UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

School of Industrial Engineering

Development of an Optimization Method for the Material Flow Computation and the Automatic Layout Generation of Conveyor Systems

Master's Thesis

Master's Degree in Industrial Engineering

AUTHOR: Kuester, Fabian

Tutor: Rubio Navarro, Gregorio

External cotutor: MESTIRI, SLAHEDDINE

Experimental director: BRANDAU, JAN

ACADEMIC YEAR: 2022/2023

Resumen

En el sector industrial, los crecientes costos logísticos y el considerable esfuerzo requerido para diseñar configuraciones óptimas de sistemas de transporte representan desafíos significativos. La tarea de idear las mejores configuraciones y calcular los flujos de materiales para estos sistemas a menudo es laboriosa y consume muchos recursos. Las metodologías existentes se centran principalmente en la optimización de los flujos de materiales, pero no abordan adecuadamente la necesidad de una disposición óptima del sistema de transporte, lo que agrava estas dificultades.
Esta tesis introduce un enfoque innovador y bifurcado para abordar estos desafíos. Se han desarrollado dos modelos de optimización: el Modelo de Optimización de Flujo de Materiales y el Modelo de Optimización de Ubicación de Transportadores. Estos modelos consideran una variedad de entradas y restricciones, con el objetivo de agilizar el proceso de diseño del esquema y el cálculo del flujo de materiales. En consecuencia, este enfoque puede contribuir a una reducción en los costos logísticos en general, particularmente aquellos asociados con el diseño del esquema.
El Modelo de Optimización de Flujo de Materiales está diseñado para minimizar los costos totales de movimiento de materiales y el uso de curvas, que suelen ser áreas de alto desgaste y consumo de energía. El objetivo es crear un flujo óptimo de materiales que sea rentable y minimice el desgaste del sistema. Una vez calculado el flujo de materiales óptimo, se emplea el Modelo de Optimización de Ubicación de Transportadores. Notablemente, este modelo emplea la geometría de los transportadores de la vida real, reflejando con precisión las restricciones y requisitos prácticos. Su objetivo es minimizar la distancia total entre los componentes del transportador y el flujo de materiales óptimo, según lo determinado por el modelo anterior. Este enfoque mejora la eficiencia y reduce tanto el desgaste como el consumo de energía. Ambos modelos se han implementado en C#, y se ha evaluado exhaustivamente su rendimiento y calidad de la solución. Esta tesis presenta los resultados de estas evaluaciones, demostrando el potencial de los modelos propuestos para reducir significativamente los costos logísticos y optimizar los esquemas de sistemas de transporte. Al incorporar la geometría de los transportadores del mundo real y crear flujos de materiales optimizados, estos modelos aportan un enfoque revolucionado para el diseño y optimización del esquema del sistema de transportadores, estableciendo un nuevo estándar en la industria.

Palabras Clave: Optimización del Flujo de Materiales, Diseño de Sistemas de Transportadores, Reducción de Costos Logísticos, Generación Automática de Diseño, Optimización de Ubicación de Transportadores, Eficiencia Operativa.

Abstract

In the industrial sector, escalating logistics costs and the considerable effort required for designing optimal layouts of conveyor systems pose significant challenges. The task of devising the best configurations and calculating material flows for these systems is often laborious and resource-intensive. Existing methodologies primarily focus on optimizing material flows, but they inadequately address the need for optimal conveyor system layout, thus exacerbating these difficulties.

This thesis introduces an innovative, two-pronged approach to tackle these challenges. Two optimization models have been developed: the Material Flow Optimization Model and the Conveyor Placement Optimization Model. These models consider a variety of inputs and constraints, aiming to streamline the layout design process and material flow computation. Consequently, this approach can contribute to a reduction in overall logistics costs, particularly those associated with the layout design.

The Material Flow Optimization Model is designed to minimize the total costs of material movement and the usage of curves, which are typically areas of high wear and energy consumption. The goal is to create an optimal flow of materials that is cost-effective and minimizes system wear. Upon computation of the optimal material flow, the Conveyor Placement Optimization Model is employed. Notably, this model employs the geometry of real-life conveyors, accurately reflecting practical constraints and requirements. It aims to minimize the total distance between the conveyor components and the optimal material flow, as determined by the prior model. This approach enhances efficiency and reduces both wear and energy consumption. Both models have been implemented in C#, and their performance and solution quality have been thoroughly evaluated. This thesis presents the results of these evaluations, demonstrating the potential of the proposed models to significantly reduce logistics costs and optimize conveyor system layouts. By incorporating real-world conveyor geometry and creating optimized material flows, these models bring a revolutionized approach to conveyor system design and layout optimization, setting a new standard in the industry.

Keywords: Material Flow Optimization, Conveyor System Layout, Logistics Cost Reduction, Automatic Layout Generation, Conveyor Placement Optimization, Operational Efficiency.

Contents

Chapter 1

Introduction

Manufacturing operations entail an array of complex activities, the utility of which often varies widely. Precise assessment indicates that only around 5% of these activities contribute significantly to value addition. A further 35% are vital to operational flow, but do not directly enhance the product's value. More strikingly, a substantial 60% of activities provide no immediate value addition. These observations emphasize the immense potential for refining operational efficiency within the manufacturing sector. [Ler-2004]
Indeed, a significant proportion of operational expenditure in manufacturing is consumed by material handling, which constitutes between 30% and 75% of a product's cost and accounts for 20% to 50% of a company's total operating budget]Tom-1984]. Facility design and conveyor system planning, while necessary for production, do not inherently add value and are often classified as a form of waste [Sul-1991].
In addition to facility design, logistical considerations also play a pivotal role in operational efficiency. The costs associated with transportation represent a significant proportion of logistics expenses and can be classified as waste since they do not directly add value for the customer. Factors such as transportation type, travel distance, and time are crucial determinants of transportation costs. These costs can be significantly reduced by effective material flow management and strategic department distribution, underscoring the critical role of plant design and layout in minimizing waste and enhancing operational leanness. [Kov-2017]

A comprehensive investigation of both plant design and material flow is thus indispensable to optimizing manufacturing operations and reducing transportation costs.

1.1 Problem Definition

The endeavor of facility planning and material flow planning is frequently marked by substantial costs and complexity. The main contributors to these issues can be traced back to a multitude of influential factors and parameters, often making it challenging to realize an effective automation process for facility design. A prominent problem in this field is the intricacy involved in transforming real-world challenges into accurate mathematical models, a process crucial for efficient decision-making and optimization [Her-2008]. Practical problems are typically characterized by a high degree of complexity, marked by multiple, often conflicting objectives, constraints, and uncertainty in the parameters. Such problems often involve decisions about location, size, and arrangement of departments, allocation of resources, or routing of material flow, each associated with their specific sets of parameters and constraints [Ghi-2013].
Moreover, even if a satisfactory mathematical model can be derived, the ensuing problem often falls into the category of combinatorial optimization problems, specifically the NP-hard problems. These problems are notorious for their complexity, where the time required for finding the optimal solution grows exponentially with the problem size. As a result, seeking exact solutions may not be feasible within a reasonable timeframe, especially for larger and more complex facilities [Bat-2014].

1.2 Research Objectives

The complexity and gaps identified in the realms of facility planning and material flow optimization within the Problem Definition section offer opportunity to develop innovative solutions. With that in mind, the research objectives are as follows:
The primary goal is to bridge the existing gap between material flow optimization and the facility layout problem by developing an integrative and comprehensive approach. This approach will aim to encapsulate the intricate geometrical constraints of conveyor systems within optimization models, embracing non-rectangular configurations and other complex geometrical attributes typical in real-world scenarios.
This approach should be capable of managing the escalated complexity emerging from the integration of facility layout and material flow optimization, and it should also account for their respective geometric characteristics. While grappling with these complexities, it is vital to maintain a fine balance between computational efficiency and the high quality of solutions. By narrowing the gap between model projections and on-ground realities while enhancing the accuracy and applicability of models and incorporating real-world constraints, a more comprehensive and effective model can be developed.

1.3 Research Design

The study commences with Chapter 2 where a meticulous examination of the fundamental aspects of optimization problems and algorithm complexity is conducted, leading up to the identification of a conspicuous research gap. Following this, Chapter 3 forms the crux of the thesis, featuring a detailed presentation of two essential models, namely, the Material Flow Optimization Model and the Conveyor Placement Optimization Model. These models are scrutinized intricately, starting from their underlying conceptual framework, right up to their mathematical intricacies.
Subsequently, in Chapter 4, the study embarks on an exploration of the practical dimensions of the research. This chapter is dedicated to detailing the implementation of the two pivotal optimization models and offering a comprehensive evaluation of the respective algorithms. This exploration commences with a theoretical assessment, where a range of influences and conditions are critically examined. The evaluation process encapsulates a broad spectrum of factors, leading to a synthesized presentation of the performance metrics. This offers a valuable understanding of the operational nuances of the algorithms. In the latter part of this chapter, the theoretical perspectives are juxtaposed with their practical implications through a pragmatic evaluation based on two practical use cases. This process substantiates the theoretical findings, demonstrating the practicality and viability of the proposed models in practical scenarios, thus enriching the overall study.
Finally, the study culminates in Chapter 5, where a comprehensive conclusion of the research conducted will be provided, summarizing the findings and their implications. This concluding section serves as a pivotal point of reflection on the investigation thus far. Following this conclusion, the discussion then transitions into a speculative exploration of potential future developments and applications of the models and algorithms. This part of the chapter explores prospective enhancements, applicability, and potential impact in relevant domains, laying the foundation for further research in this field.

The optimization problem developed and described in this thesis consists of two parts. One is to determine the optimal material flow in a discrete facility, taking into account fixed costs (e.g., cost of installing the machine handling system) and variable costs such as routing. The other part is to place actual material handling systems, by considering their geometric feasibility and minimizing the distance from the optimal material flow. The functionality of the described optimization problem is evaluated both theoretically and practically considering the performance and the solution quality.

Chapter 2

Literature Review

This chapter unfolds with the 'Fundamentals' section, providing necessary theoretical context regarding optimization problems and the computational complexity of algorithms. This serves as groundwork for understanding the study's context and depth. The chapter then turns to 'Relevant Optimization Algorithms', where the spotlight falls on two specific optimization problems pivotal to the study: the Material Flow Optimization Problem and the Facility Layout Problem. An in-depth exploration of the existing literature on these problems underpins their relevance and establishes the study's connection to these areas.
In the 'Research Gap' section, deficiencies in current literature are highlighted, thereby identifying the space this research aims to fill. This narrative of missing links in the literature shapes the direction of this study and its objectives, indicating how it seeks to offer new solutions to known challenges. By the chapter's conclusion, a comprehensive understanding of the theoretical foundations, relevant optimization problems, and this study's driving motivation should be solidly in place.

2.1 Fundamentals

2.1.1 Optimization Problems

Optimization provides a mathematical framework for making the most effective decision given a certain objective within defined constraints. At its core, it aims to identify the best solution from all feasible options. Optimization problems pervade a broad range of fields, including but not limited to, economics, engineering, logistics, manufacturing, computer science, and physics. Each field employs optimization techniques to optimize conditions such as efficiency, cost, or any other parameter to align better with their desired goals. [Kre-1986]
Consider the logistics sector, where an optimization problem might involve determining the most efficient route for a delivery vehicle to deliver goods, given constraints such as road networks and delivery times - an instance of the renowned Traveling Salesman Problem. In manufacturing, optimization aids in identifying the most cost-effective allocation of resources in a production process to minimize waste and cost while maximizing output.
Every optimization problem comprises three primary components: the decision variables, the

objective function, and constraints. The basic structure of an optimization problem, demonstrated in Equation (2.1), encompasses the objective function $F(x)$, the decision variable x, and a set of constraints $g(x)$ [Ger-2011].

$$\min z = F(x) \qquad g_i(x) = \left\{ \begin{matrix} \leq \\ = \\ \geq \end{matrix} \right\} \quad 0 \quad i = 1, \ldots, m. \tag{2.1}$$

With the principle of duality, a minimization problem can be easily converted to a maximization problem and vice versa [Gri-2018].

Decision Variables: These represent adjustable quantities to optimize the objective function. For instance, in a production optimization problem, decision variables could denote the quantities of various products manufactured. In logistics, the routes taken by delivery vehicles could serve as decision variables.

Objective Function: The objective function symbolizes the quantity to be optimized. Depending on the problem context, this quantity needs to be either minimized or maximized. Objective functions vary widely, ranging from minimizing operational costs in a business or distance traveled in a logistics problem to maximizing profits or process efficiency.

Constraints: Constraints, expressed as equations, represent the limitations or restrictions of the problem. They could be physical limitations (like resources or capacity limits), financial restrictions (such as budgets), or defined by rules and regulations related to the problem context. Constraints define the boundaries of the feasible solution space, which means they stipulate the allowable combinations of decision variable values.

Despite their apparent simplicity, optimization problems can rapidly evolve into complex NP-Complete or NP-Hard problems due to the size of the feasible set, the nature of the objective function, or the constraints involved. Their solutions often require a myriad of mathematical and computational techniques. Furthermore, real-world problems often involve multiple, potentially conflicting objectives. For instance, cost minimization could conflict with maximizing quality since improving quality might lead to increased costs. [Mie-2012]

2.1.2 Computational Complexity of Algorithms

In computational theory, problems are often classified into categories, which are primarily determined by their inherent computational complexity. This complexity signifies the time required for a deterministic Turing machine to verify a solution rather than find one. Among these categories, the most fundamental are Polynomial Time (P) and Nondeterministic Polynomial Time (NP), with the latter further subdivided into NP-Complete and NP-Hard. [Aro-2009]

Problems belonging to the P class are those for which solutions can be verified within polynomial time, effectively making them amenable to efficient algorithmic solutions. Some representative problems falling under this class include linear search, matrix multiplication, and identification of minimum or maximum elements within an array. [Coo-1971]

Contrarily, the NP class encapsulates decision problems where a solution, once postulated, can be verified in polynomial time. However, discovering these solutions may necessitate an exponential amount of time. Among the NP class, a subset called NP-Complete contains the most 'difficult'

problems. A polynomial-time solution to any NP-Complete problem would imply a polynomial-time solution to all problems in NP, serving as a benchmark for the hardness of other problems in NP [Kar-1972]. Notable examples of NP-Complete problems consist of the partition problem, the knapsack problem, and the decision version of the traveling salesman problem [Gar-1978].

Finally, NP-Hard problems constitute those that are at least as challenging as the most complex problems in NP. These problems may or may not fall under NP themselves and are not necessarily decision problems. Prime examples of NP-Hard problems include the job sequencing problem, the problem of non-existence of a Hamiltonian cycle, and the optimization version of the traveling salesman problem [Gar-1979].

Typically, the optimization variant of a problem is inherently more complex than its decision-based counterpart. However, it is essential to note that the complexity class of a problem doesn't always dictate its practical difficulty. Even P class problems can be demanding in a practical scenario if the polynomial's degree is high. Conversely, certain NP-Hard problems can often be solved in a relatively shorter duration using heuristics or approximation algorithms, despite the absence of a universally applicable polynomial-time algorithm. [Aro-2009]

2.2 Relevant Optimization Algorithms

In this section, a detailed exploration into two foundational optimization problems is conducted: the Material Flow Optimization Problem and the Facility Layout Problem. These problems have been chosen due to their fundamental relevance to the research presented in this thesis. A comprehensive literature review of both problems is undertaken, providing an understanding of the current state of the art, and insight into how these problems have been addressed previously. This is pivotal in forming the contextual backdrop against which this thesis is positioned. Not only is the basic theory of these problems elucidated, but the scope is broadened to include an examination of significant extensions and applications. This holistic approach ensures a robust understanding of the problems, necessary for the subsequent development of novel solutions and approaches.

2.2.1 Material Flow Optimization Problem

Material flow optimization problems can be traced back to various classical optimization problems, the most notable of which is probably the Transportation Problem, first formulated and solved by George Dantzig [Dan-1951]. The Transportation Problem, in its simplest form, aims to minimize the cost associated with the transport of a product from several origins, to numerous destinations. This is accomplished while keeping into account the known supply at each origin and the demand at each destination.

When viewed in the context of material flow optimization, this problem can be formulated as a multicommodity fixed-charge capacitated network. This particular concept was introduced and heuristically solved by J. Wo Herrmann et al. [Her-1995]. The problem involves optimizing the flow of materials between various departments within a specific area, often a manufacturing plant or warehouse. It's formulated on a unit grid, where the points (or nodes) on the grid represent departments or storage areas and the lines (or arcs) connecting these nodes represent possible paths for material flow.

Each arc has a predefined maximum capacity, which represents the maximum quantity of material that can be transported along that path. Furthermore, there are two types of costs associated with each arc - a variable transportation cost that depends on the amount of material being moved and a fixed cost that is incurred whenever the arc is used, regardless of the amount of material transported.
These elements can be mathematically described using several variables:

N	is a set of grid nodes
$k \in K$	Each k represents a commodity or type of material that is being moved.
O_k	Origin of commodity (k)
D_k	Destination of commodity (k)
f_k	The flow of commodity k
$(i,j) \in A$	A directed arc between departments i and j
c_{ij}^k	The per-unit transportation cost of moving commodity k along arc (i,j)
B_{ij}	The capacity of arc (i,j)
x_{ij}^k	The amount of commodity k that is transported along arc (i,j).
$\{i,j\} \in \bar{A}$	An undirected arc between departments i and j
F_{ij}	The fixed charge associated with $\{i,j\} \in \bar{A}$
y_{ij}	A binary decision variable that equals 1 if $\{i,j\}$ is used, and 0 otherwise

The objective of the problem, as given in Equation 2.2, is to minimize the total cost of transporting all commodities by varying the decision variables x_{ij}^k and y_{ij}. This cost is composed of two parts: the total variable cost of moving each commodity along the direct arcs, and the total fixed cost associated with the use of the indirect arcs.

$$\min z = \sum_{k \in K} f_k * \sum_{(i,j) \in A} c_{ij}^k * x_{ij}^k + \sum_{\{i,j\} \in \bar{A}} F_{ij} * y_{ij} \tag{2.2}$$

The problem also includes several constraints to ensure that the solution is feasible and practical, namely the flow conservation (Equation 2.3), the capacity limit of arcs (Equation 2.4), the arc selection (Equation 2.5), non-negativity of the flow (Equation 2.6), and binary decision variable constraints (Equation 2.7).

$$\sum_{j \in N(i)} x_{ji}^k - \sum_{l \in N(i)} x_{il}^k = \begin{cases} -1 & \text{if } i = O_k \\ 1 & \text{if } i = D_k \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in N, \forall k \in K \tag{2.3}$$

$$\sum_{k \in K} f_k * \left(x_{ij}^k + x_{ji}^k \right) \leq B_{ij} \quad \forall \{i,j\} \in \bar{A} \tag{2.4}$$

$$x_{ij}^k, x_{ji}^k \leq y_{ij} \quad \forall \{i,j\} \in \bar{A}, k \in K \tag{2.5}$$

$$x_{ij}^k \geq 0 \quad \forall (i,j) \in A, k \in K \tag{2.6}$$

$$y_{ij} \in \{0,1\} \quad \forall \{i,j\} \in \bar{A} \tag{2.7}$$

To delve deeper, the goal of the objective function (Equation 2.2) is to drive the total transportation costs to a minimum, which comprises both variable costs associated with moving each commodity along directed arcs and fixed costs affiliated with the usage of undirected arcs. The set of constraints act as guardrails to ascertain the feasibility of the proposed solution. More specifically, the flow conservation constraint (Equation 2.3) maintains a balance of inflow and outflow at each node, differentiated by whether the node acts as an origin, destination, or neither. The capacity constraint (Equation 2.4) imposes an upper bound on the flow passing through each arc, aligned with its capacity. The arc selection constraint (Equation 2.5) stipulates that transportation along an arc is permissible only if the arc is activated. Further, the non-negativity condition (Equation 2.6) mandates that the flow of commodities is always non-negative, i.e., commodities can only be transported in one direction along an arc. Lastly, the binary constraint (Equation 2.7) establishes that the decision variable y_{ij} is binary in nature, indicating whether a specific arc is utilized or not.

Different methods offer diverse approaches to address complex models in material flow problems. Mixed-integer programming is valuable when managing discrete variables, such as the number of machines or vehicles, that are integral to such problems [Wol-1998]. Heuristic algorithms, inspired by biological processes, are extensively used to solve complex optimization problems. For instance, genetic algorithms have been effective in scheduling and layout optimization [Shc-2020]. Ant colony optimization, rooted in the foraging behavior of ants, is adept at solving routing issues [Bul-1999]. Additionally, the artificial bee colony algorithm, as shown in the study of Alvarado-Iniesta et al., can optimize the distribution of materials in a manufacturing plant [Alv-2013]. All these techniques underline the potential of utilizing biologically inspired algorithms in material flow optimization.

Modifications of the basic model introduced the concepts of network flow problems and multi-commodity flow problems. In these problems, the material is often conceptualized as flowing through a network of nodes and arcs, with the objective being to maximize the overall flow (in a max-flow problem), or to minimize the cost of achieving a certain flow (in a min-cost flow problem) [Gol-1989; Cha-2019]. Simulation models have been increasingly utilized to handle complex scenarios where analytical models are hard to formulate or solve. Simulation models can capture the dynamic interactions among system components, making them ideal for understanding and optimizing complex material flow systems [Dra-2017; Pek-2020; Moe-2009].

Over the years, there has been a growing trend to integrate material flow optimization with other decision-making areas, for example, facility layout or facility location planning. In an integrated approach, the layout of facilities is determined simultaneously with the optimization of material flow. This approach allows for a holistic view of the system and often leads to more efficient solutions [Ioa-2007]. The "Milk run" system is another example where material flow optimization is combined with transportation planning. In a Milk run system, a single transport vehicle stops at several locations to pick up or deliver goods, as opposed to sending a vehicle to each location separately. This approach is common in the automotive industry and has been shown to streamline the material collection and distribution process [Sim-2021].

In recent years, the advent of Industry 4.0 has brought about the application of machine learning and data analytics to material flow optimization. These advanced technologies enable predictive modeling, real-time decision-making, and automation of complex processes [Mon-2016]. There is

also a growing awareness of the environmental impact of manufacturing and logistics operations. This has led to the development of new models and methods that aim to reduce environmental impact, for example, by minimizing energy consumption or reducing waste [Kan-2003], or by combining dynamic material flow analysis with life cycle optimization for a sustainable design of energy systems, resulting in a multi-objective mixed-integer linear fractional programing problem [Gao-2018].

2.2.2 Facility Layout Problem

The Facility Layout Problem (FLP), originally formulated by Armour and Buffa, seeks to establish the optimal arrangement of distinct departments within a given space, such as a factory or warehouse, in order to minimize the overall material handling cost [Gar-1963]. The problem has been iteratively refined over time, incorporating critical factors such as worker safety, ergonomics, operational efficiency, and flexibility.
In terms of mathematical formulation, the departments within the facility are denoted as nodes, while the material flows between these departments are represented as arcs. Every arc is associated with a certain cost, embodying the expense incurred in transferring materials between departments. Importantly, this cost is dependent on two variables: the quantity of material being transported and the spatial distance between the departments involved. Consequently, the central objective of the FLP is to find an optimal configuration of the departments (nodes) that leads to the least total cost. This optimal configuration should take into account not only the cost of material transfer but also the distances between departments and the quantity of material flow. The ultimate goal is a layout that efficiently integrates these variables to minimize overall expenses and improve operational effectiveness. Usually the used layout is considered either discret or continous as seen in Figure 2.1. In the following two common methods of defining a FLP, discrete and continuous as introduced by J. Balakrishnan et al. and M. Mir and M.H. Imam respectively, will be illustrated [Bal-2003; Mir-2001].

Figure 2.1: FLP Layout Considerations (a) Discrete (b) Continuous. (cf. [Dri-2007])

Discrete Formulation
Lets denote:

N	is the number of facilities in the layout
X_{ij}	is a binary variable that equals 1 if facility i is located at location j, otherwise 0
f_{ik}	is the flow cost between two facilities i and k
d_{jl}	is the distance between two locations j and l

When formulating the problem using a discrete layout it becomes a Quadratic Assignment Problem, (QAP) with an objective function as seen in Equation (2.8).

$$\min z_d = \sum_{\substack{i=1 \\ i\neq k}}^{N} \sum_{\substack{j=1 \\ j\neq l}}^{N} \sum_{k=1}^{N} \sum_{l=1}^{N} f_{ik} * d_{jl} * X_{ij} * X_{kl} \tag{2.8}$$

$$\sum_{i=1}^{N} X_{ij} = 1 \qquad \forall j \in N \tag{2.9}$$

$$\sum_{j=1}^{N} X_{ij} = 1 \qquad \forall i \in N \tag{2.10}$$

Equation (2.9) and (2.10) are two constraints that ensure that each location contains only one facility (2.9) and that each facility is placed only in one location (2.9).

Continuous Formulation

In the case that the layout is defined continuously as seen in Figure 2.1b, the objective function can be written in a Mixed Integer Programming formulation (MIP), seen in Equation (2.11) using the following variables:

N	is the number of rectangular facilities of unequal areas
(x_i, y_i)	are the coordinates of the centroid of facility i
L_i	is the length of facility i
W_i	is the width of facility i
f_{ij}	is the flow between two facilities i and j
d_{ij}	is the distance measured between the centroids of facilities i and j
A_{ij}	is the overlap area between two facilities i and j

$$\min z_c = \sum_{i=1}^{N} \sum_{j=i}^{N} f_{ij} * d_{ij}((x_i, y_i), (x_j, y_j)) \tag{2.11}$$

$$A_{ij} \leq 0 \tag{2.12}$$

$$A_{ij} = \lambda_{ij} * (\Delta X_{ij})\,(\Delta Y_{ij}) \tag{2.13}$$

$$\Delta X_{ij} = \lambda_{ij} * \left(\frac{L_i + L_j}{2}\right) - |x_i - x_j| \tag{2.14}$$

$$\Delta Y_{ij} = \lambda_{ij} * \left(\frac{W_i + W_j}{2}\right) - |y_i - y_j| \tag{2.15}$$

$$\lambda_{ij} = \begin{cases} -1 & \text{for } \Delta X_{ij} \leq 0 \text{ and } \Delta Y_{ij} \leq 0 \\ +1 & \text{otherwise} \end{cases} \tag{2.16}$$

The distance d_{ij} is usually defined in on of the three ways seen in Equations (2.17) to (2.19).

Euclidean Distance: $$d_{ij} = ((x_i - x_y)^2 + (y_i - y_j)^2)^{1/2} \tag{2.17}$$

Squared Euclidean Distance: $$d_{ij} = (x_i - x_y)^2 + (y_i - y_j)^2 \tag{2.18}$$

Rectilinear Distance: $$d_{ij} = |x_j^{\mathrm{I}} - x_i^{\mathrm{O}}| \ + |y_j^{\mathrm{I}} - y_i^{\mathrm{O}}| \tag{2.19}$$

As demonstrated in Equations (2.13) to (2.16), if facility i and facility j are overlapping the defined overlap area will be positive, but to avoid exactly this case, constraint (2.12) has been introduced.

Facility Layout Problems aim at the optimal arrangement of facilities within a specific area to minimize transportation cost or distance. Originating from simple deterministic and static models with few facilities and constraints [Boz-1997], FLP has evolved considerably, reflecting the escalating complexity and uncertainty of real-world circumstances. A multitude of factors influence the characterization of FLP. These include, but are not limited to, workshop characteristics like specifics of manufacturing systems, facility shapes, material handling systems, and layout evolution. Layout configuration, problem formulation, objectives, constraints, and resolution approaches are also central aspects to consider. [Dri-2007; Pal-2017]

In response to the dynamic nature of supply and demand, probabilistic models have been developed [Kul-2012; Aze-2017]. Moreover, various techniques such as mixed-integer programming and genetic algorithms have been leveraged to address more complex FLPs [Dri-2007]. A noteworthy contribution is the Island Model Genetic Algorithm (IMGA) proposed by J.M. Palomo-Romero et al., capable of handling a wide range of problem sizes in reasonable computational time [Pal-2017].

The field has seen an increasing integration of FLP with other areas such as material flow optimization, fostering more comprehensive and efficient solutions [Ioa-2007]. The layout of warehouses or manufacturing plants, for instance, profoundly influences their operational efficiency and cost [Tom-2010; Her-2008]. In a similar vein, the design of material handling systems interplays with facility layout, both significantly affecting the manufacturing system performance [All-1984]. Peters and Yang extensively explored this integration problem [Bre-1997].

Industry 4.0 heralds new developments in FLP. Enabling technologies like Internet of Things, machine learning, and data analytics are being harnessed to facilitate real-time decision-making and automation. In this context, Kumar et al. proposed a simulated annealing-based approach integrated with principal component analysis for sustainable robust stochastic cellular FLP [Kum-2018], and Tayal et al. put forth a three-stage methodology involving data envelopment analysis coupled with machine learning [Tay-2020]. R. Wang et al. demonstrated the development of a digital twin-based approach with the novel concept of iterative optimization between static design and dynamic execution, strengthening the predictive capabilities of digital simulations [Wan-2019]. Further, the rising importance of sustainability has led to the emergence of green facility layout models aiming to minimize environmental impact, such as energy consumption or waste generation [She-2021]. These promising directions are set to continually enhance FLP's capacity and performance in the future.

2.3 Research Gap

The existing literature on facility layout and material flow optimization highlights many modifications and applications addressing various logistical and layout issues. While most research focuses on either material flow or facility layout, a few studies integrate both aspects. This bifurcated focus creates a conspicuous research gap, particularly when the geometric aspects of layout design and conveyor systems come into consideration.

In the context of combined facility layout and material flow optimization problems, the prevailing approach often overlooks the geometric attributes of the material handling system or transportation system. Instead, the focus remains on the geometry of the facilities or departments, utilizing material flow merely as a flow variable, such as costs or weights. This simplification facilitates mathematical formulation and computation, but it inadequately reflects the complexities inherent in real-world scenarios.

Regarding facility layout optimization, many studies assume all facilities or departments to be rectangular. However, real-world facilities often exhibit irregular shapes that are not addressed by this assumption. While some researchers have broadened their investigations to include non-rectangular facilities — including those with L-shaped, circular, or elliptical configurations — the real-world physical constraints imposed by facility geometry remain largely unexplored.

Similarly, material flow optimization traditionally abstracts the problem to a graph, composed of arcs and nodes, which does not take into account the actual geometry of the facilities. Even though certain works, such as G. Ioannou's, have attempted to bridge this gap by integrating different optimization problems — including quadratic assignment, fixed charge capacitated network design, and non-depot distance-constrained vehicle routing — into a comprehensive shop design problem, these attempts still fall short of the intricacies of practical applications [Ioa-2007].

In essence, there is a conspicuous absence of research that marries both layout optimization with non-rectangular facilities and material flow optimization that considers the geometry of the conveyor systems. Ideally, a holistic approach would incorporate not only theoretical material flow and facility layout but also the actual physical conveyor system. Such an integrated perspective is vital to formulate solutions that are both practical and optimal.

The lack of consideration for specific conveyor attributes and constraints, including straight sections, curves, merges, and diverts, simplifies the problem from a computational standpoint. However, it diminishes the practical utility and applicability of the derived solutions. As a result, the layouts proposed by existing models may prove suboptimal or even unfeasible when implemented in real-world settings. Thus, there is a compelling need for research that bridges this gap, addressing the geometric complexities of both facility layouts and conveyor systems within an integrated optimization framework.

To address this gap, this research proposes the creation of an optimization model that first computes an optimal material flow and then uses this result to guide the placement of actual conveyor components, with careful consideration of their unique geometrical attributes. This model is visualized in Figure 2.2.

By minimizing the total distance between the placement of these components and the optimal material flow line as illustrated by the arrow in the presented visualization, the proposed model

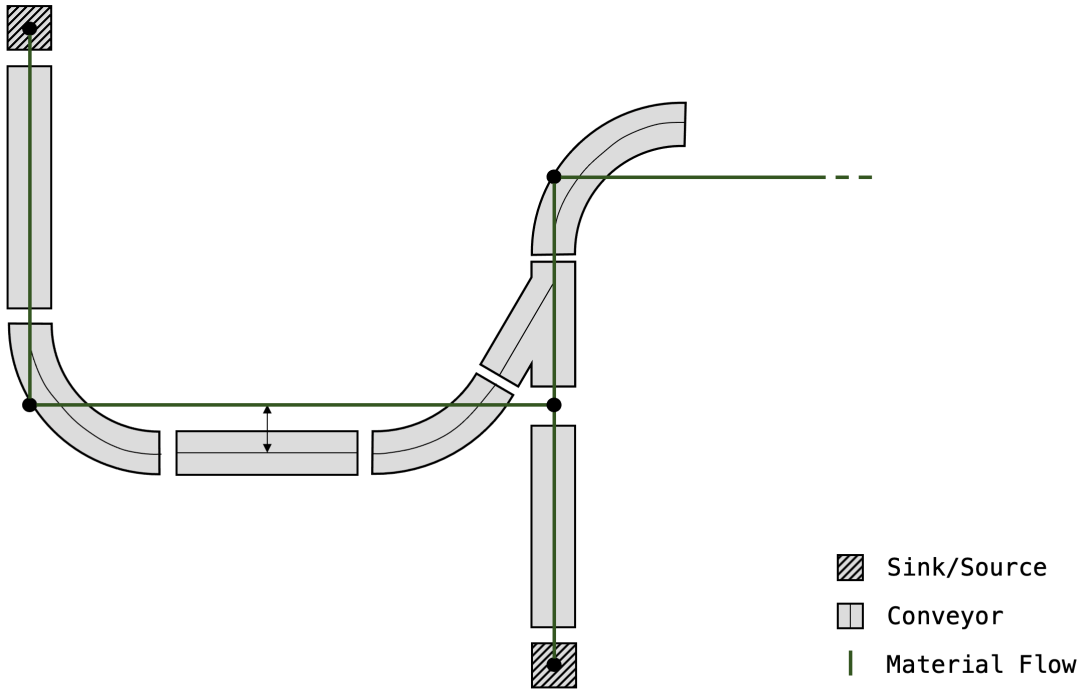


Figure 2.2: Visualization of the Basic Structure of the Optimization Model

aims to provide a more realistic, efficient, and implementable solution to conveyor system layout design. In doing so, this work seeks to bridge the gap between the theoretical elegance of mathematical optimization and the practical constraints of physical design.

Chapter 3

Mathematical Formulation of the Optimization Problems

In this chapter, we delve into the mathematical optimization models. The objective is twofold: first, to optimize the flow of materials within a complex system, and second, to optimize the placement of conveyors along the material flow lines. Each of these issues forms the core of an individual section in this chapter.
The first section focuses on the enhanced material flow optimization model. This model builds on prior work but introduces novel aspects, such as curve usage minimization and customizable objective functions. Representing the system as a unit grid network, it offers a robust mathematical tool for resource management, ensuring efficient flow of multiple commodities across the network. The second section shifts the attention to the conveyor placement problem. Leveraging the output of the material flow optimization model, it formulates a mathematical model to place actual conveyors along the computed material flow lines. The goal is to streamline the transfer process, placing the conveyors as close as possible to the material flow, thereby increasing operational efficiency. Each section will detail the mathematical formulation of these models, their intrinsic characteristics, constraints, and the underlying assumptions. Together, they form a comprehensive approach to optimize resource management in complex systems, from the abstract flow of commodities to the physical placement of conveyors.

3.1 Material Flow Optimization Model

Optimization techniques, particularly those addressing material flow, are pivotal in efficient resource management and waste minimization within complex systems. The mathematical model for material flow optimization, initially introduced by George Dantzig [Dan-1951], has been integral to identifying optimal solutions to complex logistical challenges.
However, traditional models primarily focus on cost per flow fraction and the fixed charge of an arc. To address this limitation, an enhanced model has been developed, which incorporates curve usage minimization. This refers to the minimization of the usage of combinations of two arcs forming a curve. Moreover, additional weights have been integrated to customize the objective

function, tailoring the model to specific use cases. This enhanced model is designed to calculate the most efficient material flow within a multi-commodity system, represented by a unit grid network. This network comprises nodes, each node symbolizing a key point within the material flow process, such as a source, sink, or transfer point. In this design, every node is interconnected to its neighbor-node via two directed arcs, establishing bidirectional flow. This configuration enables the movement of commodities in any desired direction within the system, effectively modeling the complexity of the material flow process.
The proposed graph representation allows the exploration of multiple routes and accounts for the nuanced logistical and operational constraints inherent in the material flow optimization problem. This section will elucidate the formulation of this enhanced model.

Main Variables:
Let's denote:

N	is a set of grid nodes
$\bar{A}$	is a set of undirected arcs $\{i, j\}$
A	is a set of directed arcs $(i, j), (j, i)$
$\mathcal{C}$	is the set of all existing curves c
(a, b)	is a pair of arcs that form a curve
K	is a set of commodities k
f_k	is the flow for commodity k within a certain time horizon
O_k	is the origin of commodity k
D_k	is the destination of commodity k
u_{ij}	is the capacity of arc $\{i, j\}$
F_{ij}	is the fixed charge of construction the arc $\{i, j\}$ if chosen
L_{ij}	is the length of arc $\{i, j\}$
c_{ij}^k	is the cost of routing parts of the commodity k across the arc $\{i, j\}$
P	is the penalty for placing a curve
α	is the weight for the cost part in the objective function
β	is the weight for the curve part in the objective function
γ	is the weight for the length part in the objective function

Decision variables
In order to formulate the enhanced material flow optimization model, several decision variables are introduced. First, the variable x_{ij}^k is defined to represent the fraction of the flow of commodity k that transits through the arc (i, j). Next, we introduce the variable y_{ij}, which is a binary variable designed to denote the utilization of a specific arc in the final solution. It is set to 1 if the arc is employed, and 0 otherwise. Lastly, to capture the usage of a curve, we define another binary variable, z_c. This variable serves as an indicator for curve usage in the solution.
These decision variables provide the foundational structure necessary to formulate and solve the enhanced material flow optimization problem.

$$x_{ij}^k \quad \in [0, \text{inf}] \quad \forall k \in K, \forall (i, j) \in A \tag{3.1}$$

$$y_{ij} \quad \in [0, 1] \quad \forall (i, j) \in A \tag{3.2}$$

$$z_c \quad \in [0, 1] \quad \forall c \in \mathcal{C} \tag{3.3}$$

Objective Function

The following objective function can be adjusted by setting the weights to either minimize the total costs, the amount of curves, the total length of the conveyor system or a combination of it.

$$\min z_1 = \sum_{k \in K} f_k * \sum_{(i,j) \in A} c_{ij}^k * x_{ij}^k + \alpha * \sum_{(i,j) \in A} F_{ij} * y_{ij} + \beta * \sum_{c \in Z} P * z_c + \gamma * \sum_{(i,j) \in A} L_{ij} * y_{ij} \quad (3.4)$$

Subject to

Flow conservation constraint:

$$\sum_{j \in N(i)} x_{ji}^k - \sum_{l \in N(i)} x_{il}^k = \begin{cases} -f_k & \text{if } i = O_k \\ f_k & \text{if } i = D_k \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in N, \forall k \in K \quad (3.5)$$

Flow direction constraint:

$$y_{ij} + y_{ji} \leq 1, \quad \forall (i,j) \in A \quad (3.6)$$

Capacity constraint:

$$\sum_{k \in K} * \sum_{(i,j) \in A} x_{ij}^k \leq y_{ij} \cdot u_{ij}, \quad \forall (i,j) \in A \quad (3.7)$$

Curve selection constraint:

$$y_a - z_c \geq 0 \qquad \forall c \in \mathcal{C} \quad (3.8)$$

$$y_b - z_c \geq 0 \qquad \forall c \in \mathcal{C} \quad (3.9)$$

$$y_a + y_b - 2 * z_c \leq 1 \quad \forall c \in \mathcal{C} \quad (3.10)$$

with y_a and y_b being the binary decision variables that represents the selection of the two arcs that form the curve c.

Flow conservation constraints

Equation (3.5) ensures that the total flow of commodities entering and exiting each node in the network adheres to the laws of conservation. This is achieved by iterating through each commodity and node, computing the total inflow and outflow at each node, and ensuring it adheres to specific rules. For each node and each commodity, the constraint calculates a flow expression. If an arc starts at the node, the associated flow of the commodity along this arc is subtracted from the flow expression. If the arc ends at the node, the associated flow is added. This captures the net flow of each commodity at each node. The nature of the constraint differs depending on whether the node is a source node, a sink node, or neither: At a source node of a commodity (where the commodity is produced), the net flow equals the negative total flow of the commodity. This is because the total flow is leaving the source and there is no incoming flow. At a sink node of a commodity (where the commodity is consumed), the net flow equals the total flow of the commodity, signifying that the total flow of the commodity is incoming and there is no outgoing flow. At any other node, the net flow of each commodity must equal zero, meaning the total flow entering the node must equal the total flow exiting the node, signifying that there is no

accumulation or shortage of the commodity at the node. The constraint ensures that the total incoming flow minus the total outgoing flow of commodity k at each node i must be equal to $-f_k$ at the source node, f_k at the sink node, and 0 at all other nodes.

Flow-direction constraint
Another important constraint is the flow-direction constraint. It is implemented to ensure that within the network flow can only occur in one direction between any two nodes. This prevents any situation where commodities are flowing in both directions between two nodes. This constraint, formulated in Equation (3.6) is implemented by iterating through all pairs of arcs, looking for pairs where one arc represents a flow from node i to node j and the other represents a flow from node j to node i. If such a pair of arcs is found, a constraint is added to the model to ensure that at most one of these arcs can be selected for flow. The sum of the binary variables representing the selection of these arcs is constrained to be less than or equal to 1, meaning that either none of the arcs or only one of them can be selected.

Capacity constraint
In Equation (3.7), the capacity constraint is shown. It ensures that the total flow of commodities along any given arc in the network does not exceed the arc's capacity. The constraint is implemented by iterating through all the arcs in the network. For each arc, a linear expression is created that sums up the flow of each commodity along the arc. Each term in the sum is the product of the flow of the commodity and a binary variable representing whether the arc is selected for that flow. This sum represents the total flow of commodities along the arc.
The model then constrains this total flow to be less than or equal to the product of the arc's capacity and the binary variable indicating whether the arc is selected. If the arc is not selected (binary variable equals 0), the right-hand side of the constraint becomes 0, and no flow is allowed along the arc. If the arc is selected (binary variable equals 1), the right-hand side of the constraint equals the arc's capacity, meaning the total flow along the arc must not exceed this value.

Curve selection constraint
Equations (3.8) to (3.10) constraints ensure that if curve c is selected (i.e., $z_c = 1$), then both arcs a and b that form this curve are also selected (i.e., $y_a = y_b = 1$). If either of the arcs a or b is not selected, then the curve cannot be selected either. The third constraint ensures that if both arcs are selected, then the curve is also selected. This encourages the optimization process to avoid selecting curves when the objective function minimizes the number of curves used.

3.2 Conveyor Placement Optimization Model

In the second section of this chapter, we concentrate on the optimization of conveyor placement in the context of the computed material flow. This problem is considerably complex due to the intricate geometrical configurations of the different types of conveyors - straight, curves, merges, diverts, as well as source and sink conveyors - each with their unique properties. The goal of this model is to optimize the placement of these conveyors such that the overall distance to the material flow is minimized, thereby enhancing efficiency. A vital aspect of this process involves

approximating trigonometric methods to calculate this distance. Moreover, the model must take into account various constraints to ensure feasible placement of the conveyors. This includes accounting for the physical reality of the system, like preventing intersection of conveyors with obstacles.
This section will elaborate on the mathematical formulation of this model, focusing on how it calculates the distance to the material flow for each conveyor, the approximation methods used, and the constraints incorporated to ensure practical feasibility. The detailed description will illustrate how this model can solve real-world conveyor placement problems, taking into account the geometry of the conveyors, the characteristics of the material flow, and the physical constraints of the environment.

In Figure 3.1, the main decision variables needed to formulate the optimization model are demonstrated using a divert conveyor as an example. The model enables the selection of the conveyor?s start position, in the case of the divert defined as the beginning of the center line of the straight part of the conveyor, using the x and y coordinates, and determines its orientation at the starting point by the angle between the x-axis and its center line. All the additional points, required to formulate the constraints, can be calculated using these decision variables and the conveyor's specific geometry.

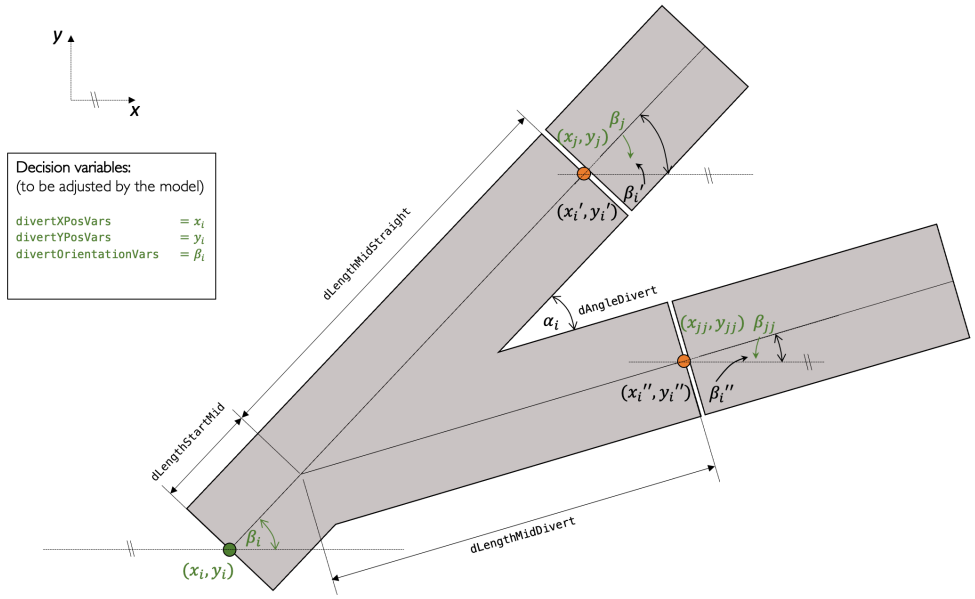


Figure 3.1: Main Decision Variables illustrated on the Example of a Divert Conveyor

Main Variables:

Let's denote:

N_{ML}	is a set of merge left nodes
N_{MR}	is a set of merge right nodes
N_{DL}	is a set of divert left nodes
N_{DR}	is a set of divert right nodes
N_{CL}	is a set of curve left nodes
N_{CR}	is a set of curve right nodes
N_{So}	is a set of source nodes

N_{Si}	is a set of sink nodes
A_{adj}	is a set of the adjusted arcs
tol_{Conv}	is the tolerance around a node, specified individually for each type of conveyor
$dist^{conv}$	is the distance of a specific conveyor to its respective node
$dist_{Conv}$	is the sum of the distance of a specific conveyor group $dist^{conv}$
$iLimX$	is the limit in the x axis
$iLimY$	is the limit in the y axis
$w_{so,si}$	is the penalty for the distance from sources and sinks to their respective nodes
tol	is the tolerance for the position and orientation difference between two conveyors
ϵ	is a small positive number to prevent boundary conditions
M	is a large positive number to formulate constraints using the "Big-M" formulation

Decision variables

For each specific type of conveyor, three distinct variables are required: one for the x-coordinate, one for the y-coordinate, and another for its orientation. As previously mentioned, for the case of merges, diverts, and curves, these variables are further differentiated into right and left.
To guarantee the accurate placement of all conveyors in proximity to their respective nodes, we define the following lower and upper bounds. In the case that these tolerance areas intersect with obstacles they get adjusted.

$$convXPosLB = \max(XPosNode - tol_{Conv}, 0) \tag{3.11}$$
$$convXPosUB = \min(XPosNode + tol_{Conv}, iLimX) \tag{3.12}$$
$$convYPosLB = \max(YPosNode - tol_{Conv}, 0) \tag{3.13}$$
$$convYPosUB = \min(YPosNode + tol_{Conv}, iLimY) \tag{3.14}$$

Merge variables:

$$mergeLeftXPosVars_m \in [mergeXPosLB, mergeXPosUB], \forall m \in N_{ML} \tag{3.15}$$
$$mergeLeftYPosVars_m \in [mergeYPosLB, mergeYPosUB], \forall m \in N_{ML} \tag{3.16}$$
$$mergeLeftOrientationVars_m \in [-\pi, \pi], \quad \forall m \in N_{ML} \tag{3.17}$$
$$mergeRightXPosVars_m \in [mergeXPosLB, mergeXPosUB] \forall m \in N_{MR} \tag{3.18}$$
$$mergeRightYPosVars_m \in [mergeYPosLB, mergeYPosUB], \forall m \in N_{MR} \tag{3.19}$$
$$mergeRightOrientationVars_m \in [-\pi, \pi], \quad \forall m \in N_{MR} \tag{3.20}$$

Divert variables:

$$divertLeftXPosVars_d \in [divertXPosLB, divertXPosUB], \forall d \in N_{DL} \tag{3.21}$$
$$divertLeftYPosVars_d \in [divertYPosLB, divertYPosUB], \forall d \in N_{DL} \tag{3.22}$$
$$divertLeftOrientationVars_d \in [-\pi, \pi], \quad \forall d \in N_{DL} \tag{3.23}$$
$$divertRightXPosVars_d \in [divertXPosLB, divertXPosUB], \forall d \in N_{DR} \tag{3.24}$$
$$divertRightYPosVars_d \in [divertYPosLB, divertYPosUB], \forall d \in N_{DR} \tag{3.25}$$
$$divertRightOrientationVars_d \in [-\pi, \pi], \quad \forall d \in N_{DR} \tag{3.26}$$

Curve variables:

$$curveLeftXPosVars_c \in [curveXPosLB, curveXPosUB], \forall c \in N_{CL} \quad (3.27)$$
$$curveLeftYPosVars_c \in [curveYPosLB, curveYPosUB], \forall c \in N_{CL} \quad (3.28)$$
$$curveLeftOrientationVars_c \in [-\pi, \pi], \quad \forall c \in N_{CL} \quad (3.29)$$
$$curveRightXPosVars_c \in [curveXPosLB, curveXPosUB], \forall c \in N_{CR} \quad (3.30)$$
$$curveRightYPosVars_c \in [curveYPosLB, curveYPosUB], \forall c \in N_{CR} \quad (3.31)$$
$$curveRightOrientationVars_c \in [-\pi, \pi], \quad \forall c \in N_{CR} \quad (3.32)$$

Source variables:

$$sourceXPosVars_s \in [sourceXPosLB, sourceXPosUB], \forall s \in N_{So} \quad (3.33)$$
$$sourceYPosVars_s \in [sourceYPosLB, sourceYPosUB], \forall s \in N_{So} \quad (3.34)$$
$$sourceOrientationVars_s \in [-\pi, \pi], \quad \forall s \in N_{So} \quad (3.35)$$

Sink variables:

$$sinkXPosVars_s \in [sinkXPosLB, sinkXPosUB], \forall s \in N_{Si} \quad (3.36)$$
$$sinkYPosVars_s \in [sinkYPosLB, sinkYPosUB], \forall s \in N_{Si} \quad (3.37)$$
$$sinkOrientationVars_s \in [-\pi, \pi], \quad \forall s \in N_{Si} \quad (3.38)$$

Straight variables:

$$straightVars_{ij} \in [0, \inf], \quad \forall (i,j) \in A_{adj} \quad (3.39)$$

Objective Function

The main objective of the optimization problem lies in minimizing the cumulative distance between the conveyor system and the pre-optimized material flow graph. To achieve this goal, a unique reference point is designated for each conveyor type, from which the distance to the material flow line is computed. Additionally, an artificial penalty $w_{so,si}$ is introduced. This penalty impacts the distance from the sources and sinks to their corresponding nodes, shifting the optimization's focus towards minimizing these specific distances.

$$\min z_2 = dist_{merges} + dist_{diverts} + dist_{curves} + w_{so,si} * (dist_{sources} + dist_{sinks}) \quad (3.40)$$

$$dist_{merges} = \sum_{m \in N_{MR}} dist^m + \sum_{m \in N_{ML}} dist^m \quad (3.41)$$
$$dist_{diverts} = \sum_{d \in N_{DR}} dist^d + \sum_{d \in N_{DL}} dist^d \quad (3.42)$$
$$dist_{curves} = \sum_{c \in N_{CR}} dist^c + \sum_{c \in N_{CL}} dist^c \quad (3.43)$$
$$dist_{sources} = \sum_{s \in N_{So}} dist^s \quad (3.44)$$
$$dist_{sinks} = \sum_{s \in N_{Si}} dist^s \quad (3.45)$$

Subject to

Orientation:

$$\theta^i_{end} - \theta^j_{start} \leq tol, \qquad \forall (i,j) \in A_{adj} \tag{3.46}$$
$$\theta^i_{end} - \theta^j_{start} \geq -tol, \qquad \forall (i,j) \in A_{adj} \tag{3.47}$$

Position:

$$(xPos^i_{end} + straightVars_{ij} * \cos(\theta^i_{end})) - xPos^j_{start} \leq tol, \qquad \forall (i,j) \in A_{adj} \tag{3.48}$$
$$(xPos^i_{end} + straightVars_{ij} * \cos(\theta^i_{end})) - xPos^j_{start} \geq -tol, \quad \forall (i,j) \in A_{adj} \tag{3.49}$$
$$(yPos^i_{end} + straightVars_{ij} * \sin(\theta^i_{end})) - yPos^j_{start} \leq tol, \qquad \forall (i,j) \in A_{adj} \tag{3.50}$$
$$(yPos^i_{end} + straightVars_{ij} * \sin(\theta^i_{end})) - yPos^j_{start} \geq -tol, \quad \forall (i,j) \in A_{adj} \tag{3.51}$$

Constraints:

The optimization process starts with analyzing the graph of the material flow, which is generated in the prior step. In this phase, all arcs with the same direction are merged together to form a streamlined graph comprised of linear combined arcs, source nodes, sink nodes, as well as curve, merge, and divert nodes. The resulting graph is then analyzed to classify the nodes. For each identified source and sink node, corresponding variables are established. Curve, merge, and divert nodes are bifurcated into left and right nodes, and their relevant variables are introduced. In addition, a $straightVars_{ij}$ variable is generated for every arc (i,j) in A_{adj} that connects these mentioned nodes.
Subsequently, a set of constraints is applied to each pair of nodes.

Orientation constraint

The end orientation of the conveyor at the start (i) of the arc (i,j) and the start orientation of the conveyor at the arc's end (j) must be within a defined tolerance, as seen in Equations (3.46) and (3.47). For curves, diverts, sources and sinks, the start orientation θ^j_{start} represents the decision variable of the conveyor at the end node (j) . For merges, as they have two inputs, an additional start orientation of the merging line is calculated using the start orientation of the straight line θ^j_{start} and its specific geometry. The end orientation θ^i_{end} of the conveyor at the start node (i) is also calculated using its decision variable and specific geometry. In the case of a divert, two end orientations are calculated. This ensures the continuity and consistency of the material flow throughout the network.

Position constraint

In addition to the orientation constraint, the difference in x and y coordinates calculated between the end point of the start conveyor and the start point of the end conveyor must be within the specified tolerance range after subtracting the corresponding $straightVars_{ij}$ variable. The required constraints are seen in Equations (3.48) and (3.49) for the x position and in Equations (3.50) and (3.51) for the y position. Similar to the orientation constraints, $xPos^j_{start}$ and $yPos^j_{start}$ are the decision variables for the position of the conveyor at the end node j and the coordinates, $xPos^i_{end}$ and $yPos^i_{end}$, of the conveyor at the start node i are calculated using its decision variables

and specific geometry. By adjusting the decision variable $straightVars_{ij}$ the model ensures the connection of the two conveyors using a straight conveyor with variable length.

3.2.1 Constraint Adding Helper Methods

In order to effectively leverage trigonometric functions such as sine and cosine, which are necessary for the alignment of conveyors and for evading obstacles with straight conveyors, additional constraints must be incorporated into the model. To efficiently introduce these constraints, three fundamental methods have been developed.

Approximation of the Sines and Cosines

Given that most solvers used to calculate these optimization models lack built-in methods for handling sine and cosine functions, these trigonometric functions have been approximated using piecewise linear approximations. This process necessitates the addition of extra constraints to the model. The custom developed methods tackle this by defining a series of "breakpoints" across the range of either the sine or cosine function.
These breakpoints partition the functions into multiple segments. For each segment, the corresponding sine or cosine value at the breakpoint is calculated. Using these pairs of breakpoints and the corresponding trigonometric values, a piecewise linear function to approximate the original sine or cosine function is defined. A constraint is added to the model for each of these piecewise linear functions, enforcing that the function must match the calculated trigonometric values at the respective breakpoints.
The output of these methods is a new variable representing the piecewise linear approximation of the trigonometric function. This variable is constrained to align with the range of the original function, i.e., between -1 and 1 for the sine function. These approximations allow the inclusion of trigonometric functionality within the optimization model while preserving the model's linearity, thus facilitating a more efficient solution process.
Let's denote:

s	is a segment
b	is a breakpoint
ϕ	is the input angle
S_ϕ	is the number of the segments

Create a new continuous decision variable for the piecewise linear approximation of the trigonometric function.

$$sinCosApprox \qquad \in [-1, 1] \tag{3.52}$$

$$sinCosApprox = s_i + \frac{s_{i+1} - s_i}{b_{i+1} - b_i} \cdot (\phi - b_i) \tag{3.53}$$

$$if \quad b_i \leq \phi < b_{i+1}, \qquad \forall i \in \{0, \ldots, S_\phi - 1\} \tag{3.54}$$

Normalization of the calculated Angles

To enhance the model's performance by refining the decision variables' boundaries, the orientation variables for the conveyors are constrained to lie within the range of $-\pi$ and π. Bearing in mind the fact that angles can be described by multiples of π and due to the calculations of the orientation of the conveyors, the resulting angles can be outside this range, and must therefore be normalized after each calculation.

Let's denote:

ϑ	is the input angle
c_ϑ	is the number of full 2π cycles in the input angle

Create a new continuous decision variable for the normalized angle.

$$\vartheta_{norm} \quad \in [-\pi, \pi] \tag{3.55}$$

Add constraints to ensure the input angle is normalized in the range of $-\pi$ and π.

$$\vartheta_{norm} = \vartheta - c_\vartheta * 2\pi \tag{3.56}$$

$$\vartheta_{norm} \leq \quad \pi \tag{3.57}$$

$$\vartheta_{norm} \geq -\pi \tag{3.58}$$

Obstacle Constraints for Straight Conveyors

To prevent potential intersections between straight conveyors and rectangularly defined obstacles, conveyors are partitioned into multiple segments. Constraints are then applied to the endpoints of these resulting segments to ensure they maintain a safe distance from the obstacles. To formally express these constraints, the introduction of the following variables is necessary:

S_{ob}	is the number of segments the straight conveyors are divided in
$xEnd_s$	is the x position of the end of the segment s
$yEnd_s$	is the y position of the end of the segment s
$isLeftOfObstacle_s$	binary, indicating if the segment's end is to the left of an obstacle
$isRightOfObstacle_s$	binary, indicating if the segment's end is to the right of an obstacle
$isBelowObstacle_s$	binary, indicating if the segment's end is below an obstacle
$isAboveObstacle_s$	binary, indicating if the segment's end is above an obstacle

The four binary variables indicate the relative position of a conveyor segment's end with respect to an obstacle. If the end of a segment is completely to the left, right, below, or above an obstacle, the corresponding binary variable takes the value 1. Otherwise, the variable remains 0.

Figure 3.2 illustrates this scenario, where the endpoint of segment 2 falls within the obstacle area, thus violating the obstacle constraint.

The variables introduced above are integral to our mathematical formulation of the problem, especially when we employ the "Big-M" method, a standard technique used in mixed-integer programming to handle certain types of constraints. In the "Big-M" method, we introduce an

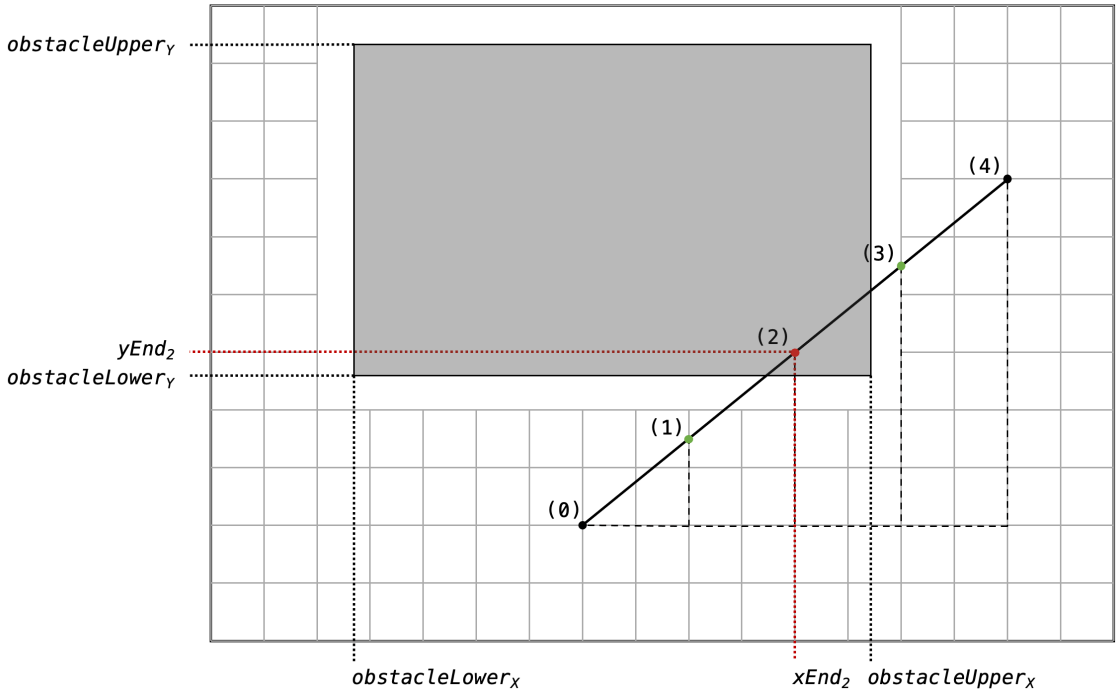


Figure 3.2: Division of the Straight Conveyor into four Segments to formulate the Constraint

auxiliary variable M, a "large" number that is sufficiently large to have a significant impact on the system or constraint but not so large as to lead to numerical instability or solution infeasibility. Essentially, M is as large as it needs to be for the specific context of the problem, hence the term "Big-M". In addition, a small tolerance, denoted as ϵ, is used. This small positive value is employed to ensure numerical stability and precision when dealing with real numbers, allowing us to handle the approximation errors that may arise in computational operations.
The application of M and ϵ in the "Big-"M method facilitates the construction of essential constraints for the positioning of a segment's endpoint in relation to an obstacle. The obstacle is characterized by its lower and upper coordinate bounds. Thus, the constraints below are formulated to ensure that the endpoint of the segment remains outside the bounds of the obstacle.
For any given segment and obstacle:

$$xEnd_s \leq obstacleLower_x - \epsilon + M \cdot (1 - isLeftOfObstacle_s) \tag{3.59}$$

$$xEnd_s \geq obstacleUpper_x + \epsilon - M \cdot (1 - isRightOfObstacle_s) \tag{3.60}$$

$$yEnd_s \leq obstacleLower_y - \epsilon + M \cdot (1 - isBelowObstacle_s) \tag{3.61}$$

$$yEnd_s \geq obstacleUpper_y + \epsilon - M \cdot (1 - isAboveObstacle_s) \tag{3.62}$$

Each constraint ensures that the end of the segment is either left, right, below, or above an obstacle by maintaining a minimum distance of ϵ. In the cases where the segment's end is indeed in the respective position (left, right, below, or above) relative to the obstacle, the term involving the "Big-M" method will become zero, thereby neutralizing its effect on the constraint. This way, the constraint is only effective when the segment's end is not in the intended position relative to

the obstacle.
Ultimately, to guarantee that the end position of each segment lies outside of all obstacles, we formulate the following constraint for all segments and obstacles:

$$isLeftOfObstacle + isRightOfObstacle + isBelowObstacle + isAboveObstacle \geq 1 \quad \forall s \in \{0, \ldots, S_{ob} - 1\} \tag{3.63}$$

In the specific situation depicted in Figure 3.2, the decision variable for segment (1) should have $isBelowObstacle$ equal to 1. For segments (3) and (4), the $isRightOfObstacle$ variable must be set to 1. However, since segment (2) falls within the obstacle area, none of the four decision variables can assume a value of 1, thereby leading to a violation of the constraints.
It is noteworthy that the total number of conveyor segments determines the additional constraints and decision variables count. While an excess of segments could potentially impair the model's performance, a limited count may not sufficiently capture all intersections of conveyors and obstacles, presenting a careful balance to strike.

Chapter 4

Evaluation of the Algorithm

This chapter is dedicated to the scrutiny of the implemented optimization models and their respective performance metrics. This chapter can be considered the practical culmination of the concepts and models developed and discussed in the preceding sections. First, an examination of the algorithm's implementation is conducted. The strategies and processes followed to transform the theoretical models into usable algorithms are elucidated, including a detailed walkthrough using UML diagrams. The chapter then transitions to a theoretical evaluation of the models. Here, the influences and conditions that may impact the model's performance are put to the test. This section culminates in a presentation of various performance metrics and solution quality parameters. The final part of this chapter encompasses the practical evaluation of the models. The theoretical insights gained are applied to two extensive, realistic use cases, thus probing the suitability and practicability of the models under real-world-like conditions. This rigorous, two-pronged evaluation approach ultimately serves to demonstrate the validity and applicability of the presented algorithms.

4.1 Implementation of the Algorithm

To address the problems elucidated in the previous chapter, the selection process involves more than just determining the appropriate programming language. It also requires the choice of a solver capable of managing such types of problems. Irrespective of the solver, the overall structure of these problems remains constant. Initially, the decision variables must be declared, followed by the constraints and the objective function. However, the declaration syntax differs across solvers, hence emphasizing the importance of selecting a suitable solver before implementing the problem. One critical factor to consider is the ability of the solver to handle possible quadratic constraints, as these constraint types pose unique challenges and are not universally supported by all solvers. For instance, the constraints for the difference in x and y result in quadratic constraints as two decision variables, the length of the straight conveyor and the output of the trigonometric method, are multiplied. Given these considerations and drawing from research conducted by B. Meindl and M. Templ, the Gurobi solver was selected. Their study suggests that the Gurobi solver's performance often surpasses that of other solvers analyzed, particularly for problems of higher

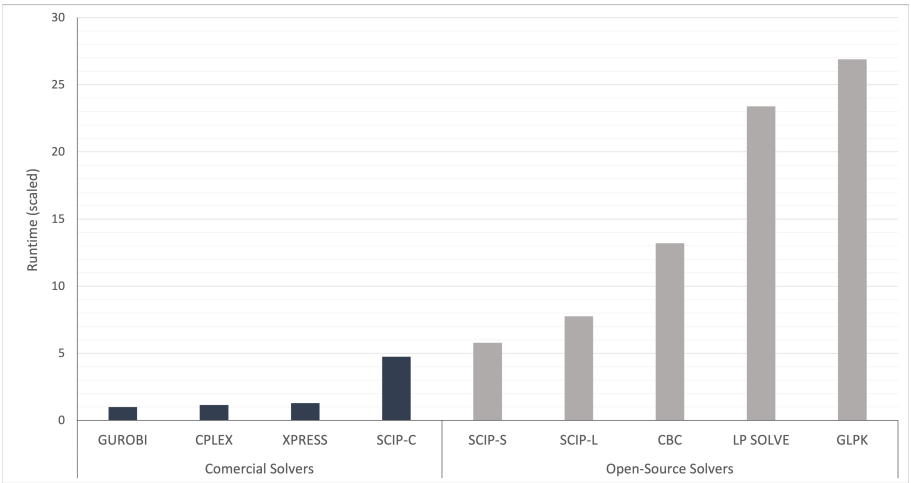


Figure 4.1: Scaled Runtime of Comercial and Open-Source Solvers (cf. [Mei-2013])

complexity. They examined nine solvers, consisting four comercial and 5 open-source solvers, comparing their runtime and success rate across a benchmark dataset consisting of 87 problem instances. The outcomes of their investigation are illustrated in Figure 4.1 [Mei-2013].

The architectural design of the program is visually represented in Figure 4.2 using a UML diagram. The 'Program' class is designed as a user interface, granting the user access to select a prior created example instance. Once selected, the user can decide upon the type of optimization to perform: this may involve a material flow optimization, the optimization of conveyor placement, or a combination of the two.
Additional utilities are provided in the form of helper methods. These are designed to facilitate user interaction and improve the user experience. For instance, these tools can automatically generate graphical representations for the grid, commodities, and computed material flows. Users are also empowered to specify crucial Gurobi parameters such as the desired gap to the optimum and performance-specific variables like the Heuristics and MIPFocus parameter.

When optimizing the material flow across the unit grid, users are offered a choice from three distinct objective functions. The first option targets the minimization of total costs (both variable and fixed). The second objective function aims to minimize curve usage. Finally, a third function allows users to employ a combination of the first two options. This is achieved by setting weights to adjust focus towards a specific objective (either costs or curve usage). Although a fourth function is technically implemented to minimize the total length of material flow, it computes the same output as the cost function when a unit grid with equal arc costs is used.
To initiate the optimization process, a use case or example must first be created. This is accomplished by generating a new instance of the IExample class. This class requires various parameters including the grid's dimensions and resolution (as defined by the grid size), potential obstacle locations, the commodity system complete with source and sink locations and corresponding flows, as well as the chosen conveyor catalogues. Using this information, the IExample class generates the desired unit grid by creating a graph-like structure comprised of Nodes and Arcs. Each node is connected to its neighboring nodes via two directed arcs, the size of which corresponds to the chosen grid size. Nodes and their corresponding arcs that lie behind defined obstacles are deleted to simplify the model's complexity. Furthermore, to continue simplifying the complexity,

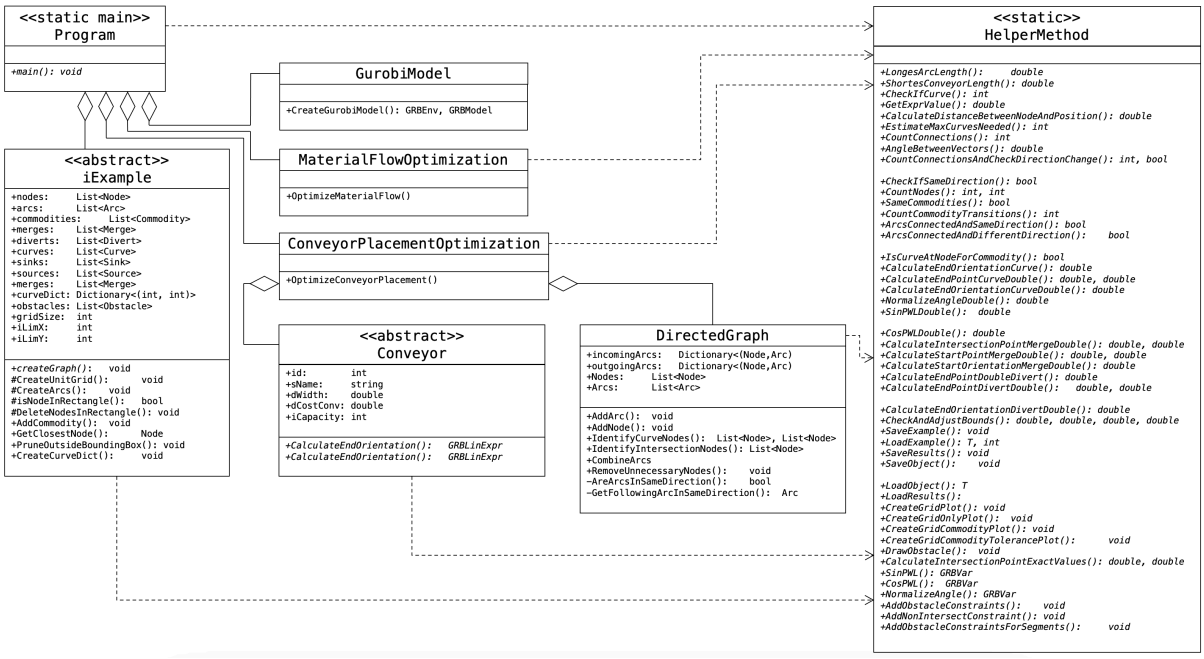


Figure 4.2: Basic Structure of the Program

the smallest bounding box surrounding the sources and sinks is calculated. All Nodes and Arcs located outside of this rectangle are eliminated, as the optimal material flow will be contained within this bounding box. The corresponding UML diagram is visualized in Figure 4.3.

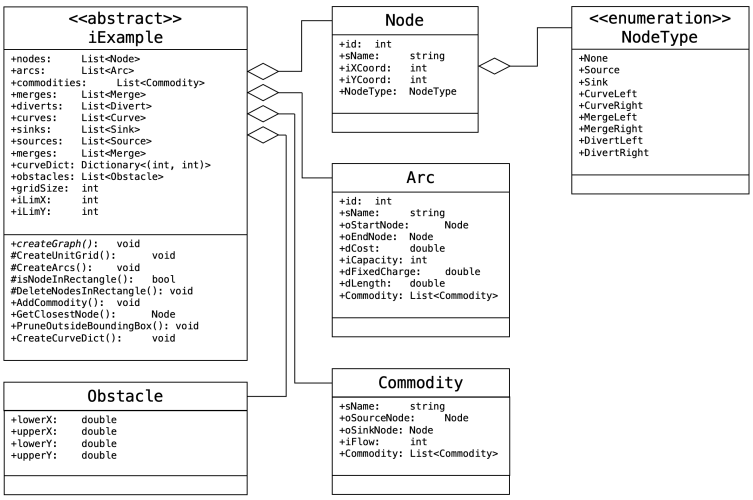


Figure 4.3: Basic Structure Use Case Creation

Prior to computing the optimal placement of conveyors, it is essential to generate the relevant conveyors. These fall into four categories: straights, curves, merges, and diverts. The sources and sinks essentially serve as minimal straight conveyors.Their dimensions could be adjusted as needed. For curves, merges, and diverts, two variants exist, determined by their directional orientation. This encompasses right and left curves, merges that merge from the right or the left, and diverts that redirect towards the right or the left. This differentiation allows for a comprehensive and

flexible conveyor system, accommodating various logistical requirements. Merges and diverts essentially comprise two straight conveyors positioned at an angle to each other. Each type of conveyor is created as a specific subclass of the superclass 'Conveyor', since each one requires different methods for calculating the necessary positions and orientations. At its core, the optimal placement computation requires the model to be capable of determining the start and end positions and orientations. For merges and diverts, two start positions or two end positions must be calculable due to their distinctive configuration. These relationships are illustrated in Figure 4.4.

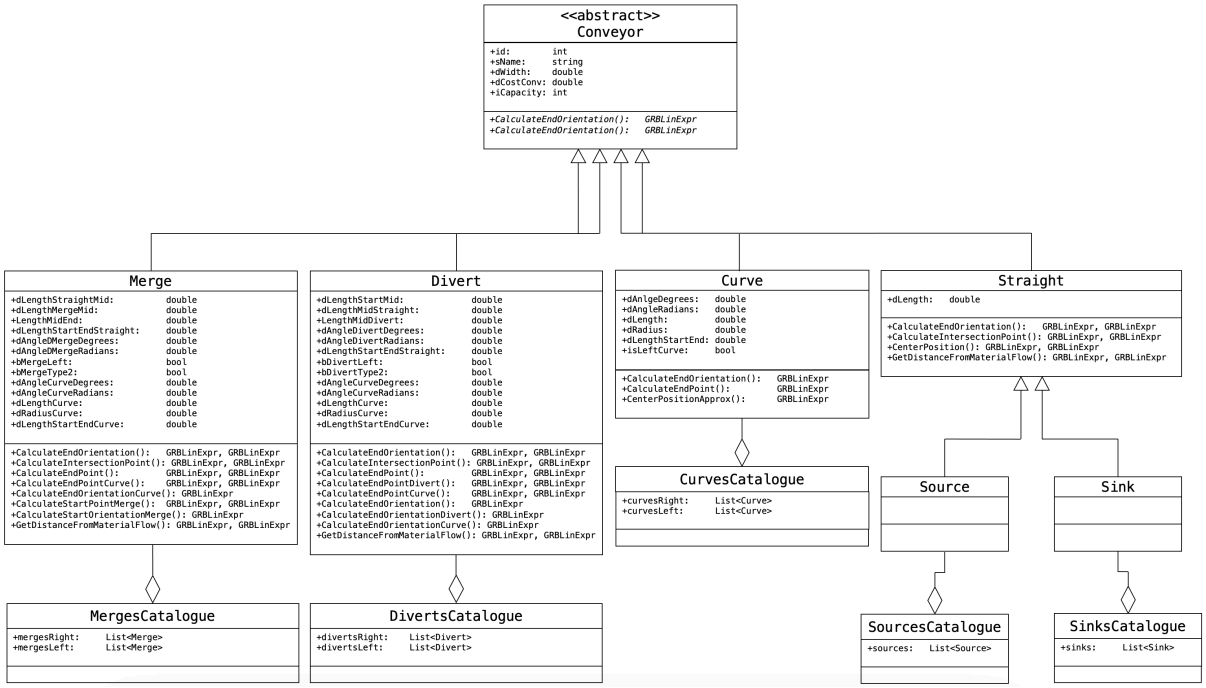


Figure 4.4: Basic Structure of the different Conveyor Types

To facilitate the computation of optimal conveyor placement, the output from the material flow optimization must be adapted. This output consists of a list of Nodes and a list of Arcs. For the conveyor placement, the unit grid is no longer necessary as the model solely needs the terminal nodes from the optimal material flow and the precise locations of the obstacles. The model constructs a new structure by creating an instance of the DirectedGraph class. This structure enables the model to perform a more in-depth investigation of the nodes. For the model to function, it must understand the types of each node. There are four basic types of Nodes: Sink-Nodes, Source-Nodes, Curve-Nodes, and Intersection-Nodes. All nodes that do not belong to any of these categories are eliminated, and the arcs connected to them are combined into a larger one to enhance performance and decrease complexity. Further classifications are made among the Curve- and Intersection-Nodes to discard inappropriate conveyors. Curve-Nodes are divided into RightCurve-Nodes and LeftCurve-Nodes, while Intersection-Nodes are first split into Merge- and Divert-Nodes, then further distinguished by their orientation. This distinction allows the model to preemptively eliminate nonsensical solutions, such as attempting to place a RightCurve onto a LeftCurve-Node. All remaining Nodes are assigned a flag from the enum class NodeType for use in the subsequent optimization model. The corresponding UML diagram is visualized in Figure 4.5.

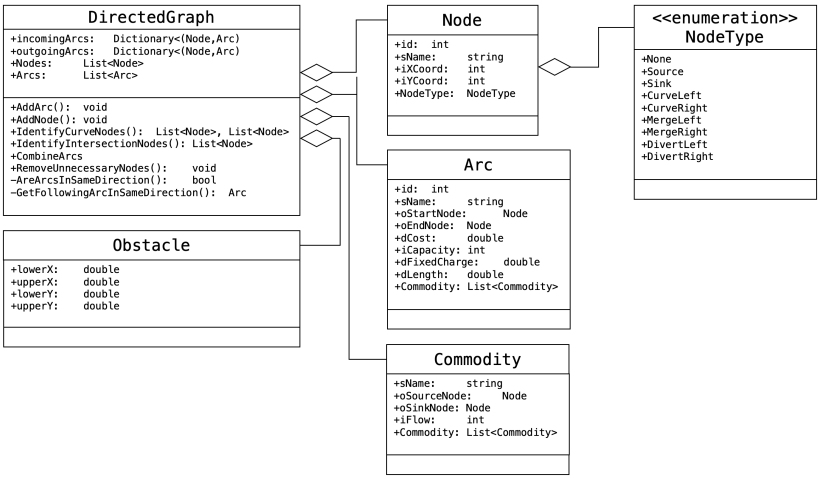


Figure 4.5: Basic Structure of the directed Graph Class

Following the creation of the DirectedGraph instance and the classification of all nodes, the model iterates through the remaining combined arcs, verifying the type of both the Start-Node and End-Node. Depending on these types, it selects the required conveyor, for instance, a Right-Curve at the start and a Left-Curve at the end. To connect the end position of the Start-Conveyor with the start position of the End-Conveyor, the model formulates the constraints mentioned in Section 3.2. To achieve this, the model differentiates among four possible scenarios. These are: both the Start- and End-Node belong to a single type; the Start-Node is of a single type and the End-Node encompasses two types; the Start-Node has multiple types and the End-Node is of a single type; or lastly, both Nodes are characterized by multiple Node-Types. These situations can arise if a source or a sink is positioned at a curve or an intersection. In such cases, the model must place both conveyor types as close as possible to the corresponding Node while simultaneously connecting the Start- and End-Node with the variable straight conveyor.
In order to formulate the objective function introduced in Section 3.2, each conveyor type has its own method for calculating the distance to the respective Node. For sinks and sources, the distance is calculated between the start position and the Node. For curves, the center point on the arc's centerline is approximated by determining the midpoint of the straight line connecting the start and end points of the curve. For merges and diverts, the intersection point of the two centerlines is calculated, and the distance to the respective nodes is determined. All of these distances are subsequently incorporated into the objective function.

4.2 Theoretical Evaluation of the Algorithm

To evaluate the efficacy of the two optimization models, a series of test scenarios have been defined. The criteria for a qualitative assessment include performance, as well as the influence of various design aspects innate to both the model and the examples produced.

Several key factors are examined to assess the qualitative evaluation of the models. These include the chosen grid size, representative of the distances between two nodes, the existence of grid obstacles, the predetermined objective function (cost or the quantity of curves utilized), and the

efficiency of a warm start when aiming to minimize the curve count. Additionally, two Gurobi-specific parameters - 'Heuristics' and 'MIPFocus' - are under investigation. For each test scenario, the evolution of the gap and the objective value in relation to the runtime as well as the quality of the solution are evaluated.

Used Hardware Specification:

CPU:	11th Gen Intel Core i7-11850H, 2.50GHz 8 Core
Memory:	32.0GB
GPU:	NVIDIA RTX A2000

4.2.1 Influence of the Grid Size

The optimization of the material flow is achieved by overlaying a uniform grid on the intended layout. The distance in both the x and y directions between all nodes is determined by the grid size parameter. The number of nodes is inversely proportional to the square of the grid size demonstrated in the following formula:

$$nodes_x = (limit_x + 1)/GridSize \tag{4.1}$$

$$nodes_y = (limit_y + 1)/GridSize \tag{4.2}$$

$$nodes = nodes_x * nodes_y \tag{4.3}$$

As it decreases (implying a higher grid resolution), the number of nodes in each dimension, calculated as the limit in x divided by the Grid Size times the limit in y divided by the grid size, increases approximately linearly, assuming that the limits in x and y remain constant. Hence, the number of arcs, which is proportional to the square of the number of nodes, escalates in an approximately quadratic manner. The exact number of arcs can be computed using the following formula:

$$arcs = (nodes_x - 1) * nodes_y + (nodes_y - 1) * nodes_x \tag{4.4}$$

It is obvious that increasing the number of nodes, hence the number of arcs, the number of possible solutions increase as well and therefore the complexity of the model increases. In the following this correlation is investigated. In Table 4.1 the test scenarios for investigating the influence of the grid size on the performance of the model is illustrated. For these test runs the cost minimizing objective function is used ($\beta = \gamma = 0$), as only the influence of the grid size is investigated. The number of nodes and arcs seen in the table are the ones resulting after reducing the grid using the smallest bounding box as described in the section before.

The selected commodities and their corresponding flows are detailed in Table 4.2. Given the unit length nature of the grid, the distances between the nodes precisely equal the grid size.

Table 4.1: Test Scenarios for the Influence of the Grid Size

id	Limit X	Limit Y	Nodes	Arcs	Comms.	Grid Size	α
1_1	60	40	750	2890	5	1	1
1_2	60	40	180	774	5	2	1
1_3	60	40	99	356	5	3	1
1_4	60	40	56	194	5	4	1
1_5	60	40	42	142	5	5	1

Additionally, in these examples, each arc has an identical capacity of 100 units. Pertaining to the fixed charge associated with each arc, this remains constant across all arcs, standing at 100.

Table 4.2: Commodities for the Test Scenarios for the Influence of the Grid Size

Name	X Source	Y Source	X Sink	Y Sink	Flow
C1	8	18	82	46	10
C2	45	30	86	20	20
C3	6	34	88	50	20
C4	32	14	80	10	20
C5	66	15	86	20	20

Figure 4.6 provides a visual representation of the grid and the associated commodities for the case of a grid size of 3. An orange bounding box visualizes the operational area the model considers during the computation. Any material flow occurring outside of this defined area is disregarded, based on the premise that it would not contribute to the optimal solution. This approach ensures computational efficiency by focusing resources only on the pertinent area. With an increase in the grid size, it's possible that the precise coordinates of a commodity's sink or source node may no longer correspond to an actual node. Under such circumstances, the nearest node is computed and the respective sink or source is consequently relocated.

In the following, we will delve into more details about how these adjustments in grid size impact the performance of the optimization process. Table 4.3 shows the results of the five test runs. In the scenario where the grid size is set to 1, the minimal gap achievable by the model, after a

Table 4.3: Results of the Test Scenarios for the Influence of the Grid Size

id	GridSize	Lowest Gap.	Runtime total	Lowest Obj. Value	Runtime Best Solution
1_1	1	35.34%	2688.51	13h 41min	35min
1_2	2	10.43%	1377.42	13h 00min	29s
1_3	3	0.00%	905.85	3min 50s	12s
1_4	4	0.00%	1229.45	7s	0.07s
1_5	5	0.00%	1160.44	3s	0.35s

significant runtime of 13 hours and 41 minutes, was 35.34%. Both the objective value and the gap seemed to reach a state of near-stagnation after only an hour into the runtime. The objective

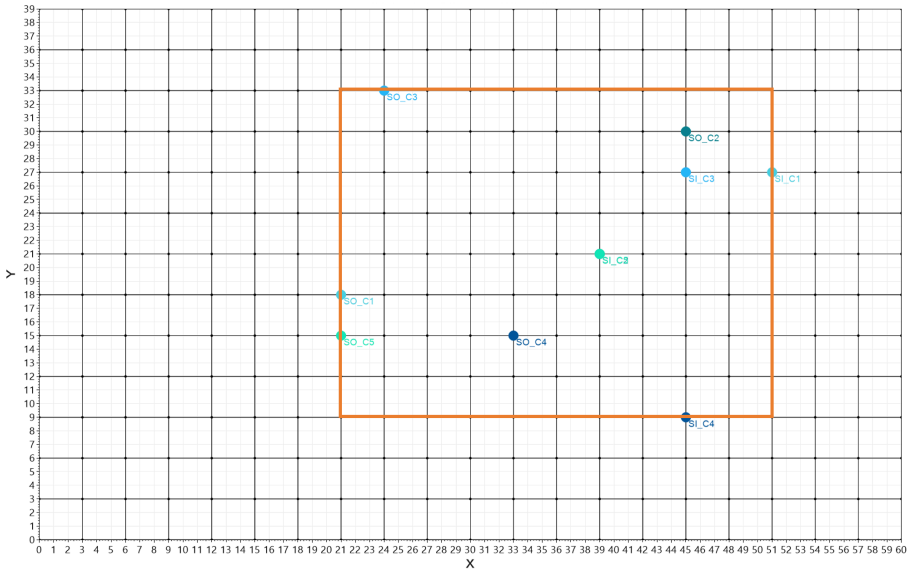


Figure 4.6: Unit grid with Grid Size 3 including the Commodity System used

value arrived at a figure of 2688.51 after 35 minutes. Despite the extended optimization period of nearly 14 hours, this value remained constant, similar for the gap.
A similar pattern is seen when setting the grid size to 2. The objective value drops in the first 30 seconds of the optimization to a level of 1377.42. This value stays constant for the rest of the optimization process, which was stopped after 13 hours. The same happens for the gap which drops in the first seconds down to 18% and has another drop to 14% after 6 minutes. During the remaining 13 hours of optimization,this value can only be reduced to 10.43%.
With the grid size increased to 3, thereby reducing the number of nodes to 99 and the corresponding number of arcs to 356, the model was capable of reducing the gap to the desired level of 0% in a mere 3 minutes and 50 seconds. The definitive objective value of 905.85 was attained after just 12 seconds into the optimization process. The remaining runtime was devoted to affirming the optimality of this solution.
Upon further simplification of the model by increasing the grid size to 4, a significant enhancement in performance can be observed. The model established its optimality by shrinking the gap to 0% within a brief span of 7 seconds. The final objective value of 1229.45 was discerned just 0.07 seconds into the optimization process.
In the final test run, the grid size was increased to 5, yielding a total of 42 nodes and the corresponding 142 arcs. This enhancement led to an even more efficient performance. A gap of 0% was swiftly achieved in just 3 seconds, while the final objective value was identified in a mere 0.35 seconds. Figure 4.7 illustrates the results for the grid size of 3.

As predicted, the runtime increased in proportion to the grid's complexity. For a grid size set to 1, the narrowest gap the model could achieve, after an extensive runtime of 13 hours and 41 minutes, was 35.34%. With a slightly reduced complexity by setting grid size to 2, the gap fell to 10.43% after a similar 13-hour duration. Strikingly, when the grid size was raised to 3, 4, or 5, the model was able to achieve the desired gap of 0%. This reduction was accomplished in less than 4 minutes with a grid size of 3, in 7 seconds with a grid size of 4, and astoundingly, in just 3 seconds with a grid size of 5. The variance in the optimal objective value arises from the unique

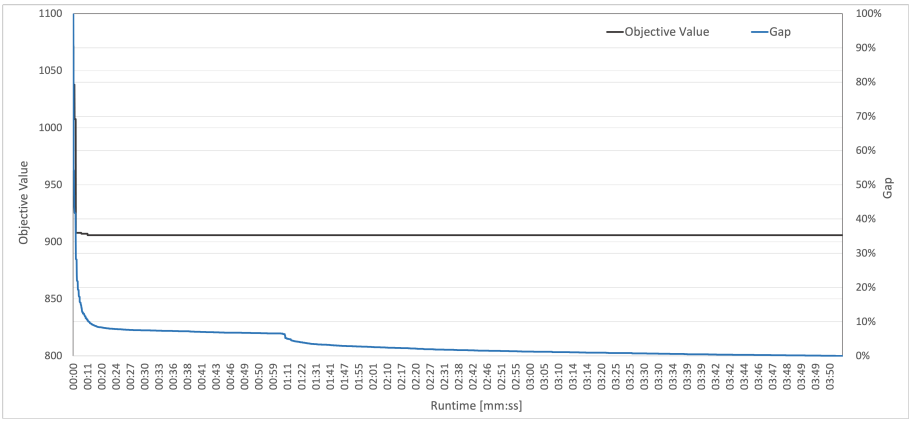


Figure 4.7: Objective Value and Gap over Runtime with a Grid Size of 3

optimal solutions dictated by distinct grid configurations. The required movement of commodities following an increase in grid size further contributes to these differences.
The optimization process also displayed a compelling pattern. The objective values sharply declined at the commencement of the optimization, indicating a rapid approach towards a near-optimal or optimal solution. However, this swift descent was followed by a plateau, indicating stagnation. This suggests that, while the optimal or near-optimal solution could be determined rather quickly, the validation of its optimality required considerably more time. Figure 4.8 illus-

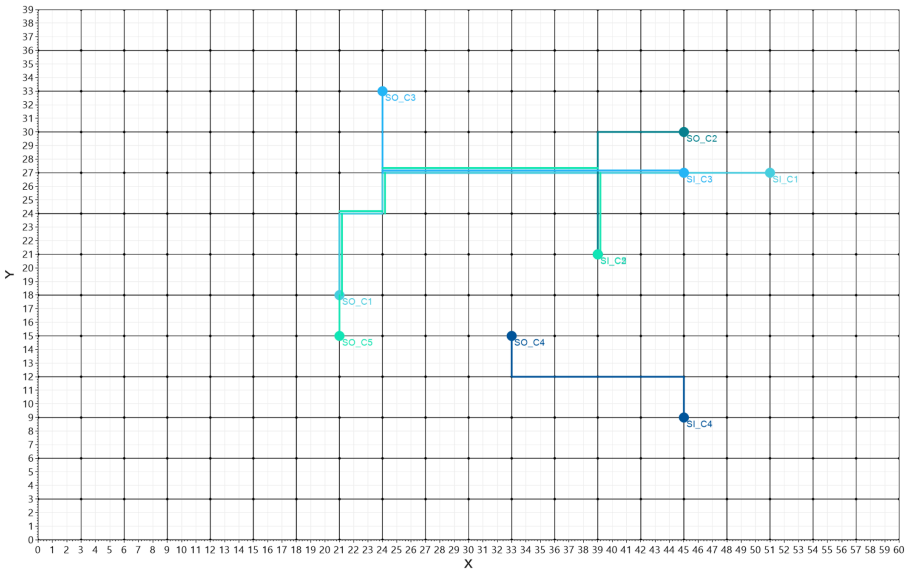


Figure 4.8: Optimal Material Flow computed for a Grid Size of 3

trates the computed material flow for the case of a grid size of 3.

The effect of the grid size on the performance of the conveyor placement optimization model was not examined, as the nodes under consideration are solely those that remain after combining all the single arcs with the same direction.

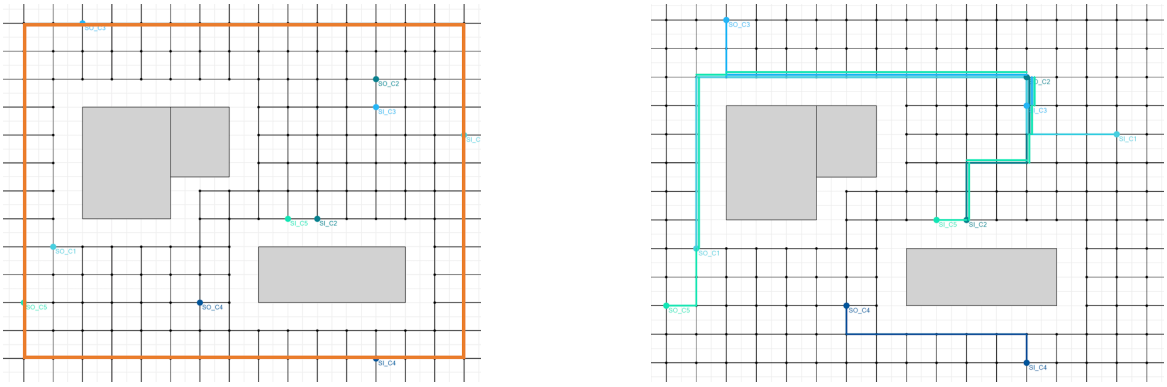
4.2.2 Influence of Obstacles

To be able to represent real life scenarios more accurate, obstacles must be introduced. In the following, their influence on the performance, for both models, the material flow and the conveyor placement model are examined, since it is expected that both models will be affected.
For the material flow model, all the nodes, and arcs that are behind given obstacles are removed and by that the number of solutions decreased. For the conveyor placement model on the other hand, every obstacle leads to an increase of the constraints to ensure that the conveyor does not intersect with any obstacle, as described in Section 3.2.1.
The commodity system employed for these tests replicates the one used in the previous test runs as depicted in Table 4.2. In these experiments, the area is populated with a percentage of obstacles situated in the region among the commodities. The specific test runs are illustrated in Table 4.4. Similar to earlier tests, to preemptively eliminate superfluous nodes and arcs, we calculate the bounding box around the commodity system. The number of nodes and arcs indicated in the table are confined to this bounding box. To ensure the obstacles exert an effect, they are positioned within this bounding box and the occupied area percentage is calculated based on this bounding box. To improve performance but still not loosing accuracy the chosen grid size is 2. As before the total cost minimization function is used ($\beta = \gamma = 0$).

Table 4.4: Test Scenarios for the Influence of Obstacles

id	Limit X	Limit Y	Area Obstacles	Number Nodes	Number Arcs	α
$2_{1,1}$	60	40	0%	208	774	1
$2_{1,2}$	60	40	15%	161	538	1
$2_{1,3}$	60	40	30%	133	432	1
$2_{1,4}$	60	40	50%	78	230	1

The scenario with 0% obstacle occupancy corresponds to the second test run of the previous section (id = 1_2). In the context of a scenario with 15 percent obstacle occupancy, the layout is demonstrated in Figure 4.9a. The orange rectangle, delineating the bounding box, encapsulates an area of 720 units. The three obstacles depicted within the bounding box, possessing a cumulative area of 108 units (comprising areas of 48, 40, and 20 units respectively), result in significant modifications to the grid. These modifications, instigated by the presence of obstacles, lead to a reduction of nodes by 23% and a decrease in arcs by 30%. The influence of the obstacles which decreased the complexity is clearly visible as this time the model was able to compute the optimal solution in 5 hours and 52 minutes. Astonishingly, the later to be proved to be the optimal value was found in the first second to the optimization, but proving its optimality took additional 5 hours and 52 minutes as seen in Table 4.5. This progress compared to the test run without obstacles underscores the effect that the reduced grid complexity has on the performance, with a decrease in the number of nodes and arcs leading to an enhanced optimization process. The fact that the optimal solution was found in the first second once again underscores the trend observed in previous scenarios, where the model identifies a near-optimal solution swiftly, but spends a significantly longer time confirming its optimality. It becomes evident that the challenge for the model does not lie in discovering the optimal solution but in establishing its optimality



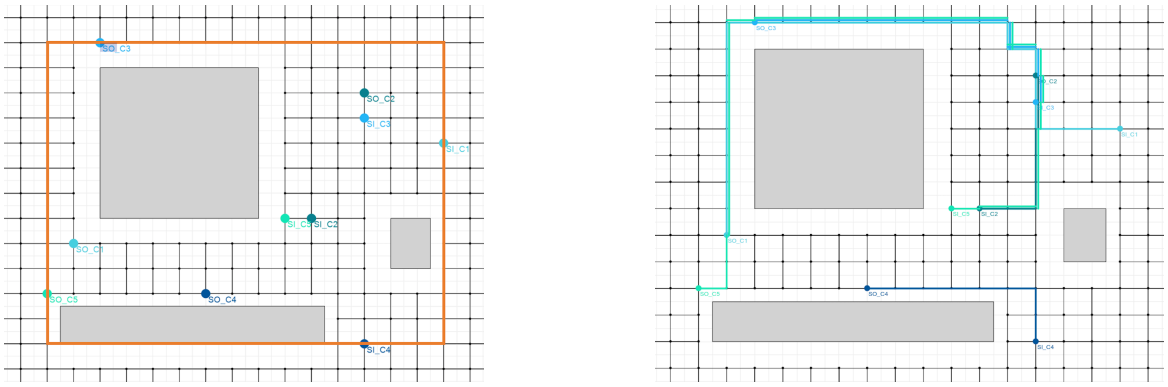
(a) (b)

Figure 4.9: 15% Obstacle Occupancy (a) Grid Layout, (b) Computed Material Flow

Table 4.5: Results of the Test Runs for the Influence of Obstacles on the Material Flow Model

id	Area Obstacles	Lowest Gap.	Runtime total	Lowest Obj. Value	Runtime Best Solution
$2_{1,1}$	0%	10.43%	13h 00min	1377.42	29s
$2_{1,2}$	15%	0.00%	5h 52min	1378.74	1s
$2_{1,3}$	30%	0.00%	13h 2min	2938.62	24s
$2_{1,4}$	50%	0.00%	9s	3466.58	3s

with certainty. The resulting material flow is illustrated in Figure 4.9b.
In Figure 4.10a the layout for the test run with a 30 percent obstacle occupancy is illustrated. As seen in Table 4.4, with that occupancy the number of nodes was reduced by roughly 35% and the number of arcs by 45% leading to a much simpler model. In this scenario, the model was once



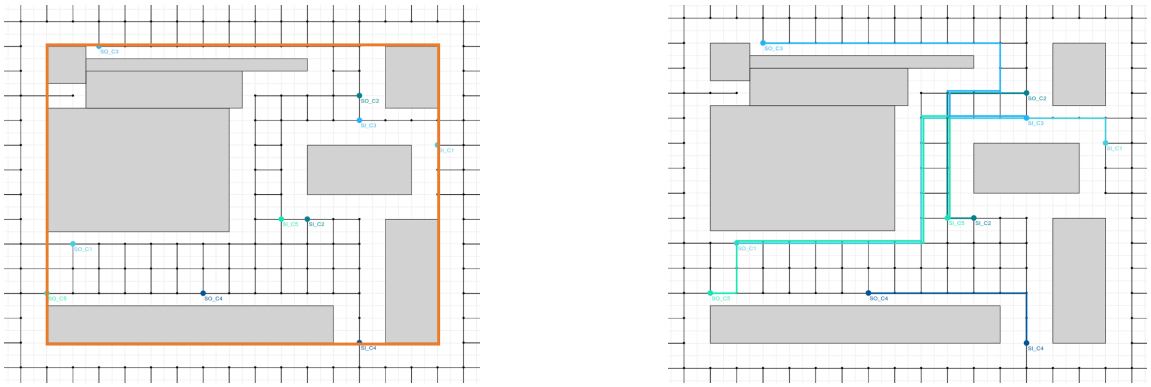
(a) (b)

Figure 4.10: 30% Obstacle Occupancy (a) Grid Layout, (b) Computed Material Flow

again able to reduce the gap to less than 15% within the first 34 seconds of runtime, but then went on to attain the optimal gap of 0% in 13 hours, more than twice as long as with an obstacle occupancy of 15%. This is surprisingly as the complexity of the model seems to be less when looking soley on the number of remaining nodes and arcs. This shows that not just the existing

of obstacles and therefore less nodes and arcs influence the performance but also the resulting layout. The computed material flow is seen in Figure 4.10b.

In the case of a 50% obstacle occupancy the resulting grid layout is visualized in Figure 4.11a. As seen in Table 4.4, with that occupancy the number of nodes was reduced by roughly 63% and the number of arcs by 70% leading to a much simpler model. The performance difference when



(a) **(b)**

Figure 4.11: 50% Obstacle Occupancy (a) Grid Layout, (b) Computed Material Flow

computing the test run for a 50% obstacle occupancy, compared to other two is immense as seen in Table 4.5. After 2 seconds runtime the gap was already below 10% and after just 9 seconds the desired gap of 0% was reached. The objective value 3466.58 - proved to be optimal - was found after 3 seconds. The corresponding computed material flow is visualized in Figure 4.11b.

To probe the impact of obstacles on the conveyor placement model, the computed material flows of the test runs were fed to the conveyor placement model. The results are shown in Table 4.6. For the case of the 0% obstacle occupancy, where no optimal material flow could be computed in a reasonable runtime the near optimal solution with a remaining gap of 10.43% shown in Figure 4.12 was used. The tolerances for sources sinks and curves was set to 1 and the tolerance for merges

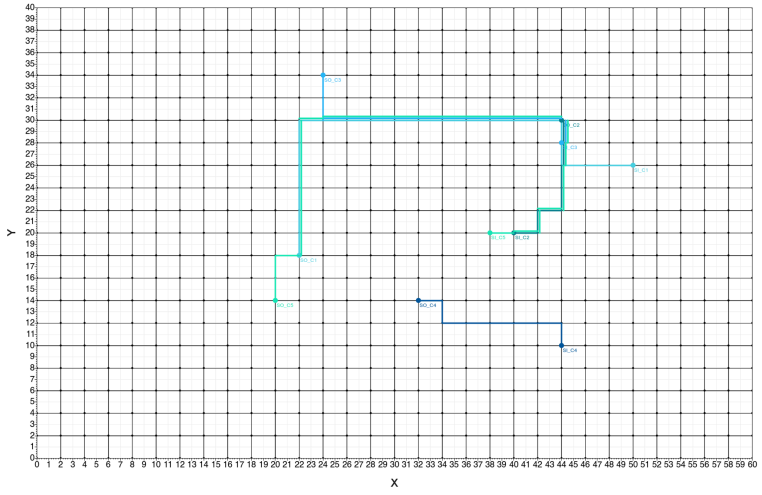
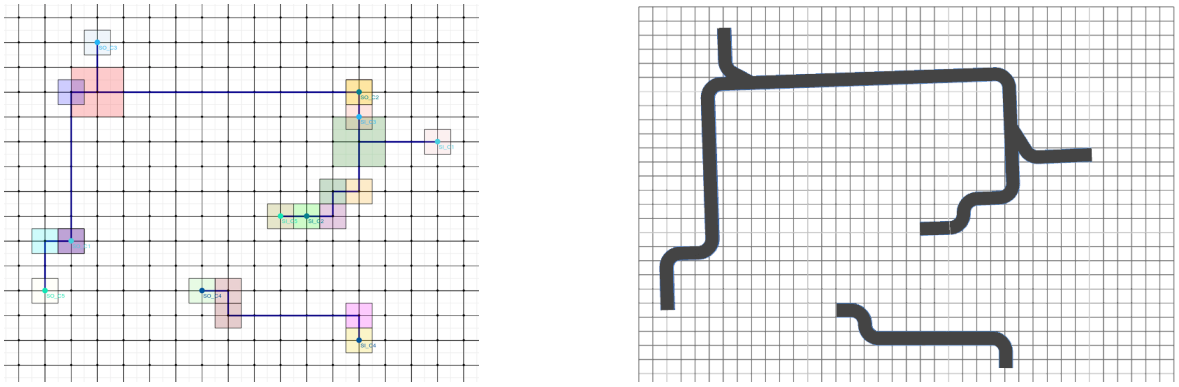


Figure 4.12: Computed Material Flow without Obstacles - 10.43% remaining Gap

and diverts was set to 2 as their geometry is more complex and their dimensions are comparable

greater. When feeding the material flow shown in Figure 4.12 to the conveyor placement model the optimal solution for the conveyor system was found in just 15 seconds. The model needed 14 seconds to find a feasible solution and then less than 1 second to improve it and by that finding the optimal solution. Due to no obstacles, no additional constraints nor variables had to be created as it was not necessary to ensure collision between conveyors and obstacles.
The defined tolerances and the computed conveyor system are visualized in Figure 4.13. The conveyor system was implemented in the simulation program Tecnomatix Plant Simulation.



(a) **(b)**

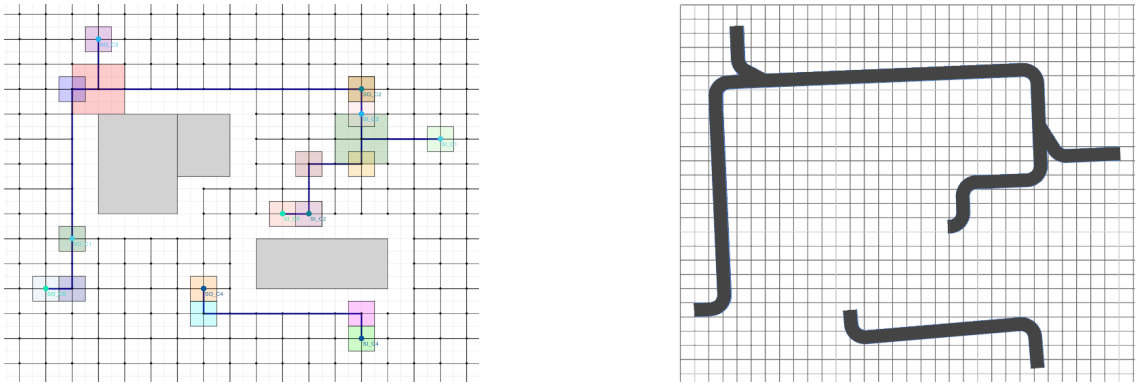
Figure 4.13: 0% Obstacle Occupancy (a) Defined Tolerances (b) Computed Conveyor System

Table 4.6: Results of the Test Runs for the Influence of Obstacles on the Conveyor Placement Model

id	Area Obstacles	Lowest Gap.	Runtime total	Lowest Obj. Value	Runtime Best Solution
$2_{2,1}$	0%	0.00%	15s	653.65	14s
$2_{2,2}$	15%	0.00%	20s	256.48	20s
$2_{2,3}$	30%	0.00%	25s	427.02	24s
$2_{2,4}$	50%	0.00%	16s	351.88	16s

In the case of a 15% obstacle occupancy, with the same tolerances, the model needed 20 seconds to compute the optimal placement of the conveyors, 33% longer. The graph with the tolerances and the conveyor placement can be observed in Figure 4.14. The model initially grapples with finding a feasible solution. However, once such a solution is discovered after a brief 20 seconds, the gap reduces to 0% almost instantaneously. This underscores the model's efficiency in identifying and verifying the optimal value in a relatively short span of time once a feasible solution is located. When comparing just the material flows used, without and with 15% obstacle occupancy, they appear to be the same, leaving the only difference the obstacles around it. This proves the complexity increase due to the additional needed constraints and variables. For the test runs with obstacles the number of segments, as defined in Section 3.2.1, was set to 5.

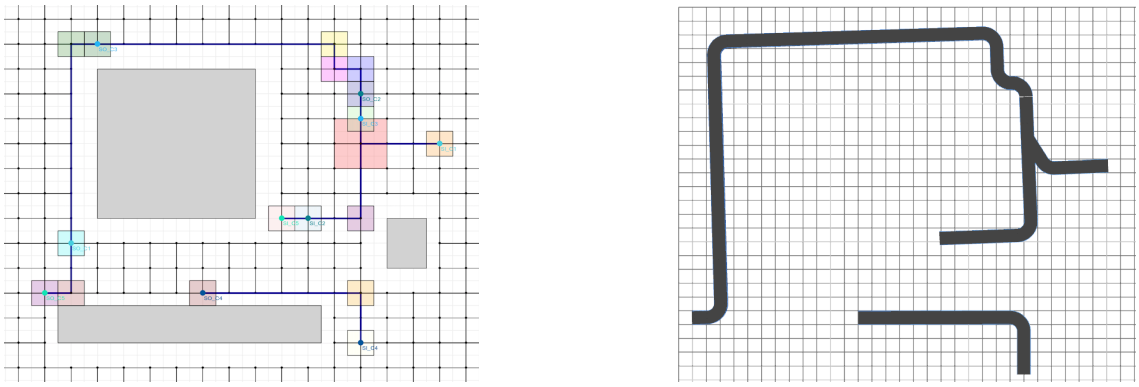
The computed optimal material flow in the case of a 30% obstacle occupancy as seen in Figure 4.10b was subsequently fed into the conveyor placement optimization model. The tolerances for the specific conveyors stayed the same as for the previous test runs, as seen in Figure 4.15a. When looking at the runtimes the same pattern as before is visible. The additional obstacles seem to increase the complexity and by that increasing the runtime again by 25%. The model



(a) **(b)**

Figure 4.14: 15% Obstacle Occupancy (a) Defined Tolerances (b) Computed Conveyor System

once again struggles to find a feasible solution but when found its optimality is proven almost instantaneously. The resulting conveyor system drawn in the simulation model is seen in Figure 4.15b.

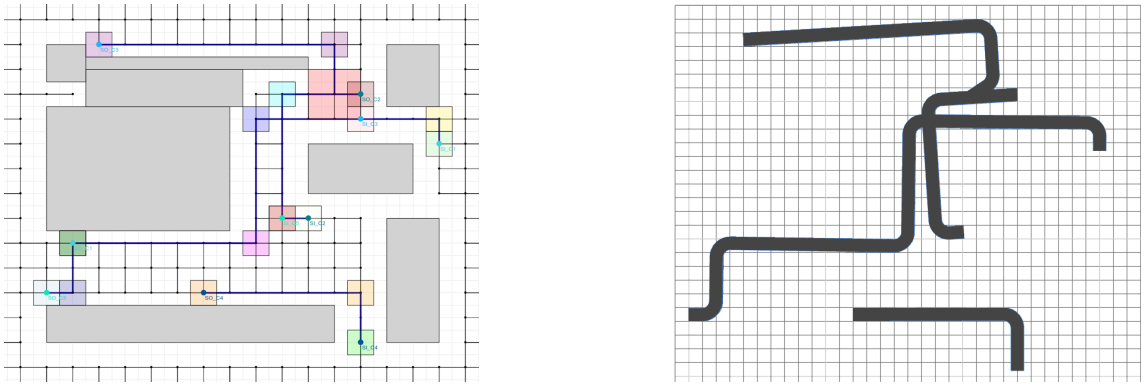


(a) **(b)**

Figure 4.15: 30% Obstacle Occupancy (a) Defined Tolerances (b) Computed Conveyor System

For the final test run - 50% obstacle occupancy - the defined tolerances, same as before, are visualized in Figure 4.16a and the computed placement of the conveyor system is seen in Figure 4.16b. Surprisingly, this time the model only needed 16 seconds to compute the optimal solution, only 1 second slower than for the test run without any obstacles. Even though every obstacle adds additional constraints the model was able to outperform the 15% and 30% test runs. A possible reason for that might be the different material flow graph of the 50% example compared to the other test runs. The for this test run resulted different material flow graph seems to positively influence the performance more than the added complexity due to the obstacles influences it negatively.

Upon comparing the test runs concerning the material flow model, the model's performance shows expected characteristics when obstacles are introduced. Each added obstacle effectively decreases the model's complexity by limiting the number of possible solutions, given that enough arcs with sufficient capacities remain to connect the commodities. Therefore, the model's feasibility



(a) (b)

Figure 4.16: 50% Obstacle Occupancy (a) Defined Tolerances, (b) Computed Conveyor System

generally remains undisturbed by these alterations. However, when examining the runtime of the third test run, it's evident that additional obstacles can have negative impact on the performance. Surprisingly, doubling the obstacles from the second test run didn't improve performance as expected, but instead, it doubled the runtime. The most substantial influence is observed in the fourth test run, which took only nine seconds to find and prove optimality.

On the other hand, the conveyor placement model exhibits a different response to the incorporation of obstacles. In comparing the first three test runs, each extra obstacle imposes more constraints on the model, thereby amplifying its complexity and lengthening the runtime. The number of these additional constraints depends on the number of segments into which each straight conveyor is divided. Since the computed material flow of the first three test runs is fairly similar, the increase in runtime can be primarily attributed to the additional constraints and variables induced by more obstacles. However, the final test run reveals that the structure of the material flow graph itself can have a more significant, in this case positive, influence on the performance.
Lastly, it should be noted that an excess of poorly placed obstacles can render the model infeasible or necessitate a larger tolerance area around nodes due to the inherent geometric restrictions of the conveyors used.

4.2.3 Influence of the Objective Function

As described in Section 4.1, a variety of objective functions have been incorporated. These span from minimizing total costs, decreasing curve usage, reducing total length, to a combination of all three using weighted factors to modify the focus of the objective function. Subsequent test runs are designed to examine the impacts of the cost and curve usage objective functions on both the performance and the quality of the solutions generated. The specifics of these tests are thoroughly outlined in Table 4.7. It's noteworthy to mention that the component that minimizes the total length of the material flow was not investigated. This is because for the case of a unit grid with unit costs, the total cost and total length functions yield identical results. Consequently, the γ value is set to 0.
All five test runs will analyze the material flow through an identical layout and commodity system, depicted in Figure 4.17.

Table 4.7: Test Scenarios for the Influence of the Objective Function

id	Comms.	Objective Function	α	β
3_1	4	Total Costs	1	0
3_2	4	Curve Usage	0	1
3_3	4	Combination	0.5	0.5
3_4	4	Combination	0.33	0.67
3_5	4	Combination	0.67	0.33

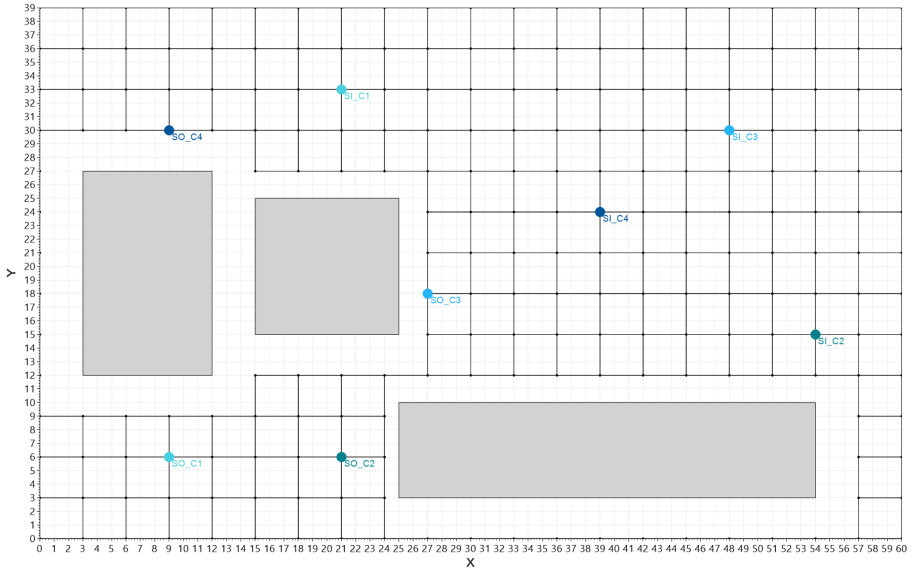


Figure 4.17: Layout and Commodity System to test the Influence of the Objective Function

The first test run will concentrate solely on minimizing total costs, encompassing both fixed and variable costs. The second test will aim to reduce curve usage, disregarding costs. The final trio of tests will consider both aspects with varying weights: initially, the model will assign equal priority to both; next, it will place a greater emphasis on curve reduction; and finally, it will prioritize cost minimization. In Table 4.8 are the results of these test runs compared. For the first run, a viable solution is identified in the first seconds of the optimization, yet validating its optimality poses a greater challenge. Although the model rapidly diminishes the gap during the initial moments of optimization, it grapples with further reductions once the gap value falls below 12%. Within the first second, the model successfully computes an objective value of 1308.12, accompanied by a gap of 20.5%. However, the ensuing 25 minutes of the optimization process yielded a reduction in the objective value to 1307.46 - a relatively marginal improvement considering the time invested. This highlights an interesting aspect of the model's performance, revealing a potential plateau in its optimization capabilities once a certain threshold in the objective value is reached. The optimal material flow, calculated with a sole focus on costs, is depicted in Figure 4.18. The orientation of the selected objective function is unmistakably visible, as the model evidently strives to combine as many arcs as possible to minimize the fixed charge associated with arc selection. Simultaneously, given the unit length of the grid, the objective function remains indifferent to avoiding curves. This pattern is particularly noticeable on the right-hand side of the layout, where the model opts

Table 4.8: Results of the Test Scenarios for the Influence of the Objective Function

id	Objective Function	α	β	Lowest Gap.	Runtime total	Lowest Obj. Value	Runtime Best Solution
3_1	Total Costs	1	0	0.00%	25min 28s	1307.46	15s
3_2	Curve Usage	0	1	0.00%	16h 22min	600	17s
3_3	Combination	0.50	0.50	0.00%	16h 40min	5736.00	23s
3_4	Combination	0.33	0.67	0.00%	13h 32min	5825.76	27s
3_5	Combination	0.67	0.33	0.00%	5h 41min	5295.88	3min 26s

for multiple curves instead of a singular curve. This choice reflects the model's objective function - purely from a cost perspective, there is no differential between utilizing various curves or one single curve.

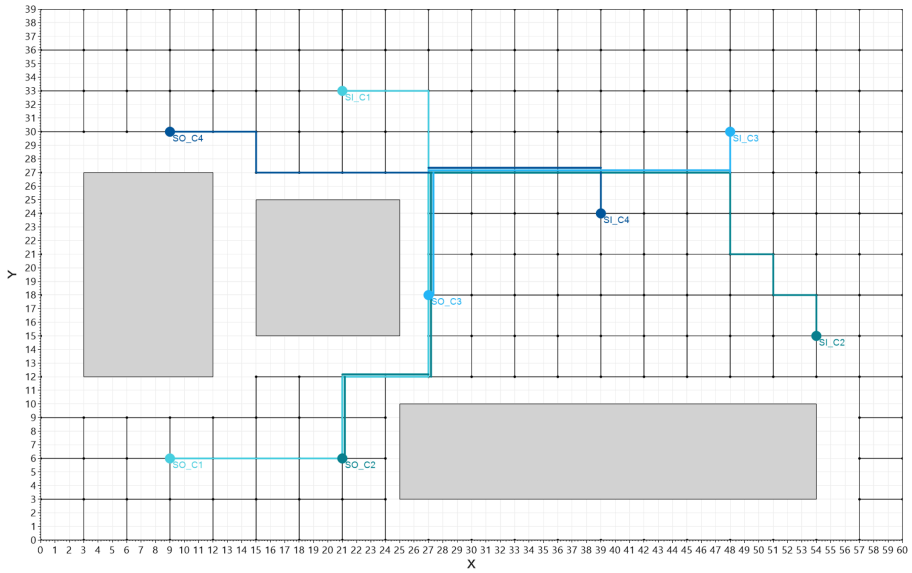


Figure 4.18: Optimal Material Flow when minimizing Total Costs only

Employing the curve-avoiding objective function leads to a significant drop in performance. To drive the gap to 0% and thereby substantiate the optimality of the discovered objective value, the model required a staggering 16 hours and 22 minutes. However, akin to previous observations, the viable solution found within the first seconds - 17s to be precise - is determined to be the optimal solution. Thus, the model expends virtually all of its time confirming the optimality of the solution it had identified within the initial seconds. This observation points to an interesting dynamic in the model's performance: despite shifting the focus to curve avoidance and subsequently facing a performance decline, it still manages to pinpoint the optimal solution almost instantaneously. However, the assurance of this solution's optimality appears to be the primary bottleneck, consuming nearly all of the model's computational efforts.
The resultant optimal material flow is illustrated in Figure 4.19. Here, the model disregards the combination of arcs for cost reduction and concentrates solely on minimizing curve usage, as each curve incurs a penalty. This approach favors the extensive usage of straight conveyors;

however, in this particular case, it resulted in the creation of three distinct graphs, which might not constitute the most practical solution in real-world applications. This outcome highlights the importance of balancing multiple objectives in the optimization process. Purely focusing on one aspect - whether it's cost reduction or curve avoidance - may not always yield the most effective or feasible results. Consequently, a combined approach leveraging both objective functions might offer a more robust and practical solution, facilitating the generation of layouts that balance cost efficiency and operational feasibility.

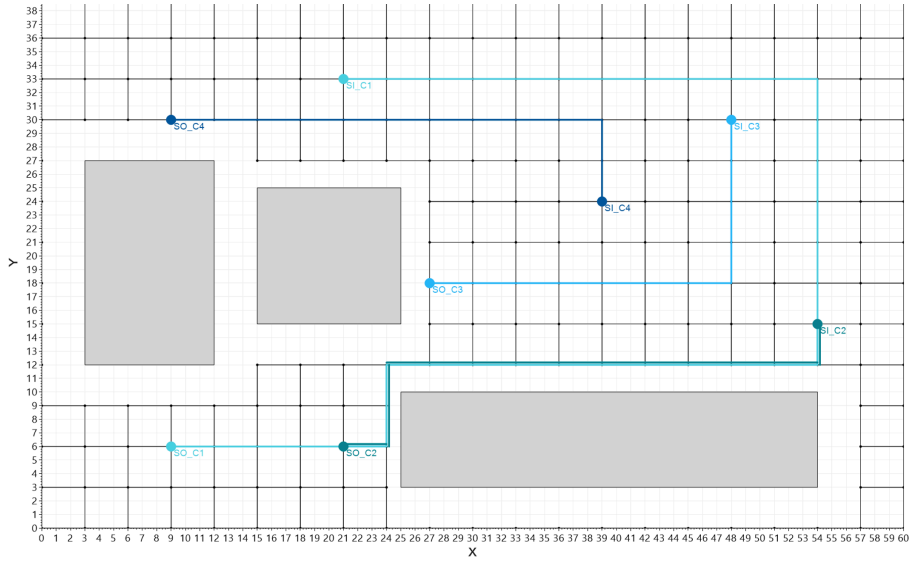


Figure 4.19: Optimal Material Flow when minimizing Curve Usage only

To experiment with this methodology, a blend of both objective functions was assembled, utilizing weights to provide the flexibility to shift the focus. The inaugural trial under this amalgamated setup was run with identical weights assigned to both objective functions. The trend of gap reduction and objective value evolution mirrors that seen in the previous optimization run, with the model identifying the optimal in the first seconds but struggling to progressively reduce the gap to 0%.
Nevertheless, a marginal deterioration in performance was noticeable when compared to the scenario utilizing only curve usage minimization. To completely eliminate the gap down to 0%, the model necessitated a duration of 16 hours and 40 minutes. Yet, the optimal objective value of 5736 was rapidly found within a mere 23 seconds of optimization, however with a prevailing gap of 45% at that point and proving its optimality took the remaining computing time.
The optimal material flow computed with the combination of the two objective functions is illustrated in Figure 4.20. When contrasted with the previous two optimizations, the influences of both objectives are distinctly apparent. While the model continues its strategy of curve avoidance, it now incorporates a conscious effort to combine arcs for different commodities in a bid to mitigate additional costs.

In the subsequent test run, the weighting parameters were reconfigured to focus more on curve usage ($\alpha = 0.33$ and $\beta = 0.67$). Once again, the model was able to decrease the objective value to 5825.76 while maintaining a gap of 46.21% in a mere 27 seconds. Nonetheless, verifying that this result was indeed the optimal solution required the model to run for a protracted period of 13

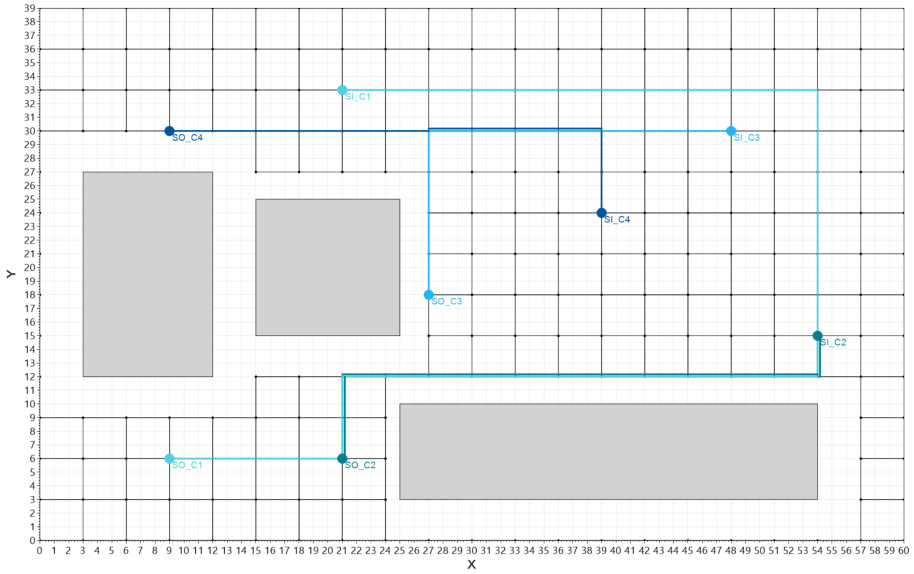


Figure 4.20: Optimal Material Flow - Combined Objective Function with equal weights

hours and 32 minutes. Shifting the focus towards the curve usage minimization led to an increase in performance.

The material flow outcome from this run is presented in Figure 4.21. Interestingly, it mirrors the flow calculated with equal weights, but was computed 3 hours and 8 minutes faster. This correlation could potentially be attributed to the interrelationship between the costs and curve penalty values. For these specific tests, both, the curve penalty and the fixed charge of employing an arc were set to 100, while the variable cost was established at 1 per unit flow. Alterations to these parameters will invariably influence the resulting material flow and the final objective value.

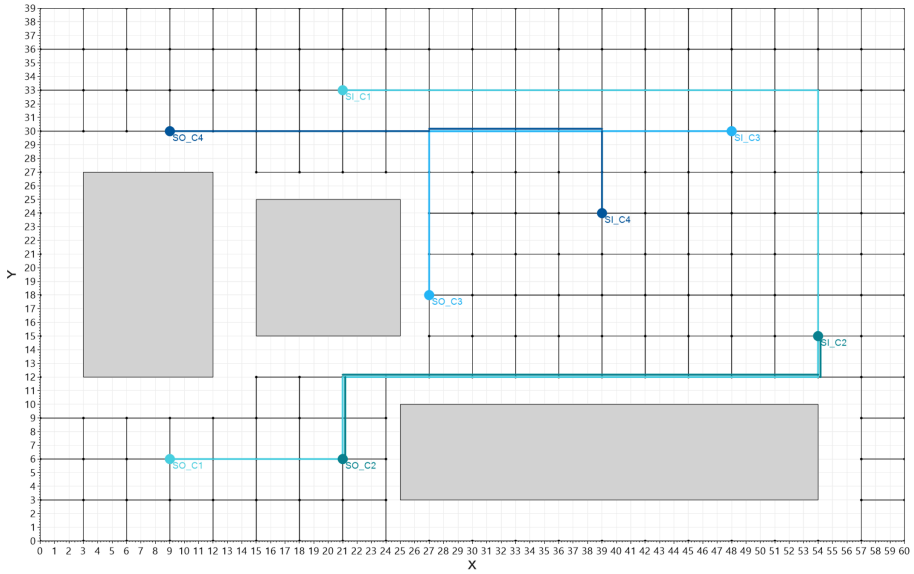


Figure 4.21: Optimal Material Flow - Combined Objective Function with Focus on Curve Usage

In the final test run, the combined objective function was once again utilized, but this time with a tilt towards cost minimization ($\alpha = 0.67$ and $\beta = 0.33$). Notably, the model was not able to determine the optimal value within the first 30 seconds as for the other test run, instead it took 3 minutes and 26 seconds with a corresponding gap of 17.35%. Nevertheless, validation of its optimality proved significantly faster compared to other test runs utilizing the combined objective function. The refocused emphasis on cost minimization led to a total computational runtime of 5 hours and 41 minutes, highlighting the efficiency gained in this aspect.
Figure 4.22 vividly illustrates the impact of shifting the focus towards cost minimization. The model strives to amalgamate as many arcs as feasible while simultaneously limiting the use of curves.

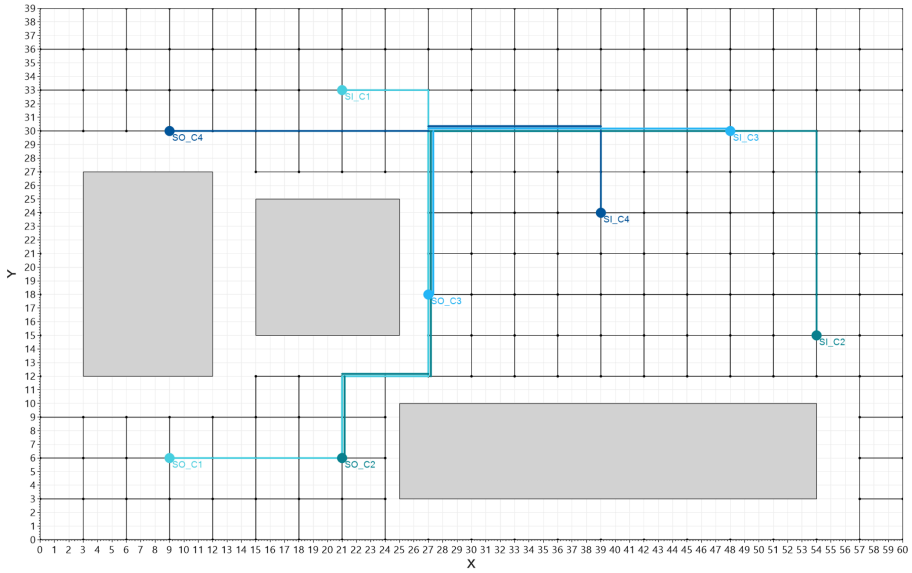


Figure 4.22: Optimal Material Flow - Combined Objective Function with Focus on Cost Minimization

The chosen objective function has a significant impact not only on the runtime but also on the resultant solution. The minimization of curve usage results in an increased number of constraints and decision variables, which in turn exacerbates the runtime due to added complexity. From a practical standpoint, minimizing the number of curves could be crucial to achieve a simpler and more streamlined solution, which favors longer straight conveyors over potentially superfluous curves.
It's noteworthy that despite the immense runtime increase when utilizing the curve minimizing function, the optimal objective value is often achieved in the early stages of the optimization process. However, this poses a challenge since one can only conjecture that a value is optimal if it remains unchanged for an extended period. Regrettably, without further computation, this presumed optimality remains an assumption rather than a validated conclusion.

4.2.4 *Influence of a Warm-Start*

From prior test runs, it is evident that minimizing the use of curves is computationally intensive, yet it can be pivotal in many practical applications. To enhance the performance of this optimization, implementing a warm start could prove beneficial. This is accomplished through a two-step optimization process. Initially, the model is optimized using an alternate objective function, in this case, for the first test run the total cost function, and for the second run the combination using the introduced weights. The derived solution is then retained and employed as the initial solution for the secondary optimization, where the curve usage is minimized. The potential enhancements via this method are explored through the test runs denoted in Table 4.9.

Table 4.9: Test Scenarios for the influence of a WarmStart

id	Comms.	Two Step Opt.	Obj. 1st Run	Obj. 2nd Run
4_1	3	false	Curve Usage	/
4_2	3	true	Min Total Costs	Curve Usage
4_3	3	true	Comb. Cost/Curves	Curve Usage
4_4	4	false	Curve Usage	/
4_5	4	true	Min Total Costs	Curve Usage
4_6	4	true	Comb. Cost/Curves	Curve Usage

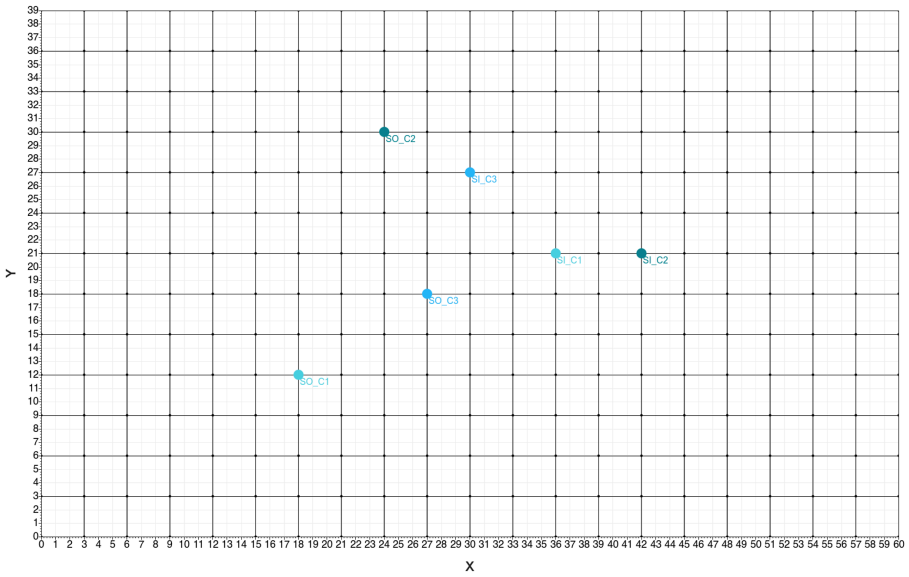


Figure 4.23: Layout and Commodity System - First Example Warmstart

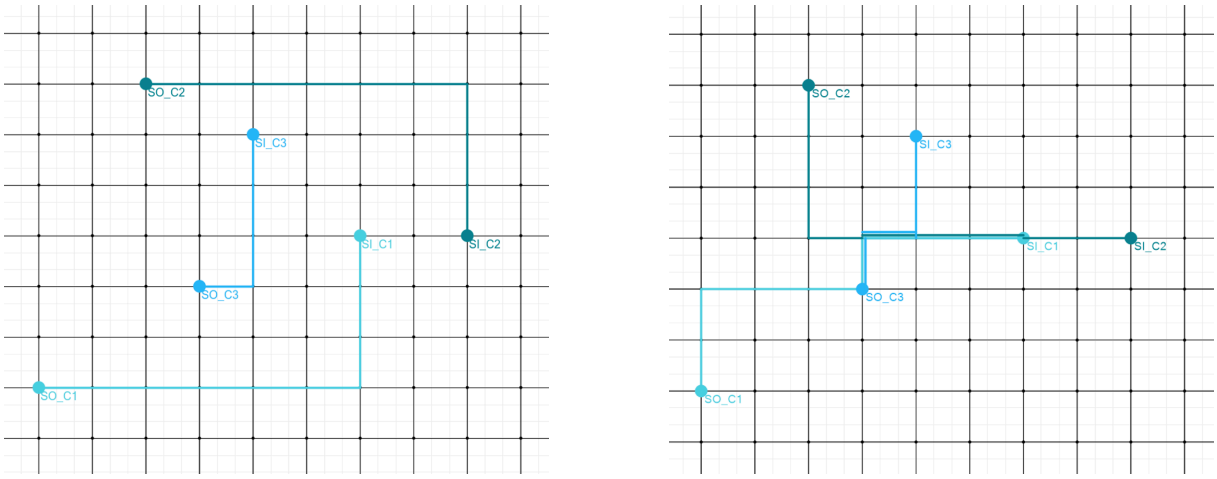
In Figure 4.23 the layout and the three commodities used for the first example are visualized. In an effort to establish a baseline for the duration required to minimize curve usage for a particular problem, the model initially attempts curve minimization without a preliminary test run - referred to as a 'warm start'. The resulting material flow can be observed in Figure 4.24a. As discussed in the preceding chapter, the objective function consists solely of penalties for each selected curve. As noted in Section 4.2.3, the complexity of minimizing curves exceeds that of the relatively simpler approach of merely minimizing costs.

This observation is again corroborated by the results of the initial test run, seen in Table 4.10. The model is quick to find a feasible solution in the opening seconds of the optimization process; however, it grapples with confirming its optimality, with the gap reducing at a glacial pace. It is worth mentioning that despite the relatively simple nature of the problem, successfully identifying the optimal solution took the model 25 minutes 43 seconds.

Table 4.10: Results of the Test Scenarios for the influence of a WarmStart

id	Obj. 1st Run	Gap 1st Run	Runtime 1st Run	Obj. 2nd Run	Gap 2nd Run	Runtime 2nd Run
4_1	Curve Usage	0.00%	25min 43s	/	/	/
4_2	Total Cost	0.00%	9s	Curve Usage	0.00%	28min 40s
4_3	Cost/Curve	0.00%	29s	Curve Usage	0.00%	9min 14s
4_4	Curve Usage	0.00%	12min 54s	/	/	/
4_5	Total Cost	0.00%	21s	Curve Usage	0.00%	20min 16s
4_6	Cost/Curve	0.00%	46s	Curve Usage	0.00%	16min 0s

Two distinct strategies can be employed while utilizing the WarmStart feature, each differing in the optimization objective of the initial run. The first strategy involves optimizing the total costs, whereas the second encompasses a combination of cost and curve penalties applied through weighting. Both strategies merit examination.
The second test run is the initial run of the two-step optimization process with an emphasis on total cost. The first feasible solution discovered turned out to be the optimal one, but yet again, proving its optimality proved to be the most challenging aspect. Notably, reducing the gap for this comparatively simple example required a mere 9 seconds.
The contrasting results of the curve only and the cost only minimization are shown in Figure 4.24. The material flow on the right clearly shows the focus on minimizing solely the costs.

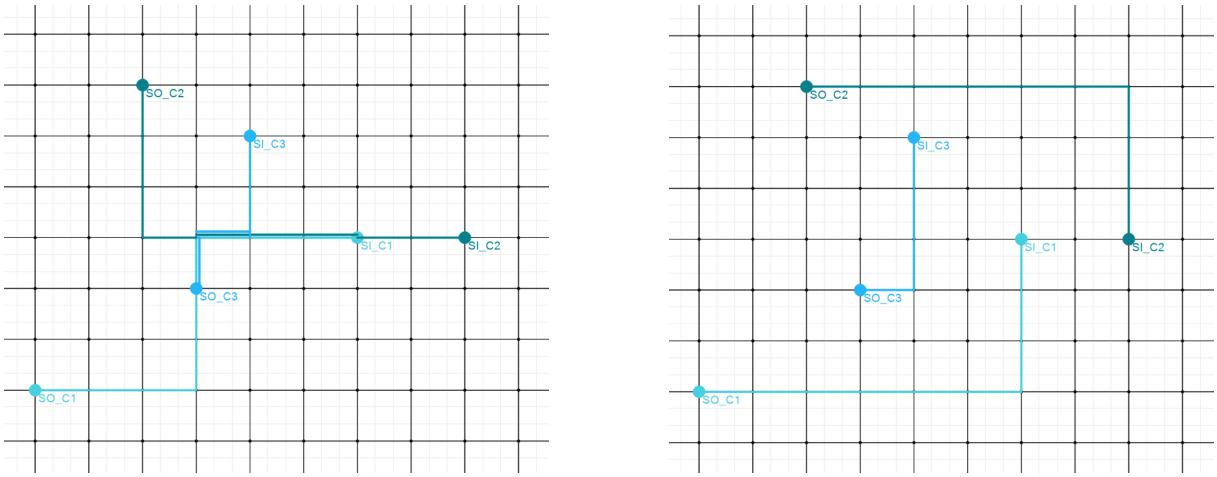


(a) **(b)**

Figure 4.24: Computed Material Flows First Example: (a) Curve avoiding without WarmStart, (b) Minimizing Total Costs

The results obtained from the previous step, achieved by minimizing total costs, are employed as a WarmStart for the subsequent minimization of curve usage.

One remarkable observation is the unexpected increase in runtime needed to prove optimality. Achieving a 0% gap took the model 28 minutes and 40 seconds, approximately 3 minutes longer than without the WarmStart. This increase in runtime could potentially be attributed to the differing objectives between the first and second run. The first run of the two-step optimization primarily focuses on total costs, which may inadvertently hinder the goal of minimizing curve usage in the second step.
To explore this further, the same example was subjected to a two-step optimization, but this time the initial run applied a combined objective function, which incorporated weights for total costs and curve penalties ($\alpha = 0.67$, $\beta = 0.33$ and $\gamma = 0$). This time the optimal solution was found nearly instantaneously, after just 2 seconds. The reduction of the gap to 0% required an additional 27 seconds amounting to a total of 29 seconds. This solution is again used to feed it back to the optimization model to minimize the curve usage. This time the model only needed roughly 9 minutes to find and prove the optimal solution, 19 minutes faster than the first two step optimization and 16 minutes faster than without using a WarmStart. The reason for that is most likely the more adequate first solution as input for the second run. Again, the first feasible solution found turned out to be the optimal solution. The computed material flows of the two runs (id = 4_3) are seen in Figure 4.25.



(a) **(b)**

Figure 4.25: Computed Material Flows First Example: (a) First Run - Combination, (b) Second Run - Curve Usage only

The influence of the first solution when using the WarmStart feature is explored further using another example. The corresponding layout and commodity system is shown in Figure 4.26. Once again, the baseline is obtained by minimizing the curve usage without a WarmStart and afterwards two test runs using the two step optimization process, first with the cost function, secondly using the combination of cost and curve penalties. When minimizing the curve usage without any pre-solution the gap and objective reduction takes nearly 13 minutes, which is rather unexpected as for this example an additional commodity was added. The same example is now optimized using the two step optimization with the total cost function as the prior optimization. The total computation time for that test run was 21 seconds, more than twice as long as for the first example.
The two computed material flows are plotted next to each other in Figure 4.27.

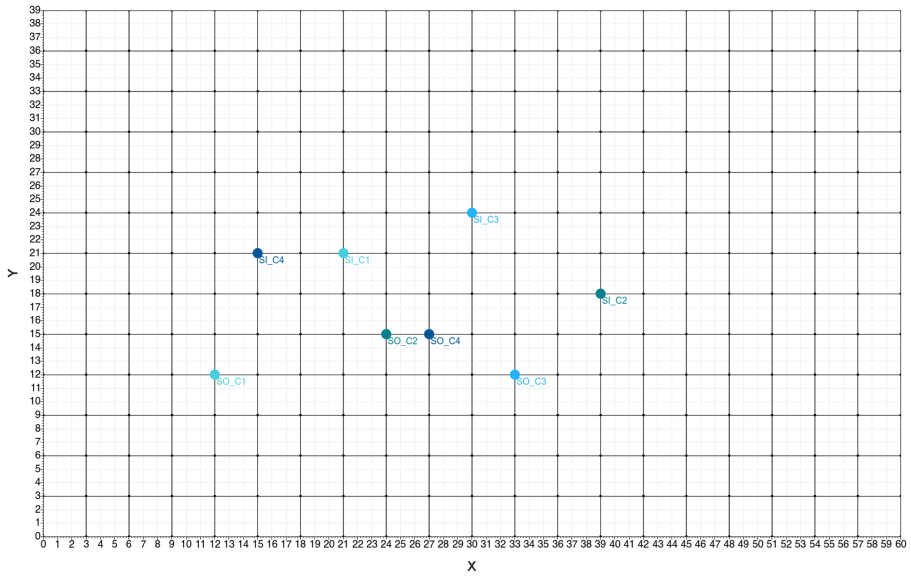


Figure 4.26: Layout and Commodity System - Second Example Warmstart

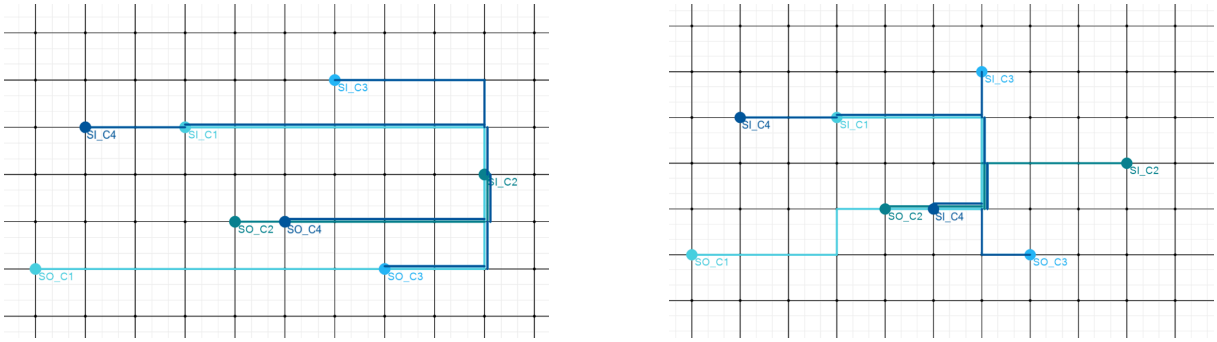
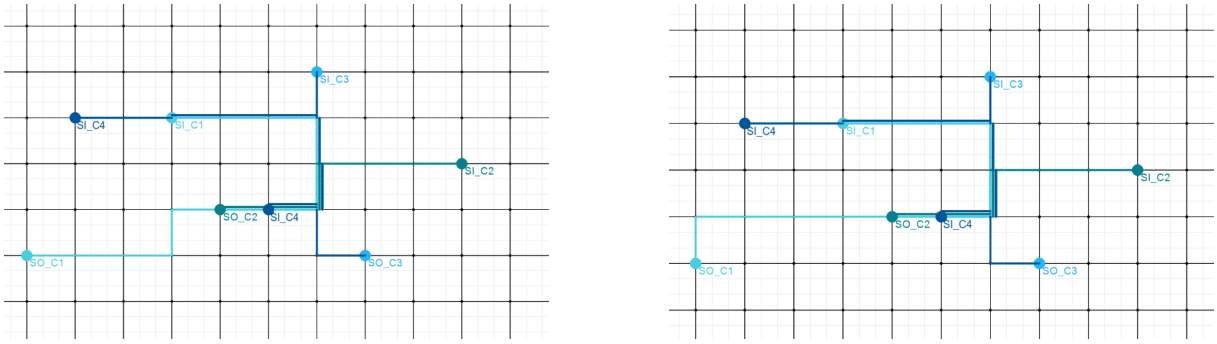


Figure 4.27: Computed Material Flows Second Example: (a) Curve avoiding without WarmStart, (b) Minimizing Total Costs

This material flow computed with the focus solely on minimizing costs is then used for the second step optimization focusing once more on minimizing curve usage. Similar to the first example, the solution of the first step seems not adequate when using the total cost function. Instead of improving the performance of the curve optimization it worsens it and results in a roughly 55% runtime increase.

As before the two step approach is tested again with a most likely more adequate first objective function by focusing on both at the same time, the costs as well as the curve penalties ($\alpha = 0.67$, $\beta = 0.33$ and $\gamma = 0$). Similar to the first example this optimization takes longer than the minimizing of just the total costs. The overall runtime is 46 seconds and the later to be proven as optimal objective value was found after just 9 seconds. In Figure 4.28 the computed material flows of the two initial runs with the two different objective functions are compared.

The material flow resulting from the optimization using the combined objective function is once more used as a WarmStart for the curve usage minimization. As seen in Table 4.10, the influence of a more adequate first solution as a WarmStart is clearly visible when comparing the runtimes of



(a) **(b)**

Figure 4.28: Computed Material Flows Second Example: (a) First Run - Minimizing Total Costs, (b) First Run - Combination

the two step optimization with the cost function. When using the combined objective function as a WarmStart, the runtime decreases by nearly 20%. However in this example using the two step optimization could not improve the performance compared to optimizing the curve usage directly. It is noteworthy that the direct curve usage optimization was surprisingly faster than in the first example even though the complexity increased. This proves once more the heavy assessability of MLP (NP-Hard) models.

Both examples highlight the critical role of the causal connection between the optimization objectives in the first and second runs. If we exclusively optimize costs in the first run and use its solution as a warmstart for the curve usage in the second run, the total runtime for both examples is negatively affected.
When considering the total runtimes of the two runs with a causally linked warmstart, the first problem is solved in 35% of the time compared to the scenario without the warmstart, indicating a clear enhancement in performance. Conversely, for the second problem, solving the model directly with the curve-minimizing method takes approximately 13 minutes, whereas employing the warmstart extends the model's runtime to nearly 17 minutes, thus demonstrating a decrease in performance.

4.2.5 Influence of the Heuristics and MIPFocus Parameter

The optimization tool 'Gurobi solver', offers an array of tunable parameters which can be individually calibrated to improve the performance of the model. The ideal configuration, however, relies heavily on the specifics of the model under consideration. Given the abundance of parameters - around 50 unique ones, many of which feature multiple settings, an extensive range of combinations is conceivable.
In this analysis, two distinctive models, the material flow model and the conveyor placement model, are under examination. Each model poses its unique challenges, therefore requiring separate considerations. The primary focus lies on two parameters: the Heuristics and the MIPFocus parameters.

During the testing phase of both models, distinct performance-related challenges were identified. While the material flow model is generally quick to find a feasible solution, it has a tendency to struggle when demonstrating optimality. On the other hand, the conveyor placement model

tends to find an initial feasible solution more slowly, yet once such a solution is identified, it demonstrates a capacity for rapid gap reduction.
The Gurobi solver usually employs an initial heuristic approach to expedite the discovery of a feasible solution, then shifts its focus towards gap reduction. Consequently, adjusting the percentage of Heuristics and MIPFocus parameters could potentially influence performance. The Heuristics parameter in Gurobi solver varies between 0 and 1, with a default value of 0.05, while the MIPFocus parameter is an integer ranging from 0 to 3, with a default setting of 0. To understand the influence of these parameters on both models, a series of test runs as delineated in Table 4.11 were performed for each model.

Table 4.11: Test Scenarios for the influence of the Heuristics and MIPFocus Parameters. The (*) next to a number indicates its default value

id	Limit X	Limit Y	Comms.	Heuristics	MIPFocus
5_1	60	40	4	0.05*	0*
5_2	60	40	4	0	0
5_3	60	40	4	0.5	0
5_4	60	40	4	1	0
5_5	60	40	4	0	1
5_6	60	40	4	0	2
5_7	60	40	4	0	3
5_8	60	40	4	1	3

Figure 4.29 illustrates the commodity system used for these test runs.

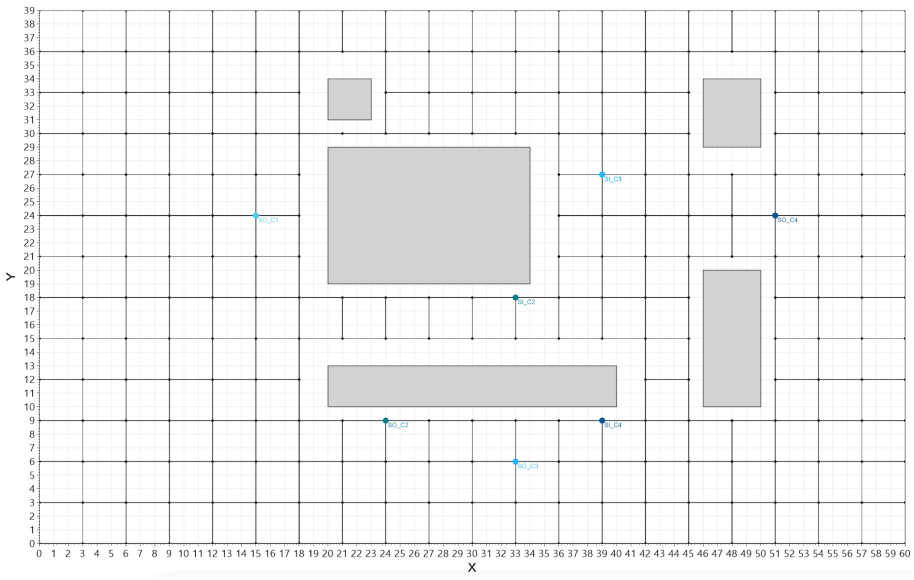


Figure 4.29: Layout and Commodity System for Test Runs to evaluate Influence of Heuristics and MIPFocus Parameters

The tests were carried out with a grid size of 3, 4 commodities, and 5 differently-sized obstacles. Both the material flow optimization model and the conveyor placement optimization model were

evaluated. The results of the 8 test runs for the material flow model, sorted by their total runtime from fastest to slowest, are summarized in Table 4.12.

Table 4.12: Results of the Test Runs for the Influence of the Heuristics and MIPFocus Parameter on the Material Flow Model. The (*) next to a number indicates its default value

id	Heuristics	MIPFocus	Lowest Gap.	Runtime total	Lowest Obj. Value	Runtime Best Solution
5_1	0.05*	0*	0.00%	**16s**	1039.83	**0.04s**
5_2	0	0	0.00%	**16s**	1039.83	15s
5_4	1	0	0.00%	17s	1039.83	0.41s
5_6	0	2	0.00%	20s	1039.83	13s
5_3	0.5	0	0.00%	28s	1039.83	0.35s
5_7	0	3	0.00%	42s	1039.83	41s
5_5	0	1	0.00%	49s	1039.83	8s
5_8	1	3	0.00%	1min 12s	1039.83	0.24s

The most efficient performance was obtained during the first two test runs, when leaving these parameters at their default values (Heuristics at 0.05 and MIPFocus at 0) or, both the Heuristics parameter and the MIPFocus parameter set to 0. Both combinations yield to a runtime of 16 seconds. Though, when leaving the paramters at their default values the optimal solution was found in just 0.04 seconds, with the remaining 16 seconds spent confirming its optimality. When the Heuristics parameter was set to 0, the optimal solution was identified just before its optimality was confirmed at 15 seconds. Across the various configurations outlined in Table 4.11, the runtime for the material flow model ranged from 16 seconds to 1 minute and 12 seconds. Worth noting is the combination of a Heuristics parameter set to 1 and a MIPFocus parameter set to 0: while it took only a second longer than the fastest two combination overall, it also managed to pinpoint the optimal solution in less than a second during the optimization process, similar pattern as for the default combination. The remaining time ? 16 seconds ? was used to validate its optimality.

The same series of test runs were executed using the conveyor placement optimization model, with the material flows calculated in the previous tests serving as inputs. The results, displayed in Table 4.13, are sorted by total runtime from the fastest to the slowest. Interestingly, the parameter combination leading to the best performance for this model is also the default setting. However, while the runtime of the first model varied from 16 seconds to 1 minute and 12 seconds, the runtime for the conveyor placement model exhibited a much broader range, spanning from 20 seconds to over 2 hours. In these two extended scenarios, the optimization process had to be terminated without reaching a 0% gap. This underscores the substantial differences in the two models.

In light of the longer runtime required by the material flow model to establish optimality in all previous performance tests, the same tests were re-run on this model, but this time with a reduced grid size of 2. The reduction in grid size is intended to heighten the complexity of the material flow model, and thereby amplify the impact of the two parameter settings. The change is expected to provide more insightful results about the model's behavior under increased complexity and its response to different parameter configurations. The results of both test runs are compared next to each other in Table 4.14. For the smaller grid size of 2, the fastest performance was achieved

Table 4.13: Results of the Test Runs for the Influence of the Type of Objective Function on the Conveyor Placement Model. The (*) next to a number indicates its default value

id	Heuristics	MIPFocus	Lowest Gap.	Runtime total	Lowest Obj. Value	Runtime Best Solution
5_1	0.05*	0*	0.00%	**20s**	331.43	**20s**
5_2	0	0	0.00%	1min 26s	331.43	1min 26s
5_3	0.5	0	0.00%	2min 2s	331.43	2min 1s
5_7	0	3	0.00%	2min 30s	331.43	2min 29s
5_8	1	3	0.00%	4min 21s	331.43	3min 1s
5_6	0	2	0.00%	5min 14s	331.43	5min 13s
5_4	1	0	0.27%	2h 0min	331.43	1h 19min
5_5	0	1	1.53%	2h 0min	331.46	56min

Table 4.14: Comparison of the two Test Runs on the Material Flow Model. The (*) next to a number indicates its default value

id	Heuristics	MIPFocus	Lowest Gap 1st	Runtime total 1st	Lowest Gap 2nd	Runtime total 2nd
5_1	0.05*	0*	0.00%	16s	0.00%	25min 20s
5_2	0	0	0.00%	16s	0.00%	**6min 12s**
5_3	0.5	0	0.00%	28s	0.00%	11min 24s
5_4	1	0	0.00%	17s	0.00%	19min 8s
5_5	0	1	0.00%	49s	12.27%	1h 0min
5_6	0	2	0.00%	20s	0.00%	6min 30s
5_7	0	3	0.00%	42s	0.00%	48min 5s
5_8	1	3	0.00%	1min 12s	3.65%	1h 0min

by setting both parameters to 0. Contrastingly, adhering to the default settings inflated the runtime from 16 seconds to over 25 minutes. The range of runtimes for the second set of test runs was considerable, spanning from slightly over 6 minutes to beyond an hour, at which point the optimization process was terminated. The most unfavorable performance was observed when the Heuristics parameter was set to 0 and the MIPFocus parameter to 1. This configuration enabled the model to diminish the gap only to 12% over an hour-long runtime. In this particular set of examples ? where the same layout was evaluated across two different grid sizes ? it appears that the optimal configuration necessitates setting both parameters to zero. This arrangement yielded notable performance outcomes across both instances.

One downside with these parameter settings is the difficulty in determining the best configuration before executing the optimization. The Gurobi solver provides a tuning method that iteratively resolves the problem while altering the combinations of variables, thereby discerning the most effective combination. This could be beneficial for a model that must be optimized multiple times while maintaining a similar structure. However, for the two models developed in this study, it is unlikely to be useful as the model's structure and complexity dramatically shift with only slight layout changes.

This is illustrated in the example, where if the tuning process had been utilized on the smaller example (larger grid size), the optimal configuration would have been the default setting. But applying this setting to the larger example would have led to poorer performance compared to other configurations.

4.2.6 Summary of the Performance Metrics

Upon analyzing the varied factors evaluated, it is evident that the grid size and the chosen objective function substantially influence the performance of the material flow model. The complexity of the model, in terms of constraints and variables, escalates drastically when the curve usage function is employed, causing a detrimental impact on performance. However, this can be mitigated through the use of a warm-start, provided the objective function of the initial run somewhat considers curve usage. Otherwise, as observed in the test runs, the performance can worsen.

Moreover, the introduction of obstacles can significantly reduce the model's complexity, thus enhancing its performance. Consequently, it is advisable to eliminate unnecessary nodes and arcs by introducing obstacles before running the model. These obstacles could represent actual impediments like pillars or walls, or symbolize additional constraints if certain areas of the layout are known to be off-limits to the material flow in advance.
Conversely, for the conveyor placement model, obstacles can increase the complexity and negatively affect the performance. Across most of the tests, except the one evaluating the influence of parameters, the model efficiently finds the optimal solution within a reasonable timeframe. Although, compared to the material flow model, it usually struggles in locating the initial feasible solution but, once found, it quickly proves its optimality. However, due to the strict geometrical constraints of the conveyors, it might necessitate adjustment of tolerances around nodes to ensure the feasibility of the model.

When considering the practical application of these theoretical evaluations, certain factors come into play. Given that real-world layouts will likely be considerably larger than those used for evaluation, using a smaller grid size could potentially lead to unreasonable computation times. Therefore, it's essential to keep the grid as simple as possible by eliminating unnecessary nodes and arcs before optimization. If feasible, incorporating intermediate sinks for all commodities could be beneficial, as this allows the layout to be segmented into sub-problems, all while maintaining global optimization. An initial optimization run might be beneficial, focusing solely on the total cost objective function, since it provides the fastest solution. If minimizing curve usage is a necessity, consider using a warm-start approach where the first run is a combination of cost and curve usage objectives, followed by a second run focusing on curve usage. One could further enhance performance without sacrificing grid resolution by adopting an advanced grid structure - generally a larger grid size, but finely tuned at certain necessary points. However, the implementation of such a grid is yet to be explored.
Overall, the structure and size of the specific problem can escalate its complexity, making it either 'NP-Complete' or even 'NP-Hard'. When this happens, predicting the time or resources needed to find a solution becomes virtually impossible.

Furthermore, in the realm of complex optimization problems, especially those that strive to emulate real-world scenarios, the quest for absolute optimality - reducing the gap to 0% - may not always be a pragmatic or necessary pursuit. It's critical to acknowledge that these models are

simplified representations of the multifaceted reality and, as such, they can never attain 100% accuracy. Thus, while the reduction of the gap is indicative of a more optimal solution within the model's constraints, it may not significantly enhance the model's applicability or effectiveness in real-world implementation. Consequently, one must strike a balance between computational effort and the practical value of the optimization results.

4.3 Practical Evaluation of the Algorithm

In order to further investigate the performance and practical applicability of the models, two use cases were implemented. These use cases serve to apply the findings of the theoretical evaluations by optimizing the material flow and the conveyor placement. The focus was to ensure that the final conveyor system was as linear as possible to minimize unnecessary curve usage.
Given the size of the two use cases (1000m x 400m layout), a grid size greater than 10 would not provide additional value and might unnecessarily increase computational complexity. Thus, a grid size of 10 was chosen. The sources and sinks are placed so that all commodities pass through the same "tunnel". Intermediate sinks were placed at the entrance and exit of this tunnel, effectively dividing the layout into three parts. This allowed for individual optimization of each part, reducing the overall complexity while still preserving the ability to achieve a global optimum.

Table 4.15: Overview of the two Use Cases to assess the Practical Usability of the two Models

id	Lim X	Lim Y	GridSize	Heuristics	MIPFocus	Comms.	Gap MatFlow	Gap Conv.
UC_1	1000	400	10	0	0	8	10%	5%
UC_2	1000	400	10	0	0	9	10%	5%

The target gaps for the material flow model and the conveyor placement model were deliberately set at 10% and 5% respectively. While this may not necessarily yield the absolute optimal solution, the results from the theoretical evaluation suggests that optimal solutions can often be discerned at gaps exceeding 10%. This strategic tolerance allows for the efficient attainment of near-optimal solutions without the computational burden of pursuing a zero-gap outcome.
As discussed in Section 4.2.5, the most effective parameter combination for the more complex model was obtained by setting both the Heuristics and MIPFocus parameters to 0. This setting was thus also used for the use cases. Table 4.15 provides a summary of the two use cases.

The first use case involved eight commodities, with an intermediate combined source-sink relationship (Figure 4.30). The eight sources were positioned at the top left of the layout, while the eight sinks were placed at the top right.
The commodity connecting the two areas formed the intermediate source-sink connection. The initial optimization of material flow was performed over the entire layout, without considering the partition into subsystems. The cost minimization function was used as it provided the quickest results, in this case in 1 minute and 12 seconds. As can be seen in Figure 4.31, although this approach yields the near optimal solution in terms of minimizing total costs, it does not provide the best solution in terms of reducing curvature. To mitigate this, and to enhance performance, the model was divided into three submodels (top-left, bottom, and top-right) for subsequent analyses. The division was based on the intermediate source-sink connection. Additionally, the cost/curve-

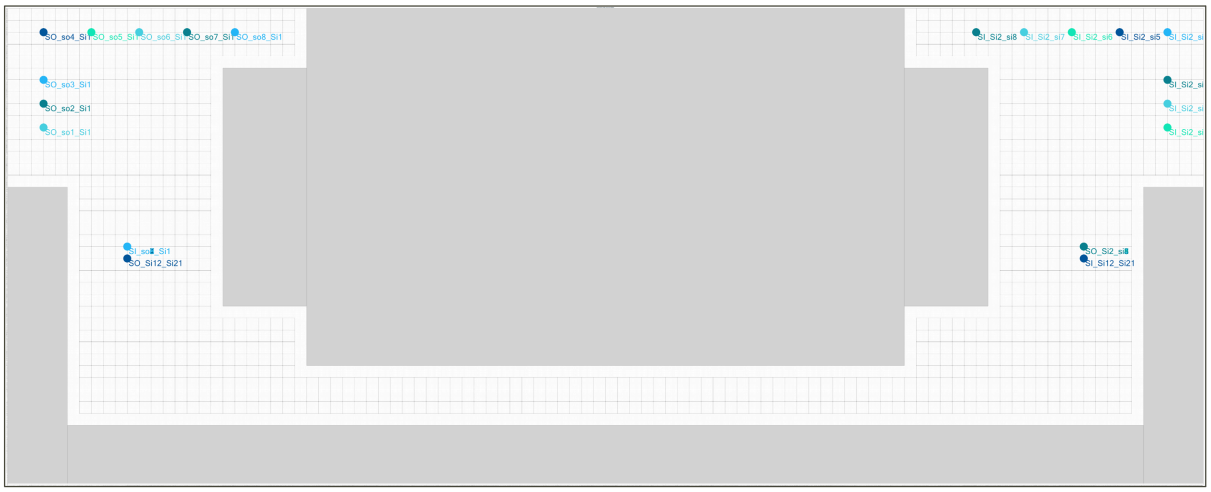


Figure 4.30: Layout and Commodity System for the first Use Case

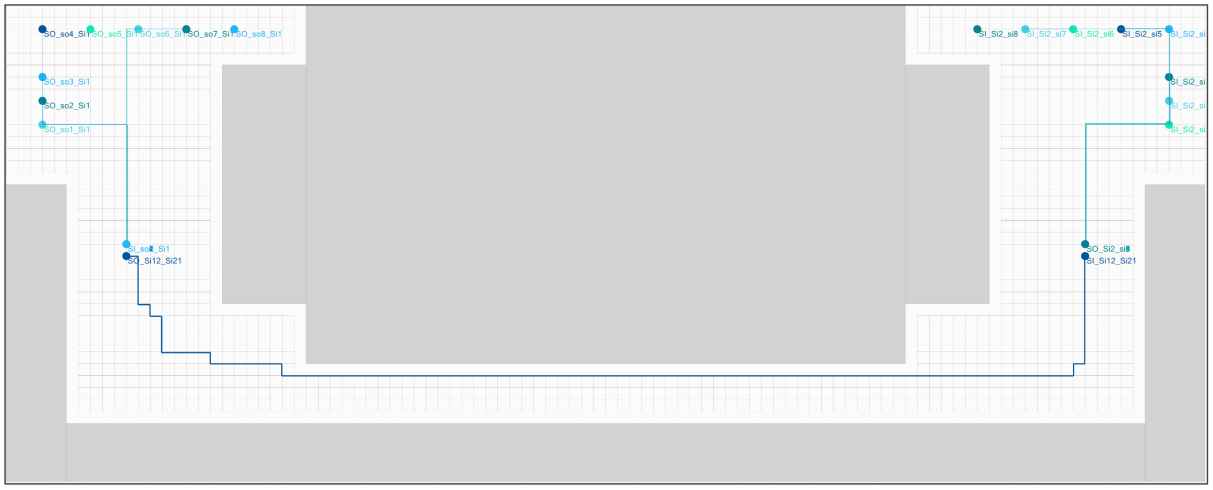


Figure 4.31: Computed Material flow for the first Use Case - Cost Minimization

usage objective function was used, with the resulting material flow serving as a warm start for a possible subsequent optimization process solely focusing on reducing curvature. The objective function was adjusted to assign weights of 0.7 to cost (α) and 0.3 to curve usage (β). The settings for the Heuristics, MIPFocus parameters, and grid size remained unchanged.

Figure 4.32 illustrates the newly computed material flow using the cost-curve combination. This flow was computed in three separate runs, with a total runtime of 9 minutes 43 seconds. Although the objective function prioritized cost minimization over reducing curve usage, the final material flow showed no unnecessary curves, making further optimization unnecessary. This solution was then used to calculate the optimal conveyor system, with default values assigned to the Heuristics and MIPFocus parameters based on the test results from Section 4.2.5. The optimization process to achieve the desired gap level of 5% took 3 seconds. The resulting conveyor system, implemented in the simulation program, can be seen in Figure 4.33.

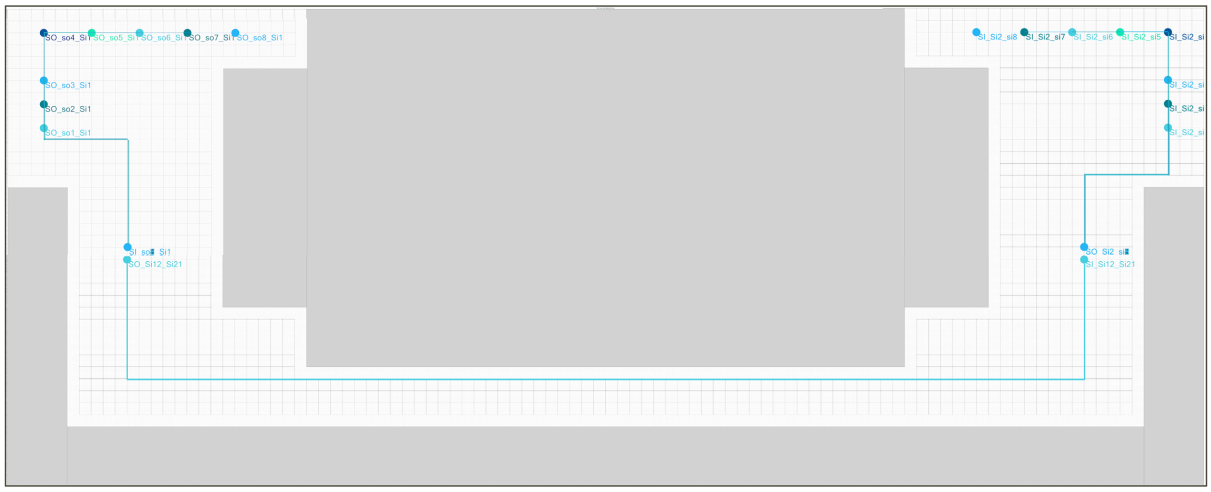


Figure 4.32: Computed Material flow for the first Use Case - Curve Minimization

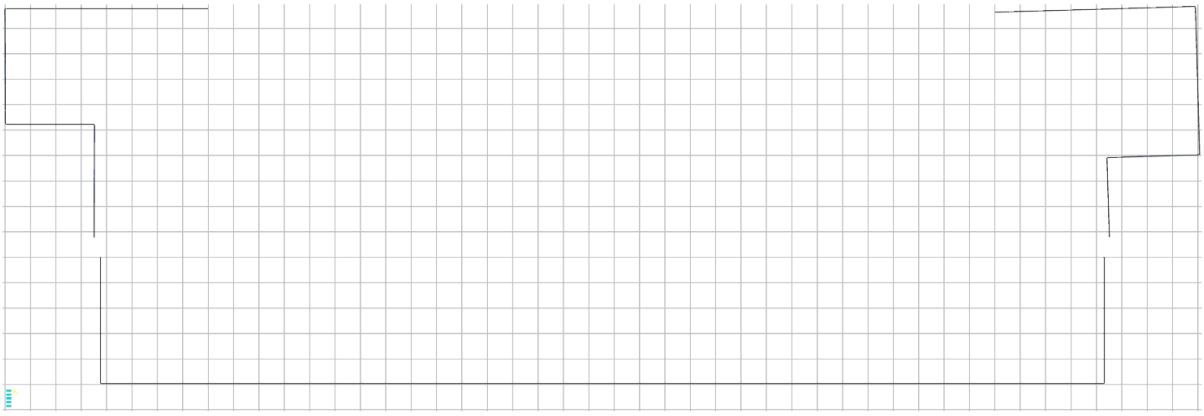


Figure 4.33: Computed Conveyor System - First Use Case

In the second use case, the layout was modified to include an additional commodity and several obstacles. However, the intermediate stops before and after the now more complex tunnel remained unchanged. The revised layout is depicted in Figure 4.34.

As with the first use case, the initial material flow optimization was carried out using the cost minimization objective function. The addition of obstacles led to a reduction in the number of nodes and arcs, but also to an increase in complexity due to the added commodity. Consequently, the runtime increased from 1 minute and 12 seconds to 6 minutes and 46 seconds. The resulting material flow is visualized in Figure 4.35.

While the computed material flow was nearly optimal concerning total cost, it contained too many curves. To enhance the material flow with regard to curve minimization, the objective function was adjusted to combine cost and curve usage minimization ($\alpha = 0.7$ and $\beta = 0.3$), and the model was re-run. The newly optimized material flow, which considers curve usage, is illustrated in Figure 4.36. The improvements in terms of curve usage are clearly evident. The total runtime for the material flow optimization (cost and cost/curve-usage optimization) was 10 minutes and 13 seconds.

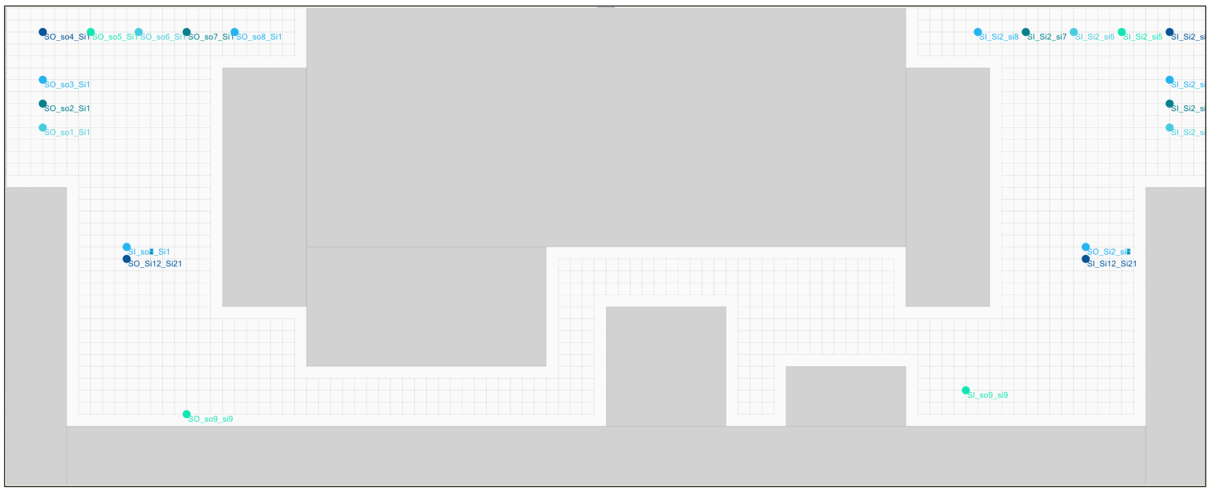


Figure 4.34: Layout and Commodity System for the second Use Case

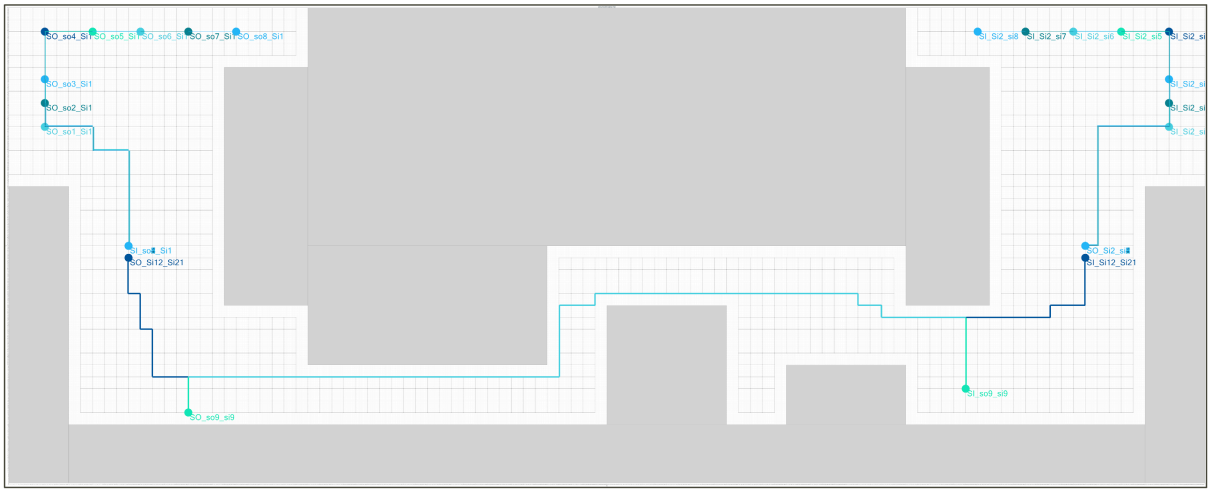


Figure 4.35: Computed Material flow for the second Use Case - Cost Minimization

Upon feeding this material flow graph to the conveyor placement model, the runtime increased slightly from 3 seconds to 4 seconds. This increase can likely be attributed to the additional constraints imposed by the new obstacles and the added commodity. The final conveyor system is presented in Figure 4.37.

Two use cases were developed to evaluate the performance and practical application of the proposed models in optimizing material flow and conveyor placement, focusing on minimizing curve usage. Each use case followed a, compared to the prior test runs much greater, layout of 1000m x 400m with a grid size of 10, with sources and sinks designed so that all commodities pass through a designated "tunnel". Intermediate sinks at the entrance and exit of this tunnel allowed for individual optimization of three distinct parts, facilitating complexity reduction while still enabling achievement of a global optimum. The first use case, with eight commodities and an intermediate combined source-sink relationship, resulted in an optimized material flow in 1 minute and 12 seconds using the cost minimization function. However, a further division into three submodels

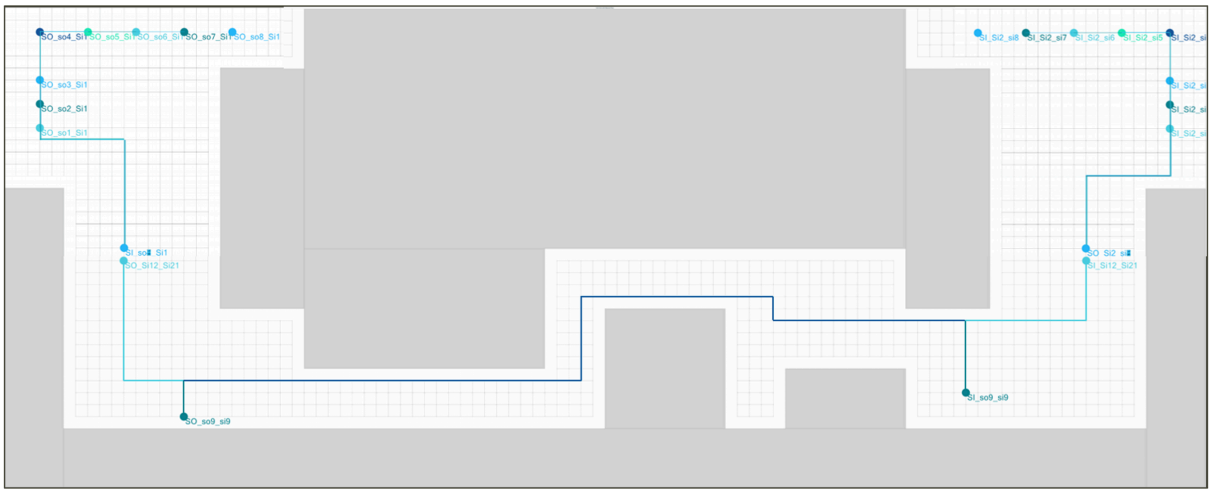


Figure 4.36: Computed Material flow for the second Use Case - Curve Minimization

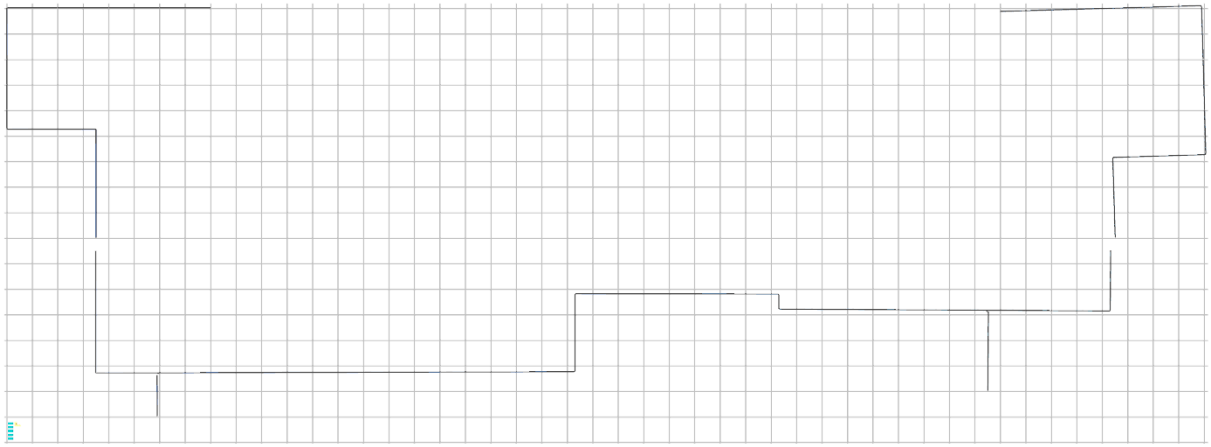


Figure 4.37: Computed Conveyor System - Second Use Case

and incorporation of a cost-curve-usage objective function improved the reduction of unnecessary curves. The optimal conveyor system was computed in 3 seconds, following a total runtime including all material flow runs, of 9 minutes 46 seconds. The second use case, which incorporated an additional commodity and several obstacles, demonstrated an increase in the total runtime to 10 minutes and 14 seconds due to added complexity, and the conveyor system computation increased to 4 seconds. Despite these changes, the adjustments to the objective function allowed for substantial improvements in curve usage.

Chapter 5

Conclusion and Future Prospects

This research project initially set out to bridge the gap between material flow optimization and the facility layout problem. To that end, a novel model was not only theoretically constructed but also practically implemented. It has successfully enhanced the material flow optimization problem by adding curve minimization to the objective function. This approach effectively transforms the abstract representation of material flow into a more tangible conveyor system, thus offering greater practical utility in real-world applications. Simultaneously, a conveyor placement model was developed, taking into account the complex geometry of real-world conveyors, including non-rectangular configurations like curves, merges, and diverts. The model strives to minimize the distance of the conveyor system from the optimal material flow, thereby forming a potent bridge between the material flow optimization and facility layout problems. Importantly, this model also accounts for obstacles, ensuring there is no intersection between conveyors and these obstructions, represented as combinations of rectangles.

The theoretical and practical performance of both models underwent evaluations, yielding valuable insights into their operational efficacy. Their practical applicability was further tested using two realistic scenarios. While the material flow model could identify feasible solutions promptly, it struggled to prove optimality when faced with larger problems. Furthermore, the inclusion of the curve usage minimization feature in the material flow model led to an increase in runtime but also significantly improved the practicality of the generated solutions. Additionally, employing a combined objective function - such as cost and curve minimization - resulted in solutions akin to those yielded by solely curve minimization, but with enhanced performance. The conveyor placement model demonstrated exceptional efficacy, finding the optimal or near-optimal solutions for the two use cases in less than five seconds. Contrary to the material flow model, the challenging part for the conveyor placement model is not proving of optimality of a feasible solution but finding one. As soon as any feasible solution is found, the model is able to identify the optimal solution almost immediately.

Looking ahead to future work, there are several promising avenues to explore. A key next step would involve enhancing the flexibility of the conveyor placement model. The introduction of additional conveyor types could offer the model more autonomy, potentially allowing it not only

to decide where but also which specific conveyor type to place, which could lead to more efficient and practical solutions. Moreover, the integration of the two optimization models into a single comprehensive model could be beneficial. Additionally, owing to the defined tolerance areas for the conveyors around their respective nodes, the model exhibits a tendency to position the conveyor system in a slightly rotated manner, diverging from perfect alignment with the x or y axes. This could pose issues in practical applications. The likelihood of such rotation decreases as the tolerance areas are defined smaller, however, excessively reducing these areas might render the model infeasible. Consequently, a more appropriate solution to mitigate this issue may lie in the introduction of an additional constraint, thereby preserving feasibility while ensuring optimal alignment.
Beyond this, it's worth considering the expansion of the problem from 2D to 3D, further enhancing its practicality. It might also be worth leveraging advanced computational methods like machine learning or artificial intelligence to boost the models' performance and help tackle large-scale problems.
Finally, continuous performance enhancement remains a perpetual goal with any computational model. By boosting the speed and efficiency of these models, their practical applicability will be significantly enhanced.

In sum, this research managed to further bridge the material flow optimization and the facility layout problem. It has resulted in the creation of a practical and efficient model capable of navigating complex geometric realities, not only in theory but also in practical implementation.

Chapter 6

Financial Projection and Analysis

The comprehensive budgetary breakdown of this Master's Thesis delineates tasks associated with the creation and critical appraisal of optimization models. Two core areas of focus within this assessment are the material flow and conveyor placement within industrial contexts. To ensure a realistic financial projection, the calculations have been predicated upon the prevailing market rates for a Simulation Engineer in Germany, a specialist skilled in modeling and simulation paradigms. The chosen Simulation Engineer, operating as a freelancer, has a competitive net hourly rate of 90 €. [Freelancermap 2023]

This compensation structure becomes particularly significant when extrapolated over a longer duration. The work model for the Simulation Engineer is configured for 7 hours each day, equating to a 35-hour workweek spread over 5 days.
A pivotal component to the functionality of the program is the commercial solver, Gurobi. For the development phase, I had the advantage of leveraging the student version, which was available without any charge. However, for commercial applications, the exact cost of Gurobi would hinge on a company-specific quote.

Given the theoretical nature of this thesis, it's essential to note that the only tangible costs incurred are those related to time. There are no additional expenses associated with materials, equipment, or external services, with the exception of potential costs for Gurobi in a commercial setting. The value of the time spent is, in turn, determined by the expertise and specialization of the professionals involved, in this case, the Simulation Engineer. Moreover, there is an implicit cost associated with the time spent by my supervisor at the company evaluating the work. While this has not been quantified and included in the present calculation, it does have intrinsic value in terms of knowledge transfer and feedback.

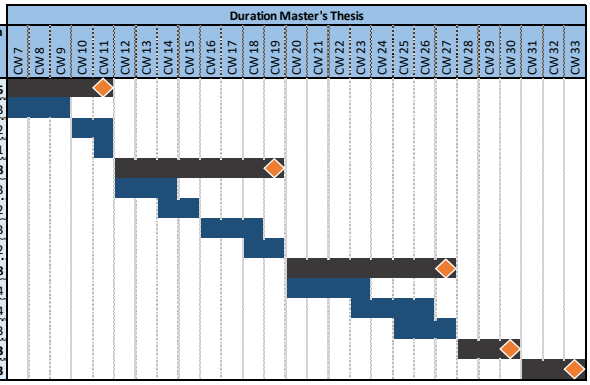
In financial terms, the cumulative cost for the engineer's services, given the specified hourly rate and the duration of the project, amasses to a substantial 85,050.00 €. However, to gain a complete understanding of the overall fiscal implications, one cannot overlook the tax component. Given a standard tax rate of 19% on professional services, there's an additional financial burden of 16,159.50 €. Consequently, the complete financial outlay, inclusive of tax, for this project ascends

to 101,209.50€. This figure embodies not just the technical expertise but also the intellectual capital invested in the project.

In the following, a detailed calculation, along with a Gantt diagram, will further illustrate the budgetary breakdown and project timeline.

Net Hourly Rate Simulation Engineer (*freelancer study*)	90 €
Working Hours per day	7
Working days per week	5
Working hours per week	35

id	Task Name	Start	End	Duration [week, 35h]
1	**Literature Research**	**CW 7**	**CW 11**	**5**
1.1	Literature Research - Optimization Problems	CW 7	CW 9	3
1.2	Literature Research - Suitable Solvers	CW 10	CW 11	2
1.3	Literature Research - Suitable Programming Language	CW 11	CW 11	1
2	**Material Flow Model**	**CW 12**	**CW 19**	**8**
2.1	Implementation of Material Flow Model	CW 12	CW 14	3
2.2	First Evaluation Runs of Material Flow Model	CW 14	CW 15	2
2.3	Theoretical Development of Curve Usage Minimization	CW 16	CW 18	3
2.4	Second Evaluation Runs of enhanced Material Flow Model	CW 18	CW 19	2
3	**Conveyor Placement Model**	**CW 20**	**CW 27**	**8**
3.1	Theoretical Development of Conveyor Placement Model	CW 20	CW 23	4
3.2	Implementation of Conveyor Placement Model	CW 23	CW 26	4
3.3	Evaluation Runs of Conveyor Placement Model	CW 25	CW 27	3
4	**Further Improvements of the Program**	**CW 28**	**CW 30**	**3**
5	**Final Testing and Evaluation of the developed Models**	**CW 31**	**CW 33**	**3**



id	Price Calculation Price Simulation Engineer
1	**15.750,00 €**
1.1	9.450,00 €
1.2	6.300,00 €
1.3	3.150,00 €
2	**25.200,00 €**
2.1	9.450,00 €
2.2	6.300,00 €
2.3	9.450,00 €
2.4	6.300,00 €
3	**25.200,00 €**
3.1	12.600,00 €
3.2	12.600,00 €
3.3	9.450,00 €
4	**9.450,00 €**
5	**9.450,00 €**
Subtotal	**85.050,00 €**
Tax [19%]	**16.159,50 €**
Balance	**101.209,50 €**

Bibliography

Allegri, T.M. (1984). *Material Handling: Principles and Practices*. New York: Van Nostrand (cit. on p. 12).

Alvarado-Iniesta, Alejandro et al. (2013). "Optimization of the material flow in a manufacturing plant by use of artificial bee colony algorithm". In: *Expert Systems with Applications* 40.12, pp. 4785–4790. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2013.02.029 (cit. on p. 9).

Armour, Garth C and Elwood S Buffa (1963). "A Heuristic Algorithm and Simulation Approach to Relative Location of Facilities". In: *Management Science* 9.2, pp. 294–309 (cit. on p. 10).

Arora, Sanjeev and Boaz Barak (2009). *Computational Complexity: A Modern Approach*. Cambridge University Press (cit. on pp. 6, 7).

Azevedo, Maria Manuela, José António Crispim, and Jorge Pinho de Sousa (2017). "A dynamic multi-objective approach for the reconfigurable multi-facility layout problem". In: *Journal of Manufacturing Systems* 42, pp. 140–152. ISSN: 0278-6125. DOI: https://doi.org/10.1016/j.jmsy.2016.12.008 (cit. on p. 12).

Balakrishnan, Jaydeep et al. (Feb. 2003). "A hybrid genetic algorithm for the dynamic plant layout problem". In: *International Journal of Production Economics* 86, pp. 107–120. DOI: 10.1016/S0925-5273(03)00027-6 (cit. on p. 10).

Battaïa, Olga and Alexandre Dolgui (2014). "A classification of line balancing problems and their solution approaches". In: *International Journal of Production Economics* 142.2, pp. 259–277 (cit. on p. 2).

Bozer, Yavuz A and Russell D Meller (1997). "A reexamination of the distance-based facility layout problem". In: *IIE transactions* 29.7, pp. 549–560 (cit. on p. 12).

Bullnheimer, Bernd, Richard F. Hartl, and Christine Strauss (1999). “An Improved Ant System Algorithm for the Vehicle Routing Problem”. In: *Annals of Operations Research* 89, pp. 319–328 (cit. on p. 9).

Charikar, Moses et al. (2019). “Multi-commodity Flow with In-Network Processing”. In: *Algorithmic Aspects of Cloud Computing*. Ed. by Yann Disser and Vassilios S. Verykios. Cham: Springer International Publishing, pp. 73–101. ISBN: 978-3-030-19759-9 (cit. on p. 9).

Cook, Stephen A. (1971). “The Complexity of Theorem-Proving Procedures”. In: *Proceedings of the third annual ACM symposium on Theory of computing*, pp. 151–158 (cit. on p. 6).

Dantzig, George B (1951). “Maximization of a linear function of variables subject to linear inequalities”. In: *Activity analysis of production and allocation* 13, pp. 339–347 (cit. on pp. 7, 15).

Drastich, Adam (2017). “Optimization of material flow by simulation methods”. In: *Acta logistica* 4.4, pp. 23–26 (cit. on p. 9).

Drira, Amine, Henri Pierreval, and Sonia Hajri-Gabouj (Dec. 2007). “Facility layout problems: A survey”. In: *Annual Reviews in Control* 31, pp. 255–267. DOI: 10.1016/j.arcontrol.2007.04.001 (cit. on pp. 10, 12).

Freelancermap (2023). *Freelancer-Studie*. Accessed: 17-08-2023 (cit. on p. 63).

Gao, Jiyao and Fengqi You (2018). “Dynamic Material Flow Analysis-Based Life Cycle Optimization Framework and Application to Sustainable Design of Shale Gas Energy Systems”. In: *ACS Sustainable Chemistry & Engineering* 6.9, pp. 11734–11752. DOI: 10.1021/acssuschemeng.8b01983. eprint: https://doi.org/10.1021/acssuschemeng.8b01983 (cit. on p. 10).

Garey, Michael R. and David S. Johnson (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman (cit. on p. 7).

— (1978). ““strong”np-completeness results: Motivation, examples, and implications”. In: *Journal of the ACM (JACM)* 25.3, pp. 499–508 (cit. on p. 7).

Gerdts, M. (2011). *Mathematische Optimierungsverfahren des Operations Research*. first. De Gruyter (cit. on p. 6).

Ghiani, Gianpaolo, Gilbert Laporte, and Roberto Musmanno (2013). *Introduction to Logistics Systems Management*. John Wiley & Sons, Incorporated (cit. on p. 2).

Goldberg, Andrew V, Éva Tardos, and Robert Tarjan (1989). *Network flow algorithm*. Tech. rep. Cornell University Operations Research and Industrial Engineering (cit. on p. 9).

Grimme, C. (2018). *Einführung in die Optimierung*. first. Springer Vieweg (cit. on p. 6).

Heragu, Sunderesh S. (2008). *Facilities Design*. CRC Press (cit. on pp. 2, 12).

Herrmann, J.W. et al. (1995). "Design of material flow networks in manufacturing facilities". In: *Journal of Manufacturing Systems* 14.4, pp. 277–289. ISSN: 0278-6125. DOI: https://doi.org/10.1016/0278-6125(95)98880-F (cit. on p. 7).

Ioannou, George (May 2007a). "An integrated model and a decomposition-based approach for concurrent layout and material handling system design". In: *Computers & Industrial Engineering* 52, pp. 459–485. DOI: 10.1016/j.cie.2007.02.003 (cit. on pp. 9, 13).

— (May 2007b). "An integrated model and a decomposition-based approach for concurrent layout and material handling system design". In: *Computers & Industrial Engineering* 52, pp. 459–485. DOI: 10.1016/j.cie.2007.02.003 (cit. on p. 12).

Kang, Jin-su, Hyeong-dong Kim, and Tai-yong Lee (2003). "Sharing benefits of Eco-Industrial Park by multiobjective material flow optimization". In: *Process Systems Engineering 2003, 8th International Symposium on Process Systems Engineering*. Ed. by Bingzhen Chen and Arthur W. Westerberg. Vol. 15. Computer Aided Chemical Engineering. Elsevier, pp. 499–504. DOI: https://doi.org/10.1016/S1570-7946(03)80594-1 (cit. on p. 10).

Karp, Richard M. (1972). "Reducibility Among Combinatorial Problems". In: pp. 85–103 (cit. on p. 7).

Kovács, György and Sebastian Kot (2017). "Facility layout redesign for efficiency improvement and cost reduction". In: *Journal of Applied Mathematics and Computational Mechanics* 16.1 (cit. on p. 1).

Krentel, Mark W (1986). "The complexity of optimization problems". In: *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pp. 69–76 (cit. on p. 5).

Kulturel-Konak, Sadan (2012). "A linear programming embedded probabilistic tabu search for the unequal-area facility layout problem with flexible bays". In: *European Journal of Operational Research* 223.3, pp. 614–625. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2012.07.019 (cit. on p. 12).

Kumar, Ravi, Surya Prakash Singh, and Kuldeep Lamba (2018). "Sustainable robust layout using Big Data approach: A key towards industry 4.0". In: *Journal of Cleaner Production* 204, pp. 643–659. ISSN: 0959-6526. DOI: https://doi.org/10.1016/j.jclepro.2018.08.327 (cit. on p. 12).

Lean Enterprise Research Centre (2004). *Cardiff Business School* (cit. on p. 1).

Meindl, Bernhard and Matthias Templ (Aug. 2013). "Analysis of Commercial and Free and Open Source Solvers for the Cell Suppression Problem". In: *Tansaction on Data Privacy* Volume 6, pp. 147–159 (cit. on p. 28).

Miettinen, Kaisa (2012). *Nonlinear Multiobjective Optimization*. Vol. 12. Springer Science & Business Media (cit. on p. 6).

Mir, M. and M.H. Imam (2001). "A hybrid optimization approach for layout design of unequal-area facilities". In: *Computers & Industrial Engineering* 39.1, pp. 49–63. ISSN: 0360-8352. DOI: https://doi.org/10.1016/S0360-8352(00)00065-6 (cit. on p. 10).

Moeller, Andreas et al. (2009). "Simulation and optimization of material and energy flow systems". In: *Proceedings of the 2009 Winter Simulation Conference (WSC)*, pp. 1444–1455. DOI: 10.1109/WSC.2009.5429292 (cit. on p. 9).

Monostori, L. et al. (2016). "Cyber-physical systems in manufacturing". In: *CIRP Annals* 65.2, pp. 621–641. ISSN: 0007-8506. DOI: https://doi.org/10.1016/j.cirp.2016.06.005 (cit. on p. 9).

Palomo-Romero, Juan M, Lorenzo Salas-Morera, and Laura Garca-Hernández (2017). "An island model genetic algorithm for unequal area facility layout problems". In: *Expert Systems with Applications* 68, pp. 151–162 (cit. on p. 12).

Pekarcikova, M et al. (2020). "Material flow optimization through e-kanban system simulation". In: *International Journal of Simulation Modelling* 19.2, pp. 243–254 (cit. on p. 9).

Peters, Brett and Taho Yang (Sept. 1997). "Integrated facility layout and material handling system design in semiconductor fabrication facilities". In: *Semiconductor Manufacturing, IEEE Transactions on* 10, pp. 360–369. DOI: 10.1109/66.618209 (cit. on p. 12).

Shchekutin, Nikita, Ludger Overmeyer, and Vyacheslav P. Shkodyrev (2020). "Layout Optimization for Cyber-Physical Material Flow Systems Using a Genetic Algorithm". In: *Cyber-Physical Systems and Control*. Ed. by Dmitry G. Arseniev et al. Cham: Springer International Publishing, pp. 27–39. ISBN: 978-3-030-34983-7 (cit. on p. 9).

Simic, D. et al. (2021). "Modelling material flow using the Milk run and Kanban systems in the automotive industry". In: *Expert Systems* 38.1. Accessed: 29-07-2023, pp. 1–15. DOI: 10.1111/exsy.12546 (cit. on p. 9).

Sulaiman, S. Sheik et al. (2021). "An evolutionary optimal green layout design for a production facility by simulated annealing algorithm". In: *Materials Today: Proceedings* 47. FRESM21, pp. 4423–4430. ISSN: 2214-7853. DOI: https://doi.org/10.1016/j.matpr.2021.05.256 (cit. on p. 12).

Sule, D.R. (1991). *Manufacturing Facilities: Location, Planning, and Design*. Boston, MA: PWS Kent (cit. on p. 1).

Tayal, Akash et al. (2020). "Efficiency analysis for stochastic dynamic facility layout problem using meta-heuristic, data envelopment analysis and machine learning". In: *Computational*

Intelligence 36.1, pp. 172–202. DOI: https://doi.org/10.1111/coin.12251. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/coin.12251 (cit. on p. 12).

Tompkins, J.A. and J.A. White (1984). *Facilities Planning*. 1st ed. New York: John Wiley & Sons (cit. on p. 1).

Tompkins, James A et al. (2010). *Facilities planning*. John Wiley & Sons (cit. on p. 12).

Wang, Rui et al. (2019). "A digital twin-based approach for designing and multi-objective optimization of hollow glass production line". In: *Enterprise Information Systems* 13.9, pp. 1328–1346 (cit. on p. 12).

Wolsey, Laurence A. (1998). *Integer Programming*. New York, NY: Wiley-Interscience (cit. on p. 9).

List of Figures

List of Tables