$ linear monomials in the pBp, so for instances in our benchmark library holds $n \leq m$. Due to these considerations it is possible to formulate the problem of constructing a permutation matrix that leads to a complex instance as a problem of finding n vertex-disjoint paths in a graph obtained from the Hasse diagram. Though this problem is known to be polynomially solvable [132], the fact that the complete Hasse diagram has 2^m vertices makes the procedure very time-consuming for large m. However, there exists a trivial solution:

$$\pi_{ij} = (i + j) \bmod m + 1. \tag{2.88}$$

In case $n = 4$, $m = 5$ this solution leads to the following permutation matrix Π:

$$\Pi = \begin{bmatrix} 3 & 4 & 5 & 1 \\ 4 & 5 & 1 & 2 \\ 5 & 1 & 2 & 3 \\ 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \end{bmatrix}.$$

Based on a representation of the monomials as a collection of chains it is possible to estimate the complexity of a PMP instance (the maximum number of monomials in the pBp). Consider a complete Hasse diagram that contains all subsets of $\{1,\ldots,m\}$. Clearly, the maximum length of a chain of embedded non-constant terms is $m-1$, as it is the maximum possible degree of the pBp. The number of chains of this maximum length is exactly $m = \binom{m}{1}$ as there exits m linear terms (as well as m terms of degree $m-1$). Each of these chains uses exactly one term of each degree from 1 to $m-1$. Once all maximum length chains are used, the next available maximum length of a chain is $m-3$ (terms of degree $2,\ldots,m-2$). The number of such chains

is $\binom{m}{2} - \binom{m}{1}$ which is exactly the number of quadratic terms that were not included in chains of length $m-1$. If we have enough columns to use all these chains (i.e., n is sufficiently large), then we switch to chains of length $m-5$ (terms of degree $3,\ldots,m-3$) and there are $\binom{m}{3} - [\binom{m}{2} - \binom{m}{1}] - \binom{m}{1} = \binom{m}{3} - \binom{m}{2}$ such chains, which is exactly the number of cubic terms not used by chains of lengths $m-2$ and $m-1$. We continue picking the longest possible chains until we have n of them. It is not hard to understand that the number of terms of some degree k ($1 \le k \le m-1$) in such a collection of n longest chains is bounded by n and, at the same time, cannot exceed $\binom{m}{k}$. Figure 2.3 gives a graphical representation of how the number of terms in such a collection of chains of maximum length can be calculated.

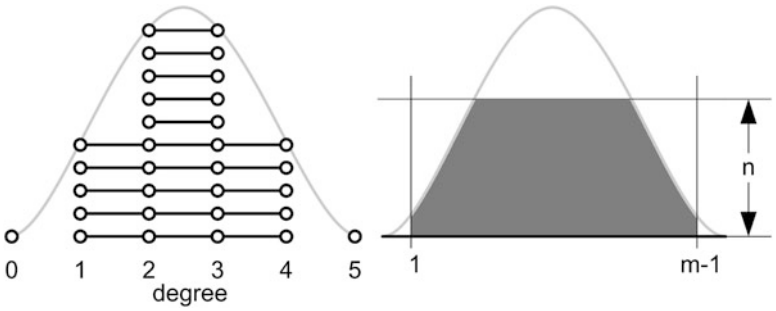

Fig. 2.3 Estimating the maximum number of nonzero terms in a pBp

In the left part of Fig. 2.3 an example for $m = 5$ is shown. Circles denote terms of a pBp that are arranged in such a way that terms of same degree are within one column. Lines correspond to possible chains of embedded terms. If one is aimed at having n chains containing the maximum number of terms, then he picks n longest chains starting from the lower part of the picture. In particular, it can be seen that it is impossible to get more than five full chains for the given example. For instance, if $n = 6$, then at least one linear monomial will be reduced. For arbitrary m and n the maximum number of monomials in the reduced pBp corresponds to the area of the shaded region in the right part of Fig. 2.3. Thus the complexity (equivalently, the number of monomials in the corresponding pBp) of a PMP instance $\mathscr{C}$ defined by an $m \times n$ cost matrix is bounded by

$$comp(\mathscr{C}) \le \sum_{i=1}^{m-1} \min\{n, \binom{m}{i}\} + 1. \tag{2.89}$$

The main peculiarity is that for a number of clients n exceeding the number of locations m addition of new clients has progressively smaller impact on the complexity of the instance that is always less than $n(m-1)+1$, while for $n \le m$ there exist instances of complexity $n(m-1)+1$.

It is easy to see that in case $n < m$ all the minima are contained in at most n rows (i.e., all minima are achieved on at most $n < m$ locations) and preprocessing will

eliminate at least $m-n$ variables (rows of the cost matrix). This leads to a conclusion that instances with $n = m$ are, potentially, the most complex ones (provided the entries of the cost matrix satisfy the considered above requirements).

Possibility of truncation of the pBp depends only on the number of medians and is not affected by the values of the cost matrix. Thus, we cannot negate this reduction by adjustment of the cost matrix and if p is fixed (2.89) can be improved in the following way:

$$comp(\mathscr{C}) \leq \sum_{i=1}^{m-p} \min\{n, \binom{m}{i}\} + 1. \tag{2.90}$$

Our benchmark library contains complex (in terms of possibility of problem size reduction) PMP instances defined on square matrices of different sizes. As costs matrices are dense, they are stored explicitly in files named "Xmatr$Y$,Z.txt," where X is "t" if the permutation matrix Π is defined by (2.88) and X is "r" if Π is a randomized permutation matrix obtained as a solution to the disjoint paths problem mentioned above. Y reflects the values of m and n (in our instances $n = m$), and costs are selected in such a way that entries of the difference matrix Δ are uniformly distributed random integers from $\{1,\ldots,Z\}$. Due to Claim 2.5.2 instances with smaller Z are harder to solve. For example, the file named "tmatr4,1.txt" defines the following instance:

$$C = \begin{bmatrix} 3 & 4 & 1 & 2 \\ 4 & 1 & 2 & 3 \\ 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \end{bmatrix}. \tag{2.91}$$

It is easy to check that the permutation matrix is the same as cost matrix C and the difference matrix has all unit entries.

The structure of the files is as follows. The first line contains the numbers of clients and potential locations (columns and rows of the cost matrix); next all entries of the cost matrix are explicitly listed row by row. The library is available at http://go.to/dkrush (under "Science"→"PMP").

We would like to finish description of our complex instances by mentioning that under certain conditions optimal objective values can be computed by a simple formula (see Lemma 2.6 below). This property is especially useful for the developers of heuristic methods and makes it possible to estimate the quality of the generated solutions.

Lemma 2.6. *If the $m \times n$ cost matrix of a PMP instance satisfies the following conditions:*

- $m = n$.
- *A permutation matrix is defined by (2.88).*
- *All entries of the difference matrix are equal to some constant d,*

then the optimal objective value can be computed as

$$d(n'+1)\left[\frac{n'p}{2} + (n \bmod p)\right], \tag{2.92}$$

where $n' = \lfloor n/p \rfloor$.

Proof. Conditions of the lemma ensure that each row (and each column) of the cost matrix can be obtained from $(d, 2d, \ldots, md)$ by a cyclic shift, i.e., each multiple of d is contained in each row exactly once. This implies that at most p clients can be served at a cost d. As well, at most p clients can be served at costs $2d$, $3d$, etc. Thus, the minimum can be obtained by serving first p clients at cost d, the next $\min\{n-p, p\}$ clients at cost $2d$, the next $\min\{n-2p, p\}$ clients at a cost $3d$, etc., until we serve all n clients. By a simple combinatorial reasoning, the total costs in this case can be computed as

$$d\left[\frac{n'(n'+1)}{2}p + (n'+1)(n \bmod p)\right] = d(n'+1)\left[\frac{n'p}{2} + (n \bmod p)\right]. \tag{2.93}$$

This minimum is achieved by the p locations (rows of the cost matrix) that fall within the following pattern:

$$\begin{array}{l} (d \quad 2d \ldots \; pd \ldots \ldots \ldots \ldots \ldots \ldots \; md) \\ (\ldots \ldots \ldots \; md \quad d \quad 2d \ldots \; pd \ldots \ldots \ldots) \\ (\ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \; md \quad d \quad 2d \ldots) \\ \ldots \end{array} \tag{2.94}$$

□

Lemma 2.6 and its constructive proof have an important corollary. It can be checked that in the instances satisfying the condition of the lemma, each location is open in p optimal solutions and the number of optimal solutions is n (does not depend on the number of medians p). This means that these instances are degenerate and may be easily solvable, irrespective of their size.

In order to check the properties of our instances we held a number of computational experiments. For the sake of comparison we used two formulations of PMP: our MBpBM and Elloumi's NF [55] (which is the most compact MILP formulation of PMP, to the best of our knowledge). Figure 2.4 shows the ranges of m and p for which the model can be loaded into the MILP solver (in our case Xpress). For different sizes of the $m \times m$ input matrix we checked for which range of p the formulation can be loaded into the MILP solver (i.e., is small enough to fit into the memory). Clearly, this range is bounded from above by the line $p = m$. As Elloumi's formulation does not account for the number of medians, there exist some critical

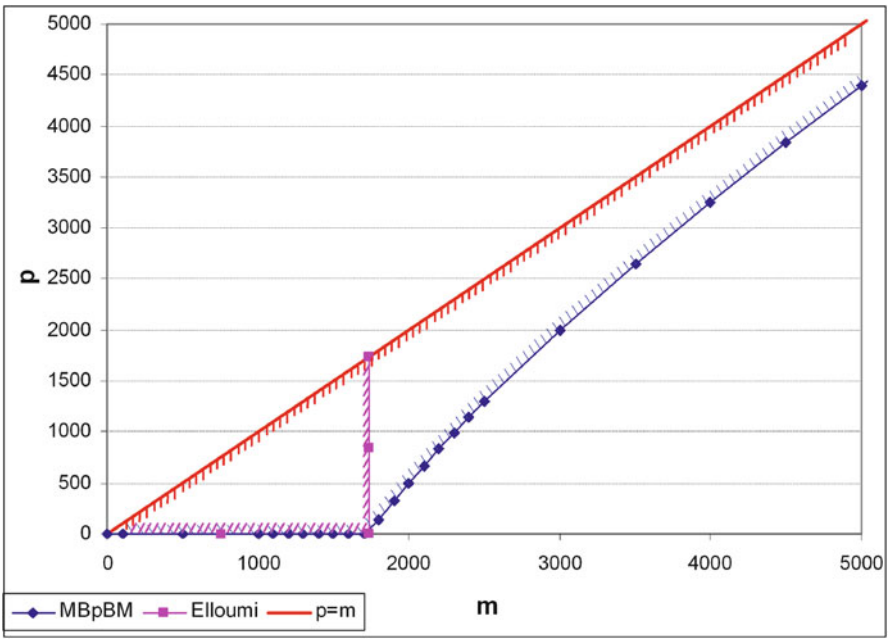

Fig. 2.4 Ranges of complex instances data size for which our MBpBM and Elloumi's NF can be loaded by Xpress

size of the cost matrix beyond which the formulation becomes prohibitively large, irrespective of p. At the same time, our formulation based on the pseudo-Boolean representation of the instance data can be loaded by a general-purpose MILP solver for some values of p even if the input matrix is of huge dimension (see Fig. 2.4).

Finally, we compared running times of two solution approaches (our MBpBM and Elloumi's NF) applied to selected OR instances and to our generated instances of the same size and with the same number of medians p. As was presumed, instances with permutations given by (2.88) are easy for the MILP solver and the running times are of the same magnitude as running times for OR instances (even though the number of coefficients in the formulation is much larger). Thus, we compared running times of OR instances and our complex instances with randomized permutation matrices and difference matrices containing all unit entries. The results of this comparison are presented in Table 2.6 and show that our benchmark instances are complex also in terms of running time. In particular, for small values of p, computation times explode even for 100×100 input matrices. Also, for the unsolved instances we compared the best found integer solutions with solutions obtained by heuristics and it was observed that heuristics produced better solutions. This contradicts the common observation that MILP solvers based on branch-and-bound procedures spend only a very small portion of the total running time on finding the optimal solution (while most of the time is spent on proving its optimality). Thus, from this point of view, our instances with randomized permutation matrices are also complex.

Table 2.6 Running times in seconds for our MBpBM and Elloumi's NF for OR instances and our complex instances of the corresponding size

		OR instances			Our instances		
m	p	MBpBM	MBpBMb	Elloumi	MBpBM	MBpBMb	Elloumi
100	5	0.22	0.20	0.25	4,434.66	3,443.13	24,684.42
	10	1.47	0.58	4.08	878.78	1,141.05	3,926.20
	20	0.11	0.06	0.14	92.95	26.25	62.94
	33	0.22	0.05	0.13	0.28	0.11	0.61
200	5	15.22	17.67	17.06	*	*	*
	10	0.73	0.55	0.77	*	*	*
	20	0.49	0.31	0.55	*	*	*
	40	0.41	0.28	0.45	1,616.33	1,218.45	1,753.47
	67	0.27	0.14	0.41	1.08	0.63	1.34
300	5	4.00	4.61	4.50	*	*	*
	10	8.59	8.33	7.36	*	*	*
	30	0.80	0.56	1.25	*	*	*
	60	1.05	1.13	2.34	*	*	*
	100	0.48	0.30	0.86	1.16	0.27	1.81
400	5	42.47	30.78	23.38	*	*	*
	10	25.16	21.19	32.02	*	*	*
	40	1.73	1.31	2.97	*	*	*
	80	0.97	0.72	1.61	*	*	*
	133	0.73	0.80	1.25	3.83	1.86	6.28
500	5	4.52	3.92	6.22	*	*	*
	10	51.63	64.05	98.59	*	*	*
	50	1.74	1.31	2.77	*	*	*
	100	1.42	0.97	2.33	*	*	*
	167	1.44	0.88	1.84	14.91	4.14	18.56
600	5	163.84	111.81	180.31	*	*	*
	10	27.59	21.31	43.73	*	*	*
	60	2.48	2.13	3.61	*	*	*
	120	1.78	1.31	2.91	*	*	*
	200	1.50	0.78	4.81	49.81	15.41	201.39
700	5	153.22	57.05	90.95	*	*	*
	10	33.13	43.39	37.64	*	*	*
	70	3.09	2.69	4.73	*	*	*
	140	3.72	1.97	7.11	*	*	*
800	5	70.30	154.41	514.72	*	*	*
	10	2,256.83	4,252.13	6,737.25	*	*	*
	80	3.91	3.08	7.00	*	*	*
900	5	1,328.34	2,041.28	1,143.97	*	*	*
	10	572.81	444.08	473.95	*	*	*
	90	5.39	4.02	8.42	*	*	*

* Not solved within 24 h

2.6 Equivalent PMP Instances

Generally speaking, there exist many different PMP instances that have the same (reduced) Hammer–Beresnev polynomial, mainly because similar monomials can be aggregated and disaggregated. Correspondingly, if two PMP instances have the same size (the same number of potential locations) and the same Hammer–Beresnev polynomial, then any solution $\mathbf{y}$ has the same objective value for both of them. This implies that a solution that is optimal for one of the instances is also optimal for the other one (provided the number of medians p are the same for both instances). This allows defining a notion of equivalence based on a pseudo-Boolean representation. Two instances of the PMP defined by cost matrices C and D are called *equivalent* if they have the same size (number of rows), the same number of medians p and $\mathscr{B}_C = \mathscr{B}_D$. The most important point here is that equivalence of two instances can be checked in time that is polynomially bounded in the size of the input cost matrix, even though the PMP is itself NP-hard. This becomes clear if one observes that a Hammer–Beresnev representation can be generated in polynomial time and contains a polynomially bounded number of monomials. However, it should be noted that such equivalence is only a sufficient condition for two PMP instances to have the same optimal solution. The following counter-example illustrates this.

Example 2.3. Consider two instances defined by costs matrices C and D:

$$C = \begin{bmatrix} 1 & 1 \\ 2 & 2 \end{bmatrix} \text{ and } D = \begin{bmatrix} 1 & 1 \\ 3 & 3 \end{bmatrix}. \tag{2.95}$$

If $p = 1$ then the Hammer–Beresnev functions $\mathscr{B}_C(\mathbf{y}) = 2 + 2y_1$ and $\mathscr{B}_D(\mathbf{y}) = 2 + 4y_1$ are different but the instances have a unique optimal solution $\mathbf{y}^* = (0,1)^{\mathrm{T}}$.

Let us consider the set of all PMP instances that are equivalent to a given instance defined by matrix C:

$$\mathscr{P}_{C,p} = \left\{ D \in \mathbb{R}_+^{mn} : \mathscr{B}_{D,p} = \mathscr{B}_{C,p} \right\}. \tag{2.96}$$

This set can be defined as

$$\mathscr{P}_{C,p} = \bigcup_{\Pi \in PERM(C)} P_{C,\Pi,p}\,, \tag{2.97}$$

where

$$P_{C,\Pi,p} = \left\{ D \in \mathbb{R}_+^{mn} : \mathscr{B}_{C,p} = \mathscr{B}_{D,\Pi,p} \right\}, \tag{2.98}$$

and $PERM(C)$ is a set of all $m \times n$ permutation matrices that can be induced by $\mathscr{B}_{C,p}$. Any set of embedded terms of Hammer–Beresnev polynomial defines at least one permutation of $\{1,\ldots,m\}$ that can be viewed as some column of permutation matrix. We say that a permutation matrix Π of equivalent instance is induced by $\mathscr{B}_{C,p}$ if it has the same size as C, each its column is defined by some set of embedded terms of $\mathscr{B}_{C,p}$ and each term is used in generation of some column of Pi.

Let us now consider the set $PERM(C)$. It should be noted that $perm(C) \subseteq PERM(C)$ and having fixed some permutation $\Pi \in perm(C)$ all other elements of this set can be obtained by application of operations from a finite set to it. First such operation is a permutation of the columns of Π—it is clear that it does not affect the polynomial. Second operation is a permutation of such elements i and k from some column j for which holds $c_{\pi_{ij}j} = c_{\pi_{kj}j}$. It is easy to check that all terms containing either of variables $y_{\pi_{ij}}$ and $y_{\pi_{kj}}$ (but not both) have zero coefficients. In order to provide the following two operations, it is useful to introduce matrix representation of the Hammer–Beresnev polynomial: each such polynomial can be represented as an $m \times n$ matrix, every entry of which corresponds to some monomial of the Hammer–Beresnev polynomial (possibly with a zero coefficient). Moreover, the following restrictions take place:

(i) Every row i contains monomials of ith degree.
(ii) Monomials within each column have embedded terms.

Having a polynomial and a fixed permutation it is possible to restore the matrix representation (the inverse is also true—having a matrix representation it is possible to restore the polynomial and some permutation(s)). For example,

$$\mathscr{B}_C(\mathbf{y}) = 7 + 5y_1 + 4y_2 + 4y_1y_2 \text{ and } \Pi = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 3 \end{bmatrix} \tag{2.99}$$

lead to many optional matrix representations, for example,

$$\begin{bmatrix} 7 & 0 \\ 5y_1 & 4y_2 \\ 3y_1y_2 & 1y_1y_2 \end{bmatrix}, \begin{bmatrix} 3 & 4 \\ 5y_1 & 4y_2 \\ 0y_1y_2 & 4y_1y_2 \end{bmatrix}. \tag{2.100}$$

It can be seen from this example that by interchanging the entries in the second row the restrictions (i) and (ii) on the matrix representation of the polynomial are not violated. At the same time, this leads to a different permutation matrix:

$$\begin{array}{cc} \text{polynomial} & \text{permutation} \\ \begin{bmatrix} 3 & 4 \\ 5y_1 & 4y_2 \\ 3y_1y_2 & 1y_1y_2 \\ 4y_1y_2y_3 & 5y_1y_2y_4 \end{bmatrix} & \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} \\ \downarrow & \downarrow \\ \begin{bmatrix} 3 & 4 \\ 4y_2 & 5y_1 \\ 3y_1y_2 & 1y_1y_2 \\ 4y_1y_2y_3 & 5y_1y_2y_4 \end{bmatrix} & \begin{bmatrix} 2 & 1 \\ 1 & 2 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} \end{array}. \tag{2.101}$$

Thus, the third operation is permutation of the entries within one row such that the embedded structure of the matrix is preserved. The fourth operation is reduction of

similar monomials; it leads to an increase in the number of zero coefficients in the matrix representation and thus provides more opportunities for application of the second operation:

$$\begin{array}{cc} \text{polynomial} & \text{permutations} \\ \begin{bmatrix} 3 & 4 \\ 5y_1 & 4y_2 \\ 3y_1y_2 & 1y_1y_2 \\ 4y_1y_2y_3 & 5y_1y_2y_4 \end{bmatrix} & \begin{bmatrix} 1\ 2 \\ 2\ 1 \\ 3\ 4 \\ 4\ 3 \end{bmatrix} \\ \downarrow & \downarrow \\ \begin{bmatrix} 3 & 4 \\ 5y_1 & 4y_2 \\ 4y_1y_2 & 0y_1y_2 \\ 4y_1y_2y_3 & 5y_1y_2y_4 \end{bmatrix} & \begin{bmatrix} 1\ 2 \\ 2\ 1 \\ 3\ 4 \\ 4\ 3 \end{bmatrix}, \begin{bmatrix} 1\ 1 \\ 2\ 4 \\ 3\ 2 \\ 4\ 3 \end{bmatrix}. \end{array} \tag{2.102}$$

Similarly to [4], we show that the set $P_{C,\Pi}$ can be described by a system of linear inequalities.

Let us assume that $\Pi, \Psi \in perm(C)$. The choice of the particular Π and Ψ is unimportant since the truncated Hammer–Beresnev polynomials for all permutations within $perm(\cdot)$ are identical (see [5]). The truncated Hammer–Beresnev polynomial for C is

$$\begin{aligned} \mathscr{B}_{C,p}(\mathbf{y}) = &\sum_{j=1}^{n} \Delta c[1,j] + \sum_{j=1}^{n} \Delta c[2,j] y_{\pi_{1j}} + \\ &\sum_{k=3}^{m-p} \sum_{j=1}^{n} \Delta c[k,j] \prod_{r=1}^{k} y_{\pi_{rj}}, \end{aligned} \tag{2.103}$$

while that for D is

$$\begin{aligned} \mathscr{B}_{D,p}(\mathbf{y}) = &\sum_{j=1}^{n} \Delta d[1,j] + \sum_{j=1}^{n} \Delta d[2,j] y_{\psi_{1j}} + \\ &\sum_{k=3}^{m-p} \sum_{j=1}^{n} \Delta d[k,j] \prod_{r=1}^{k} y_{\psi_{rj}}. \end{aligned} \tag{2.104}$$

For the PMP defined on D to be equivalent to the PMP defined on C, $\mathscr{B}_{D,p}(\mathbf{y})$ has to be equal to $\mathscr{B}_{C,p}(\mathbf{y})$. By equating similar monomials in $\mathscr{B}_{C,p}(\mathbf{y})$ and $\mathscr{B}_{D,p}(\mathbf{y})$, we see that for equivalence entries in D have to satisfy the following equations.

Equating the coefficients of the constant and linear monomials in (2.103) and (2.104) yields

$$\sum_{j=1}^{n} \Delta d[1,j] = \sum_{j=1}^{n} \Delta c[1,j], \tag{2.105}$$

$$\sum_{j:\psi_{1j}=k} \Delta d[2,j] = \sum_{j:\pi_{1j}=k} \Delta c[2,j] \quad k = 1,\ldots,m-p. \tag{2.106}$$

By equating the coefficients of non-linear monomials we get the equations

$$\sum_{\{\psi_{1j},\ldots,\psi_{kj}\}=\{\pi_{1j},\ldots,\pi_{kj}\}} \Delta d[k,j]-\Delta c[k,j]=0 \quad k=3,\ldots,m-p;\ j=1,\ldots,n. \tag{2.107}$$

Finally, since $\Pi \in perm(C)$ and $\Psi \in perm(D)$ and since all entries in the instances are assumed to be nonnegative, we have that

$$\Delta d[k,j] \geq 0 \qquad k=1,\ldots m-p;\ j=1,\ldots,n, \tag{2.108}$$
$$d_{ij} \geq 0 \qquad i=1,\ldots,m;\ j=1,\ldots,n, \tag{2.109}$$

Consider the instance in (2.18) with $p=1$ and $\mathscr{B}_C(\mathbf{y}) = \mathscr{B}_{C,p=1} = 8+0y_1+1y_2+2y_4+0y_1y_2+1y_1y_3+1y_2y_3+1y_2y_4+1y_3y_4+7y_1y_2y_3+1y_1y_3y_4+5y_2y_3y_4$. Then P_{C,Π_1} (where Π_1 is given by (2.19)) is defined by the following system. Note that by adding a specific permutation we just specify the names of entries in an equivalent matrix.

Equations corresponding to constants (2.105):

$$d_{11}+d_{32}+d_{23}+d_{14}+d_{45}=33. \tag{2.110}$$

Equations corresponding to linear monomials (2.106):

$$\begin{aligned}
y_1: &\quad (d_{31}-d_{11})+(d_{24}-d_{14})=0,\\
y_2: &\quad (d_{32}-d_{22})=1,\\
y_3: &\quad (d_{42}-d_{32})=0,\\
y_4: &\quad (d_{23}-d_{43})+(d_{35}-d_{45})=2.
\end{aligned}$$

Equations corresponding to non-linear monomials (2.107):

$$\begin{aligned}
y_1y_2: &\quad (d_{34}-d_{24})=0,\\
y_1y_3: &\quad (d_{21}-d_{31})=1,\\
y_2y_3: &\quad (d_{42}-d_{32})=1,\\
y_2y_4: &\quad (d_{33}-d_{23})=1,\\
y_3y_4: &\quad (d_{15}-d_{35})=1,\\
y_1y_2y_3: &\quad (d_{41}-d_{21})+(d_{44}-d_{34})=7,\\
y_1y_3y_4: &\quad (d_{25}-d_{15})=1,\\
y_2y_3y_4: &\quad (d_{12}-d_{42})+(d_{13}-d_{33})=5.
\end{aligned}$$

Inequalities corresponding to nonnegativity of differences (2.108):

$$\begin{aligned}
&d_{31}-d_{11}, d_{24}-d_{14}, d_{32}-d_{22}, d_{23}-d_{43}, d_{35}-d_{45} \geq 0,\\
&d_{34}-d_{24}, d_{21}-d_{31}, d_{42}-d_{32}, d_{33}-d_{23}, d_{15}-d_{35} \geq 0,\\
&d_{41}-d_{21}, d_{44}-d_{34}, d_{25}-d_{15}, d_{12}-d_{42}, d_{13}-d_{33} \geq 0.
\end{aligned}$$

Inequalities corresponding to nonnegativity of costs (2.109):

$$d_{11}, d_{12}, \ldots, d_{44}, d_{45} \geq 0. \tag{2.111}$$

For $p = 2$ we have that $\mathscr{B}_{C,p=2} = 8 + 0y_1 + 1y_2 + 2y_4 + 0y_1y_2 + 1y_1y_3 + 1y_2y_3 + 1y_2y_4 + 1y_3y_4 +$, and all equations corresponding to cubic terms will be replaced by

$$\begin{aligned} y_1y_2y_3: &\quad (d_{41} - d_{21}) + (d_{44} - d_{34}) = 0, \\ y_1y_3y_4: &\quad (d_{25} - d_{15}) = 0, \\ y_2y_3y_4: &\quad (d_{12} - d_{42}) + (d_{13} - d_{33}) = 0. \end{aligned}$$

Hence, given a cost matrix C, any solution D to the set of inequalities (2.105)–(2.109) will be a matrix for an equivalent instance.

Inequalities (2.105)–(2.109) define a family of polyhedra in $\mathbb{R}_+^{mn}$ (each polyhedron in the family corresponds to some permutation $\Pi_i \in perm(C)$). Further, we show that any two such polyhedra (2.98) either have an empty intersection or coincide.

From the following example it can be seen that in the simplest case, when each coefficient in the Hammer–Beresnev polynomial is defined by the costs of serving only one client (by entries of only one column in the cost matrix), the claimed property can be verified by considering only (2.105)–(2.107). The polynomial is presented in a matrix form, such that each column of the matrix contains monomials with embedded terms.

Let us consider two representations of the same polynomial,

$$\begin{bmatrix} 7 & 0 \\ 5y_1 & 4y_2 \\ 4y_1y_2 & 0y_1y_2 \end{bmatrix} \begin{bmatrix} 7 & 0 \\ 4y_2 & 5y_1 \\ 4y_1y_2 & 0y_1y_2 \end{bmatrix}, \tag{2.112}$$

their corresponding cost matrices

$$\begin{bmatrix} 7 & 4 \\ 12 & 0 \\ 16 & 4 \end{bmatrix} \qquad \begin{bmatrix} 11 & 0 \\ 7 & 5 \\ 15 & 5 \end{bmatrix} \tag{2.113}$$

and permutations

$$\begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 3 \end{bmatrix} \qquad \begin{bmatrix} 2 & 1 \\ 1 & 2 \\ 3 & 3 \end{bmatrix}. \tag{2.114}$$

Then, equations describing polyhedra for the two cases are

$$\begin{array}{rr} const.: d_{11}+d_{22}=7, & const.: d_{21}+d_{12}=7, \\ y_1: d_{21}-d_{11}=5, & y_2: d_{11}-d_{21}=4, \\ y_2: d_{12}-d_{22}=4, & y_1: d_{22}-d_{12}=5, \\ y_1y_2: d_{31}-d_{21}=4, & y_1y_2: d_{31}-d_{11}=4, \end{array} \tag{2.115}$$

It can be seen that equations in the second line (in the third, as well) contradict each other, so the two polyhedra have empty intersection. On the other hand, the only way to eliminate this contradiction is to put r.h.s. of these equations to 0. However, in this case we will have two equivalent systems of equations:

$$\begin{array}{ll} d_{11}+d_{22}=7, & d_{21}+d_{12}=7, \\ d_{21}-d_{11}=0, & d_{11}-d_{21}=0, \\ d_{12}-d_{22}=0, & d_{22}-d_{12}=0, \\ d_{31}-d_{21}=4, & d_{31}-d_{11}=4. \end{array} \tag{2.116}$$

In a more general case, however, by considering only the equations it is not possible to prove the claimed property of the polyhedra induced by the Hammer–Beresnev polynomial. The following counter-example illustrates this point.

Consider the two following matrix representations of the same polynomial:

$$\begin{bmatrix} 7 & 0 & 0 \\ 5y_1 & 4y_2 & 0y_1 \\ 4y_1y_2 & 0y_1y_2 & 2y_1y_3 \end{bmatrix} \begin{bmatrix} 7 & 0 & 0 \\ 4y_2 & 5y_1 & 0y_1 \\ 4y_1y_2 & 0y_1y_2 & 2y_1y_3 \end{bmatrix}, \tag{2.117}$$

their corresponding cost matrices

$$\begin{bmatrix} 7\ 4\ 0 \\ 12\ 0\ 2 \\ 16\ 4\ 0 \end{bmatrix} \qquad \begin{bmatrix} 11\ 0\ 0 \\ 7\ 5\ 2 \\ 15\ 5\ 0 \end{bmatrix} \tag{2.118}$$

and permutations

$$\begin{bmatrix} 1\ 2\ 1 \\ 2\ 1\ 3 \\ 3\ 3\ 2 \end{bmatrix} \qquad \begin{bmatrix} 2\ 1\ 1 \\ 1\ 2\ 3 \\ 3\ 3\ 2 \end{bmatrix}. \tag{2.119}$$

Then, equations describing polyhedra for the two cases are

$$\begin{array}{rr} const.: d_{11}+d_{22}+d_{13}=7, & const.: d_{21}+d_{12}+d_{13}=7, \\ y_1: d_{21}-d_{11}+d_{33}-d_{13}=5, & y_1: d_{22}-d_{12}+d_{33}-d_{13}=5, \\ y_2: d_{12}-d_{22}=4, & y_2: d_{11}-d_{21}=4, \\ y_1y_2: d_{31}-d_{21}+d_{32}-d_{12}=4, & y_1y_2: d_{31}-d_{11}+d_{32}-d_{22}=4, \\ y_1y_3: d_{23}-d_{33}=2, & y_1y_3: d_{23}-d_{33}=2, \end{array} \tag{2.120}$$

There are no contradicting equations in the two systems and it can be verified that they have common solutions. Thus, if only equalities are considered, the polyhedra can intersect in general case, but, if constraints on non-negativity (2.108)–(2.109) are added, it is possible to formulate the following lemma.

Lemma 2.7. *Polyhedra induced by different permutation matrices either do not intersect or coincide.*

Proof. Suppose, for some column j it is possible to change its permutation from $\pi_1 = (\cdots, i, k, \cdots)^{\mathrm{T}}$ to $\pi_2 = (\cdots, k, i, \cdots)^{\mathrm{T}}$, so that only two adjacent entries are interchanged. The first permutation will lead to the following inequality (among the others):

$$d_{i,j} - d_{k,j} \geq 0. \tag{2.121}$$

Similarly, the second permutation leads to a system that includes

$$d_{k,j} - d_{i,j} \geq 0. \tag{2.122}$$

It is straightforward that the polyhedra intersect only if $d_{k,j} = d_{i,j}$, but in this case the systems of equations that correspond to the two cases (permutations, matrices) are equivalent. □

2.6.1 Dimensions of PMP Equivalence Polyhedra

As mentioned above (2.97), the union $\mathscr{P}_C$ of polyhedra corresponding to possible permutations of the cost matrix C describes all equivalent PMP instances. Once the equivalence relation is established, it is natural to estimate the dimension of equivalent data (dimension of $\mathscr{P}_C$). In order to do that, we introduce additional notations. Let us denote the system of (2.105)–(2.107) by $\mathscr{E}$. In turn, $\mathscr{E}$ can be presented in a matrix form as $\mathbf{A} \cdot d = b$, where $\mathbf{A}$—$mn \times N$ matrix, d—vector of entries of the cost matrix (variables), $d = (d_{11}, d_{21}, \ldots, d_{mn})^{\mathrm{T}}$, and N—the number of equations in (2.105)–(2.107). Under these notations it is possible to formulate the following properties of $\mathscr{E}$ (we assume that $\mathbf{A}$ has at least two rows, i.e., $m \geq 2$).

Observation 1 *For any two equations eq_1 and eq_2 from $\mathscr{E}$ there exist two variables d_{ij} and d_{kl}, such that*

$$d_{ij} \in eq_1, d_{ij} \notin eq_2, \tag{2.123}$$
$$d_{kl} \in eq_2, d_{kl} \notin eq_1. \tag{2.124}$$

Observation 2 *All coefficients in the equations from $\mathscr{E}$ belong to $\{-1, 1\}$.*

Observation 3 *There exist variables $d_{mi}, i = 1, \ldots, n$ that are included in exactly one equation with coefficient +1. All the other variables are included in exactly two equations with coefficients +1 and −1, correspondingly.*

The latter becomes clear if one considers equations corresponding to coefficients of a chain of embedded terms of the Hammer–Beresnev polynomial.

Observations 1 and 2 have an important consequence formalized in the following lemma.

Lemma 2.8. *Constraint matrix* $\mathbf{A}$ *describing the polyhedron of equivalent instances is totally unimodular. The transposed minor of A excluding the first row (the one corresponding to constraints (2.105)) is also totally unimodular if all subpermutations in* Π *are different, i.e., if the pseudo-Boolean representation of the instance has no similar monomials.*

Proof. The first statement can be easily verified by observing that every column of $\mathbf{A}$ has at most two nonzero entries and they the opposite sign. Together with the statement of Observation 2 this matches the well-known sufficient conditions for total unimodularity (see, e.g., [120], Theorem 13.3).

In order to prove the second statement, observe that absence of similar monomials implies that each coefficient in the Hammer–Beresnev polynomial (except the constant) is uniquely defined as a difference of two entries of the cost matrix. This means that each row of $\mathbf{A}$ (correspondingly, each column of $\mathbf{A}^{\mathrm{T}}$ except the first one) has exactly two nonzero entries of the opposite sign. Thus, the sufficient conditions used in the first part of the proof are satisfied. □

Lemma 2.8 implies that the polyhedron of equivalent instances has only integral vertices and that each cost matrix within an equivalence class can be represented as a conic combination of some integer-valued cost matrices from the same equivalence class.

The following observations reflect some additional properties of the constraints defining the equivalence polyhedron.

Observation 4 *If some two equations* eq_1, eq_2 *from* $\mathscr{E}$ *contain the same variable* d_{ij}*, then exactly one of the following holds* (λ_1, λ_2*—some constants):*

- $d_{ij} \in \lambda_1 eq_1 + \lambda_2 eq_2$ *and there exists no other equation* $eq_3 \in \mathscr{E}$ *that contains* d_{ij}.
- $d_{ij} \notin \lambda_1 eq_1 + \lambda_2 eq_2$ *and there exists no other equation* $eq_3 \in \mathscr{E}$ *that contains* d_{ij}.

Observation 5 *(Elimination of variables) If in a linear combination* $\lambda_1 eq_1 + \lambda_2 eq_2$ *of two equations from* $\mathscr{E}$ *some variable* d_{ij} *(that was present in both of them) is eliminated, then this combination contains* $d_{(i+1)j}$ *and* $d_{(i-1)j}$*, and no other equation from* $\mathscr{E}$ *contains both these variables.*

Lemma 2.9. *The system of equations* $\mathscr{E}$ *is linearly independent.*

Proof. By definition, the system is linearly dependent if there exist such constants $\lambda_1, \ldots, \lambda_N$ ($\sum_i |\lambda_i| > 0$) that $\mathscr{L} \equiv \lambda_1 eq_1 + \lambda_2 eq_2 + \ldots + \lambda_N eq_N = 0$, or, in other words, coefficients at all variables in $\mathscr{L}$ are zero. Let us show that such set of constants $\lambda_i, i = 1 \ldots N$ does not exist by induction on the number of equations in $\mathscr{L}$.

For $N = 1$—trivially, in each equation there are variables with nonzero coefficients (see Observation 2).

Suppose, for some $N > 1$ holds $\mathscr{L} = \lambda_1 eq_1 + \lambda_2 eq_2 + \ldots + \lambda_N eq_N \neq 0$, this means that $\mathscr{L}$ contains at least one variable with a nonzero coefficient. Let us show that by adding one more equation to $\mathscr{L}$ one will not obtain 0. If $\mathscr{L}$ and eq_{N+1} do not have common variables, then for any λ_{N+1} holds

$$\mathscr{L} + \lambda_{N+1} eq_{N+1} \neq 0. \tag{2.125}$$

If $\mathscr{L}$ and eq_{N+1} have some common variable d_{ij}, then exactly one of the following cases takes place:

- $d_{(i+1)j} \in \mathscr{L},\ d_{(i+1)j} \notin eq_{N+1}$
- $d_{(i+1)j} \in eq_{N+1}, d_{(i+1)j} \notin \mathscr{L}$
- $d_{(i+1)j} \notin eq_{N+1},\ d_{(i+1)j} \notin \mathscr{L}$

In the first two cases we have a variable that cannot be eliminated by adding $\lambda_{N+1} eq_{N+1}$ to $\mathscr{L}$. The third case implies that $d_{(i+1)j}$ was eliminated in $\mathscr{L}$ and according to Observation 5 there exist some d_{kj}, $k > i$ that is in $\mathscr{L}$ and not in eq_{N+1}. This finishes the proof. □

As all equations in $\mathscr{E}$ are linearly independent, then $\mathbf{A}$ has the maximum possible rank that is equal to the number of rows in it, i.e., $rank(\mathbf{A}) = N$. If one now denotes by $|T|$ the number of monomials in the initial Hammer–Beresnev polynomial (equal to the number of nonzero entries in the difference matrix Δ) and by $|B|$—the number of terms in the reduced polynomial (with combined similar monomials), then the following bounds on $rank(\mathbf{A})$ take place.

Lemma 2.10.

$$rank(\mathbf{A}) \geq |B|. \tag{2.126}$$

$$rank(\mathbf{A}) \leq |B| + mn - |T|. \tag{2.127}$$

Proof. As every term with a nonzero coefficient corresponds to an equation in $\mathscr{E}$, then the number of equations cannot be smaller than $|B|$. However, equations for some terms of the polynomial with zero coefficients are to be included into the system as their corresponding zero entries existed in the difference matrix. So, the upper bound is obtained by adding the quantity of zeros induced by equal entries in the columns of the cost matrix. □

It should be noted that for a PMP instance defined on a cost matrix D to be equivalent to a PMP instance defined on a cost matrix C, $perm(D)$ has to be identical to $perm(C)$. Note that a choice among many equivalent matrices might be refined by adding either additional constraints reflecting some sufficient conditions for the polynomially solvable special case as well as some additional requirements to the entries included in, for the sake of simplicity, a linear objective function defined on $P_{C,p}$.

The problem of finding an equivalent matrix D with the minimum number of columns to the given matrix C can be solved using the following well-known Dilworth's decomposition theorem (see, e.g., [136], Theorem 14.2):

The set of terms T_a with positive coefficients in a pseudo-Boolean polynomial are subsets of partially ordered set T, and hence, the minimum number of chains covering T_a (nothing else as the minimum number of aggregated columns of C) is equal to the maximum size of an antichain (the maximum number of non-embedded terms).

The maximum size of an antichain found for matrix (2.18) is equal to four, and if the permutation matrix Π_E is chosen to be

$$\Pi_E = \begin{bmatrix} 1 & 2 & 4 & 3 \\ 3 & 3 & 2 & 4 \\ 2 & 1 & 1 & 1 \\ 4 & 4 & 3 & 2 \end{bmatrix}, \tag{2.128}$$

then the corresponding Hammer–Beresnev polynomial $\mathscr{B}_{C,p=2}(\mathbf{y}) = [1 + 0y_1 + 1y_1y_3 + 2y_1y_2y_3] + [1 + 1y_2 + 1y_2y_3 + 3y_2y_3y_4] + [1 + 1y_4 + 1y_2y_4 + 2y_2y_3y_4] + [3 + 0y_1 + 0y_1y_2 + 5y_1y_2y_3] + [2 + 1y_4 + 1y_3y_4 + 1y_1y_3y_4]$ yields the following (in)equalities. Equations corresponding to constants (2.105):

$$e_{11} + e_{22} + e_{43} + e_{34} = 8. \tag{2.129}$$

Equations corresponding to linear monomials (2.106):

$$\begin{aligned} y_1: &\quad (e_{31} - e_{11}) = 0, \\ y_2: &\quad (e_{32} - e_{22}) = 1, \\ y_3: &\quad (e_{44} - e_{34}) = 0, \\ y_4: &\quad (e_{23} - e_{43}) = 2. \end{aligned}$$

Equations corresponding to non-linear monomials (2.107):

$$\begin{aligned} y_1y_3: &\quad (e_{21} - e_{31}) = 1, \\ y_2y_3: &\quad (e_{12} - e_{32}) = 1, \\ y_2y_4: &\quad (e_{13} - e_{23}) = 1, \\ y_3y_4: &\quad (e_{14} - e_{44}) = 1, \\ y_1y_2y_3: &\quad (e_{41} - e_{21}) + (e_{42} - e_{12}) = 0, \\ y_1y_2y_4: &\quad (e_{33} - e_{13}) = 0, \\ y_1y_3y_4: &\quad (e_{24} - e_{14}) = 0. \end{aligned}$$

Nonnegativity inequalities corresponding to (2.108):

$$\begin{aligned} &e_{31} - e_{11}, e_{32} - e_{22}, e_{44} - e_{34} \geq 0, \\ &e_{23} - e_{43}, e_{21} - e_{31}, e_{12} - e_{32} \geq 0, \\ &e_{13} - e_{23}, e_{14} - e_{44}, e_{41} - e_{21} \geq 0, \\ &e_{42} - e_{12}, e_{33} - e_{13}, e_{24} - e_{14} \geq 0. \end{aligned}$$

Nonnegativity inequalities corresponding to (2.109):

$$e_{11}, e_{12}, \ldots, e_{43}, e_{44} \geq 0. \tag{2.130}$$

Finally, the reconstructed costs matrix E that is equivalent to matrix C (2.18) is

$$E = \begin{bmatrix} 8 & 2 & 3 & 1 \\ 9 & 0 & 2 & 1 \\ 8 & 1 & 3 & 0 \\ 9 & 2 & 0 & 0 \end{bmatrix}. \tag{2.131}$$

Note that the equivalence relation defined in this chapter for the PMP can be extended to other problems modelled via the PMP. For example, in Chap. 3 we show that the CFP can be modelled via the PMP. This implies that one can define equivalent CFP instances as those leading to equivalent PMP instances.

2.7 Summary and Future Research Directions

This chapter presents a new approach to formulation of models for the PMP. We first formulate the PMP using a pBp as the objective function and with just one constraint related to the number of medians in a feasible solution keeping all decision variables Boolean. We then reduce the size of the objective function by truncation and reducing similar monomials. After that we linearize all non-linear terms in the objective function with additional variables and linear constraints. The resulting model that we call MBpBM is within the well-studied class of mixed Boolean linear programming models. The number of nonzero coefficients in the objective function of MBpBM is minimal compared to all previously published models for the PMP. Since the number of Boolean decision variables is equal to m and cannot be further reduced (except by well-known reduction tests finding some open and/or closed sites; see, e.g., [4, 11, 68] and references within) and the number of linear constraints is equal to the number of nonnegative variables, one may conclude that the MBpBM has the smallest number of constraints related to the nonnegative variables. As we have shown, the matrix of constraints related to the nonnegative decision variables is as sparse as possible if we would be able to solve a generalization of the classical set covering problem defined on the set of all terms involved in the pseudo-Boolean formulation of PMP. Unfortunately, this set covering problem is NP-hard [60]. As shown by our computational experiments for a PMP instance with $m = n = 1,000$ the corresponding ground set of covering problem might be in magnitudes larger. From the other side, even if we might be able to find the most sparse matrix of constraints, then the created MBpBM is still in the class of mixed Boolean linear programming models which are computationally intractable (NP-hard). Anyway, if we evaluate the optimality of a model within the class of mixed Boolean linear programming models by the smallest number of nonnegative

variables and corresponding constraints, our MBpBM is an optimal one and PMP instance specific!

The main distinction between MBpBM and all the well-known mixed Boolean PMP formulations is that the number of non-negative variables in MBpBM is automatically adjusted according to the number p of medians, i.e., the number of non-negative variables as well as the number of constraints decrease linearly with increasing values of p. This feature of MBpBM implies that PMP instances with relatively large numbers of medians are easier to solve using standard MILP software applied with our MBpBM. Thus, our models (MBpBM, MBpBMb, and MBpBMb1) and pseudo-Boolean approach to their creation do not only extend the capabilities of general-purpose software in solving larger sized PMPs but also make it possible to solve smaller problems more efficiently while using general-purpose MILP software.

The MBpBM allows either to solve much larger problems by general-purpose MILP software than what is possible using previous model formulations or to speed up essentially the best-known models for the PMP. In sharp contrast, CPLEX is unable to solve the 15 largest OR test problems (see Table 2.1) in classical formulation of PMP (see [11, 18]).

The MBpBM approach may also lead to reduced model sizes for other location models like the SPLP and the capacitated SPLP (a generalization of PMP with fixed charges) as well as to improve data correcting approach to the SPLP (see [61, 68]). Together with the MBpBM we have introduced two variations of MBpBM, namely MBpBMb and MBpBMb1. The MBpBMb includes preprocessing of monomials and corresponding linear constraints based on a lower bound to a subproblem of MBpBM induced by a subspace determined by the term of the corresponding monomial in pBp. The MBpBMb1 is based on further reductions of monomials in pBp induced by a monomial with zero coefficient and embedded in all terms with higher degrees.

Computational results reported in Tables 2.4–2.5 show that MBpBMb1 outperforms the best available MILP Elloumi's model for all OR library instances except pmed38. Our MBpBM allows solving PMPs with much less execution times than required by the best-known models in the literature. It also allows solving much larger problems by general-purpose MILP software than is currently possible using previous model formulations. The MBpBM has been able to obtain an optimal solution to the fl1400 instance with $p = 400$, which remained unsolved in [11] by their state-of-the-art algorithm for PMP as well as by Beltran's et al. advanced semi-Lagrangian approach based on proximal-analytic center cutting plane method [18]. Our models MBpBM, MBpBMb, MBpBMb1 are computationally more efficient than all available in the literature MILP formulations of PMP and outperform corresponding state-of-the-art algorithms available in the literature; see [11, 18, 27, 43–45, 55, 139].

Computational results reported in Table 2.3 show that our MBpBM will be useful for p-median approach applicable to large cell formation instances in group technology such that the optimal number p of cells can be found (see, e.g., [160]). To

summarize, in this chapter we have shown that our model extends the ability to solve large-scale PMP instances to optimality by means of general-purpose software, e.g., Xpress-MP.

With regard to the main subject of this book—the cell formation problem—one may conclude that application of PMP to solving the cell formation problem is beneficial as the former can be solved to optimality very efficiently using the introduced MBpBM formulation. Taking into account that the size of the problem is quite limited (e.g., 100 machines are already too many for a typical manufacturing system) one may expect tiny solution times and this expectation will be verified in the following chapter.

http://www.springer.com/978-1-4614-8001-3

Cell Formation in Industrial Engineering
Theory, Algorithms and Experiments
Goldengorin, B.; Krushinsky, D.; Pardalos, P.M.
2013, XIV, 206 p., Hardcover
ISBN: 978-1-4614-8001-3