algorithms

MDPI

Article

A Primal-Dual Interior-Point Method for Facility Layout Problem with Relative-Positioning Constraints

Shunichi Ohmori [1,*] and Kazuho Yoshimoto [2]

[1] Graduate School of Creative Science and Engineering, Waseda University, Tokyo 169-8050, Japan
[2] School of Creative Science and Engineering, Waseda University, Tokyo 169-8050, Japan; kazuho@waseda.jp
* Correspondence: ohmori0406@aoni.waseda.jp

Abstract: We consider the facility layout problem (FLP) in which we find the arrangements of departments with the smallest material handling cost that can be expressed as the product of distance times flows between departments. It is known that FLP can be formulated as a linear programming problem if the relative positioning of departments is specified, and, thus, can be solved to optimality. In this paper, we describe a custom interior-point algorithm for solving FLP with relative positioning constraints (FLPRC) that is much faster than the standard methods used in the general-purpose solver. We build a compact formation of FLPRC and its duals, which enables us to establish the optimal condition very quickly. We use this optimality condition to implement the primal-dual interior-point method with an efficient Newton step computation that exploit special structure of a Hessian. We confirm effectiveness of our proposed model through applications to several well-known benchmark data sets. Our algorithm shows much faster speed for finding the optimal solution.

Keywords: facility layout problem; interior-point method; convex optimization

Citation: Ohmori, S.; Yoshimoto, K. A Primal-Dual Interior-Point Method for Facility Layout Problem with Relative-Positioning Constraints. *Algorithms* **2021**, *14*, 60. https://doi.org/10.3390/a14020060

Academic Editor: Akemi Galvez Tomida
Received: 29 December 2020
Accepted: 6 February 2021
Published: 13 February 2021

1. Introduction

1.1. Facility Layout Problem

Facility Layout Problem (FLP) is one of the most fundamental issues in the design of production system. The goal of FLP is "locating the different facilities or departments in a plant in order to achieve the greatest efficiency in the production of a product or service" (Tompkins et al. 2010 [1]). In an industrial situation, putting in a poor layout may well lead to serious blunders in the use of a company's available land, in costly re-arrangements in actually tearing down buildings, walls, or major structures which are still usable but which subsequently turn out to be roadblocks to efficiency and low-cost operation [2]. Therefore, it is important for the facility planners to develop the facility layouts that avoid such mistakes.

1.2. Related Research

FLP is a well studied problem and a number of different models and algorithms have been developed. See the surveys Levary and Kalchik(1985) [3], Kusiak and Heragu (1987) [4], Hassan (1994) [5], Meller and Gau (1996) [6], Drira et al.(2007) [7], Arikaran et al. (2010) [8], Anjos and Vieira (2017) [9], and Hosseini-Nasab et al. (2018) [10].

Early research in this field include Koopmans and Beckmann (1957) [11], where the FLP is formulated as QAP (Quadratic Assignment Problem), by approximating the problem as model of assigning all facilities to different locations with the goal of minimizing the sum of the distances multiplied by the corresponding flows. In the 1990s, several exact formulation was presented by representing the candidate layout continuously (called 'continuous representation'), as opposed to the QAP formulation represents the layout candidate by a grid structure (called 'discrete representation'). According to Liu and Meller (2007) [12], "By representing the FLP in a discrete fashion, the FLP is simplified, but at the

penalty of eliminating many solutions from consideration. Continuous representation is more accurate and realistic than discrete representation, and thus is capable of finding the 'real optimal' final layout solution." However, the continuous representation also increases the complexity of the FLP. In the optimization point of view, FLP belongs to the non-convex problem, and, unfortunately, there are no effective methods for solving the problem in this class. Methods for the general non-convex problem therefore take several different approaches: NLP (NonLinear Programming)-based approach; MH (Meta-Heuristics)-based approach; and MIP (Mixed-Integer-Programming)-based approach, each of which involves some *compromise*.

In the NLP-based approach, FLP is represented by purely continuous variables and optimized by NLP techniques. The compromise is to give up seeking the optimal solution, and instead they seek a point that is only a locally optimal solution. As a result, the objective value of the local solution obtained is highly dependent on the choice of an initial point. Hence, much of the research is focused on a method for producing a good enough initial point. Those research include Van Camp, Carter and Vaneli (1992) [13], Anjos and Vaneli (2002) [14], Anjos and Vaneli (2006) [15], Jankovits, Luo, Anjos and Vaneli (2011) [16], and Anjos and Vieira (2016) [17]. The general framework is based on the combination of two models: The first is a relaxation of the layout problem and intended to find a good initial point for the iterative algorithm used to solve the second model; the second model is an exact formulation of the facility layout problem and intended to find locally optimal solution. These methods can be fast, can handle large-scale problems, but still have the disadvantage of initial-dependency even with the relaxation step, since the relaxed model that has been proposed is *not* convex as well.

In the MH-based approach, FLP is represented by a string of finite length integer-variables and optimized by MH techniques (e.g., Simulated Annealing or Genetic Algorithm). The compromise is the quality of solution; an approximate solution is found by drawing some number of candidates from a probability distribution, and taking the best one found as the approximate solution. This approach includes LOGIC and FLEX-BAY. LOGIC is an improvement type algorithm developed by [18], where a candidate layout is represented as a slicing tree. A slicing tree is a binary tree which shows the process of recursive partitioning rectangular in such a way that each rectangular partition corresponds to the space allocated to a facility. In the LOGIC, the slicing tree is optimized using a Genetic Algorithm, and further extensions are conducted by Gau and Meller (1999) [19], Shayan and Chittilappilly (2004) [20], Scholz, Petrick, and Domschke (2009) [21], and Kang and Chae (2017) [22]. FLEX-BAY is an improvement-type algorithm based on a continuous representation developed by Tate and Smith [23]. A layout is represented by a flexible number of vertical bays of varying width, each divided into one or more rectangular departments. Encoding flexible bay layouts is a two-part representation: permutation of the departments and breakpoints for the bays. FLEX-BAY utilizes a genetic algorithm to search the solution space by varying department-to-bay assignments or by adding or removing a bay breakpoint. A family of this approach are Kulturel-Konak, Konak, Norman and Smith(2006) [24], Wong and Komarudin (2010) [25], Kulturel-Konak and Konak (2010) [26], Kulturel-Konak and Konak (2011) [27], Palomo Romero et al. (2017) [28], Garcia-Hernandez et al. (2019) [29], and Garcia-Hernandez et al. (2020) [30]. Recent papers study the multi-objective model. Liu et al. (2018) [31] developed a multi-objective model to minimize material handling cost, to maximize the total adjacency value, and to maximize space utilization. They applied a multi-objective particle swarm optimization. Guan et al. (2019) [32] developed a multi-objective model to minimize material handling cost, to minimize the number of available workshops required for departments, and to maximize space utilization. They also applied a multi-objective particle swarm optimization. Another research trend is the dynamic and stochastic model. Pourvaziri and Pierreval (2017) [33] developed a dynamic facility layout problem based on an open queuing network theory. Turanoğlu and Akkaya (2018) [34] introduced a hybrid heuristic algorithm based on bacterial foraging optimization. Tayal et al. (2017) [35] developed a stochastic dynamic

facility layout problem. Several studies integrated these multi-objective and dynamic models. Azevedo et al. (2017) [36] developed a multi-objective model to minimize material handling cost, to minimize reconfiguration cost, and to minimize unsuitability between departments and location. Pourhassan and Raissi (2017) [37] developed a multi-objective model to minimize the total material handling cost and machine rearrangement costs, and to minimize the number of possible transporters accident that can be evaluated by the simulation. They proposed a simulation-based optimization framework in which the genetic algorithm was applied to find an optimal arrangement. While MH-based methods work very well in practice for larger sized problem instances, these approaches have a structural disadvantage in which all these heuristics cannot consider all-feasible solutions due to the encoding scheme that represents solutions as strings for the use of MH techniques.

In the MIP-based approach, FLP is represented by a combination of the binary variables that specify the relative positioning of each pair of departments and the continuous variables that denote the department positioning and shapes. The outline of a general MIP-based-method is as follows. It alternates between two steps: determining a relative position of departments, and the optimizing department positions and the shapes under the specified relative positioning. If the relative positioning of the department is specified, FLP can be reduced to the convex optimization problems such as LP (Linear Programming), QP (Quadratic Programming), or SDP (SemiDefinite Programming), and, thus, can be solved by several convex optimization techniques developed in the past few decades (e.g., simplex method, active-set method, interior-point method). Unlike other two approaches which may miss the optimal solution, MIP-based approach is the true global optimization approach, and, thus, is absolutely reliable. The compromise is, however, the scalability; the first MIP-based method presented by Montreuil in 1990 [38] can only solve the problem with six or less departments. This model is now referred to as FLP0 and several modification for MIP-FLP formulation has been conducted such as FLP1 presented by Meller et al. (1999) [39]; FLP2 by Sherali et al. (2003) [40]; and ε-accurate scheme for controlling the department area by Castillo and Westerlund (2005) [41]; a sequence-pair representation by Liu and Meller (2007) [12]; a graph–pair representation by Bozer and Wang (2012) [42]. Even with those improvements, however, the problem size that can be solved to optimality is still less than twelve (Chae and Regan 2016 [43]).

1.3. Research Purpose

Corresponding to the above-mentioned two steps of MIP, there are two approaches for speeding up MIP: searching the relative position more efficiently, or optimizing department positions and the shapes under the specified relative positioning faster. In this study, we focus primarily on the latter class approach.

In this paper, we describe a custom interior-point method for solving the FLP with relative positioning constraints (FLPRC) that is substantially faster than the standard methods used in the general-purpose solver. We build a compact formation of FLPRC and its dual, which enables us to establish the optimal condition very quickly. We use this optimality condition to implement the primal-dual interior-point method with an efficient Newton step computation that exploits the special structure of a Hessian. The computation effort of our method scales with linear-order of the number of departments, whereas that of standard methods scales with cubic-order of the number of departments.

The outline of this paper is as follows. In Section 2, we describe the FLPRC, formulated as a linear programming in a matrix form. In Section 3, we describe the overall algorithm. We first derive a dual problem and the Karush–Kuhn–Tucker (KKT) optimal condition of the model. We use this KKT condition to implement our optimization method with a primal-dual interior-point. We describe the barrier subproblem associated with a primal-dual interior-point method for the FLPRC, and we show how the special structure of the FLPRC can be exploited to compute the search direction very efficiently. In Section 4, we give numerical results to illustrate the effectiveness of proposed method. Finally, we provide our conclusions and discuss some of the future research in Section 5.

2. Facility Layout Problem with Relative Positioning Constraints

In this section, we first review the FLPRC based on the FLP2 presented by [40]. We then describe our modified model and how our model can be faster than the FLP2.

2.1. The FLPRC Formulation

We consider N departments $i = 1, \cdots, N$, which have centroid positions $(x_i, y_i) \in \mathbb{R}^2$ and shapes $(w_i, h_i) \in \mathbb{R}^2$ of half of their widths and heights, for $i = 1, ..., N$. They must be placed in a rectangle area with width W and height H, and lower left corner at the position (0, 0), under specified relative positioning.

The objective of FLP is to find a set of positions and shapes of department $r_i = [x_i, y_i, w_i, h_i]^T$ that yield as small material handling costs as possible. In the FLP2, the material handling cost is measured as the product of distance times flows between departments as in (1)

$$\text{minimize} \quad \sum_{i=1}^{N} \sum_{j=i+1}^{N} f_{ij} d_{ij}, \tag{1}$$

where f_{ij} denotes the material flows from department i to department j and d_{ij} denotes the distance between centroids of department i and department j. The choices of distance measures are usually either rectilinear or Euclidean, but, in the FLP2, the rectilinear distances are used as in (2).

$$\begin{aligned} d_{ij} &= d_{ij}^x + d_{ij}^y \\ &= |x_i - x_j| + |y_i - y_j| \end{aligned} \tag{2}$$

The absolute values in the distance function (2) can be linearized as in (3)

$$\begin{aligned} -d_{ij}^x \le x_i - x_j \le d_{ij}^x \\ -d_{ij}^y \le y_i - y_j \le d_{ij}^y \end{aligned} \tag{3}$$

We call these inequalities the *distance constraints*.

In the FLP, it is required that the departments lie inside the bounding rectangle with width W and height H as in (4) and (5).

$$w_i \le x_i \le W - w_i, \quad i = 1, \cdots, N \tag{4}$$

$$h_i \le y_i \le H - h_i, \quad i = 1, \cdots, N \tag{5}$$

These inequalities are called the *within-boundary constraints*.

The *aspect-ratio constraints* require that the aspect ratio of a department is bounded within upper and lower limits, i.e.,

$$l_i \le h_i / w_i \le u_i, \qquad i = 1, \cdots, N, \tag{6}$$

where aspect ratio of a department is defined by h_i/w_i and the upper and lower bound of aspect ratio are defined by u_i, l_i, respectively. Multiplying both sides of each inequality by w_i reduces to these constraints into linear inequalities as in (7).

$$l_i w_i \le h_i \le u_i w_i, \qquad i = 1, \cdots, N, \tag{7}$$

In case the dimension of a department is known a priori (often called Machine Layout Problem, or MLP), we can use the fixed aspect ratio with $l_i = u_i$, which results in a linear equality constraint.

Let s_i be the quarter of minimal area of department i for $i = 1, ..., N$, thus the *area constraints* require Equation (8):

$$w_i h_i \ge s_i, \qquad i = 1, \cdots, N. \tag{8}$$

Instead of these exact constraints, we use an outer polyhedra approximation, proposed by [40], to obtain the speed-up as (9):

$$s_i w_i + 4\bar{w}_{im}^2 h_i \geq 2 s_i \bar{w}_{im}, \qquad i = 1, \cdots, N, m = 1, \cdots, M. \tag{9}$$

where affine support points are defined by $\bar{w}_{im}$ and the number of affine support points are defined by M, respectively. By using the constraints (9), instead of (8), the area constraints can be reduced to the standard linear inequalities, and thus can be solved much faster.

The relative-positioning constraints require that, for each pair of departments (i, j) with $i \neq j$, the department i must be placed to the left, right, above, or below the department j. These relations are specified with the binary variables $\mathcal{H}$ (meaning 'horizontal') and $\mathcal{V}$ (meaning 'vertical') as in (10) and (11):

$$\mathcal{H}_{ij} = \begin{cases} 1 & (\text{if } i \text{ is left of } j) \\ 0 & (otherwise) \end{cases} \tag{10}$$

$$\mathcal{V}_{ij} = \begin{cases} 1 & (\text{if } i \text{ is below } j) \\ 0 & (otherwise) \end{cases} \tag{11}$$

Using these variables, the following linear inequalities are imposed as in (12) and (13):

$$x_i + w_i \leq x_j - w_j \qquad \text{if } \mathcal{H}_{ij} = 1 \tag{12}$$

$$y_i + h_i \leq y_j - h_j \qquad \text{if } \mathcal{V}_{ij} = 1 \tag{13}$$

Although we will *not* determine the relative positioning, we mention that the following condition must hold to ensure that the departments do not overlap (14):

$$\mathcal{H}_{ij} + \mathcal{H}_{ji} + \mathcal{V}_{ij} + \mathcal{V}_{ji} = 1 \tag{14}$$

These constraints are included in the MIP-based FLP and make the problem a complicated combinational problem.

We summarize the overall problem formulation as in (15).

$$\text{minimize} \quad \sum_{i=1}^{N} \sum_{j=i+1}^{N} f_{ij} \{ d_{ij}^x + d_{ij}^y \} \tag{15a}$$

$$\text{subject to} \quad -d_{ij}^x \leq x_i - x_j \leq d_{ij}^x, \qquad i, j = 1, \cdots, N. \tag{15b}$$

$$-d_{ij}^y \leq y_i - y_j \leq d_{ij}^y, \qquad i, j = 1, \cdots, N. \tag{15c}$$

$$w_i \leq x_i \leq W - w_i, \qquad i = 1, \cdots, N. \tag{15d}$$

$$h_i \leq y_i \leq H - h_i, \qquad i = 1, \cdots, N. \tag{15e}$$

$$l_i w_i \leq h_i \leq u_i w_i, \qquad i = 1, \cdots, N, \tag{15f}$$

$$x_i + w_i \leq x_j - w_j \quad \text{if } \mathcal{H}_{ij} = 1, \qquad i, j = 1, \cdots, N. \tag{15g}$$

$$y_i + h_i \leq y_j - h_j \quad \text{if } \mathcal{V}_{ij} = 1, \qquad i, j = 1, \cdots, N. \tag{15h}$$

$$s_i w_i + 4\bar{w}_{im}^2 h_i \leq 2 s_i \bar{w}_{im}, \qquad i = 1, \cdots, N, m = 1, \cdots, M. \tag{15i}$$

- N: number of departments;
- i, j: department indices ($i, j = 1 \cdots N$);
- f_{ij}: material flow between department i and department j;
- W, H: width and height of facility;
- $\mathcal{H}_{ij}, \mathcal{V}_{ij}$: binary constants to specify the relative positioning of department i and j in the horizontal and vertical direction;
- l_i, u_i: lower and upper limits on the aspect ratio of department i;
- s_i: quarter of minimum area of department i;

- d_{ij}^x, d_{ij}^y: rectilinear distance between department i and j in x and y direction;
- x_i, y_i: x and y coordinates of centroid of department i;
- w_i, h_i: half length of width and height of department i;

2.2. Reducing Redundant Inequalities by Exploiting Relative-Positioning Constraints Structure

In this section, we describe the redundant inequalities that can be eliminated by exploiting relative-positioning constraints structure. It yields more sparsity of the matrix in the linear equation to compute the Newton step, which is the main effort of the algorithm, and thus yields the speeding-up the algorithm.

2.2.1. Redundancy in Distance Constraints

For each pair of departments (i, j) with $i \neq j$, two inequalities in (12) and (13) can be reduced to a linear equality, if $\mathcal{H}_{ij}$ and $\mathcal{V}_{ij}$ are specified as follows:

$$\begin{aligned} -d_{ij}^x \leq x_i - x_j \leq d_{ij}^x \quad &\rightarrow \quad d_{ij}^x = x_j - x_i \quad (\text{if } \mathcal{H}_{ij} = 1) \\ -d_{ij}^y \leq y_i - y_j \leq d_{ij}^y \quad &\rightarrow \quad d_{ij}^y = y_j - y_i \quad (\text{if } \mathcal{V}_{ij} = 1) \end{aligned}$$

Substituting these equalities into d_{ij}^x, d_{ij}^y in the objective function (1), these variables and equalities disappeared.

Assuming the non-overlapping condition (14) holds, at least ${}_NC_2$ variables and $2 \times {}_NC_2$ inequalities can be reduced. We can further reduce the distance variables and constraints in the similar way, for each pair of departments (i, j) with $f_{ij} = 0$.

2.2.2. Redundancy in Within-Boundary Constraints

The within-boundary constraints are also reduced by using the relative-positioning structure. The constraints $w_i \leq x_i$ in (4) only needs to be imposed on the left-most departments, i.e., for $\forall i$ s.t. $\sum_{j=1}^N \mathcal{H}_{ij} = 0$. In the similar way, the minimal set of within-boundary constraints can be reduced as follows:

$$\begin{aligned} w_i \leq x_i \quad &\text{for } \forall i \text{ s.t. } \sum_{j=1}^N \mathcal{H}_{ji} = 0 \\ x_i \leq W - w_i \quad &\text{for } \forall i \text{ s.t. } \sum_{j=1}^N \mathcal{H}_{ij} = 0 \\ h_i \leq y_i \quad &\text{for } \forall i \text{ s.t. } \sum_{j=1}^N \mathcal{V}_{ji} = 0 \\ y_i \leq H - h_i \quad &\text{for } \forall i \text{ s.t. } \sum_{j=1}^N \mathcal{V}_{ij} = 0 \end{aligned}$$

Suppose $N_{left}, N_{right}, N_{upper}, N_{lower}$ are the number of the most left, right, upper, and lower departments, respectively, the number of inequalities can be reduced to $N_{left} + N_{right} + N_{upper} + N_{lower}$. Assuming that the non-overlapping condition (14) holds, this number becomes, at most, less than $2(N + 1)$, which is less than the original $4N$ inequalities.

2.2.3. Redundancy in Relative-Positioning Constraints

Finally, we mention the redundant inequalities in relative-positioning constraints. It is obvious that, if the conditions $\mathcal{H}_{ij} = 1$ and $\mathcal{H}_{jk} = 1$ hold, then the condition $\mathcal{H}_{ik} = 1$ must hold. Using these relations, we can reduce the redundant inequalities.

A minimal set of relative-positioning constraints can easily be obtained through eliminating the indirect path in $\mathcal{H}_{ij}$ or $\mathcal{V}_{ij}$ by the *Warshall–Floyd* algorithm.

2.3. The FLPRC Formulation in a Matrix Form

In this section, we introduce simpler formulation in a matrix form. This formulation does not reduce the total number of variables nor constraints, but it removes many row/column permutation to solve the linear equation to compute the Newton step.

2.3.1. Notation

First, we describe the notation in this section. $\mathbf{0}_m$ denotes the $(m \times 1)$ vector with all entries zero, $\mathbf{1}_m$ denotes the $(m \times 1)$ vector with all entries one. I means the identity matrix, and **diag(a)** means diagonal matrix with diagonal entries $a_1 \cdots a_m$, where a_i is i-th entry of $\mathbf{a}$.

2.3.2. Decision Vector

Let $\mathbf{x}, \mathbf{y}, \mathbf{w}, \mathbf{h} \in \mathbb{R}^N$ denote the vector with i^{th} entry x_i, y_i, w_i, h_i, respectively. We define the *layout vector* $\mathbf{r} \in \mathbb{R}^{4N}$ as

$$\mathbf{r} = \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{w} \\ \mathbf{h} \end{bmatrix} \in \mathbb{R}^{4N}$$

and the *distance vector* $\mathbf{d} = [\mathbf{d_x}^T, \mathbf{d_y}^T]^T \in \mathbb{R}^{N(N-1)}$ as

$$\mathbf{d_x} = \begin{bmatrix} d_1^x \\ d_2^x \\ \vdots \\ d_K^x \end{bmatrix} = \begin{bmatrix} d_{12}^x \\ d_{13}^x \\ \vdots \\ d_{(N-1)N}^x \end{bmatrix}, \quad \mathbf{d_y} = \begin{bmatrix} d_1^y \\ d_2^x \\ \vdots \\ d_K^y \end{bmatrix} = \begin{bmatrix} d_{12}^y \\ d_{13}^y \\ \vdots \\ d_{(N-1)N}^y \end{bmatrix}$$

where $d_k^x = d_{ij}^x$ with $k = (i-1)/2 \times (2N-i) + (j-i)$. We denote the overall set of decision variables $\mathbf{v} \in \mathbb{R}^{4N+(N-1)N}$ as

$$\mathbf{v} = \begin{bmatrix} \mathbf{r} \\ \mathbf{d} \end{bmatrix}.$$

2.3.3. Objective Function

We define the *flow vector* $\mathbf{f} \in \mathbb{R}^{N(N-1)/2}$ as

$$\mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_K \end{bmatrix} = \begin{bmatrix} f_{12} \\ f_{13} \\ \vdots \\ f_{(N-1)N} \end{bmatrix}$$

and the overall *cost vector* $\mathbf{c} = [\mathbf{0}_{4N}^T, \mathbf{f}^T, \mathbf{f}^T]^T$. Using these vectors, we can write the objective function (1) as in (16):

$$\begin{aligned} \sum_{i=1}^{N} \sum_{j=i+1}^{N} f_{ij} d_{ij} &= f_1 d_1 + \cdots + f_K d_K \\ &= f_1 (d_1^x + d_1^y) + \cdots + f_K (d_K^x + d_K^y) \\ &= \mathbf{f}^T \mathbf{d_x} + \mathbf{f}^T \mathbf{d_y} \\ &= \begin{bmatrix} \mathbf{0} \\ \mathbf{f} \\ \mathbf{f} \end{bmatrix}^T \begin{bmatrix} \mathbf{r} \\ \mathbf{d_x} \\ \mathbf{d_y} \end{bmatrix} \\ &= \mathbf{c}^T \mathbf{v} \end{aligned} \tag{16}$$

If a pair of department (i, j) can be reduced with $\mathcal{H}_{ij}, \mathcal{V}_{ij}$ from the redundancy in distance constraints, then the *cost vector* becomes $\mathbf{c} = [\mathbf{c}_r^T, \mathbf{f}^T, \mathbf{f}^T]^T$ with

$$(\mathbf{c}_r)_k = \begin{cases} f_{ij} & \text{if } k = (i-1)/2 \times (2N-i) + (j-i) \\ -f_{ij} & \text{if } k = (j-1)/2 \times (2N-j) + (i-j) \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

and the kth entry corresponding (i, j) in $\mathbf{f}$ becomes 0.

2.3.4. Distance Constraints

We define the *distance constraints matrix* $A_d \in \mathbb{R}^{N(N-1)\times 4N}$ as

$$A_d = \begin{bmatrix} A_d^x & O & O & O \\ O & A_d^y & O & O \end{bmatrix},$$

where $A_d^x = A_d^y$ are defined as

$$A_d^x = A_d^y = \begin{bmatrix} A_{d1} \\ A_{d2} \\ \vdots \\ A_{dN} \end{bmatrix},$$

and A_{di} is defined as

$$A_{di} = \begin{bmatrix} O_{(N-i+1)\times(i-1)} & \mathbf{1}_{(N-i+1)} & -I_{(N-i+1)\times(N-i+1)} \end{bmatrix}.$$

Using these matrices, we can write the distance constraints (3) as in (18).

$$\begin{aligned} & -d_{ij}^x \le x_i - x_j \le d_{ij}^x, && \forall i < j \\ & -d_{ij}^y \le y_i - y_j \le d_{ij}^y, && \forall i < j \\ \Leftrightarrow & \begin{cases} x_i - x_j - d_{ij}^x \le 0 \\ y_i - y_j - d_{ij}^y \le 0 \\ -(x_i - x_j) - d_{ij}^x \le 0 \\ -(y_i - y_j) - d_{ij}^y \le 0 \end{cases}, && \forall i < j. \\ \Leftrightarrow & \begin{cases} A_d^x \mathbf{x} - \mathbf{d_x} \le \mathbf{0} \\ A_d^y \mathbf{y} - \mathbf{d_y} \le \mathbf{0} \\ -A_d^x \mathbf{x} - \mathbf{d_x} \le \mathbf{0} \\ -A_d^x \mathbf{y} - \mathbf{d_y} \le \mathbf{0} \end{cases} \\ \Leftrightarrow & \begin{cases} A_d \mathbf{r} - \mathbf{d} \le \mathbf{0} \\ -A_d \mathbf{r} - \mathbf{d} \le \mathbf{0} \end{cases} \\ \Leftrightarrow & \begin{bmatrix} A_d & -I \\ -A_d & -I \end{bmatrix} \begin{bmatrix} \mathbf{r} \\ \mathbf{d} \end{bmatrix} \le \mathbf{0} \end{aligned} \tag{18}$$

If there are redundant constraints that can be eliminated, the corresponding rows are eliminated. Instead, the corresponding term is represented by the cost vector (17).

2.3.5. Within-Boundary Constraints

We define the *within-boundary constraints matrix* $A_{wb} \in \mathbb{R}^{4N\times 4N}$ and the *within-boundary constraints vector* $\mathbf{b}_{wb} \in \mathbb{R}^{4N}$ as follows:

$$A_{wb} = \begin{bmatrix} -I & I \\ I & I \end{bmatrix}, \quad \mathbf{b}_{wb} = \begin{bmatrix} \mathbf{0}_{2N} \\ W \times \mathbf{1}_N \\ H \times \mathbf{1}_N \end{bmatrix}$$

Using these matrices, the within-boundary constraints can be written as in (19).

$$
\begin{aligned}
& w_i \le x_i \le W - w_i, \qquad i = 1, \cdots, N. \\
& h_i \le y_i \le H - h_i, \qquad i = 1, \cdots, N. \\
\Leftrightarrow & \begin{cases} -x_i + w_i \le 0, & i = 1, \cdots, N. \\ -y_i + h_i \le 0, & i = 1, \cdots, N. \\ x_i + w_i \le W, & i = 1, \cdots, N. \\ y_i + h_i \le H, & i = 1, \cdots, N. \end{cases} \\
\Leftrightarrow & \begin{bmatrix} -I & O & I & O \\ O & -I & O & I \\ I & O & I & O \\ O & I & O & I \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{w} \\ \mathbf{h} \end{bmatrix} \le \begin{bmatrix} \mathbf{0}_N \\ \mathbf{0}_N \\ W \times \mathbf{1}_N \\ H \times \mathbf{1}_N \end{bmatrix} \\
\Leftrightarrow & A_{wb}\mathbf{r} \le \mathbf{b}_{wb}.
\end{aligned} \tag{19}
$$

If there are redundant constraints that can be eliminated, the entries of corresponding department become 0.

2.3.6. Aspect-Ratio Constraints

The aspect-ratio constraints (7) can be written as

$$
\begin{aligned}
l_i w_i - h_i \le 0, \qquad & i = 1, \cdots, N, \\
-u_i w_i + h_i \le 0, \qquad & i = 1, \cdots, N.
\end{aligned}
$$

We can express these bounds in a matrix form as

$$
\begin{bmatrix} O & O & -L & I \\ O & O & U & I \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{w} \\ \mathbf{h} \end{bmatrix} \le \begin{bmatrix} \mathbf{0}_{2N} \\ \mathbf{0}_{2N} \end{bmatrix}
$$

or simply we can write as in (20)

$$
A_{ar}\mathbf{r} \le \mathbf{b}_{ar} \tag{20}
$$

where $A_{ar} \in \mathbb{R}^{4N \times 4N}, L \in \mathbb{R}^{N \times N}, U \in \mathbb{R}^{N \times N}, \mathbf{b}_{ar} \in \mathbb{R}^{4N}$ are defined as follows:

$$
A_{ar} = \begin{bmatrix} O & O & -L & I \\ O & O & U & I \end{bmatrix}, L = \mathbf{diag}(\mathbf{l}), U = \mathbf{diag}(\mathbf{u}), \mathbf{b}_{ar} = \mathbf{0}_{4N}
$$

2.3.7. Area Constraints

The approximated area constraints

$$
-s_i w_i - 4\bar{w}_{im}^2 h_i \le -2 s_i \bar{w}_{im}, \qquad i = 1, \cdots, N, m = 1 \cdots M.
$$

can be written as follows:

$$
\begin{bmatrix} O & O & -S & -\bar{W}_m \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{w} \\ \mathbf{h} \end{bmatrix} \le -2 \times S\bar{W}_m \times \mathbf{1}_N, \qquad m = 1, \cdots, M
$$

This can be compactly written as follows:

$$
A_s\mathbf{r} \le \mathbf{b}_s, \tag{21}
$$

where $A_s \in \mathbb{R}^{MN\times 4N}, A_{sm} \in \mathbb{R}^{N\times 4N}, \mathbf{b}_{sM} \in \mathbb{R}^{MN}, \mathbf{b}_{sm} \in \mathbb{R}^{N}, S \in \mathbb{R}^{N\times N}, \bar{W}_m \in \mathbb{R}^{N\times N}, \mathbf{s} \in \mathbb{R}^{N}, \bar{\mathbf{w}}_m \in \mathbb{R}^{N}$ are defined as follows:

$$A_s = \begin{bmatrix} A_{s1} \\ \vdots \\ A_{sM} \end{bmatrix}, A_{sm} = -\begin{bmatrix} O & O & S & \bar{W}_m \end{bmatrix}, \mathbf{b}_{sM} = \begin{bmatrix} \mathbf{b}_{s1} \\ \vdots \\ \mathbf{b}_{sM} \end{bmatrix}, \mathbf{b}_{sm} = -2 \times S\bar{W}_m \times \mathbf{1}_N,$$
$$S = \mathbf{diag}(\mathbf{s}), \bar{W}_m = 4\mathbf{diag}(\bar{\mathbf{w}}_\mathbf{m})^2, \mathbf{s} = \begin{bmatrix} s_1, \cdots, s_N \end{bmatrix}^T, \bar{\mathbf{w}}_\mathbf{m} = \begin{bmatrix} \bar{w}_{1m}, \cdots, \bar{w}_{Nm} \end{bmatrix}^T.$$

2.3.8. Relative-Positioning Constraints

The relative-positioning constraints in a matrix form can be written as

$$A_{rp}\mathbf{r} \leq \mathbf{b}_{rp} \tag{22}$$

where $A_{rp} \in \mathbb{R}^{N(N-1)\times 4N}, \mathbf{b}_{rp} \in \mathbb{R}^{N(N-1)}$ are defined as follows:

$$\begin{aligned}
A_{rp} &= \begin{bmatrix} A_{rp}^{xw} \\ A_{rp}^{wh} \end{bmatrix} \\
(A_{rp}^{xw})_{mn} &= \begin{cases} 1 & \text{if } (m = (i-1)/2(2N-i) + (j-i)) \wedge (n = i, i+2N, j+2N) \wedge (\mathcal{H}_{ij} = 1) \\ -1 & \text{if } (m = (i-1)/2(2N-i) + (j-i)) \wedge (n = j) \wedge (\mathcal{H}_{ij} = 1) \\ 0, & \text{otherwise} \end{cases} \\
(A_{rp}^{yh})_{mn} &= \begin{cases} 1 & \text{if } (m = (i-1)/2(2N-i) + (j-i)) \wedge (n = i+N, i+3N, j+3N) \wedge (\mathcal{V}_{ij} = 1) \\ -1 & \text{if } (m = (i-1)/2(2N-i) + (j-i)) \wedge (n = j+N) \wedge (\mathcal{V}_{ij} = 1) \\ 0 & \text{otherwise} \end{cases} \\
\mathbf{b}_{rp} &= \mathbf{0}_M
\end{aligned}$$

For a department pair (i, j) with $\mathcal{H}_{ij} = 1$, in mth row corresponding (i, j), the coefficient of x_i, w_i, w_j becomes 1 and that of x_j becomes -1. Similarly, for a department pair (i, j) with $\mathcal{V}_{ij} = 1$, in mth row corresponding (i, j), the coefficient of y_i, h_i, h_j becomes 1 and that of y_j becomes -1. We can eliminate the row corresponding the redundant inequalities that can be reduced.

2.3.9. Overall Problem Formulation in Matrix Form

We define the overall *constraint matrix* $A \in \mathbb{R}^{(5/2N^2+(11/2+M)N)\times(4N+N(N-1))}$ and *constraint vector* $\mathbf{b} \in \mathbb{R}^{(5/2N^2+(11/2+M)N)}$ as

$$A = \begin{bmatrix} A_d & -I \\ -A_d & -I \\ A_r & O \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{b_r} \end{bmatrix}$$

and $A_r \in \mathbb{R}^{(1/2N^2+(15/2+M)N)\times(4N+N(N-1))}$ and $\mathbf{b}_r \in \mathbb{R}^{(1/2N^2+(15/2+M)N)}$ are defined as follows:

$$A_r = \begin{bmatrix} A_{wb} \\ A_{ar} \\ A_s \\ A_{rp} \end{bmatrix}, \quad \mathbf{b_r} = \begin{bmatrix} \mathbf{b_{wb}} \\ \mathbf{b_{ar}} \\ \mathbf{b_s} \\ \mathbf{b_{rp}} \end{bmatrix}$$

Using these matrices and vectors, we can write the overall constraints as

$$\begin{cases} \begin{bmatrix} A_d & -I \\ -A_d & -I \end{bmatrix} \begin{bmatrix} \mathbf{r} \\ \mathbf{d} \end{bmatrix} & \leq \mathbf{0} \\ A_{wb}\mathbf{r} & \leq \mathbf{b}_{wb} \\ A_{ar}\mathbf{r} & \leq \mathbf{b}_{ar} \\ A_s\mathbf{r} & \leq \mathbf{b}_s \\ A_{rp}\mathbf{r} & \leq \mathbf{b}_{rp} \end{cases}$$
$$\Leftrightarrow A\mathbf{v} \leq \mathbf{b}$$

Thus, we can summarize the overall problem formulation as:

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T\mathbf{v} \\ \text{subject to} \quad & A\mathbf{v} \leq \mathbf{b} \end{aligned} \tag{23}$$

This problem (23) is the standard Linear Programming (LP), and, thus, can be solved to optimality.

2.4. Dual Problem and Optimality Conditions for FLPRC

In this section, we derive a dual problem for the FLPRC (23) and outline the optimality condition. We introduce Lagrangian Multiplier vectors $\lambda = [\lambda_1, \cdots \lambda_{(5/2N^2+(11/2+M)N)}]^T \in \mathbb{R}^{(5/2N^2+(11/2+M)N)}$ for the constraint corresponding to $\mathbf{a}_i^T\mathbf{v} \leq \mathbf{b}_i$, where $\mathbf{a}_i^T$ is i-th row vector of the constraints matrix A. We let $n_\lambda = \dim(\lambda) = (5/2N^2 + (11/2 + M)N)$ denote the dimensions of λ. The Lagrangian is then

$$\begin{aligned} L(\mathbf{v}, \lambda) =& \mathbf{c}^T\mathbf{v} + \sum_{i=1}^{n_\lambda} \lambda_i(\mathbf{a}_i^T\mathbf{v} - \mathbf{b}_i) \\ =& \Big(\mathbf{c} + \sum_{i=1}^{n_\lambda} \lambda_i\mathbf{a}_i\Big)^T\mathbf{v} - \lambda^T\mathbf{b} \end{aligned} \tag{24}$$

To obtain the dual function, we minimize L over the primal variables $\mathbf{v}$. We find that the minimum is $-\infty$, unless $\mathbf{c} + \sum_{i=1}^{n_\lambda} \lambda_i\mathbf{a}_i \leq 0$. The dual function is therefore given by

$$\begin{aligned} &\inf L(\mathbf{v}, \lambda) \\ &= \inf\left\{ \Big(\mathbf{c} + \sum_{i=1}^{n_\lambda} \lambda_i\mathbf{a}_i\Big)^T\mathbf{v} - \lambda^T\mathbf{b} \right\} \\ &= \begin{cases} -\lambda^T\mathbf{b} & \text{if } \mathbf{c} + \sum_{i=1}^{n_\lambda} \lambda_i\mathbf{a}_i \leq 0 \\ -\infty & \text{otherwise} \end{cases} \end{aligned}$$

Thus, we can write the dual of the FLPRC as

$$\begin{aligned} \textbf{maximize} \quad & -\lambda^T\mathbf{b} \\ \textbf{subject to} \quad & A^T\lambda + \mathbf{c} = \mathbf{0} \\ & \lambda \geq \mathbf{0} \end{aligned} \tag{25}$$

The optimality condition, i.e., the Karush–Kuhn–Tucker (KKT) condition, is given as

$$\nabla_x L(\mathbf{v}, \lambda) = \mathbf{c} + A^T\lambda = \mathbf{0} \tag{26a}$$
$$\textbf{diag}(\lambda)(\mathbf{b} - A\mathbf{v}) = \mathbf{0} \tag{26b}$$
$$A\mathbf{v} - \mathbf{b} \leq \mathbf{0} \tag{26c}$$
$$\lambda \geq \mathbf{0} \tag{26d}$$

The first equation (26a) that follows from the gradient of Lagrangian must be zero, since the optimal point **v*** minimizes the $L(\mathbf{v}, \lambda)$ over $\mathbf{v}$. The second equation (26b) is the *complementary slickness*. The third and the last inequality (26c) and (26d) are the constraints with respect to the *primal* and *dual feasibility*, respectively.

If (and only if) there are $(\mathbf{v}^*, \lambda^*)$ that satisfies the KKT condition (26), then the $\mathbf{v}^*$ is the optimal point. In other words, solving LP (23) is equivalent to the problem of finding the point $(\mathbf{v}^*, \lambda^*)$ that satisfies the KKT conditions. As is the case in most other convex problems, however, it is difficult to directly find such points due to a large number of linear inequalities to maintain the primal and dual feasibility in the KKT conditions (26).

The interior-point method is a technique used to solve those inequality-constrained problems by reducing it to a sequence of problems without inequalities. The details are described in the next section.

3. Custom Interior-Point Method for Solving FLPRC

In this chapter, we describe an efficient method for solving FLPRC (23). The method is *primal-dual interior-point method*, which is one of the most powerful techniques for solving Linear Programming (LP) or other classes of convex optimization programming. We propose an efficient Newton step computation that exploit the special structure of the problem. Our algorithm is also applicable to other types of interior-point methods with a minor change.

3.1. Barrier Subproblem

The main idea of the interior-point method is casting the inequality constraints into the objective function as

$$\begin{array}{ll} \text{minimize} & \mathbf{c}^T\mathbf{v} \\ \text{subject to} & A\mathbf{v} \leq \mathbf{b} \end{array} \quad \rightarrow \quad \text{minimize} \quad \mathbf{c}^T\mathbf{v} + \frac{1}{t}\phi(\mathbf{v}) \tag{27}$$

where t is the scaling parameter, and $\phi(\mathbf{v})$ is defined as

$$\phi(\mathbf{v}, \lambda) = \sum_{i=1}^{n_\lambda} \phi_i(\mathbf{v}) \tag{28}$$

$$\phi_i(\mathbf{v}, \lambda) = \begin{cases} -log(b_i - \mathbf{a}_i^T\mathbf{v}) \\ \infty. \end{cases} \tag{29}$$

The problem is called a *barrier subproblem* and the function ϕ is called a *barrier function*. This function is convex and smooth, and, thus, the barrier subproblem becomes convex if the original problem is convex. The first derivative of this function is

$$\nabla\phi(\mathbf{v}, \lambda) = \sum_{i=1}^{n_\lambda} \frac{\mathbf{a}_i}{b_i - \mathbf{a}_i^T\mathbf{v}} \tag{30}$$

We define $\mathbf{z}$ with elements $z_i = 1/(b_i - \mathbf{a}_i^T\mathbf{v})$, and we have

$$\nabla\phi(\mathbf{v}, \lambda) = A^T\mathbf{z}$$

The optimality condition for the *barrier subproblem* is

$$\nabla\left(\mathbf{c}^T\mathbf{v} + \frac{1}{t}\phi(\mathbf{v})\right) = \mathbf{c} + \frac{1}{t}A^T\mathbf{z} = \mathbf{0}. \tag{31}$$

Comparing this with the gradient of Lagrangian (26a),

$$\nabla_x L(\mathbf{v}, \lambda) = \mathbf{c} + A^T\lambda = \mathbf{0}$$

we have

$$\lambda = \frac{1}{t}\mathbf{z} = \frac{1}{t(\mathbf{b} - A\mathbf{v})}. \tag{32}$$

Substituting this into the *complementary slackness* (26b) and eliminating linear inequalities (26c) and (26d), we have the modified KKT condition:

$$\begin{aligned} \mathbf{c} + A^T\lambda &= \mathbf{0} \\ \mathbf{diag}(\lambda)(\mathbf{b} - A\mathbf{v}) &= (1/t)\mathbf{1} \end{aligned} \tag{33}$$

The modified KKT conditions (33) satisfy the following two properties:

- These conditions are far easier to solve than the KKT condition (26)
- As t increases, these condition approaches the KKT condition (26).

Combining these two properties, the FLPRC is solved to optimality. The modified KKT conditions (33) are solved by the Newton method, which is described in detail in the next section.

3.2. Newton Method

We rewrite the modified KKT conditions (33) as in (34).

$$R_t(\mathbf{v}, \lambda) = \begin{bmatrix} R_t^{\mathbf{v}}(\mathbf{v}, \lambda) \\ R_t^{\lambda}(\mathbf{v}, \lambda) \end{bmatrix} = \begin{bmatrix} \mathbf{c} + A^T\lambda \\ \mathbf{diag}(\lambda)(\mathbf{b} - A\mathbf{v}) - (1/t)\mathbf{1} \end{bmatrix} = \mathbf{0} \tag{34}$$

In a primal-dual interior method, we start with some initial point $(\mathbf{v_0}, \lambda_0)$ and update them in such a way that $(\mathbf{v_0}, \lambda_0) \to (\mathbf{v_1}, \lambda_1) \to \cdots \to (\mathbf{v_k}, \lambda_k)$ with $R_t(\mathbf{v_k}, \lambda_k) \to 0$ for $k \to \infty$. These updates are made by a Newton method.

At each iteration, primal and dual search directions $(\Delta\mathbf{v}_k, \Delta\lambda_k)$ and step size γ_k are computed, then the new point is obtained as $(\mathbf{v}_{k+1}, \lambda_{k+1}) = (\mathbf{v}_k + \gamma_k\Delta\mathbf{v}_k, \lambda_k + \gamma_k\Delta\lambda_k)$. In the Newton method, the search direction is characterized by the solution of the linear equation

$$R_t(\mathbf{v}_k + \Delta\mathbf{v}_k, \lambda_k + \Delta\lambda_k) \approx R_t(\mathbf{v}_k, \lambda_k) + D_{R_t}(\mathbf{v}_k, \lambda_k)[\Delta\mathbf{v}_k, \Delta\lambda_k]^T = \mathbf{0} \tag{35}$$

$$\Leftrightarrow \begin{bmatrix} \Delta\mathbf{v}_k \\ \Delta\lambda_k \end{bmatrix} = -D_{R_t}^{-1}(\mathbf{v}_k, \lambda_k)R_t(\mathbf{v}_k, \lambda_k) \tag{36}$$

where

$$\begin{aligned} D_{R_t}(\mathbf{v}_k, \lambda_k) &= \begin{bmatrix} \nabla_{\mathbf{v}} R_t^{\mathbf{v}}(\mathbf{v}, \lambda) & \nabla_{\lambda} R_t^{\mathbf{v}}(\mathbf{v}, \lambda) \\ \nabla_{\mathbf{v}} R_t^{\mathbf{v}}(\lambda, \lambda) & \nabla_{\lambda} R_t^{\lambda}(\mathbf{v}, \lambda) \end{bmatrix} \\ &= \begin{bmatrix} O & A^T \\ \mathbf{diag}(\lambda)A & \mathbf{diag}(\mathbf{b} - A\mathbf{v}) \end{bmatrix} \end{aligned}$$

The first equation (35) follows from the 1st order Taylor approximation. Thus, the solution of (35) may be a good estimate of the solution of Equation (34). After computing the Newton step, we then carry out the back-tracking line search to find the step size γ_k.

3.3. Efficient Newton Step Computation

In this section, we show how to compute the Newton step $(\Delta\mathbf{v}_k, \Delta\lambda_k)$, i.e., solve Equation (36), efficiently. These equations can be solved using standard methods for linear equations; however, by exploiting the special structure of the equations, they can be solved even faster. The method we describe in this section is based on a sequence of three elimination steps, in which particular blocks of variables are eliminated by the Schur Complement.

3.3.1. 1st Elimination Step

Our first step is to eliminate the variables $\Delta\lambda_k$ from Equation (35). From the second equation, we obtain

$$\begin{aligned}
&\mathbf{diag}(\lambda)A\Delta\mathbf{v}_k + \mathbf{diag}(\mathbf{b} - A\mathbf{v})\Delta\lambda_k = -R_\lambda \\
&\Leftrightarrow \Delta\lambda_k = -\mathbf{diag}(\mathbf{b} - A\mathbf{v})^{-1}R_\lambda - \mathbf{diag}(\mathbf{b} - A\mathbf{v})^{-1}\mathbf{diag}(\lambda)A\Delta\mathbf{v}_k \\
&\Leftrightarrow \Delta\lambda_k = -\mathbf{diag}(\mathbf{z})R_\lambda - \mathbf{diag}(\mathbf{z}\lambda)A\Delta\mathbf{v}_k
\end{aligned} \tag{37}$$

The second to third equation is derived using $z_i = 1/(b_i - \mathbf{a}_i^T\mathbf{v})$. Substituting this into the first equation in (35), we have Δv_k as follows:

$$\begin{aligned}
&A^T\Delta\lambda_k = R_\mathbf{v} \\
&\Leftrightarrow -A^T\mathbf{diag}(\mathbf{z})R_\lambda - A^T\mathbf{diag}(\mathbf{z}\lambda)A\Delta\mathbf{v}_k = R_\mathbf{v} \\
&\Leftrightarrow \Delta\mathbf{v}_k = -[A^T\mathbf{diag}(\mathbf{z}\lambda)A]^{-1}[R_\mathbf{v} + A^T\mathbf{diag}(\mathbf{z})R_\lambda] \\
&\Leftrightarrow \Delta\mathbf{v}_k = -[A^TZLA]^{-1}[R_\mathbf{v} + A^TZR_\lambda]
\end{aligned} \tag{38}$$

where $Z = \mathbf{diag}(\mathbf{z}), L = \mathbf{diag}(\lambda)$. These Equations (37) and (38) are alternative for the original Equation (36). Instead of solving the large set of linear Equations (36), we find $\Delta\mathbf{v_k}$ from the smaller Equation (38), which can be easily solved as described in the next two sections, and then calculate the $\Delta\lambda$ from Equation (37).

3.3.2. 2nd Elimination Step

We break down the equation and eliminate some of the blocks further to reduce the size of linear equation. From our definition of $\mathbf{v} = [\mathbf{r}^T, \mathbf{d}^T]^T$, and of A, we can rewrite the Equation (37) as follows:

$$\begin{aligned}
\Delta\mathbf{v}_k = \begin{bmatrix}\Delta\mathbf{r}_k \\ \Delta\mathbf{d}_k\end{bmatrix} &= -[A^TZLA]^{-1}[R_\mathbf{v} + A^TZR_\lambda] \\
&= -\left(\begin{bmatrix}A_d^T & -A_d^T & A_r^T \\ -I & -I & O\end{bmatrix}\begin{bmatrix}D_d^+ & O & O \\ O & D_d^- & O \\ O & O & D_r\end{bmatrix}\begin{bmatrix}A_d & -I \\ -A_d & -I \\ A_r & O\end{bmatrix}\right)^{-1}\begin{bmatrix}g_\mathbf{r} \\ g_\mathbf{d}\end{bmatrix} \\
&= -\begin{bmatrix}A_d^T(D_d^+ + D_d^-)A_d + A_r^TD_rA_r & -A_d^T(D_d^+ - D_d^-) \\ -(D_d^+ - D_d^-)A_d & D_d^+ + D_d^-\end{bmatrix}^{-1}\begin{bmatrix}g_\mathbf{r} \\ g_\mathbf{d}\end{bmatrix}
\end{aligned} \tag{39}$$

where $D_\mathbf{d}^+, D_\mathbf{d}^-, D_\mathbf{r}$ are defined as follows

$$\begin{aligned}
D_\mathbf{d}^+ &= \mathbf{diag}\{\lambda_d^{+T}(A_d\mathbf{r} - \mathbf{d})^{-1}\} \\
D_\mathbf{d}^- &= \mathbf{diag}\{\lambda_d^{-T}(-A_d\mathbf{r} - \mathbf{d})^{-1}\} \\
D_\mathbf{r} &= \mathbf{diag}\{\lambda_r^T(A_r\mathbf{r} - \mathbf{b_r})^{-1}\}
\end{aligned}$$

and $g_\mathbf{r}, g_\mathbf{d}$ are defined as follows:

$$\begin{aligned}
g_\mathbf{r} &= 2(A_d^T\lambda_d^+ - A_d^T\lambda_d^- + A_r^T\lambda_r) - (1/t)(A_d^T\mathbf{z}_d^{+T} - A_d^T\mathbf{z}_d^{-T} + A_r\mathbf{z}_r^T) \\
g_\mathbf{d} &= [\mathbf{f}^T, \mathbf{f}^T]^T + 2(-\lambda_d^+ - \lambda_d^-) - (1/t)(-\mathbf{z}_d^+ - \mathbf{z}_d^-)
\end{aligned}$$

with $(\lambda_d^+, \lambda_d^-, \lambda_r)$ are Lagrangian multipliers and (z_d^+, z_d^-, z_r) are inverse vectors for the constraints $(A_d^T\mathbf{r} - \mathbf{d} \leq 0, -A_d^T\mathbf{r} - \mathbf{d} \leq 0, A_r\mathbf{r} - \mathbf{b_r} \leq 0)$, respectively.

$(g_{\mathbf{r}}, g_{\mathbf{d}})$ are derived from the right-hand-side as (40)

$$\begin{aligned}
[R_{\mathbf{v}} + A^T Z R_\lambda] &= [(\mathbf{c} + A^T\lambda) + A^T Z(L\mathbf{z}^{-1} - (1/t)\mathbf{1})] \\
&= [\mathbf{c} + A^T(\lambda + ZL\mathbf{z}^{-1} - (1/t)\mathbf{z})] \\
&= [\mathbf{c} + A^T(\lambda + LZ\mathbf{z}^{-1} - (1/t)\mathbf{z})] \\
&= [\mathbf{c} + A^T(\lambda + L\mathbf{1} - (1/t)\mathbf{z})] \\
&= [\mathbf{c} + A^T(\lambda + \lambda - (1/t)\mathbf{z})] \\
&= [\mathbf{c} + A^T(2\lambda - (1/t)\mathbf{z})] \\
&= \left[\begin{bmatrix}\mathbf{0}\\ \mathbf{f}\\ \mathbf{f}\end{bmatrix} + \begin{bmatrix}A_d^T & -A_d^T & A_r^T\\ -I & -I & O\end{bmatrix}\left(2\begin{bmatrix}\lambda_d^+\\ \lambda_d^-\\ \lambda_r\end{bmatrix} - (1/t)\begin{bmatrix}z_d^+\\ z_d^-\\ z_r\end{bmatrix}\right)\right] \\
&= \begin{bmatrix}g_{\mathbf{r}}\\ g_{\mathbf{d}}\end{bmatrix}
\end{aligned} \tag{40}$$

From the second equation in (40), we have $\Delta\mathbf{d}_k$ as (41).

$$\begin{aligned}
-g_{\mathbf{d}} &= -(D_d^+ - D_d^-)A_d\Delta\mathbf{r}_k + (D_d^+ + D_d^-)\Delta\mathbf{d}_k \\
\Leftrightarrow\quad \Delta\mathbf{d}_k &= -(D_d^+ + D_d^-)^{-1}g_{\mathbf{d}} + (D_d^+ + D_d^-)^{-1}(D_d^+ - D_d^-)A_d\Delta\mathbf{r}_k
\end{aligned} \tag{41}$$

Substituting this into the first equation in (40), we have Δr as follows:

$$\begin{aligned}
-g_{\mathbf{r}} &= \left(A_d^T(D_d^+ + D_d^-)A_d + A_r^T D_r A_r\right)\Delta\mathbf{r}_k - A_d^T(D_d^+ - D_d^-)\Delta\mathbf{d}_k \\
&= A_d^T(D_d^+ + D_d^-)A_d\Delta\mathbf{r}_k + A_r^T D_r A_r\Delta\mathbf{r}_k \\
&\quad + A_d^T(D_d^+ - D_d^-)(D_d^+ + D_d^-)^{-1}g_{\mathbf{d}} \\
&\quad - A_d^T(D_d^+ - D_d^-)(D_d^+ + D_d^-)^{-1}(D_d^+ - D_d^-)A_d\Delta\mathbf{r}_k
\end{aligned}$$

The sum of the first and forth terms can be summarized as follows:

$$\begin{aligned}
& A_d^T\left((D_d^+ + D_d^-) - (D_d^+ - D_d^-)(D_d^+ + D_d^-)^{-1}(D_d^+ - D_d^-)\right)A_d\mathbf{r}_k \\
= \; & A_d^T(D_d^+ + D_d^-)^{-1}\left((D_d^+ + D_d^-)^2 - (D_d^+ - D_d^-)^2\right)A_d\Delta\mathbf{r}_k \\
= \; & A_d^T\left(4D_d^+ D_d^-(D_d^+ + D_d^-)^{-1}\right)A_d\Delta\mathbf{r}_k
\end{aligned}$$

We define D_d and g as follows:

$$\begin{aligned}
D_d &= 4D_d^+ D_d^-(D_d^+ + D_d^-)^{-1} \\
g &= g_{\mathbf{r}} + A_d^T(D_d^+ - D_d^-)(D_d^+ + D_d^-)^{-1}g_{\mathbf{d}}
\end{aligned}$$

Using these expressions, we have Δr as in (42).

$$(A_d^T D_d A_d + A_r^T D_r A_r)\Delta\mathbf{r}_k = -g \tag{42}$$

3.3.3. Third Elimination Step

Equation (42) can be further broken down into smaller pieces, and solved efficiently. We rewrite Equation (42) as in (43)

$$\begin{aligned}
E\Delta\mathbf{r}_k &= -g \\
E &= E_d + E_r \\
E_d &= A_d^T D_d A_d \\
E_r &= A_r^T D_r A_r
\end{aligned} \tag{43}$$

The term $A_r^T D_r A_r$ can be factorized as in Equation (44)

$$\begin{aligned} A_r^T D_r A_r &= \begin{bmatrix} A_{wb}^T & A_{ar}^T & A_s^T & A_{rp}^T \end{bmatrix} \begin{bmatrix} D_{wb} & O & O & O \\ O & D_{ar} & O & O \\ O & O & D_s & O \\ O & O & O & D_{rp} \end{bmatrix} \begin{bmatrix} A_{wb} \\ A_{ar} \\ A_s \\ A_{rp} \end{bmatrix} \\ &= A_{wb}^T D_{wb} A_{wb} + A_{ar}^T D_{ar} A_{ar} + A_s^T D_s A_s + A_{rp}^T D_{rp} A_{rp} \\ &= E_{wb} + E_{ar} + E_s + E_{rp}, \end{aligned} \tag{44}$$

where $D_{wb}, D_{ar}, D_s, D_{rp}$ are the diagonal matrices with entries with respect to *within-boundary*, *aspect-ratio*, *area*, and *relative-positioning* constraints. We decompose $\Delta \mathbf{r}_k$ with respect to $(\Delta x, \Delta y)$ and $(\Delta w, \Delta h)$ as $\Delta \mathbf{r}_k = [\Delta \mathbf{r}_{(xy)k}^T, \Delta \mathbf{r}_{(wh)k}^T]^T$. Factorizing (43) into block elements with respect to $(\Delta x, \Delta y)$ and $(\Delta w, \Delta h)$ vectors, we have the following Equation (45)

$$\begin{bmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{bmatrix} \begin{bmatrix} \Delta r_{(xy)k} \\ \Delta r_{(wh)k} \end{bmatrix} = \begin{bmatrix} g_{xy} \\ g_{wh} \end{bmatrix}, \tag{45}$$

We discuss the (m, n) block element $E_{mn}, m = 1, 2, n = 1, 2$. We define the (m, n) block element of $E_d, E_r, E_{wb}, E_{ar}, E_s, E_{rp}$ as $E_{mn}^d, E_{mn}^r, E_{mn}^{wb}, E_{mn}^{ar}, E_{mn}^s, E_{mn}^{rp}$.

First, $E_{mn}^{wb}, E_{mn}^{ar}, E_{mn}^s, m = 1, 2, n = 1, 2$ are all diagonal matrices because the block elements of $A_{wb}, A_{ar}, D_{wb}, D_{ar}$ are diagonal matrices. As for E_{mn}^{rp}, the block elements of A_{rp} is a *sparse* matrix, and D_{rp} is a diagonal matrix, so $E_{mn}^{rp}, m = 1, 2, n = 1, 2$ is a sparse symmetric matrix. Since $E_{mn}^r, m = 1, 2, n = 1, 2$ is the sum of these diagonal matrices and a sparse symmetric matrix, $E_{mn}^r, m = 1, 2, n = 1, 2$ is a sparse symmetric matrix. On the other hand, for E_{mn}^d, since it has only the components related to (x, y), E_{11}^d is a dense matrix, and the other block elements are $E_{12}^d, E_{21}^d, E_{22}^d = O$. From the above, by taking the sum of each block element, E_{11} is a dense matrix, and the other block elements E_{12}, E_{21}, and E_{22} are sparse symmetric matrices.

Speed-up is achieved using this structure. From the first equation in (45), we can obtain $\Delta \mathbf{r}_{wh}$ as Equation (46):

$$\begin{aligned} & E_{11} \Delta \mathbf{r}_{(xy)k} + E_{12} \Delta \mathbf{r}_{(wh)k} = g_{xy} \\ \Leftrightarrow\ & \Delta \mathbf{r}_{(wh)k} = E_{12}^{-1} (g_{xy} - E_{11} \Delta \mathbf{r}_{(xy)k}). \end{aligned} \tag{46}$$

Substituting this into the second equation in (45), we have Equation (47) to get $\Delta \mathbf{r}_{xy}$:

$$\begin{aligned} & E_{21} \Delta \mathbf{r}_{(xy)k} + E_{22} (E_{12}^{-1} (g_{xy} - E_{11} \Delta \mathbf{r}_{(xy)k})) = g_{wh} \\ \Leftrightarrow\ & (E_{21} - E_{22} E_{12}^{-1} E_{11}) \Delta \mathbf{r}_{(xy)k} = E_{22} E_{12}^{-1} g_{xy} - g_{wh}. \end{aligned} \tag{47}$$

Using this elimination, we can compute $\Delta \mathbf{r}_{(xy)k}$ efficiently, we then compute the remaining variables $\Delta \mathbf{r}_{(xy)k}, \Delta \mathbf{d}, \Delta \lambda$, which is faster than just solving the larger set of equations (33) using a standard factorization method.

Note that all of the matrices are non-singular, if there exists an optimal solution of problem (15).

3.4. Complexity Analysis

In this section, we analyze the calculation complexity. The complexity is measured by the number of floating operations (flops), and all of the calculation in this section is based on Boyd and Vandenberghe [44]. Let $n_v = \dim(v) = 4N + N(N-1)$ denote the dimension of v. The size of the coefficient matrix of the original Equation (36) is $(n_v + n_\lambda)$.

Using the standard LU decomposition, the computational complexity of the Equation (36) is $(2/3)(n_v + n_\lambda)^3$. Expanding this, the amount of calculation is expressed as follows:

$$\begin{aligned}(2/3)(n_v + n_\lambda)^3 &= \tfrac{343}{8}N^6 + (\tfrac{2793}{8} + \tfrac{147}{4}M)N^5 + (\tfrac{7581}{8} + \tfrac{399}{2}M + \tfrac{21}{2}M^2)N^4 \\ &\quad + (\tfrac{6859}{3} + \tfrac{1083}{4}M + \tfrac{57}{2}M^2 + M^3)N^3\end{aligned} \tag{48}$$

On the other hand, the calculation effort after 1st elimination is performed is $2n_v^2 n_\lambda + (2/3)n_v^3$. In this equation, the complexity is proportional to n_λ. Since $n_\lambda > n_v$, the amount of calculation is greatly reduced. The amount of calculation after 1st elimination is performed is as follows:

$$2n_v^2 n_\lambda + (2/3)n_v^3 = \tfrac{17}{3}N^6 + (47 + 2M)N^5 + (129 + 12M)N^4 + (117 + 18M)N^3 \tag{49}$$

The calculation effort in the second term can be further reduced by performing 2nd elimination. Let $n_r = \dim(r) = 4N$ be the dimensions of r, and let $n_d = \dim(d) = N(N-1)$ denote the dimensions of d. The calculation effort after 1st elimination is performed is $(2/3)n_v^3 = (2/3)(n_r + n_d)^3$. The calculation effort after 2nd elimination is performed is as shown in $2n_r^2 n_d + (2/3)n_r^3$. In this equation, the complexity is proportional to n_d. Since $n_d > n_r$ in the range of $N > 4$, the amount of calculation is greatly reduced. The amount of calculation after 2nd elimination that is performed is as follows:

$$2n_v^2 n_\lambda + 2n_r^2 n_d + (2/3)n_r^3 = \tfrac{4}{5}N^6 + (\tfrac{79}{5} + 2M)N^5 + (\tfrac{526}{5} + 12M)N^4 + (\tfrac{329}{3} + 18M)N^3 \tag{50}$$

The calculation effort in the second term can be further reduced by performing 3rd elimination. Let $n_{xy} = \dim(x) + \dim(y) = 2N$ denote the dimensions of x and y, and $n_{wh} = \dim(w) + \dim(h) = 2N$ denote the dimensions of w and h. The calculation effort after 2nd elimination is performed is as shown in $(2/3)n_r^3 = (2/3)(n_{xy} + n_{wh})^3$. The calculation effort after 3rd elimination is performed is $n_{wh} + 2n_{xy}^2 n_{wh} + (2/3)n_{xy}^3$. Here, the first term is the amount of calculation when sparse factorization is used by utilizing the fact that the matrix E_{22} of Equation (47) is a symmetric and sparse matrix. The third term is a block diagonal matrix. Therefore, it can be decomposed into equations related to x and y and the calculation effort is $(2/3)N^3 \times 2 = (4/3)N^3$. The amount of calculation after 3rd elimination is performed is as follows:

$$\begin{aligned}&2n_v^2 n_\lambda + 2n_r^2 n_d + n_{wh} + 2n_{xy}^2 n_{wh} + (4/3)N^3 \\ &= \tfrac{4}{5}N^6 + (\tfrac{79}{5} + 2M)N^5 + (\tfrac{526}{5} + 12M)N^4 + (\tfrac{52}{3} + 18M)N^3 + 2N\end{aligned} \tag{51}$$

These complexity is summarized in Table 1. From the above analysis, the problem can be reduced to solving smaller equations by decomposing the matrix rather than solving the original matrix.

Table 1. Summary of complexity to solve Newton equation.

Equation		Complexity (# of Flops)
Original Newton Equation (36)	$(2/3)(n_v + n_\lambda)^3$	$\tfrac{343}{8}N^6 + (\tfrac{2793}{8} + \tfrac{147}{4}M)N^5$ $+(\tfrac{7581}{8} + \tfrac{399}{2}M + \tfrac{21}{2}M^2)N^4$ $+(\tfrac{6859}{3} + \tfrac{1083}{4}M + \tfrac{57}{2}M^2 + M^3)N^3$
After 1st elimination (38)	$2n_v^2 n_\lambda + (2/3)n_v^3$	$\tfrac{17}{3}N^6 + (47 + 2M)N^5$ $+(129 + 12M)N^4$ $+(117 + 18M)N^3$
After 2nd elimination (42)	$2n_v^2 n_\lambda + 2n_r^2 n_d$ $+(2/3)n_r^3$	$\tfrac{4}{5}N^6 + (\tfrac{79}{5} + 2M)N^5$ $+(\tfrac{526}{5} + 12M)N^4$ $+(\tfrac{329}{3} + 18M)N^3$
After 3rd elimination (47)	$2n_v^2 n_\lambda + 2n_r^2 n_d$ $+n_{wh} + 2n_{xy}^2 n_{wh}$ $+(4/3)N^3$	$\tfrac{4}{5}N^6 + (\tfrac{79}{5} + 2M)N^5$ $+(\tfrac{526}{5} + 12M)N^4$ $+(\tfrac{52}{3} + 18M)N^3 + 2N$

3.5. Starting-Point Using Flexible Bay Structure

In this section, we present a technique to create the starting-point for further speeding-up the computation. While our proposed primal-dual interior point method has enough computation effectiveness in itself, there is a potential advantage in utilizing a 'good' starting point from a practical perspective. In the context of using interior point method, a starting point should be strictly feasible. Therefore, we define a 'good' starting point as points that are *close to optimality*, yet is *sufficiently far from the boundary of the feasible solution*. We describe a technique to create such starting point in this section.

Our method is based on a relaxed *Flexible-Bay Structure* (FBS), in which the departments are located in parallel bays with the varying width associated the total area of the departments in that bay, using two part representation: permutation of the departments and breakpoints for the bays. Figure 1 presents an example illustrating the bay assignment and the obtained layout from the horizontal and vertical relative-positioning graphs.

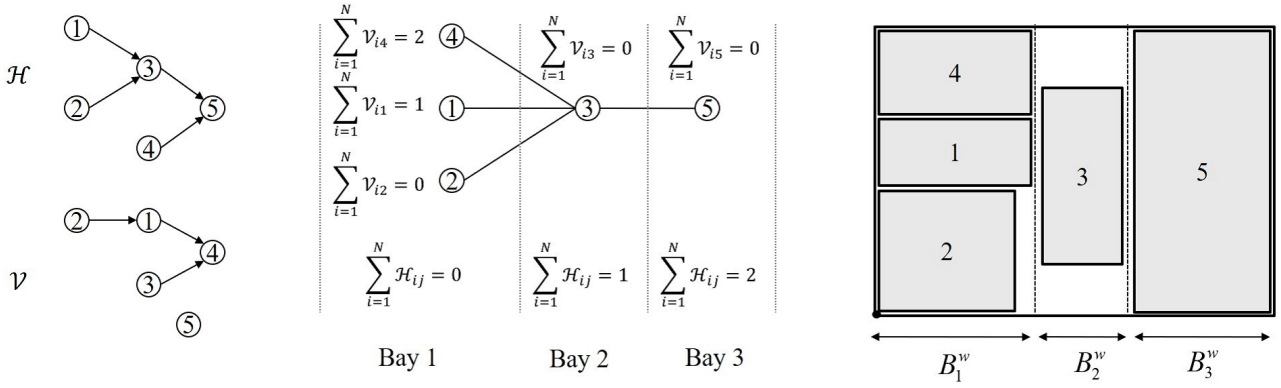

Figure 1. Example illustrating the bay assignment (shown at center) and the obtained layout (shown at right) from the horizontal and vertical relative-positioning graphs (shown at left).

In our proposal, the permutation and breakpoints are specified using the relative-positioning $\mathcal{H}_{ij}, \mathcal{V}_{ij}$, and the shapes of department (w_i, h_i) and the positioning of departments (x_i, y_i) are determined through the relaxed-FBS decoding scheme. However, those obtained from the FBS procedure is near optimality and feasible, but not strictly feasible in general, since each pair of adjacent departments has no separation between them. Therefore, an additional small clearance ρ (say, 10^{-5}) is used to make them strictly feasible. The outline of starting-point generation using the relaxed-FBS is described as Algorithm 1.

Algorithm 1: Starting-point generation using the relaxed-FBS

given $\mathcal{H}_{ij}, \mathcal{V}_{ij}, \rho$
assign department $\forall i$ to the bay k with the horizontal precedence $\sum_{i=1}^{N} \mathcal{H}_{ij} = k$.
sort department $\forall i$ in the bay $\forall k$ in the order of vertical precedence $\sum_{i=1}^{N} \mathcal{V}_{ij} = k$
get bay area B_k as : $B_k = \sum_{i \in k} S_i$
get bay width W_k^B as : $W_k^B = \max_{\forall i \in k}(H/B_k, w_i^{min})$
get department widths and heights as $w_i = \min(W_k^B, w_i^{max})$, $h_i = S_i/w_i$
get department coordinates as $w_i = \min(W_k^B, w_i^{max})$, $h_i = S_i/w_i$

$$x_i = w_i + \rho, \quad \forall i \text{ with } \sum_{i=1}^{N} \mathcal{H}_{ij} = 0 \text{ (root node)}$$
$$y_i = h_i + \rho, \quad \forall i \text{ with } \sum_{i=1}^{N} \mathcal{V}_{ij} = 0 \text{ (root node)}$$
$$x_j = x_i + (w_i + w_j) + \rho, \quad \forall (i,j) \text{ with } \mathcal{H}_{ij} = 1$$
$$y_j = x_i + (h_i + h_j) + \rho, \quad \forall (i,j) \text{ with } \mathcal{V}_{ij} = 1,$$

where $\rho \in \mathbb{R}$ is an additional clearance between departments.

3.6. Overall Algorithm

We summarize the overall algorithm is as Algorithm 2.

Algorithm 2: Overall Algorithm

given t_0, A, **b**, **c**, ϵ
find initial points $(\mathbf{v}_0, \lambda_0)$ by warm-start algorithm.
repeat 1-4:
1. Find primal and dual search directions $[\Delta\mathbf{v}_k^T, \Delta\lambda_k^T]^T$ by Newton equation.
2. Find step size $\gamma_k \in \mathbb{R}$ by back-tracking
3. Update primal and dual variables $[\mathbf{v}_{k+1}^T, \lambda_{k+1}^T] := [\mathbf{v}_k^T, \lambda_k^T] + \gamma_k[\Delta\mathbf{v}_k^T, \Delta\lambda_k^T]^T$
4. Update $t_k \to t_{k+1}$
until $||R_t(\mathbf{v}_k, \lambda_k)|| \leq \epsilon$

Theoretically, the convergence proof for the primal-dual interior-point algorithm is given in several books and papers such as Boyd and Vandenberghe (2004) [44]. In practice, it is numerically difficult to converge when ϵ is very small. However, the extreme accuracy is not necessary in the context of the intent of this study. The intent of the research is to provide the decision-making support tool for the facility designers, in the phase of the *block layout* design which the "macro" flows in the facility is primarily concerned with, while the *detailed layout* is often concerned with the "micro" flows (Tompkins et al. (2010) [1]).

4. Numerical Experiments

4.1. Experimental Overview

In this chapter, we give some numerical examples to illustrate the effectiveness of the algorithm described before. All of our examples use the data following from FLP benchmark problem instances, summarized in Liu and Meller (2007) [12]. We generated the randomized relative positioning constraints for each problem, using the following two steps: giving the randomly distributed location first, then specifying the $\mathcal{H}_{ij}, \mathcal{V}_{ij}$ for $\forall i, j$ with $i \neq j$ as follows:

- if $|x_i - x_j| \geq |y_i - y_j|, \mathcal{H}_{ij} = 1$
- otherwise $\mathcal{V}_{ij} = 1$

We abandoned infeasible combinations.

The FLPRC is a convex problem and the optimal solution can be obtained. Therefore, the material handling cost is the same for all algorithms. For this reason, we only compared the results in terms of the computation efficiency. To illustrate the efficiency of our proposed elimination algorithm, we have compared the average CPU TIME for computing a Newton step for each problem with two mathematical solvers (CPLEX, NEOS). We then have compared the average CPU TIME for finding an optimal arrangement under relative positioning constraints for each problem. The parameter settings and the computer specs used to code and run are summarized in Table 2.

Table 2. The outline of experimental conditions.

OS	Mac OS X (ver. 10.7.3)
Processor	3.4GHz Intel Core i7
Memory	8GB 1333MHz DDR3
Language	MATLAB 2011Ra
α	0.5
β	0.5
t_0	10^{-4}
ϵ	10^{-3}

4.2. Experimental Results

The results of the comparison of the average CPU TIME for computing a Newton step for each problem with or without eliminations are summarized in Table 3. The results of

the comparison of the average CPU TIME for computing an optimal arrangement under relative-constraints are summarized in Table 4.

Table 3. Average CPU TIME for computing Newton Steps (μ sec).

Problem	*with* Eliminations	*without* Eliminations
O7	0.1	4.2
O8	0.1	4.5
O9	0.1	4.7
VC10	0.2	4.8
Ba10	0.2	5.3
M15a	0.2	7.8
M15s	0.2	8.7
AB20	0.2	8.9
Tam30	0.2	10.1
SC30	0.2	12.6
SC35	0.2	13.9

Table 4. Average CPU TIME for computing an optimal arrangement under relative-constraints (μ sec).

Problem	Proposal	CPLEX	NEOS
O7	14	14	15
O8	14	13	22
O9	14	22	22
VC10	15	34	59
Ba10	15	37	89
M15a	15	47	94
M15s	15	66	136
AB20	15	77	162
Tam30	16	87	207
SC30	16	102	225
SC35	16	117	320

These examples show that, by exploiting the special structure of the problem, the proposed method reduces the CPU TIME to compute Newton Steps. In our implementation, it corresponds to around [0.1–0.2] μ sec. In particular, the proposed method is much faster for large sized problems. For a FLPRC with $N = 35$, the proposed method solves the Newton equations in 0.2 μ sec, approximately 70 times faster than the generic solvers.

The sparsity pattern of the coefficient matrix of the Newton equation for SC35 is shown in Figure 2. In this example, the dimension of the coefficient matrix is $n_v + n_\lambda = 1330 + 3955 = 5285$ based on the (36) equation. However, due to the removal of some constraints from redundancy, the size of the matrix is reduced to $n_v + n_\lambda = 1330 + 3360 = 4690$. We see that the lower right block is a diagonal matrix. Therefore, by using this structure, the main computational effort is reduced to the effort of solving Equation (38) whose coefficient matrix size is $n_v = 1330$. Therefore, it is possible to find the solution much more efficiently than solving the original Equation (36).

The sparsity pattern of the coefficient matrix of Equation (38) is shown in Figure 3. As is the same with Figure 2, the lower right block is diagonal, so using this structure can be more efficient than solving Equation (38).

Furthermore, the sparsity pattern of the coefficient matrix of the Equation (42) is as shown in Figure 4. This matrix is a dense matrix only for $m = n = 1, \cdots, N$ component and $m = n = N + 1, \cdots, 2N$ component, and is a sparse matrix for the others. Therefore, the main computational load is consistent with solving this dense matrix.

Table 5 shows the calculation effort based on the calculation shown in Table 1. The required flops to compute the Newton steps are reduced a lot by the proposed block eliminations. This reduction leads to the faster CPU Time for computing Newton steps as

shown in Table 3, and then leads to the faster CPU Time for finding an optimal solution of the FLPRC as shown in Table 4.

These results indicate that the proposed method reduction solves the problem faster, by exploiting the special structure of the matrix. Since the dimension of the final dense matrix matches the number of departments, no further decomposition is possible. Therefore, there is little room for further improvement in this problem.

Table 5. Summary of complexity to solve Newton equation for SC35.

Equation	Complexity (# of Flops)
Original Newton Equation (36)	68,774,472,667
After 1st elimination (38)	133,541,333
After 2nd elimination (42)	48,477,333
After 3rd elimination (47)	971,903

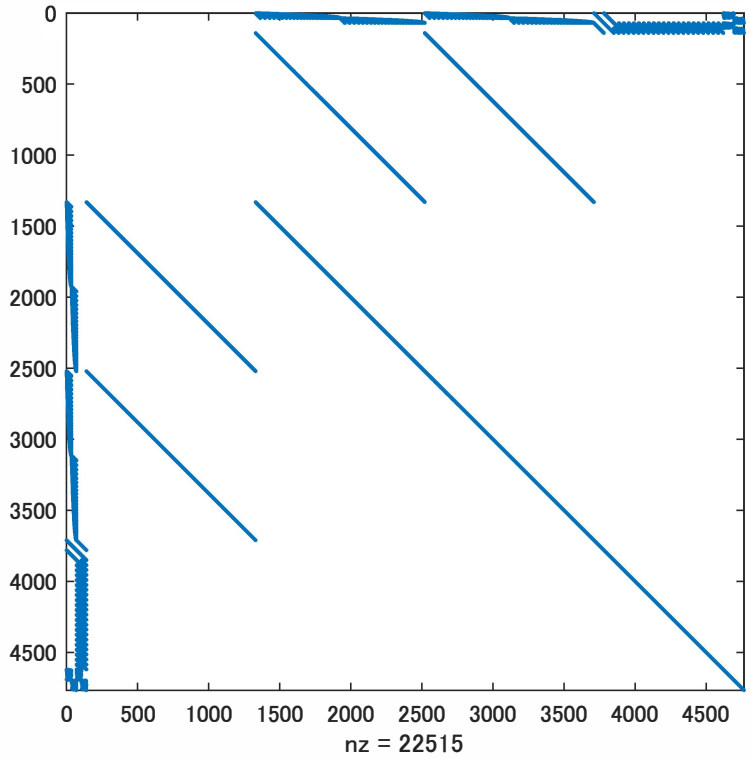

Figure 2. Sparsity pattern of the coefficient matrix of the Newton equation for SC35.

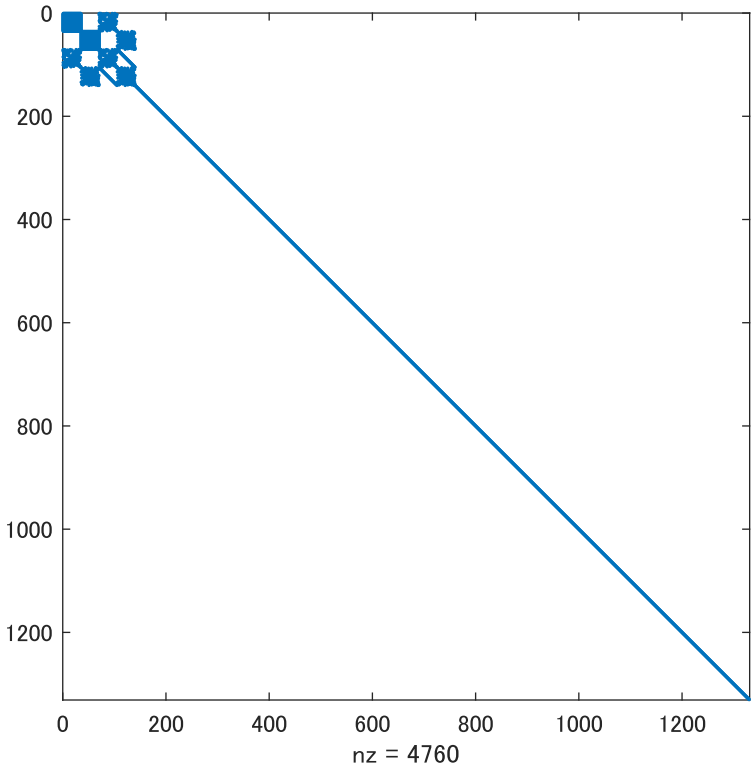

Figure 3. Sparsity pattern of the coefficient matrix after the 1st elimination for SC35.

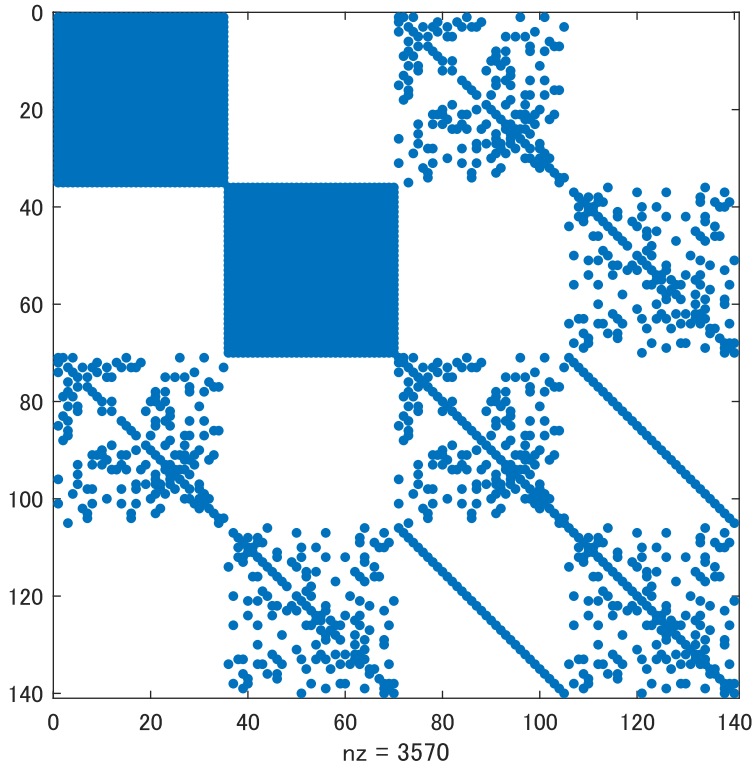

Figure 4. Sparsity pattern of the coefficient matrix after the 2nd elimination for SC35.

4.3. Discussion

The method proposed in this study is the optimization of department position and shape given a relative position $\mathcal{H}_{ij}, \mathcal{V}_{ij}$. This technique can be used as a subroutine for a solution method using MIP. This approach also includes Liu and Meller [12] and Bozer and

Wang [42], which combines MIP and MH. Among these algorithms, FLPRC is solved many times. Therefore, by using the proposed method, the calculation time can be shortened and more time can be devoted to the search of the relative positions.

The proposed method can also be used as a subroutine of the layout technique using FBS, such as [30] because FBS is used to derive the starting point in the proposed method. FBS has an encoding that divides the building into bays. Although this approach is simple, it does not necessarily guarantee optimality due to the layout that cannot be expressed by encoding. On the other hand, in the process of searching, if the proposed method is applied based on the relative position obtained by FBS, the layout of FBS can be *improved*, and there is a possibility that a better layout can be searched. In recent years, many MH approaches have used FBS and have achieved very good results. Further improvement can be considered by applying this technique.

5. Conclusions

We have considered the problem of computing an optimal arrangement of department within a facility, as measured by the minimum material handling cost, subject to the constraints for departments with respect to within-boundary, aspect-ratio, non-overlapping, and relative positioning. We have developed a primal-dual interior-point method that exploits the special structure of a problem, and is much faster than a standard method. We have proposed an efficient computation of the Newton step, using three types of block eliminations. To illustrate the effectiveness of our algorithm, we have given numerical results, using problem instances from several benchmark problems for FLP with specified relative-positioning given from randomized based rule. For a typical problem with 35 departments, the proposed algorithm is roughly 10 times faster. Using the proposed algorithm, we can accelerate a MIP (Mixed-Integer Programming) based algorithm for solving FLP, which is one of the most efficient methods for a continuous representation based problem.

Future research includes applying the proposed method to the MIP-based approach and MH-based approach. When applied to the MIP-based approach, the relaxed problem in the branch-and-bound method can be solved faster, which may solve a larger problem. The existing MIP-based approach can only solve problems for up to 12 departments. When applied to the MH-based approach, the proposed technique can be applied using the relative positional relationship of the layout obtained by FLEX-BAY or LOGIC. By doing so, the better layout can be obtained at each iteration, and thus there is possibility to get a better solution.

Author Contributions: Conceptualization, S.O. and K.Y.; methodology, S.O.; software, S.O.; validation, S.O.; writing—original draft preparation, S.O.; writing—review and editing, S.O. and K.Y.; supervision, K.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tompkins, J.A.; White, J.A.; Bozer, Y.A.; Tanchoco, J.M.A. *Facilities Planning*; John Wiley & Sons: Hoboken, NJ, USA, 2010.
2. Richard, M.; Hales, L. Systematic Layout Planning. Available online: http://hpcinc.com/wp-content/uploads/2016/07/Systematic-Layout-Planning-SLP-4th-edition-soft-copy.pdf (accessed on 9 February 2021).
3. Levary, R.R.; Kalchik, S. Facilities layout: A survey of solution procedures. *Comput. Ind. Eng.* **1985**, *9*, 141–148. [CrossRef]
4. Kusiak, A.; Heragu, S.S. The facility layout problem. *Eur. J. Oper. Res.* **1987**, *29*, 229–251. [CrossRef]
5. Hassan, M.M. Machine layout problem in modern manufacturing facilities. *Int. J. Prod. Res.* **1994**, *32*, 2559–2584. [CrossRef]
6. Meller, R.D.; Gau, K.Y. The facility layout problem: Recent and emerging trends and perspectives. *J. Manuf. Syst.* **1996**, *15*, 351–366. [CrossRef]
7. Drira, A.; Pierreval, H.; Hajri-Gabouj, S. Facility layout problems: A survey. *Annu. Rev. Control* **2007**, *31*, 255–267. [CrossRef]
8. Arikaran, P.; Jayabalan, V.; Senthilkumar, R. Analysis of unequal areas facility layout problems. *Int. J. Eng.* **2010**, *4*, 44–51.
9. Anjos, M.F.; Vieira, M.V. Mathematical optimization approaches for facility layout problems: The state-of-the-art and future research directions. *Eur. J. Oper. Res.* **2017**, *261*, 1–16. [CrossRef]

10. Hosseini-Nasab, H.; Fereidouni, S.; Ghomi, S.M.T.F.; Fakhrzad, M.B. Classification of facility layout problems: A review study. *Int. J. Adv. Manuf. Technol.* **2018**, *94*, 957–977. [CrossRef]
11. Koopmans T.C.; Beckmann M. Assignment Problems and the Location of Econmic Activities. *Econometrica* **1957**, *25*, 53–76. [CrossRef]
12. Liu, Q.; Meller, R.D. A sequence-pair representation and MIP-model-based heuristic for the facility layout problem with rectangular departments. *IIE Trans.* **2007**, *39*, 377–394. [CrossRef]
13. Van Camp, D.J.; Carter, M.W.; Vannelli, A. A nonlinear optimization approach for solving facility layout problems. *Eur. J. Oper. Res.* **1992**, *57*, 174–189. [CrossRef]
14. Anjos, M.F.; Vannelli, A. An attractor-repeller approach to floorplanning. *Math. Methods Oper. Res.* **2002**, *56*, 3–27. [CrossRef]
15. Anjos, M.F.; Vannelli, A. A new mathematical-programming framework for facility-layout design. *INFORMS J. Comput.* **2006**, *18*, 111–118. [CrossRef]
16. Jankovits, I.; Luo, C.; Anjos, M.F.; Vannelli, A. A convex optimisation framework for the unequal-areas facility layout problem. *Eur. J. Oper. Res.* **2011**, *214*, 199–215. [CrossRef]
17. Anjos, M.F.; Vieira, M.V. An improved two-stage optimization-based framework for unequal-areas facility layout. *Optim. Lett.* **2016**, *10*, 1379–1392. [CrossRef]
18. Tam, K.Y. Genetic algorithms, function optimization, and facility layout design. *Eur. J. Oper. Res.* **1992**, *63*, 322–346.
19. Gau, K.Y.; Meller, R.D. An iterative facility layout algorithm. International *J. Prod. Res.* **1999**, *37*, 3739–3758. [CrossRef]
20. Shayan, E.; Chittilappilly, A. Genetic algorithm for facilities layout problems based on slicing tree structure. *Int. J. Prod. Res.* **2004**, *42*, 4055–4067. [CrossRef]
21. Scholz, D.; Petrick, A.; Domschke, W. STaTS: A slicing tree and tabu search based heuristic for the unequal area facility layout problem. *Eur. J. Oper. Res.* **2009**, *197*, 166–178. [CrossRef]
22. Kang, S.; Chae, J. Harmony search for the layout design of an unequal area facility. *Expert Syst. Appl.* **2017**, *79*, 269–281. [CrossRef]
23. Tate, D.M.; Smith, A.E. Unequal-area facility layout by genetic search. *IIE Trans.* **1995**, *27*, 465–472. [CrossRef]
24. Kulturel-Konak, S.; Smith, A.E.; Norman, B.A. Multi-objective tabu search using a multinomial probability mass function. *Eur. J. Oper. Res.* **2006**, *169*, 918–931. [CrossRef]
25. Wong, K.Y. Solving facility layout problems using flexible bay structure representation and ant system algorithm. *Expert Syst. Appl.* **2010**, *37*, 5523–5527. [CrossRef]
26. Kulturel-Konak, S.; Konak, A. A new relaxed flexible bay structure representation and particle swarm optimization for the unequal area facility layout problem. *Eng. Optim.* **2011**, *43*, 1263–1287. [CrossRef]
27. Kulturel-Konak, S.; Konak, A. Unequal area flexible bay facility layout using ant colony optimisation. *Int. J. Prod. Res.* **2011**, *49*, 1877–1902. [CrossRef]
28. Palomo-Romero, J.M.; Salas-Morera, L.; Garcia-Hernandez, L. An island model genetic algorithm for unequal area facility layout problems. *Expert Syst. Appl.* **2017**, *68*, 151–162. [CrossRef]
29. Garcia-Hernandez, L.; Salas-Morera, L.; Garcia-Hernandez, J.A.; Salcedo-Sanz, S.; de Oliveira, J.V. Applying the coral reefs optimization algorithm for solving unequal area facility layout problems. *Expert Syst. Appl.* **2019**, *138*, 112819. [CrossRef]
30. Garcia-Hernandez, L.; Salas-Morera, L.; Carmona-Munoz, C.; Garcia-Hernandez, J.A.; Salcedo-Sanz, S. A novel Island Model based on Coral Reefs Optimization algorithm for solving the unequal area facility layout problem. *Eng. Appl. Artif. Intell.* **2020**, *89*, 103445. [CrossRef]
31. Liu, J.; Zhang, H.; He, K.; Jiang, S. Multi-objective particle swarm optimization algorithm based on objective space division for the unequal-area facility layout problem. *Expert Syst. Appl.* **2018**, *102*, 179–192. [CrossRef]
32. Guan, C.; Zhang, Z.; Liu, S.; Gong, J. Multi-objective particle swarm optimization for multi-workshop facility layout problem. *J. Manuf. Syst.* **2019**, *53*, 32–48. [CrossRef]
33. Pourvaziri, H.; Pierreval, H. Dynamic facility layout problem based on open queuing network theory. *Eur. J. Oper. Res.* **2017**, *259*, 538–553. [CrossRef]
34. Turanoglu, B.; Akkaya, G. A new hybrid heuristic algorithm based on bacterial foraging optimization for the dynamic facility layout problem. *Expert Syst. Appl.* **2018**, *98*, 93–104. [CrossRef]
35. Tayal, A.; Gunasekaran, A.; Singh, S.P.; Dubey, R.; Papadopoulos, T. Formulating and solving sustainable stochastic dynamic facility layout problem: A key to sustainable operations. *Ann. Oper. Res.* **2017**, *253*, 621–655. [CrossRef]
36. Azevedo, M.M., Crispim, J.A.; de Sousa, J.P. A dynamic multi-objective approach for the reconfigurable multi-facility layout problem. *J. Manuf. Syst.* **2017**, *42*, 140–152. [CrossRef]
37. Pourhassan, M.R.; Raissi, S. An integrated simulation-based optimization technique for multi-objective dynamic facility layout problem. *J. Ind. Inf. Integr.* **2017**, *8*, 49–58. [CrossRef]
38. Montreuil, B. A modelling framework for integrating layout design and flow network design. In *Material Handling' 90*; Springer: Berlin/Heidelberg, Germany, 1991; pp. 95–115.
39. Meller, R.D.; Narayanan, V.; Vance, P.H. Optimal facility layout design. *Oper. Res. Lett.* **1998**, *23*, 117–127. [CrossRef]
40. Sherali, H.D.; Fraticelli, B.M.; Meller, R.D. Enhanced model formulations for optimal facility layout. *Oper. Res.* **2003**, *51*, 629–644. [CrossRef]
41. Castillo, I.; Westerlund, T. An ε-accurate model for optimal unequal-area block layout design. *Comput. Oper. Res.* **2005**, *32*, 429–447. [CrossRef]

42. Bozer, Y.A.; Wang, C.T. A graph-pair representation and MIP-model-based heuristic for the unequal-area facility layout problem. *Eur. J. Oper. Res.* **2012**, *218*, 382–391. [CrossRef]
43. Chae, J.; Regan, A.C. Layout design problems with heterogeneous area constraints. *Comput. Ind. Eng.* **2016**, *102*, 198–207. [CrossRef]
44. Boyd, S.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK, 2004.