Kerstin Maier

Solving the Combined Cell Layout Problem with Exact Approaches

Master Thesis

submitted in fulfilment of the requirements for the degree of
Diplom-Ingenieurin

Programme: Master's programme Applied Informatics
Alpen-Adria-Universität Klagenfurt

Evaluator:
O.Univ.-Prof. Dipl.-Ing. Dr. Gerhard Friedrich
Alpen-Adria-Universität Klagenfurt
Institut für Angewandte Informatik

Klagenfurt, February 21, 2019

Affidavit

I hereby declare in lieu of an oath that

- the submitted academic thesis is entirely my own work and that no auxiliary materials have been used other than those indicated,
- I have fully disclosed all assistance received from third parties during the process of writing the thesis, including any significant advice from supervisors,
- any contents taken from the works of third parties or my own works that have been included either literally or in spirit have been appropriately marked and the respective source of the information has been clearly identified with precise bibliographical references (e.g. in footnotes),
- to date, I have not submitted this thesis to an examining authority either in Austria or abroad and that
- when passing on copies of the academic thesis (e.g. in bound, printed or digital form), I will ensure that each copy is fully consistent with the submitted digital version.

I understand that the digital version of the academic thesis submitted will be used for the purpose of conducting a plagiarism assessment.
I am aware that a declaration contrary to the facts will have legal consequences.

Kerstin Maier e.h. Klagenfurt, February 21, 2019

Abstract

In this thesis the Combined Cell Layout Problem (CCLP) is considered. The CCLP aims to minimize the material handling costs in a cellular manufacturing system with at least two cells where processing occurs, and in the presence of pieces that need to be processed in more than one cell. The alignment of the machines in each cell can follow a row or a circular layout. The CCLP was first introduced by Hungerländer and Anjos (2013) who suggested a Semidefinite Programming (SDP) approach for solving it.

We propose an Integer Linear Programming approach as well as an Answer Set Programming approach for solving the CCLP. Further, we create an extensive benchmark library to encourage further research in this field. In a computational study we demonstrate the efficiency of our two exact approaches for solving this problem and compare it to the SDP by Hungerländer and Anjos, which was the only available exact approach to date.

Acknowledgements

First, I would like to express my sincere gratitude to my supervisor O. Univ.-Prof. Dr. Gerhard Friedrich for his valuable support while writing this master thesis.

I also would like to thank Assoc. Prof. DDr. Philipp Hungerländer for introducing me to the topic of this thesis. He gave me the opportunity to join his research team more than three years ago, which enabled me to improve my research skills and to work with a highly motivated team on interesting topics in the field of routing and scheduling. I look forward to further years of common research with Mr. Hungerländer and the whole team. In addition, thanks to Prof. Dr. Miguel F. Anjos for his fruitful collaboration while working on the corresponding conference and journal papers.

Further, I would like to thank my family, especially my parents, for supporting me throughout my life. My university education would not have been possible without their help. Last but not least, lot of thanks to my friends for making my years of study an enjoyable journey.

Content

Chapter 1

Preface

In this thesis the Combined Cell Layout Problem CCLP is discussed. The CCLP aims to minimize the material handling costs in a cellular manufacturing system with at least two cells where processing occurs, and in the presence of pieces that need to be processed in more than one cell. The alignment of the machines in each cell can follow a row or a circular layout. The CCLP was first introduced by Hungerländer and Anjos [22] who suggested a Semidefinite Programming (SDP) approach for solving it. In this thesis we propose an Integer Linear Programming (ILP) approach as well as an Answer Set Programming (ASP) approach for solving the CCLP.

An extended abstract covering the key results from our ILP approach has been published in proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management 2018 [6]. Additionally, an extended version of this abstract will be submitted to a leading international journal in the near future. My contributions to these publications are the following: I modeled and implemented the ILP. The implementation was done in Java using Gurobi 8.0 as ILP solver. I created a new benchmark library with a great variety of different CCLP instances. Further, I ran experiments with the ILP on the generated benchmark instances and analyzed the produced results. Finally, I took part in the preparation of our publications.

The thesis is organized as follows. We introduce the CCLP and give an overview of related literature in Chapter 2. In Chapter 3 we give an overview of the problem setup and the input parameters and notations used for our exact approaches. In Chapter 4 we repeat the SDP approach suggested by Hungerländer and Anjos [22]. We

propose our ILP approach for solving the CCLP in Chapter 5. In Chapter 6 we present our two ASP formulations for the CCLP. We report the results of our computational experiments in Chapter 7. Finally, a conclusion can be found in Chapter 8 and an outlook for future research directions is given in Chapter 9.

Chapter 2

Introduction

In cellular manufacturing systems pieces with similar processing requirements are collected into piece families, and the machines required to process these pieces are clustered into machine cells. In an ideal world, cells are planed in such a way that the individual piece families are completely processed within only one machine cell, which results in pairwise independent cells without inter-cell material flow. However, in practice it may be uneconomical or impractical to consider only pairwise independent cells. Therefore, some pieces need to be processed in more than one cell.

Cellular manufacturing is a well-recognized paradigm for organizing factory operations. A real-world success story is described in [15], and a study of 46 plants involving a total of 126 cells is reported in [41]. The cell structure in manufacturing continues to be relevant in the current Industry 4.0 paradigm, see e.g. [36].

We focus on the problem of determining optimal layouts of the machines within each cell in the presence of pieces that need to be processed in more than one cell. Cell layout normally takes place after the determination of machine cells, see e.g., [12]. Until now, much research has been done on cell formation, however, the layout of machines within the cells has gained less attention, yet it is critical for maximizing the practical performance within the cells as well as for the production system as a whole.

In this thesis we propose an Integer Linear Programming (`ILP`) formulation as well as two Answer Set Programming formulations (`ASP`) that minimize the total of intra-cell and inter-cell material handling costs for a given machine-cell assignment and

with the machines within each cell aligned in a row or in a circle. We depict in Figure 2.1 a sketch of a cellular manufacturing system for further clarifying our considered problem. This problem is denoted as the Combined Cell Layout Problem (CCLP)

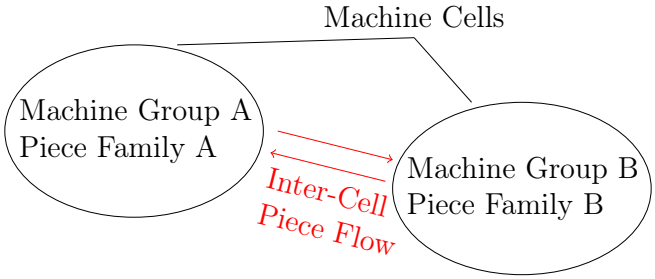

Figure 2.1: Illustration of a cellular manufacturing system where intra-cell and inter-cell material handling costs occour.

and was first suggested by Hungerländer and Anjos [22] who proposed a Semidefinite Programming (SDP) approach for solving the CCLP. As far as we know, this is the first work that uses an ILP or an ASP for solving the CCLP. In this work we consider two different cell layouts, namely single-row layout and circular layout. Both layout types have been studied comprehensively by the research community but except in [22] always independent of each other. Let us give a short review of results and applications known for these types of layouts.

Row Layout. The Single-Row Facility Layout Problem (SRFLP) tries to determine an optimal alignment of machines in a row such that the total weighted sum of the distances between the centroids of all machine pairs is minimal [35]. One of many practical applications can be found in the alignment of machines on a production line where the material flow is controlled by an automated guided vehicle moving in both directions along a straight line path [20]. Other applications for the SRFLP can be found when

- arranging airplanes to gates within an airport terminal [39],
- assigning rooms along a corridor in supermarkets, hospitals or offices [38],
- locating disk cylinders to files and aligning books on a shelf [35].

An illustration of the SRFLP can be found in Figure 2.2.

A SRFLP instance is made up of M one-dimensional machines, with fixed lengths $\ell_1, \ldots, \ell_M \in \mathbb{N}$ and pairwise connectivities $c_{ij},\ i, j \in \mathcal{M} = \{1, \ldots, M\}$. The corresponding optimization problem can be stated as

$$\min_{\pi \in \Pi} \sum_{\substack{i,j \in \mathcal{M} \\ i<j}} c_{ij} \zeta_{ij}^{\pi},$$

where Π is the set of all feasible layouts of the machines in $\mathcal{M}$ and ζ_{ij}^{π} is the distance between the centroids of machine i and machine j regarding a specific permutation $\pi \in \Pi$.

It is possible to compute tight global bounds and even optimal layouts for SRFLP instances of reasonable size. The strongest ILP formulation to date is an LP-based cutting plane algorithm that uses betweenness variables [3] and is able to find optimal layouts for instances with up to 35 machines. The strongest SDP approach that uses products of ordering variables [24] is able to solve instances with up to 42 machines to optimality and obtains tight global bounds for instances with up to 100 machines. Alternatively, various well-performing heuristics exist for the SRFLP; the best ones are suggested by Datta et al. [14], Kothari and Ghosh [27], and Samarghandi and Eshghi [37].

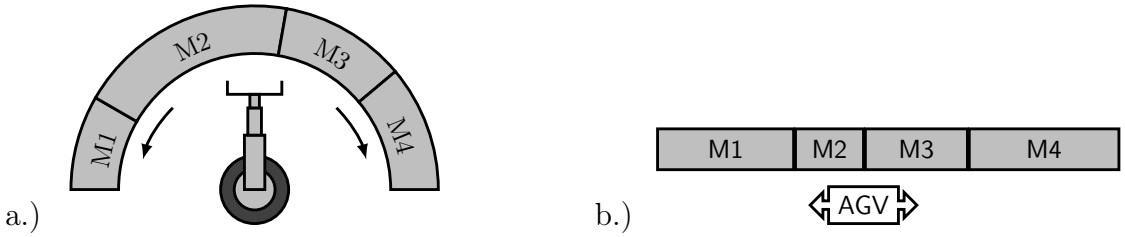

Figure 2.2: In a.) an industrial robot and in b.) an AGV is used to transport pieces from one machine to another.

The Multi-Row Facility Layout Problem (MRFLP), as displayed in Figure 2.3, is a generalization of the SRFLP and aims to determine an optimal alignment of machines to 2 or more parallel rows. We refer to [13] (see also [43]) and [4] for Mixed Integer Linear Programming (MILP) approaches, which find optimal layouts for instances with up to 12 machines allocated to 2 rows. An SDP approach for the MRFLP that obtains tight global bounds for instances with up to 12 departments considering 2 to

5 rows was proposed by Hungerländer and Anjos [23]. Recently, Fischer et al. [16] suggested a MILP formulation that is able to find optimal layouts for instances with up to 16 machines arranged within 2 rows.

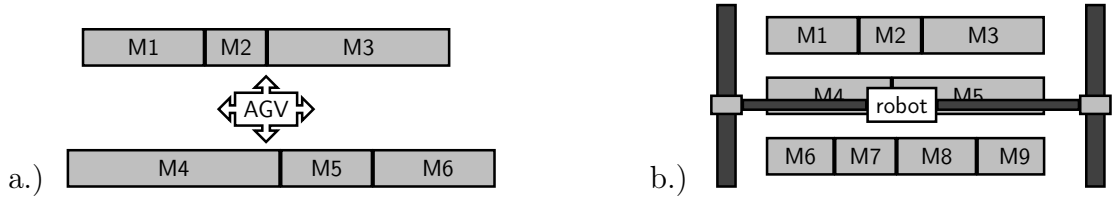

Figure 2.3: In a.) again an AGV moves pieces between the machines. For areas with limited space a gantry robot is required as displayed in b.).

Circular Layout. The Directed Circular Facility Layout Problem (DCFLP) (see also Figure 2.4) asks for an optimal alignment of machines along a circle such that the total weighted sum of the distances between the centroids of all machine pairs calculated in clockwise direction is minimal.

A DCFLP instance is made up of M one-dimensional machines placed next to each other along a circle with fixed lengths $\ell_1, \ldots, \ell_M \in \mathbb{N}$ and pairwise flows $f_{ij},\ i, j \in \mathcal{M},\ i \neq j$. The corresponding optimization problem can be stated as

$$\min_{\pi \in \Pi} \sum_{\substack{i,j \in \mathcal{M} \\ i \neq j}} f_{ij} z_{ij}^{\pi},$$

where Π is the set of all feasible layouts of the machines in $\mathcal{M}$ and z_{ij}^{π} gives the center-to-center distance between machine i and machine j in the circular layout π calculated in clockwise direction.

Circular layouts are more preferable in practical applications than row layouts due to higher material handling flexibility, lower initial investment costs and easier adaption to future introductions of additional pieces and process changes [1, 29].

Besides its practical advantages, the DCFLP is a fascinating problem from a theoretical perspective as it is related to various layout problems, which have been comprehensively studied by the research community. First, the DCFLP generalizes the Directed

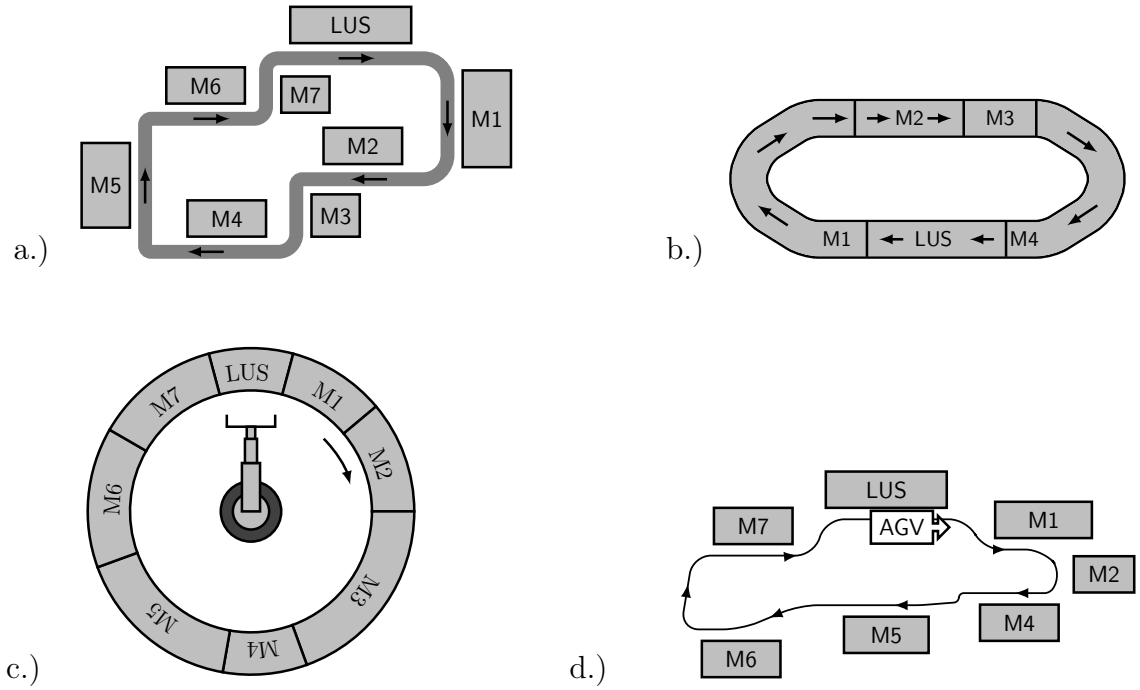

Figure 2.4: A conveyor belt moving in a closed-loop rail in one direction transports pieces between the machines in a.) and b.). In c.) an industrial robot moves in clockwise direction and a single loop AGVs carries pieces from one machine to another in d.).

Circular Arrangement Problem (DCAP) to arbitrary machine lengths. Liberatore [30] first proposed the DCAP and proved its NP-hardness. Second, the DCFLP is a generalization of the NP-hard [28] Unidirectional Cyclic Layout Problem (UCFLP) which also takes into account a circular layout and aims to determine an allocation of machines to fixed candidate locations such that the total handling cost is minimal. Note that the DCFLP is NP-hard due to generalizing NP-hard problems. The following layout problems are two well-discussed special cases of the UCFLP that are at the same time special cases of the DCFLP:

- In the Balanced Unidirectional Cyclic Layout Problem (BUCFLP) the piece flow is preserved at each machine, i.e., at each machine the total inflow equals the total outflow.
- In the Equidistant Unidirectional Cyclic Layout Problem (EUCFLP) the pairwise distances between all candidate locations are equal.

Obviously the EUCFLP and the DCAP are equivalent. In addition, the BUCFLP is equivalent to the EUCFLP [8]. We refer the reader to [2, 33, 34] for an overview of existing exact and heuristic approaches for the UCFLP and its special cases.

The DCFLP was introduced by Hungerländer [21], who suggested an SDP approach for solving it. Recently [25, 26, 32] showed that it is possible to formulate the DCFLP as Linear Ordering Problem and exploited this connection to obtain efficient heuristics and exact algorithms that determine optimal layouts for instances with up to 40 machines and tight global bounds for larger instances.

At the end of this chapter, we would like to clarify the structure and the differences of the SRFLP, the DCFLP and the CCLP with the help of a toy example: We consider 4 machines with lengths $\ell_{LUS} = 1,\ \ell_1 = \ell_{\mathrm{I}} = 1,\ \ell_2 = \ell_{\mathrm{II}} = 2,\ \ell_3 = \ell_{\mathrm{III}} = 3$. Additionally we have given the pairwise connectivities (for the SRFLP) and the pairwise flows (for the DCFLP) $c_{12} = c_{\mathrm{I\,II}} = 1,\ c_{13} = f_{\mathrm{III\,I}} = 2,\ c_{23} = f_{\mathrm{II\,III}} = 2$. When combining these two problems into one CCLP, we additionally consider the distance $u_{C_1C_2} = 3$ between the two cells c_1 (SRFLP cell) and c_2 (DCFLP cell) and an inter-cell flow $f_{1\mathrm{I}} = 1$. In Figure 2.5 the optimal layouts and the corresponding costs for the three layout problems are displayed.

In the following chapter we give an overview of our problem setup considered. Further, we introduce notations and input parameters that are used to describe our approaches throughout Chapters 4-6.

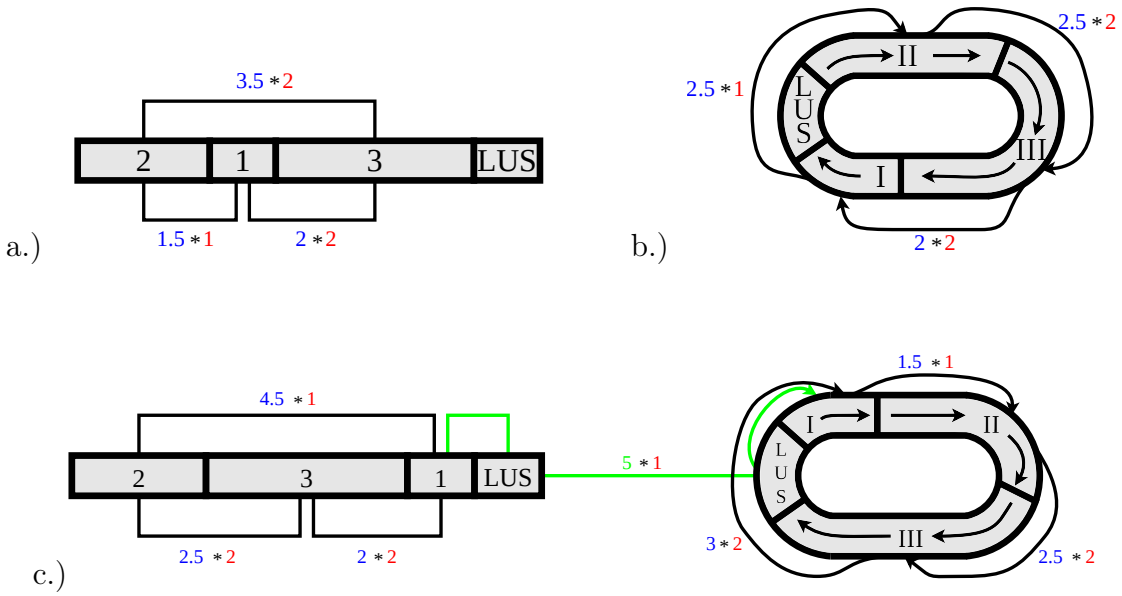

Figure 2.5: We have given the following data: $\ell_{LUS} = 1$, $\ell_1 = \ell_{\mathrm{I}} = 1$, $\ell_2 = \ell_{\mathrm{II}} = 2$, $\ell_3 = \ell_{\mathrm{III}} = 3$, $c_{12} = c_{\mathrm{I\,II}} = 1$, $c_{13} = f_{\mathrm{III\,I}} = 2$, $c_{23} = f_{\mathrm{II\,III}} = 2$, $f_{1\mathrm{I}} = 1$, $u_{C_1C_2} = 3$. In a.) we give the optimal SRFLP layout with corresponding costs of $1.5 \cdot 1 + 2 \cdot 2 + 3.5 \cdot 2 = 12.5$. In b.) we depict the optimal DCFLP layout with associated costs of $2.5 \cdot 1 + 2.5 \cdot 2 + 2 \cdot 2 = 11.5$. In c.) we display the optimal CCLP layout with associated costs of $4.5 \cdot 1 + 2 \cdot 2 + 2.5 \cdot 2 + 1.5 \cdot 1 + 2.5 \cdot 2 + 3 \cdot 2 + 5 \cdot 1 = 31$. LUS denotes the loading and unloading stations.

Chapter 3

Problem Setup

In a material handling system pieces move from one machine to another following a predefined sequence specified by the process plan. In most such systems, there exist loading and unloading stations (LUSs) that are used by all pieces for entering and exiting the system. In general LUSs do not perform any operation on these pieces. However, for an unbiased comparison in Chapter 7, the LUSs are allowed to process pieces in our setup. It is assumed that more than one piece is produced using the same layout but only one piece is produced at a time. Our goal is to determine the optimal alignment of the machines such that the total material flow is minimal.

We define the given piece types as $\mathcal{P} = \{1, \ldots, P\}$. A piece type $p \in \mathcal{P}$ is connected to a specific process plan S_p, which defines the sequence in which p is processed by the machines. The cost function for each piece is different and dependent on n_{ij}^p, which denotes the amount of moves piece type p performs from machine i to machine j per time period. The value of n_{ij}^p can be easily calculated from S_p. Furthermore, n_p specifies the amount of pieces of type p that are processed per time period. Each piece makes up a part of the total produced output and therefore represents a weight. Using this weights, the distinct cost functions can be combined into a function that considers the layout of all pieces simultaneously. Hence, the total amount of pieces of type $p \in \mathcal{P}$ that are moved from machine i to machine j per time period equals $f_{ij}^p = n_p n_{ij}^p,\ i, j \in \mathcal{M},\ i \neq j,\ p \in \mathcal{P}$, where $\mathcal{M} = \{1, \ldots, M\}$ defines all available machines. The total amount of pieces which are transported from machine i to machine j per time period is thus $f_{ij} = \sum_{p \in \mathcal{P}} f_{ij}^p,\ i, j \in \mathcal{M},\ i \neq j$.

Each machine has an integer length $\ell_i,\ i \in \mathcal{M}$ and is preassigned to one cell $c \in \mathcal{C} := \{1, \ldots, C\}$ by the function $z : \mathcal{M} \to \mathcal{C}$. Moreover, the function $t : \mathcal{C} \to \{(\texttt{SRFLP}), (\texttt{DCFLP})\}$ specifies the associated layout type of each cell. The LUS, where all pieces entering and exiting a cell, is defined by the function $s : \mathcal{C} \to \mathcal{M}$, where w.l.o.g. $s(c) \leq m,\ c \in \mathcal{C},\ \ i \in \mathcal{M},\ z(s(c)) = z(i)$. Finally, $u_{ij},\ i, j \in \mathcal{C},\ i < j$ denotes the integer distance for each pair of cells.

In the next chapters we describe our considered exact solution approaches. We start with the `SDP` approach by [22] in the following chapter. In Chapter 5 we introduce our `ILP` approach and in Chapter 6 we suggest two `ASP` formulations.

Chapter 4

Semidefinite Programming

In this chapter we present the Semiefinite Programming approach proposed by Hungerländer and Anjos [22]. To model the arrangement of the machines in multiple cells the most natural and sensible way seems to be the usage of ordering variables because the most competitive exact approaches for both the SRFLP and the DCFLP to date are based on semidefinite relaxations in ordering variables. Hence, bivalent variables

$$y_{ij} \in \{-1, 1\},\ i, j \in \mathcal{M},\ i < j,\ z(i) = z(j), \tag{1}$$

with the following interpretation are introduced:

$$y_{ij} = \begin{cases} 1, & \text{if machine } i \text{ is located left of machine } j, \\ -1, & \text{otherwise.} \end{cases}$$

Every feasible alignment of the machines must satisfy the following 3-cycle inequalities:

$$-1 \leq y_{ij} + y_{jk} - y_{ik} \leq 1, \quad i, j, k \in \mathcal{M},\ i < j < k,\ z(i) = z(j) = z(k), \tag{2}$$

It is common-knowledge that the inequalities (14) combined with the integrality of the y-variables are sufficient to describe feasible alignments, see e.g. [40, 42].

The center-to-center distances $d_{ij}^{(\texttt{SRFLP})}$ between machines within the same single-row

layout cell can be computed with the help of products of ordering variables [5]:

$$d_{ij}^{(\texttt{SRFLP})} = \frac{\ell_i + \ell_j}{2} + \sum_{\substack{k \in \mathcal{M} \\ k<i \\ z(k)=z(i)}} \ell_k \frac{1 - y_{ki}y_{kj}}{2} + \sum_{\substack{k \in \mathcal{M} \\ i<k<j \\ z(k)=z(i)}} \ell_k \frac{1 + y_{ik}y_{kj}}{2} + \sum_{\substack{k \in \mathcal{M} \\ k>j \\ z(k)=z(i)}} \ell_k \frac{1 - y_{ik}y_{jk}}{2},$$

where $i, j \in \mathcal{M},\ i < j,\ z(i) = z(j),\ t(z(i)) = (\texttt{SRFLP})$. One machine is fixed to be "first" in each cell (the LUS) and hence, every $y_{ij},\ j \in \mathcal{M},\ s(z(j)) = i,\ t(z(i)) = (\texttt{SRFLP})$ is set to one. Now the distance variables can be rewritten as follows:

$$d_{ij}^{(\texttt{SRFLP})} = \frac{\ell_i + \ell_j}{2} + \sum_{\substack{k \in \mathcal{M} \\ i<k<j \\ z(k)=z(j)}} \ell_k \frac{1 + y_{kj}}{2} + \sum_{\substack{k \in \mathcal{M} \\ k>j \\ z(k)=z(j)}} \ell_k \frac{1 - y_{jk}}{2},$$

$$j \in \mathcal{M},\ s(z(j)) = i,\ t(z(i)) = (\texttt{SRFLP}),$$

$$d_{ij}^{(\texttt{SRFLP})} = \frac{\ell_i + \ell_j}{2} + \sum_{\substack{k \in \mathcal{M} \\ s(z(i))<k<i \\ z(k)=z(i)}} \ell_k \frac{1 - y_{ki}y_{kj}}{2} + \sum_{\substack{k \in \mathcal{M} \\ i<k<j \\ z(k)=z(i)}} \ell_k \frac{1 + y_{ik}y_{kj}}{2} + \sum_{\substack{k \in \mathcal{M} \\ k>j \\ z(k)=z(i)}} \ell_k \frac{1 - y_{ik}y_{jk}}{2},$$

$$i, j \in \mathcal{M},\ s(z(i)) < i < j,\ z(i) = z(j),\ t(z(i)) = (\texttt{SRFLP}).$$

Hence, the objective function for machines located in the same cell $c \in \mathcal{C}$ with $t(c) = (\texttt{SRFLP})$ can be written as a linear-quadratic function f in ordering variables:

$$f_{(\texttt{SRFLP})} = K_c + \sum_{\substack{i,j \in \mathcal{M} \\ s(z(i))<i<j \\ z(i)=z(j)=c}} \frac{c_{ij}}{2} \left(- \sum_{\substack{k \in \mathcal{M} \\ s(z(i))<k<i \\ z(k)=z(i)}} \ell_k y_{ki} y_{kj} \right.$$

$$+ \sum_{\substack{k\in\mathcal{M}\\ i<k<j\\ z(k)=z(i)}} \ell_k y_{ik} y_{kj} - \sum_{\substack{k\in\mathcal{M}\\ k>j\\ z(k)=z(i)}} \ell_k y_{ik} y_{jk} \Bigg) .$$

where

$$K_c = \left(\sum_{\substack{i,j\in\mathcal{M},\ s(z(i))<i<j\\ z(i)=z(j)=c}} \frac{c_{ij}}{2} \right) \left(\sum_{\substack{k\in\mathcal{M},\ s(z(k))<k\\ z(k)=c}} \ell_k \right),\ c\in\mathcal{C},\ t(c)=(\texttt{SRFLP}).$$

For a circular layout cell, the distance variables $D_{ij}^{(\texttt{DCFLP})},\ i,j\in\mathcal{M},\ i<j$, represent the differences of the sums of the machine lengths placed in front of machine i and machine j respectively [21, 32].

$$D_{ij}^{(\texttt{DCFLP})} = \left(\frac{\ell_i}{2} + \sum_{\substack{k\in\mathcal{M},\ k<i\\ z(k)=z(i)}} \ell_k \frac{1+y_{ki}}{2} + \sum_{\substack{k\in\mathcal{M}\\ k>i\\ z(k)=z(i)}} \ell_k \frac{1-y_{ik}}{2} \right)$$
$$- \left(\frac{\ell_j}{2} + \sum_{\substack{k\in\mathcal{M}\\ k<j\\ z(k)=z(i)}} \ell_k \frac{1+y_{kj}}{2} + \sum_{\substack{k\in\mathcal{M}\\ k>j\\ z(k)=z(i)}} \ell_k \frac{1-y_{jk}}{2} \right),$$
$$i,j\in\mathcal{M},\ i<j,\ z(i)=z(j),\ t(z(i))=(\texttt{DCFLP}).$$

The $D_{ij}^{(\texttt{DCFLP})}$ are linear expressions in the ordering variables and can be simplified to

$$D_{ij}^{(\texttt{DCFLP})} = \frac{1}{2} \left(\sum_{\substack{k\in\mathcal{M},\ k<i\\ z(k)=z(i)}} \ell_k y_{ki} - \sum_{\substack{k\in\mathcal{M}\\ k>i\\ z(k)=z(i)}} \ell_k y_{ik} - \sum_{\substack{k\in\mathcal{M}\\ k<j\\ z(k)=z(i)}} \ell_k y_{kj} + \sum_{\substack{k\in\mathcal{M}\\ k>j\\ z(k)=z(i)}} \ell_k y_{jk} \right),$$
$$i,j\in\mathcal{M},\ i<j,\ z(i)=z(j),\ t(z(i))=(\texttt{DCFLP}).$$

Again we fix one machine (the LUS) to be "first" in the circular layout, hence we

set $y_{ij} = 1,\ j \in \mathcal{M},\ s(z(j)) = i,\ t(z(i)) = (\texttt{DCFLP})$. This results in the following adaption of the distance variables:

$$D_{ij}^{(\texttt{DCFLP})} = \frac{1}{2}\left(- \sum_{\substack{k\in\mathcal{M}\\ i<k<j\\ z(k)=z(j)=c}} \ell_k y_{kj} + \sum_{\substack{k\in\mathcal{M}\\ k>j\\ z(k)=z(j)=c}} \ell_k y_{jk} - L_c \right), \qquad L_c = \sum_{\substack{k\in\mathcal{M}\\ z(k)=c}} \ell_k,$$

$$j \in \mathcal{M},\ j \neq i,\ s(z(j)) = i,\ c \in \mathcal{C},\ t(c) = (\texttt{DCFLP}),$$

$$D_{ij}^{(\texttt{DCFLP})} = \frac{1}{2}\left(\sum_{\substack{k\in\mathcal{M}\\ s(z(i))<k<i\\ z(k)=z(i)}} \ell_k y_{ki} - \sum_{\substack{k\in\mathcal{M}\\ k>i\\ z(k)=z(i)}} \ell_k y_{ik} \right.$$

$$\left. - \sum_{\substack{k\in\mathcal{M}\\ s(z(i))<k<j\\ z(k)=z(i)}} \ell_k y_{kj} + \sum_{\substack{k\in\mathcal{M}\\ k>j\\ z(k)=z(i)}} \ell_k y_{jk} \right),$$

$$i, j \in \mathcal{M},\ s(z(i)) < i < j,\ z(i) = z(j),\ t(z(i)) = (\texttt{DCFLP}).$$

Now with the help of the distance variables $D_{ij}^{(\texttt{DCFLP})},\ i, j \in \mathcal{M},\ i < j$, it is possible to define the distances between machine i and machine j on the circle (calculated in clockwise direction), denoted by $d_{ij}^{(\texttt{DCFLP})},\ i, j \in \mathcal{M},\ i \neq j$, via the distance variables

$$d_{ij}^{(\texttt{DCFLP})} = -D_{ij}^{(\texttt{DCFLP})},\ d_{ji}^{(\texttt{DCFLP})} = L_c + D_{ij}^{(\texttt{DCFLP})}$$

$$j \in \mathcal{M},\ j \neq i,\ s(z(j)) = i,\ z(j) = c,\ t(c) = (\texttt{DCFLP}),$$

$$d_{ij}^{(\texttt{DCFLP})} = -D_{ij}^{(\texttt{DCFLP})} + \frac{1 - y_{ij}}{2} L_c,\ d_{ji}^{(\texttt{DCFLP})} = D_{ij}^{(\texttt{DCFLP})} + \frac{1 + y_{ij}}{2} L_c,$$

$$i, j \in \mathcal{M},\ s(z(i)) < i < j,\ z(i) = z(j) = c,\ t(c) = (\texttt{DCFLP}),$$

where L_c denotes the sum of the lengths of the machines in the cell c. Finally, the objective function for machines located in the same cell $c \in \mathcal{C}$ with $t(c) = (\texttt{DCFLP})$

can be written as a linear function g in ordering variables:

$$g_{(\texttt{DCFLP})} = K_c + \sum_{\substack{i,j\in\mathcal{M}\\ s(z(i))<i<j\\ z(i)=z(j)=c}} (f_{ij}-f_{ji})\left(D_{ij}^{(\texttt{DCFLP})} - \frac{L_c y_{ij}}{2}\right). \tag{3}$$

where

$$K_c := \frac{L_c}{2} \sum_{\substack{i,j\in\mathcal{M}\\ s(z(i))<i<j\\ z(i)=z(j)=c}} (f_{ij}+f_{ji}), \qquad c\in\mathcal{C},\ t(c)=(\texttt{DCFLP}).$$

The distances of machines $i,j\in\mathcal{M}$, located in different cells $z(i)=c_1,\ z(j)=c_2,\ c_1\neq c_2\in\mathcal{C},\ s(z(i))\neq i,\ s(z(j)\neq j$, are given by

$$d_{ij}^{(\texttt{IC})} = u_{c_1c_2} + \begin{cases} d_{s(c_1)i}^{(\texttt{SRFLP})} + d_{s(c_2)j}^{(\texttt{SRFLP})}, & t(c_1)=t(c_2)=(\texttt{SRFLP}),\\ d_{is(c_1)}^{(\texttt{DCFLP})} + d_{s(c_2)j}^{(\texttt{DCFLP})}, & t(c_1)=t(c_2)=(\texttt{DCFLP}),\\ d_{s(c_1)i}^{(\texttt{SRFLP})} + d_{s(c_2)j}^{(\texttt{DCFLP})}, & t(c_1)=(\texttt{SRFLP}),\ t(c_2)=(\texttt{DCFLP}),\\ d_{is(c_1)}^{(\texttt{DCFLP})} + d_{s(c_2)j}^{(\texttt{SRFLP})}, & t(c_1)=(\texttt{DCFLP}),\ t(c_2)=(\texttt{SRFLP}). \end{cases} \tag{4}$$

As we assume that the LUSs are allowed to process pieces, we additionally need the following distances of machines $i,j\in\mathcal{M}$, located in different cells $z(i)=c_1,\ z(j)=c_2,\ c_1\neq c_2\in\mathcal{C}$:

$$d_{ij}^{(\texttt{IC})} = u_{c_1c_2} + \begin{cases} 0, & s(z(i))=i,\ s(z(j)=j,\\ d_{s(c_1)i}^{(\texttt{SRFLP})}, & t(c_1)=(\texttt{SRFLP}),\ s(z(i))\neq i,\ s(z(j)=j,\\ d_{is(c_1)}^{(\texttt{DCFLP})}, & t(c_1)=(\texttt{DCFLP}),\ s(z(i))\neq i,\ s(z(j)=j,\\ d_{s(c_2)j}^{(\texttt{DCFLP})}, & t(c_2)=(\texttt{DCFLP}),\ s(z(i))=i,\ s(z(j)\neq j,\\ d_{s(c_2)j}^{(\texttt{SRFLP})}, & t(c_2)=(\texttt{SRFLP}),\ s(z(i))=i,\ s(z(j)\neq j. \end{cases} \tag{5}$$

Hence, in summary the objective function for the inter-cell flow (IC) can also be formulated as a linear function h in ordering variables:

$$h_{(\texttt{IC})} = K_{c_1,c_2} + \sum_{\substack{i,j\in\mathcal{M}\\ i\neq j\\ c_1=z(i)\neq z(j)=c_2}} f_{ij}d_{ij}^{(\texttt{IC})}, \tag{6}$$

where

$$d_{ij}^{(\texttt{IC})} = \frac{1}{2}\left(\sum_{\substack{k\in\mathcal{M}\\ s(c_1)<k<i\\ z(k)=c_1}} \ell_k y_{ki} - \sum_{\substack{k\in\mathcal{M}\\ k>i\\ z(k)=c_1}} \ell_k y_{ik}\right) + \frac{1}{2}\left(\sum_{\substack{k\in\mathcal{M}\\ s(c_2)<k<j\\ z(k)=c_2}} \ell_k y_{kj} - \sum_{\substack{k\in\mathcal{M}\\ k>j\\ z(k)=c_2}} \ell_k y_{jk}\right),$$

$$i,j\in\mathcal{M}, i\neq j,\ c_1,c_2\in\mathcal{C},\ c_1\neq c_2,\ t(c_1)=(\texttt{SRFLP}),$$

$$d_{ij}^{(\texttt{IC})} = -\frac{1}{2}\left(\sum_{\substack{k\in\mathcal{M}\\ s(c_1)<k<i\\ z(k)=c_1}} \ell_k y_{ki} + \sum_{\substack{k\in\mathcal{M}\\ k>i\\ z(k)=c_1}} \ell_k y_{ik}\right) + \frac{1}{2}\left(\sum_{\substack{k\in\mathcal{M}\\ s(c_2)<k<j\\ z(k)=c_2}} \ell_k y_{kj} - \sum_{\substack{k\in\mathcal{M}\\ k>j\\ z(k)=c_2}} \ell_k y_{jk}\right),$$

$$i,j\in\mathcal{M}, i\neq j,\ c_1,c_2\in\mathcal{C},\ c_1\neq c_2,\ t(c_1)=(\texttt{DCFLP}),$$

and

$$K_{c_1,c_2} := \sum_{\substack{i,j\in\mathcal{M}\\ i\neq j\\ c_1=z(i)\neq z(j)=c_2}} f_{ij}\left(u_{c_1c_2} + \frac{L_{c_1}+L_{c_2}}{2}\right).$$

Let us collect all ordering variables introduced in a vector y of length $t := \sum_{c\in\mathcal{C}} \binom{|\mathcal{M}_c|}{2}$, where $\mathcal{M}_c,\ c\in\mathcal{C}$, gives the set of all machines assigned to cell c. Finally, we define

$$K = \sum_{c\in\mathcal{C}} K_c + \sum_{\substack{c_1,c_2\in\mathcal{C}\\ c_1\neq c_2}} K_{c_1,c_2}.$$

In summary we have deduced the following quadratic programming formulation in ordering variables of the CCLP.

Theorem 4.1 The problem

$$K + \min_{y\in\{-1,1\}^t} f_{(\texttt{SRFLP})} + g_{(\texttt{DCFLP})} + h_{(\texttt{IC})}$$

subject to (1) and (2) is equivalent to the CCLP.

Proof. The integrality condition (1) of the y variables together with Inequalities (2) induce a feasible layout in all cells, and the formulation of the objective function guar-

anty that the center-to-center distances between all pairs of machines are calculated correctly. □

Next matrix-based relaxations are used to get tight lower bounds for the CCLP. Related relaxations have also been successfully applied to combinatorial optimization problems that arise in the area of graph drawing, see [9, 10, 11] for details. Hungerländer and Anjos [22] elaborate on the SDP-based approaches by [24] for the SRFLP and [21] for the DCFLP and show how to combine and generalize them for finally applying them to the CCLP (defined and modeled above). First, the intractable quadratic programming formulation from Theorem 4.1 is rewritten in matrix notation and then standard techniques are used for relaxing it. Hence, an basic semidefinite relaxation is obtained that can be tightened by adding several classes of valid constraints. At the end an appropriate combination of optimization methods is described, which is taken to produce tight lower bounds and feasible layouts from the semidefinite relaxation.

Matrix Notation and Two Reformulations. When looking at matrix-based formulations, the object of interest is the multi-level linear-quadratic ordering polytope

$$\mathcal{P}_{MQO} := \text{conv}\left\{ Z : y \in \{-1,1\}, y \text{ satisfies (2)} \right\}.$$

where $Z := \begin{pmatrix} 1 \\ y \end{pmatrix} \begin{pmatrix} 1 \\ y \end{pmatrix}^\top = \begin{pmatrix} 1 & y \\ y^\top & Y \end{pmatrix}$ with $Y = yy^\top$ contains all linear and quadratic terms in ordering variables. Next an appropriate cost matrix C is defined from

$$\langle C, Z \rangle \stackrel{\triangle}{=} f_{(\texttt{SRFLP})} + g_{(\texttt{DCFLP})} + h_{(\texttt{IC})},$$

to ensure that the distances between all machines are computed correctly.

To take a closer look at the structure of C let us recall our toy example from the introduction with the input data: $\ell_{LUS} = 1,\ \ell_1 = \ell_{\text{I}} = 1,\ \ell_2 = \ell_{\text{II}} = 2,\ \ell_3 = \ell_{\text{III}} = 3,\ c_{12} = c_{\text{I II}} = 1,\ c_{13} = f_{\text{III I}} = 2,\ c_{23} = f_{\text{II III}} = 2,\ f_{1\text{I}} = 1,\ u_{C_1C_2} = 3$. The matrix C is indexed row- and column-wise by the vector $[1, y]$, i.e., in our example $y = [1, y_{12}, y_{13}, y_{23}, y_{\text{I II}}, y_{\text{I III}}, y_{\text{II III}}]^\top$. This means that e.g. the entry in the third row

and fifth column of C corresponds to the costs assigned to the term $y_{23} \cdot y_{\mathrm{I\,II}}$. Hence the entry in the first row and column of C denotes the constant costs K. For our toy example from the introduction the optimal layout encoded in ordering variables reads $y = [1, 1, 1, -1, 1, 1, 1]^\top$ and the matrices $C = (c_{ij})$ and $Z = (z_{ij})$ look as follows:

$$C = \begin{pmatrix} 42.5 & -0.5 & -0.75 & 0 & -3 & 2 & -2.75 \\ -0.5 & 0 & -0.5 & 1 & 0 & 0 & 0 \\ -0.75 & -0.5 & 0 & -0.75 & 0 & 0 & 0 \\ 0 & 1 & -0.75 & 0 & 0 & 0 & 0 \\ -3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ -2.75 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$Z = \begin{pmatrix} 1 & 1 & 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 1 & 1 \\ -1 & -1 & -1 & 1 & -1 & -1 & -1 \\ 1 & 1 & 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 1 & 1 \end{pmatrix}.$$

Let us explain the deduction of the above matrix C in more detail:

- The constant term is computed as $K = K_{C_1} + K_{C_2} + K_{C_1,C_2} = \frac{5}{2} \cdot 6 + \frac{7}{2} \cdot 6 + 1 \cdot (3 + \frac{7}{2}) = 42.5$.
- The other entries are deduced from
  - $f_{(\mathtt{SRFLP})} = -\frac{1}{2} \cdot 3y_{13}y_{23} + \frac{1}{2} \cdot 2y_{12}y_{23} - \frac{1}{2} \cdot y_{12}y_{13}$,
  - $g_{(\mathtt{DCFLP})} = 1\left(y_{\mathrm{I\,II}} + \frac{3}{2}y_{\mathrm{I\,III}} + \frac{1}{2}y_{\mathrm{I\,II}} - \frac{3}{2}y_{\mathrm{II\,III}} - \frac{7}{2}y_{\mathrm{I\,II}}\right)$
    $- 2\left(y_{\mathrm{I\,II}} + \frac{3}{2}y_{\mathrm{I\,III}} - \frac{1}{2}y_{\mathrm{I\,III}} - y_{\mathrm{II\,III}} - \frac{7}{2}y_{\mathrm{I\,III}}\right)$
    $+ 2\left(-\frac{1}{2}y_{\mathrm{I\,II}} + \frac{3}{2}y_{\mathrm{II\,III}} - \frac{1}{2}y_{\mathrm{I\,III}} - y_{\mathrm{II\,III}} - \frac{7}{2}y_{\mathrm{II\,III}}\right)$,
  - $h_{(\mathtt{IC})} = \frac{1}{2}\left(-2y_{12} - 3y_{13}\right) + \frac{1}{2}\left(-2y_{\mathrm{I\,II}} - 3y_{\mathrm{I\,III}}\right)$

  that yields all together

$$f_{(\mathtt{SRFLP})} + g_{(\mathtt{DCFLP})} + h_{(\mathtt{IC})} = -0.5y_{12}y_{13} + 2y_{12}y_{23} - 1.5y_{13}y_{23} - y_{12}$$

$$- 1.5y_{13} - 6y_{\mathrm{I\,II}} + 4y_{\mathrm{I\,III}} - 5.5y_{\mathrm{II\,III}}.$$

Evaluating the objective function gives $\langle C, Z\rangle = \sum_{i,j\in[7]} c_{ij}z_{ij} = 31$, the cost of the optimal layout. Finally, let us draft the general structure of the cost matrix C for the case of one cell with n_1 departments arranged in a single-row layout and one cell with n_2 departments arranged in a circular layout: C has 6 blocks (with the given dimensions) that are determined by different parts of the total cost function

$$C = \begin{pmatrix} (K)_{[1,1]} & \left(h_{(\mathtt{IC})}\right)_{\left[1,\binom{n_1}{2}\right]} & \left(g_{(\mathtt{DCFLP})} + h_{(\mathtt{IC})}\right)_{\left[1,\binom{n_2}{2}\right]} \\ \left(h_{(\mathtt{IC})}\right)_{\left[\binom{n_1}{2},1\right]} & \left(f_{(\mathtt{SRFLP})}\right)_{\left[\binom{n_1}{2},\binom{n_1}{2}\right]} & (0)_{\left[\binom{n_1}{2},\binom{n_2}{2}\right]} \\ \left(g_{(\mathtt{DCFLP})} + h_{(\mathtt{IC})}\right)_{\left[\binom{n_2}{2},1\right]} & (0)_{\left[\binom{n_2}{2},\binom{n_1}{2}\right]} & (0)_{\left[\binom{n_2}{2},\binom{n_2}{2}\right]} \end{pmatrix}.$$

Now we are able to reformulate Theorem 4.1 in matrix notation:

Theorem 4.2 Minimizing $\langle C, Z\rangle$ over $y \in \{-1, 1\}^t$ fulfilling (1) and (2) solves the `CCLP`.

Proof. Conditions (1) together with Inequalities (2) are sufficient to generate feasible single-row and circular layouts and the representation of C guaranty that the distances between machines are calculated exactly. □

Next the nonconvex equation $Y - yy^\top = 0$ can be relaxed to the positive semidefinite constraint

$$Y - yy^\top \succcurlyeq 0.$$

The Schur complement lemma [7, Appendix A.5.5] implies $Y - yy^\top \succcurlyeq 0 \Leftrightarrow Z \succcurlyeq 0$. Furthermore, in Y the main diagonal entries represent squared $\{-1, 1\}$ variables, which results in $\mathrm{diag}(Y) = e$, the vector of all ones. We therefore conclude that $\mathcal{P}_{MQO}$ is contained in the elliptope

$$\mathcal{E} := \{\ Z :\ \mathrm{diag}(Z) = e,\ Z \succcurlyeq 0\ \}.$$

As a next step, the constraints need to be reformulated as quadratic conditions in y to express them in terms of Y. This can be ensured for the 3-cycle inequalities $|y_{ij} + y_{jk} - y_{ik}| = 1$ by squaring both sides. Additionally using $y_{ij}^2 = 1$ obtains

$$y_{ij,jk} - y_{ij,ik} - y_{ik,jk} = -1, \quad i, j, k \in \mathcal{M},\ i < j < k,\ z(i) = z(j) = z(k). \qquad (7)$$

In [11] it is shown that these 3-cycle equations describe the smallest linear subspace that contains $\mathcal{P}_{MQO}$. Now we can formulate the CCLP as a semidefinite optimization problem in binary variables.

Theorem 4.3 The problem

$$\min \left\{ K + \langle C, Z \rangle : Z \text{ satisfies (7)}, \ Z \in \mathcal{E}, \ y \in \{-1,1\}^t \right\}$$

is equivalent to the CCLP.

Proof. Since $y_{ij}^2 = 1, \ i,j \in \mathcal{M}, \ z(i) = z(j)$ the $\operatorname{diag}(Y - yy^\top) = 0$, which together with $Y - yy^\top \succcurlyeq 0$ reveals that $Y = yy^\top$ is integral. The 3-cycle equations (7) ensure that (2) holds. Finally, the objective value displays the total costs of the layout described by y due to the representation of the cost matrix C and the constant K. □

Next we apply standard techniques to the SDP formulation of the CCLP proposed in the above theorem to construct SDP relaxations over $\mathcal{P}_{MQO}$.

Establishing and Solving Semidefinite Relaxations

Removing the integrality condition on the first row and column of Z results in the basic semidefinite relaxation of the CCLP:

$$\min \left\{ K + \langle C_Z, Z \rangle : Z \text{ satisfies (7)}, \ Z \in \mathcal{E} \right\}. \qquad (\text{SDP}_{\texttt{basic}})$$

This relaxation can be tightened by a variety of ways. In this work we consider two of them, which have been sufficiently applied for the SRFLP and the DCFLP.
First it can be noticed that Z is produced as the outer product of the vector $\begin{pmatrix} 1 & y \end{pmatrix}^\top$, which has just $\{-1,1\}$ entries in the non-relaxed SDP formulation. Therefore any feasible CCLP solution is contained in the metric polytope $\mathcal{M}$, which is formulated through $4\binom{\Delta}{3} \approx \frac{1}{12}n^6$ facets.

$$\mathcal{M} = \left\{ Z : \begin{pmatrix} -1 & -1 & -1 \\ -1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \end{pmatrix} \begin{pmatrix} z_{ij} \\ z_{jk} \\ z_{ik} \end{pmatrix} \leq e, \ \ 1 \leq i < j < k \leq \Delta \right\}. \qquad (8)$$

Another way for strengthening $\texttt{SDP}_{\texttt{basic}}$ was proposed by Lovász and Schrijver [31]. They propose to multiply the 3-cycle inequalities

$$1 - y_{ij} - y_{jk} + y_{ik} \geq 0, \qquad 1 + y_{ij} + y_{jk} - y_{ik} \geq 0. \tag{9}$$

by the nonnegative expressions

$$1 - y_{lo} \geq 0, \qquad 1 + y_{lo} \geq 0, \qquad l, o \in \mathcal{M},\ l < o, z(l) = z(o). \tag{10}$$

This results in the following inequalities:

$$\begin{aligned} -1 - y_{lo} \leq y_{ij} + y_{jk} - y_{ik} + y_{ij,lo} + y_{jk,lo} - y_{ik,lo} \leq 1 + y_{lo}, \\ -1 + y_{lo} \leq y_{ij} + y_{jk} - y_{ik} - y_{ij,lo} - y_{jk,lo} + y_{ik,lo} \leq 1 - y_{lo}, \end{aligned} \tag{11}$$

for $i, j, k, l, o \in \mathcal{M},\ i < j < k,\ l < o,\ z(i) = z(j) = z(k),\ z(l) = z(o)$. The corresponding polytope $\mathcal{LS}$ can be defined as follows:

$$\mathcal{LS} := \{\ Z : Z \text{ satisfies (11) }\}. \tag{12}$$

Summarizing yields the succeeding tractable semidefinite relaxation of $\mathcal{P}_{MQO}$:

$$\min \{K + \langle C_Z, Z\rangle : Z \text{ satisfies (7)},\ Z \in (\mathcal{E} \cap \mathcal{M} \cap \mathcal{LS})\ \}\,. \tag{$\texttt{SDP}_{\texttt{strong}}$}$$

Each variable in Z with a positive cost coefficient occurs in one of the 3-cycle equalities (7) and hence, is strongly constrained in $\texttt{SDP}_{\texttt{strong}}$. That's the reason for obtaining strong bounds, even for large instances, in the root node relaxation with the help of several linear and semidefinite relaxations for the SRFLP or the DCFLP, which are based on ordering or betweenness variables. Notice that $\texttt{SDP}_{\texttt{strong}}$ can be easily applied to general Quadratic Ordering Problems by adapting its objective function, see e.g. combinatorial optimization problems that arise in the field of graph drawing [9, 10, 11]. For the experiments in Chapter 7 we will apply $\texttt{SDP}_{\texttt{strong}}$ to several CCLP instances with up to 240 machines.

Let us now discuss how $\texttt{SDP}_{\texttt{strong}}$ can be solved efficiently and how these solutions can be exploited. The proposed semidefinite approach aims to solve the semidefinite relaxation $\texttt{SDP}_{\texttt{strong}}$ with the help of a bundle method [17] in combination with interior point methods. The resulting fractional solutions constitute lower bounds.

This fractional solutions can be exploited to get upper bounds, i.e., integer solutions that reflect feasible machine layouts, with the help of a rounding strategy. Finally, a feasible solution, in conjunction with a proof how far this solution may be from the true optimum is obtained. Next, these two steps will be discussed in more detail.

Solving $\texttt{SDP}_{\texttt{strong}}$ It is obvious that the large number of constraints in $\texttt{SDP}_{\texttt{strong}}$ are not tractable in practice. Already considering $O(n^3)$ constraints is not reasonable for instances with at least 20 machines. Hence, Hungerländer and Anjos [22] modified an approach originally suggested in [17]. At the beginning, it is only explicitly ensured that Z is obtained in the elliptope $\mathcal{E}$, which is reached with the help of standard interior-point methods, see e.g. [19]. The remaining constraints are managed through Lagrangian duality in the objective function f. This results in a non-smooth objective function f. A bundle method, which iteratively conducts function evaluations of f and performs improvement steps, is used to obtain an approximate minimizer of f that reflects a lower bound to the optimal $\texttt{SDP}_{\texttt{strong}}$ solution.

Obtaining Feasible Layouts Applying the hyperplane rounding algorithm of Goemans-Williamson [18] to the approximate solution of $\texttt{SDP}_{\texttt{strong}}$ yields feasible layouts. The signs of some of the entries in the resulting vector are switched to make it feasible with respect to the 3-cycle inequalities (2). Computational results showed that the repair strategy is not as critical as some might suppose, e.g. the performance of the SDP-based rounding heuristic for the SRFLP is comparable to the best heuristics [14, 27, 37] and much better as the SDP-based heuristic suggested by [5].

In more detail the heuristic looks as follows: Initially, a vector y' is considered, which displays a random, but feasible ordering within all cells. The algorithm stops after 1000 executions[1] of step 2; y' is then the heuristic solution. If it was not possible to close the duality gap, further bundle iterations follow and then the heuristic is retried (keeping the last vector y').

1. Let Z'' be the current primal fractional solution of $\texttt{SDP}_{\texttt{strong}}$ obtained by the bundle method. Compute the convex combination $R := \lambda(y'y'^{\top}) + (1-\lambda)Z''$, using some random $\lambda \in [0.3, 0.7]$. Compute the Cholesky decomposition $DD^{\top}$ of R.

[1]Before its 501st execution, step 1 is performed again. As step 1 is very expensive, it is not executed too often.

2. Apply the hyperplane rounding by [18] to D and obtain a $-1/+1$ vector $\overline{y}$.

3. Compute the induced layout costs $\langle C, \overline{y}\overline{y}^\top \rangle$. If $\langle C, \overline{y}\overline{y}^\top \rangle \geq \langle C, y'y'^\top \rangle$: goto step 2.

4. If all 3-cycle inequalities are satisfied by $\overline{y}$: set $y' := \overline{y}$ and goto 2. Else: adapt $\overline{y}$ by switching the sign of one of three variables in every unsatisfied inequality and goto step 3.

Chapter 5

Integer Linear Programming

In this chapter we formulate the CCLP as an ILP. First, we need $O(n^2)$ binary ordering variables

$$x_{ij} \in \{0,1\},\ i,j \in \mathcal{M},\ i<j,\ z(i)=z(j), \tag{13}$$

with the following interpretation:

$$x_{ij} = \begin{cases} 1, & \text{if machine } i \text{ is located left of machine } j, \\ 0, & \text{otherwise.} \end{cases}$$

As described in Chapter 4, every feasible alignment of the machines must satisfy the 3-cycle inequalities. Rewriting the 3-cycle inequalities (2) to $\{0,1\}$ ordering variables results in:

$$0 \leq x_{ij} + x_{jk} - x_{ik} \leq 1, \quad i,j,k \in \mathcal{M},\ i<j<k,\ z(i)=z(j)=z(k). \tag{14}$$

Anjos et al. [5] showed that the distances $d_{ij}^{(\mathtt{SRFLP})}$ between the centroids of all pairs of machines for the single-row layout can be computed using products of $\{-1,1\}$ ordering variables. Fixing one machine i (the LUS) to be "first" in any cell with $x_{ij}=1,\ j \in \mathcal{M},\ i \neq j,\ s(z(j))=i,\ t(z(i))=$ (SRFLP) and using $\{0,1\}$ instead of

$\{-1, 1\}$ variables yields:

$$d_{ij}^{(\texttt{SRFLP})} = \frac{\ell_i + \ell_j}{2} + \sum_{\substack{k \in \mathcal{M} \\ i<k<j \\ z(k)=z(j)}} \ell_k x_{kj} + \sum_{\substack{k \in \mathcal{M} \\ k>j \\ z(k)=z(j)}} \ell_k (1 - x_{jk}),$$

$$k \in \mathcal{M},\ s(z(j)) = i,\ t(z(i)) = (\texttt{SRFLP}),$$

$$D_{ij}^{(\texttt{SRFLP})} = \sum_{\substack{k \in \mathcal{M} \\ s(z(i))<k<j \\ z(k)=z(i)}} \ell_k (1 - x_{ki}) x_{kj}$$

$$+ \sum_{\substack{k \in \mathcal{M} \\ i<k<j \\ z(k)=z(i)}} \ell_k x_{ik} x_{kj} + \sum_{\substack{k \in \mathcal{M} \\ k>i \\ z(k)=z(i)}} \ell_k x_{ik} (1 - x_{jk}),$$

$$i, j \in \mathcal{M},\ s(z(i)) < i < j,\ z(i) = z(j),$$

$$t(z(i)) = (\texttt{SRFLP}),$$

$$D_{ij}^{(\texttt{SRFLP})} = \sum_{\substack{k \in \mathcal{M} \\ j<k<i \\ z(k)=z(i)}} \ell_k (1 - x_{ki})(1 - x_{jk})$$

$$+ \sum_{\substack{k \in \mathcal{M} \\ s(z(i))<k<j \\ z(k)=z(i)}} \ell_k (1 - x_{ki}) x_{kj} + \sum_{\substack{k \in \mathcal{M} \\ k>i \\ z(k)=z(i)}} \ell_k x_{ik} (1 - x_{jk}),$$

$$i, j \in \mathcal{M},\ s(z(i)) < j < i,\ z(i) = z(j),$$

$$t(z(i)) = (\texttt{SRFLP}),$$

$$d_{ij}^{(\texttt{SRFLP})} = \frac{\ell_i + \ell_j}{2} + D_{ij}^{(\texttt{SRFLP})} + D_{ji}^{(\texttt{SRFLP})},$$

$$i, j \in \mathcal{M},\ s(z(i)) < i < j,\ z(i) = z(j),$$

$$t(z(i)) = (\texttt{SRFLP}).$$

Note that for all $i, j \in \mathcal{M}$, $i < j$, either D_{ij} or D_{ji} is set to zero and hence, d_{ij} correctly calculates the distance between machine i and machine j. The objective function for machines located in the same cell $c \in \mathcal{C}$ with $t(c) = (\texttt{SRFLP})$ can be written as a linear-quadratic function in ordering variables. For modeling the SRFLP

as an ILP we apply a standard linearization and introduce new variables

$$\begin{aligned} &y_{ikj} \in \{0,1\},\ i,j,k \in \mathcal{M},\ i \neq k \neq j, \\ &z(i) = z(k) = z(j),\ t(z(i)) = (\texttt{SRFLP}), \end{aligned} \tag{15}$$

for all occuring products of ordering variables:

$$\begin{aligned} y_{ikj} &= x_{ik}x_{kj}, \quad i,j,k \in \mathcal{M},\ i < k < j, \\ y_{ikj} &= (1 - x_{ki})x_{kj}, \quad i,j,k \in \mathcal{M},\ k < i < j, \\ y_{ikj} &= x_{ik}(1 - x_{jk}), \quad i,j,k \in \mathcal{M},\ i < j < k, \\ y_{ikj} &= (1 - x_{ki})(1 - x_{jk}), \quad i,j,k \in \mathcal{M},\ j < k < i, \\ y_{ikj} &= (1 - x_{ki})x_{kj}, \quad i,j,k \in \mathcal{M},\ k < j < i, \\ y_{ikj} &= x_{ik}(1 - x_{jk}), \quad i,j,k \in \mathcal{M},\ j < i < k. \end{aligned} \tag{16}$$

The following standard constraints are used to relate the ordering variables and their products:

$$\begin{aligned} y_{ikj} &\leq x_{ik}, \quad i,j,k \in \mathcal{M},\ i < k, \\ y_{ikj} &\leq 1 - x_{ki}, \quad i,j,k \in \mathcal{M},\ i > k, \\ y_{ikj} &\leq 1 - x_{jk}, \quad i,j,k \in \mathcal{M},\ j < k, \\ y_{ikj} &\leq x_{kj}, \quad i,j,k \in \mathcal{M},\ k > j, \\ y_{ikj} &\geq x_{ik} + x_{kj} - 1, \quad i,j,k \in \mathcal{M},\ i < k < j, \\ y_{ikj} &\geq -x_{ki} + x_{kj}, \quad i,j,k \in \mathcal{M},\ i > k,\ j > k, \\ y_{ikj} &\geq x_{ik} - x_{jk}, \quad i,j,k \in \mathcal{M},\ i < k,\ j < k, \\ y_{ikj} &\geq 1 - x_{ki} - x_{jk}, \quad i,j,k \in \mathcal{M},\ j < k < i. \end{aligned} \tag{17}$$

Thus we obtain the linearized objective function

$$f_{(\texttt{SRFLP})} = \sum_{\substack{i,j \in \mathcal{M} \\ s(z(i)) \leq i < j \\ z(i) = z(j)}} c_{ij} d_{ij}^{(\texttt{SRFLP})}.$$

Similar to Chapter 4, the distance variables $D_{ij}^{(\texttt{DCFLP})},\ i,j \in \mathcal{M},\ i < j$, for machines

within the same circular layout cell represent the differences of the sums of the machine lengths placed in front of machine i and machine j respectively [25, 32]. Again we fix the LUS machine to be "first" in the circular layout, and we set $x_{ij} = 1,\ j \in \mathcal{M},\ s(z(j)) = i,\ t(z(i)) = (\texttt{DCFLP})$. The distance variables can now be expressed as

$$D_{ij}^{(\texttt{DCFLP})} = \frac{\ell_i - \ell_j}{2} - \sum_{\substack{k \in \mathcal{M} \\ i \le k < j \\ z(k) = z(j)}} \ell_k x_{kj} - \sum_{\substack{k \in \mathcal{M} \\ k > j \\ z(k) = z(j)}} \ell_k (1 - x_{jk}),$$

$$j \in \mathcal{M},\ s(z(j)) = i,\ t(z(i)) = (\texttt{DCFLP}),$$

$$D_{ij}^{(\texttt{DCFLP})} = \left(\frac{\ell_i}{2} + \sum_{\substack{k \in \mathcal{M} \\ s(z(i)) < k < i \\ z(k) = z(i)}} \ell_k x_{ki} + \sum_{\substack{k \in \mathcal{M} \\ k > i \\ z(k) = z(i)}} \ell_k (1 - x_{ik}) \right)$$

$$- \left(\frac{\ell_j}{2} + \sum_{\substack{k \in \mathcal{M} \\ s(z(i)) < k < j \\ z(k) = z(j)}} \ell_k x_{kj} + \sum_{\substack{k \in \mathcal{M} \\ k > j \\ z(k) = z(j)}} \ell_k (1 - x_{jk}) \right),$$

$$i, j \in \mathcal{M},\ s(z(i)) < i < j,\ z(i) = z(j),$$

$$t(z(i)) = (\texttt{DCFLP}).$$

The distances between machine i and machine j along the circle, denoted by $d_{ij}^{(\texttt{DCFLP})},\ i, j \in \mathcal{M},\ i \neq j$, can be calculated via the distance variables $D_{ij}^{(\texttt{DCFLP})}$ as follows:

$$d_{ij}^{(\texttt{DCFLP})} = -D_{ij}^{(\texttt{DCFLP})},\ d_{ji}^{(\texttt{DCFLP})} = L_c + D_{ij}^{(\texttt{DCFLP})},\ j \in \mathcal{M},$$

$$j \neq i,\ s(z(j)) = i,\ z(j) = c,\ t(s(z(j)) = (\texttt{DCFLP}),$$

$$d_{ij}^{(\texttt{DCFLP})} = (1 - x_{ij}) L_c - D_{ij}^{(\texttt{DCFLP})},\ d_{ji}^{(\texttt{DCFLP})} = x_{ij} L_c + D_{ij}^{(\texttt{DCFLP})},$$

$$i, j \in \mathcal{M},\ s(z(i)) < i < k,\ z(i) = z(j) = c,\ t(c) = (\texttt{DCFLP}),$$

where

$$L_c = \sum_{\substack{k \in \mathcal{M} \\ z(k) = c}} \ell_k,\ c \in \mathcal{C},$$

defines the sum of the lengths of all machines. Hence, the objective function can be written as a linear function of ordering variables:

$$g_{(\texttt{DCFLP})} = \sum_{\substack{i,j \in \mathcal{M} \\ i \neq j \\ z(i)=z(j)}} f_{ij} d_{ij}^{(\texttt{DCFLP})}.$$

The distances of machines $i, j \in \mathcal{M}$, located in different cells $z(i) = c_1,\ z(j) = c_2,\ c_1 \neq c_2 \in \mathcal{C}$ can be computed analogous to Chapter 4 (see Equalities (4) and (5)). Hence, the objective function for the inter-cell (IC) flow can also be formulated as a linear function h of ordering variables:

$$h_{(\texttt{IC})} = \sum_{\substack{i,j \in \mathcal{M} \\ i \neq j \\ z(i)\neq z(j)}} f_{ij} d_{ij}^{(\texttt{IC})}.$$

Summarizing, we have deduced the following ILP formulation of the CCLP.

Theorem 5.1 The problem

$$\min\ f_{(\texttt{SRFLP})} + g_{(\texttt{DCFLP})} + h_{(\texttt{IC})}$$

subject to (13), (14), (15), (16), (17) is equivalent to the CCLP.

Proof. The integrality condition (13) of the x variables together with Inequalities (14) induce a feasible layout in all cells, and the formulation of the objective function guaranty that the center-to-center distances between all pairs of machines are calculated correctly. □

Chapter 6

Answer Set Programming

In this chapter we suggest two Answer Set Programming formulations for solving the CCLP. First, we introduce the necessary input parameters for both formulations. We are given the following constants:

$$\begin{aligned}
&\#const\, n = |\mathcal{M}|.\\
&\#const\, m = |\mathcal{C}|.\\
&\#const\, p = \frac{n}{m}.\\
&\#const\, s = |\{c \in \mathcal{C} \,|\, t(c) = (\texttt{SRFLP})\}|.
\end{aligned}$$

We define n as the amount of machines, m as the amount of cells, p as the amount of machines per cell and s as the amount of single row cells. In addition we need the following two sets of facts:

$$machine(i, \ell_i, z(i)). \qquad i \in \mathcal{M}, \tag{18}$$

$$weight(i, j, c_{ij}). \qquad i < j \in \mathcal{M},\ z(i) = z(j),\ t(z(i) = (\texttt{SRFLP}). \tag{19}$$

$$weight(i, j, f_{ij}). \qquad i \neq j \in \mathcal{M},\ z(i) = z(j),\ t(z(i) = (\texttt{DCFLP}). \tag{20}$$

$$weight(i, j, f_{ij}). \qquad i \neq j \in \mathcal{M},\ z(i) \neq z(j). \tag{21}$$

Facts (18) add all machines $\mathcal{M}$ with its associated length and cell to the ASP model. The flows or connectivities of each pair of machines are set by the facts (19) - (21).

ASP I: The first formulation tries to guess for each pair of machines $i \neq j \in \mathcal{M}$, if machine i is in front of machine j or otherwise. We define the guessing part as follows:

$$\{before(X, Y, C) : machine(X, \_, C), machine(Y, \_, C)\} = (p-1) * p/2 \\ :- C = 1..m. \tag{22}$$

$$\{before(((C-1) * p) + 1, Y, C) : machine(Y, \_, C)\} = p - 1 :- C = 1..m. \tag{23}$$

Due to fixing the LUS to be first in each machine cell, it is possible to consider also the arrangement within DCFLP cells as linear. Hence, exactly $(p-1)\frac{p}{2}$ *before(_,_,_)* atoms must be considered, which is ensured by (22). (23) guaranty that each LUS is before each other machine in the same cell.

$$before(X, Z, C) :- before(X, Y, C), before(Y, Z, C). \tag{24}$$

$$between(X, Z, Y, C) :- before(X, Z, C), before(Z, Y, C), X < Y, C = 1..s. \tag{25}$$

$$between(X, Z, Y, C) :- before(Z, X, C), before(Y, Z, C), X < Y, C = 1..s. \tag{26}$$

$$:- before(X, Y, C), before(Y, X, C). \tag{27}$$

Rule (24) ensures transitivity of the machine arrangement. With the help of Rules (25) and (26) we derive if a machine is located between two other machines. Rule (27) guaranty that for each pair of machines $i, j \in \mathcal{M}$ not both atoms ($before(i, j, z(i))$ and $before(i, j, z(i))$) are added to the answer set. The following rule derives for each pair of machines $i < j \in \mathcal{M},\ z(i) = z(j),\ t(i) =$ (SRFLP), the sum of the lengths of the machines that are placed between i and j within the same single row layout cell:

$$distance(X, Y, S) :- S = \\ \#sum\{W, Z : between(X, Z, Y, C), machine(Z, W, C)\}, \\ X = ((C-1) * p) + 1..C * p, \\ Y = ((C-1) * p) + 1..C * p, X < Y, C = 1..s. \tag{28}$$

With the help of Rules (29) - (33) we get the for each pair of machines $i \neq j \in \mathcal{M},\ z(i) = z(j),\ t(i) =$ (DCFLP), the sum of the lengths of the machines that are

placed between i and j considering the circular layout in clockwise direction:

$$totalLength(C,S) :- S = \#sum\{L, X : machine(X,L,C)\}, C = s+1..m. \quad (29)$$

$$distance(((C-1)*p)+1, Y, S) :- S = \#sum\{W, X : before(X,Y,C), machine(X,W,C), X! = ((C-1)*p)+1\}, Y = ((C-1)*p)+1..C*p, Y! = ((C-1)*p)+1, C = s+1..m. \quad (30)$$

$$distance(Y, ((C-1)*p)+1, T-S-W1-W2) :- totalLength(C,T), distance(((C-1)*p)+1, Y, S), machine(((C-1)*p)+1, W1, C), machine(Y, W2, C), Y = 2..n, C = s+1..m. \quad (31)$$

$$distance(X, Y, T-S-W1) :- distance(((C-1)*p)+1, X, S), distance(((C-1)*p)+1, Y, T), before(X,Y,C), machine(X,W1,C), machine(Y,W2,C), X = 2..n, Y = 2..n, C = s+1..m. \quad (32)$$

$$distance(X, Y, T-D-W1-W2) :- totalLength(C,T), distance(Y, X, D), not\ before(X,Y,C), machine(X,W1,C), machine(Y,W2,C), X! = Y, X = 2..n, Y = 2..n, C = s+1..m. \quad (33)$$

Finally, the Inter-Cell Distances between two machines $i \neq j \in \mathcal{M},\ z(i) \neq z(j)$, are calculated as follows:

$$distance(X, Y, S1+S2) :- distance(((C1-1)*p)+1, X, S1), distance(((C2-1)*p)+1, Y, S2), machine(X, \_, C1), machine(Y, \_, C2), C1! = C2, C1 <= s. \quad (34)$$

$$distance(X, Y, S1+S2) :- distance(X, ((C1-1)*p)+1, S1), distance(((C2-1)*p)+1, Y, S2), machine(X, \_, C1),$$

$$machine(Y, \_, C2), C1! = C2, C1 > s. \tag{35}$$

$$distance(((C1-1)*p)+1, Y, S) :- distance(((C2-1)*p)+1, Y, S),$$
$$machine(\_, \_, C1), machine(Y, \_, C2), C1! = C2. \tag{36}$$

$$distance(X, ((C2-1)*p)+1, S) :- distance(((C1-1)*p)+1, X, S),$$
$$machine(X, \_, C1), machine(\_, \_, C2), C1 <= s, C1! = C2. \tag{37}$$

$$distance(X, ((C2-1)*p)+1, S) :- distance(X, ((C1-1)*p)+1, S),$$
$$machine(X, \_, C1), machine(\_, \_, C2), C1 > s, C1! = C2. \tag{38}$$

The objective function minimizes over the total sum of all weighted distances:

$$\#minimize\{S*W, X, Y : distance(X, Y, S), weight(X, Y, W)\}. \tag{39}$$

ASP II: The second formulation aims to guess the location of each machine $i \in \mathcal{M}$. We formulate the guessing part as follows:

$$\{location(X, L, C) : machine(X, \_, C)\} = 1 :- C = 1..m, L = 1..p. \tag{40}$$

$$\{location(((C-1)*p)+1, 1, C) : machine(Y, \_, C)\} = 1 :- C = 1..m. \tag{41}$$

With the help of (40) we ensure that each machine is allocated to exactly one location and due to (41) it is guarantied that each LUS is positioned first in its associated cell.

$$:- location(X, L, C), location(Y, L, C), X! = Y. \tag{42}$$

$$:- location(X, L, C), location(X, M, C), L! = M. \tag{43}$$

Rules (42) and (43) ensure that two machines $i \neq j \in \mathcal{M},\ z(i) = z(j)$, within the same cell are not assigned to the same position in the machine arrangement. Similarly to ASP I, the remaining rules define for each pair of machines $i < j \in \mathcal{M},\ z(i) = z(j)$, the sum of the lengths of the machines that are placed between i and j.

$$distance(X, Y, S) :- S = \#sum\{W, Z : location(Z, LZ, C),$$
$$machine(Z, W, C), LZ < LY, LZ > LX\}, location(X, LX, C),$$
$$location(Y, LY, C), LX < LY, X < Y, C = 1..s. \tag{44}$$

$$distance(Y, X, S) :- S = \#sum\{W, Z : location(Z, LZ, C),$$
$$machine(Z, W, C), LZ < LY, LZ > LX\}, location(X, LX, C),$$
$$location(Y, LY, C), LX < LY, X > Y, C = 1..s. \quad (45)$$

$$distance(X, Y, S) :- S = \#sum\{W, Z : location(Z, LZ, C),$$
$$machine(Z, W, C), LZ < LY, LZ > LX\}, location(X, LX, C),$$
$$location(Y, LY, C), LX < LY, C = s + 1..m. \quad (46)$$

$$totalLength(C, S) :- S = \#sum\{L, X : machine(X, L, C)\}, C = s + 1..m. \quad (47)$$

$$distance(Y, X, S - D - W1 - W2) :- totalLength(C, S),$$
$$distance(X, Y, D), machine(X, W1, C), machine(Y, W2, C),$$
$$location(X, LX, C), location(Y, LY, C), LX < LY, C = s + 1..m. \quad (48)$$

Rules (44) and (45) replace Rule (28) for machines within the same single row layout cell and Rules (46) - (48) replace Rules (29) - (33) for machines within the same circular layout cell. The inter-cell distances and the objective function is equivalent to those in ASP I. Hence, we add (34) - (39) to our ASP II as well.

To get the correct cost of the optimal CCLP layout as described in the previous chapters we need to add the following constant to the objective value returned by our ASP approach:

$$\sum_{\substack{i,j\in\mathcal{M}\\ i<j\\ z(i)=z(j)\\ t(i)=(\texttt{SRFLP})}} \frac{\ell_i+\ell_j}{2}c_{ij} + \sum_{\substack{i,j\in\mathcal{M}\\ i\neq j\\ z(i)=z(j)\\ t(i)=(\texttt{DCFLP})}} \frac{\ell_i+\ell_j}{2}f_{ij} + \sum_{\substack{i,j\in\mathcal{M}\\ i\neq j,\ z(i)\neq z(j)\\ s(z(i))\neq i\\ s(z(j))\neq j}} \frac{\ell_i+\ell_j+\ell_{s(z(i))}+\ell_{s(z(j))}}{2}f_{ij}$$
$$+ \sum_{\substack{i,j\in\mathcal{M}\\ i\neq j,\ z(i)\neq z(j)\\ s(z(i))=i\\ s(z(j))\neq j}} \frac{\ell_i+\ell_j+\ell_{s(z(j))}}{2}f_{ij} + \sum_{\substack{i,j\in\mathcal{M}\\ i\neq j,\ z(i)\neq z(j)\\ s(z(i))\neq i\\ s(z(j))=j}} \frac{\ell_i+\ell_j+\ell_{s(z(i))}}{2}f_{ij} + \sum_{\substack{i,j\in\mathcal{M}\\ i\neq j,\ z(i)\neq z(j)\\ s(z(i))=i\\ s(z(j))=j}} \frac{\ell_i+\ell_j}{2}f_{ij}$$

We compare the efficiency of the SDP by Hungerländer and Anjos [22], our ILP and our two ASP formulations in the next chapter.

Chapter 7

Computational Experiments

In this chapter we state the results of our computational experiments with the SDP, ILP and ASP approaches proposed in the previous chapters. All benchmark instances considered are available to download from https://tinyurl.com/cclp-instances. All computations were performed on a 2.8 GHz Intel Core i7 with 16 GB RAM, running macOS. We implemented the semidefinite relaxation $\texttt{SDP}_{\texttt{strong}}$ in Matlab 7.7. We used Gurobi 8.0 as ILP solver and Clingo 5.2.2 as ASP solver. Each solver was started in single thread mode.

For generating the instances we set the amount of pieces P to 50. The amount of machines is set to $M \in \{10, 20, 30, 40, 50, 60, 70, 160, 180, 200, 220, 240\}$. The process plan S_p for each piece type $p \in \mathcal{P}$ was created as follows:

- First, the amount of machines to be visited is determined by sampling a random number $r_p \in \mathbb{N}$ between 2 and M from a uniform distribution.
- Second, r_p further numbers, which represent the process sequence of type p, were sampled from $U(1, M)$. Note that it is possible that a machine occurs more than once in the sequence but not consecutively.
- Third, $n_p \in \mathbb{N}$, the total amount of pieces from type p, was sampled from $U(1, 50)$.

The machine lengths $\ell_i \in \mathbb{N},\ i \in \mathcal{M}$ were sampled from $U(1, 10)$. Each machine is allocated to one of the cells according to their predefined index, i.e., for an instance with 50 machines and 10 cells the first 10 machines are assigned to the first cell, the

next 10 machines are assigned to the second cell and so on. Hence, we simply took one large instance and split it up into the according amount of cells. In this work we only solve instances where all cells have an identical amount of machines. However, our approaches are able to deal with an arbitrary amount of machines per cell. In each cell the LUS is assigned to be the machine with the lowest index. To facilitate an preferably even-handed unbiased comparison for the different numbers of cells the LUS are also allowed to process pieces, i.e., the flow matrix is the same, independent of the amount of cells considered. Finally, the integer distances between the cells were sampled from $U(20, 40)$.

We indicate the amount of cells and the corresponding layout types in each table using the notation (a, b, c), where a denotes the total amount of cells, b the amount of single-row layouts, and c the amount of circular layouts (hence, $a = b + c$).

Tables 7.1 - 7.4 summarize the results of all considered approaches for instances consisting of 10 to 40 machines that are allocated to 2 cells. While the `SDP` and the `ILP` approaches solve all instances considered to optimality, both `ASP` formulations are only able to solve instances with 10 machines to optimality. Our `ILP` approach generates all optimal solutions faster than the `SDP` approach. There is no clear winner between our two `ASP` formulations. The solving time for optimal layouts for instances with 10 machines is better for the first one. In addition, it generates more better layouts for instances with 20 machines. Especially, for 20 machines considering two circular layout cells the first formulation performs much better. However, `ASP` I is not able to produce any feasible layout within the given time limit for one instance with 30 machines and 7 instances with 40 machines. In addition, each layout obtained by `ASP` I for the remaining instances with 40 machines is not as good as the one returned by `ASP` II.

In Tables 7.5 - 7.10 we state the results of our `ILP` and `ASP` approaches for instances with 20 to 40 machines that are assigned to 5 cells. We solve all instances considered to global optimality within 14 millisecond with the help of our `ILP`. Both `ASP` formulations obtain optimal layouts only for instances with 20 machines. The first formulation is faster than the second one on all those instances. For the remaining instances with 30 and 40 machines it depends on the instance if the obtained layout by the first or the second formulation is better.

We represent the results of our ILP and ASP approaches for instances with 30 and 40 machines that are assigned to 10 cells in Tables 7.11 - 7.13 . Our ILP approach solves all instances considered to global optimality within one millisecond. The first ASP formulation produces optimal values for all instances considered, but is not able to prove the optimality for instances with 40 machines within the given time limit. The second ASP formulation solves instances with 30 machines to global optimality and instances with 40 machines close to optimality within the given time limit.

We omit to state the SDP results for the instances discussed in Tables 7.5 - 7.13. Note that the SDP is able to solve all those instances to global optimality. However, with an increased solving time compared to the ILP.

Finally, we summarize the results for the SDP and the ILP for instances with 50 to 240 machines assigned to 10 cells in Tables 7.14 - 7.17. Our ILP approach generates optimal layouts for all instances considered within one minute, while it took up to six hours for the SDP approach. It is easier to solve the DCFLP than the SRFLP for the ILP, as all optimal solutions were produced within one second for 10 circular layout cells, while it took up to nearly one minute for 10 single-row layout cells. In contrast to that, our experiments showed that it is easier to solve the SRFLP than the DCFLP for the SDP.

In summary, our ILP approach is able to solve instances with up to 240 machines assigned to 10 cells to global optimality within one minute, while it took up to six hours for the SDP approach proposed in [22]. Hence, we were able to improve the current best available exact approach for solving the CCLP. Further, we noticed that it is not beneficial to solve CCLP instances with an ASP approach, as both considered formulations are not able to solve instances with only 30 machines assigned to 5 cells to global optimality.

Instance	n	(2,2,0)		(2,0,2)		(2,1,1)	
		Optimum	Time	Optimum	Time	Optimum	Time
AnHuMa2019_10_2_50_set1	10	228626.5	0:01	252576.5	0:01	237970.5	0:01
AnHuMa2019_10_2_50_set2	10	160229.0	0:01	184876.0	0:01	167952.0	0:01
AnHuMa2019_10_2_50_set3	10	209208.0	0:01	225035.0	0:01	216690.0	0:01
AnHuMa2019_10_2_50_set4	10	160683.0	0:01	185014.0	0:01	171587.0	0:01
AnHuMa2019_10_2_50_set5	10	160291.0	0:01	185056.0	0:01	170466.0	0:01
AnHuMa2019_20_2_50_set1	20	392226.0	0:01	458654.0	0:25	423783.0	0:06
AnHuMa2019_20_2_50_set2	20	535746.5	0:01	635999.5	0:23	575307.5	0:05
AnHuMa2019_20_2_50_set3	20	613093.0	0:01	720433.0	0:34	669431.0	0:07
AnHuMa2019_20_2_50_set4	20	447236.0	0:01	574588.0	0:30	502399.0	0:04
AnHuMa2019_20_2_50_set5	20	381747.0	0:01	457277.0	0:40	417865.0	0:07
AnHuMa2019_30_2_50_set1	30	1092418.0	0:22	1298628.0	2:05	1200254.0	1:09
AnHuMa2019_30_2_50_set2	30	1069078.0	0:23	1315993.0	1:33	1187255.0	0:24
AnHuMa2019_30_2_50_set3	30	1191520.5	0:36	1488056.5	2:35	1340489.5	1:45
AnHuMa2019_30_2_50_set4	30	679752.5	0:12	845085.5	1:01	743178.5	0:28
AnHuMa2019_30_2_50_set5	30	1198741.0	0:05	1499673.0	1:12	1358601.0	0:18
AnHuMa2019_40_2_50_set1	40	1484881.0	0:56	1829107.0	4:23	1665471.0	3:00
AnHuMa2019_40_2_50_set2	40	1542343.5	2:03	1917527.5	15:54	1690584.5	10:38
AnHuMa2019_40_2_50_set3	40	2045226.5	1:02	2656180.5	10:23	2370303.5	4:40
AnHuMa2019_40_2_50_set4	40	1827206.5	2:24	2405535.5	10:23	2126104.5	7:00
AnHuMa2019_40_2_50_set5	40	1650568.0	1:12	1992067.0	5:67	1801935.0	2:38

Table 7.1: Results obtained by the `SDP` approach for instances with 10 to 40 machines that are allocated to 2 cells, each containing the same amount of machines. All instances could be solved to global optimality. The running times are given in min:sec.

Instance	n	(2,2,0)		(2,0,2)		(2,1,1)	
		Optimum	Time	Optimum	Time	Optimum	Time
AnHuMa2019_10_2_50_set1	10	228626.5	0.01	252576.5	0.01	237970.5	0.01
AnHuMa2019_10_2_50_set2	10	160229.0	0.01	184876.0	0.01	167952.0	0.01
AnHuMa2019_10_2_50_set3	10	209208.0	0.01	225035.0	0.01	216690.0	0.01
AnHuMa2019_10_2_50_set4	10	160683.0	0.01	185014.0	0.01	171587.0	0.01
AnHuMa2019_10_2_50_set5	10	160291.0	0.01	185056.0	0.01	170466.0	0.01
AnHuMa2019_20_2_50_set1	20	392226.0	0.17	458654.0	0.01	423783.0	0.04
AnHuMa2019_20_2_50_set2	20	535746.5	0.07	635999.5	0.01	575307.5	0.04
AnHuMa2019_20_2_50_set3	20	613093.0	0.07	720433.0	0.01	669431.0	0.04
AnHuMa2019_20_2_50_set4	20	447236.0	0.07	574588.0	0.01	502399.0	0.04
AnHuMa2019_20_2_50_set5	20	381747.0	0.14	457277.0	0.01	417865.0	0.03
AnHuMa2019_30_2_50_set1	30	1092418.0	3.10	1298628.0	0.02	1200254.0	2.73
AnHuMa2019_30_2_50_set2	30	1069078.0	0.63	1315993.0	0.01	1187255.0	0.27
AnHuMa2019_30_2_50_set3	30	1191520.5	1.12	1488056.5	0.02	1340489.5	1.21
AnHuMa2019_30_2_50_set4	30	679752.5	1.61	845085.5	0.02	743178.5	1.10
AnHuMa2019_30_2_50_set5	30	1198741.0	3.02	1499673.0	0.01	1358601.0	2.31
AnHuMa2019_40_2_50_set1	40	1484881.0	5.68	1829107.0	0.04	1665471.0	1.26
AnHuMa2019_40_2_50_set2	40	1542343.5	3.72	1917527.5	0.20	1690584.5	1.25
AnHuMa2019_40_2_50_set3	40	2045226.5	2.19	2656180.5	0.05	2370303.5	1.00
AnHuMa2019_40_2_50_set4	40	1827206.5	5.54	2405535.5	0.09	2126104.5	1.40
AnHuMa2019_40_2_50_set5	40	1650568.0	7.44	1992067.0	0.03	1801935.0	3.55

Table 7.2: Results obtained by our `ILP` approach for instances with 10 to 40 machines that are allocated to 2 cells, each containing the same amount of machines. All instances could be solved to global optimality. The running times are given in sec.msec.

Instance	(2,2,0)			(2,0,2)			(2,1,1)		
	Best Layout	Gap [%]	Time	Best Layout	Gap [%]	Time Time	Best Layout	Gap [%]	Time
AnHuMa2019_10_2_50_set1	228626.5	0.00	0.06	252576.5	0.00	0.28	237970.5	0.00	0.13
AnHuMa2019_10_2_50_set2	160229.0	0.00	0.07	184876.0	0.00	0.27	167952.0	0.00	0.11
AnHuMa2019_10_2_50_set3	209208.0	0.00	0.07	225035.0	0.00	0.22	216690.0	0.00	0.11
AnHuMa2019_10_2_50_set4	160683.0	0.00	0.05	185014.0	0.00	0.19	171587.0	0.00	0.09
AnHuMa2019_10_2_50_set5	160291.0	0.00	0.07	185056.0	0.00	0.23	170466.0	0.00	0.11
AnHuMa2019_20_2_50_set1	399780.0	1.89	6:00:00	470572.0	0.03	6:00:00	432185.0	1.94	6:00:00
AnHuMa2019_20_2_50_set2	551226.5	2.81	6:00:00	648204.5	0.02	6:00:00	584730.5	1.61	6:00:00
AnHuMa2019_20_2_50_set3	629699.0	2.64	6:00:00	728932.0	0.01	6:00:00	683484.0	2.06	6:00:00
AnHuMa2019_20_2_50_set4	455197.0	1.75	6:00:00	580673.0	0.01	6:00:00	511003.0	1.68	6:00:00
AnHuMa2019_20_2_50_set5	389748.0	2.05	6:00:00	463345.0	0.01	6:00:00	423090.0	1.23	6:00:00
AnHuMa2019_30_2_50_set1	1188375.0	8.07	6:00:00	1394401.0	6.87	6:00:00	1312269.0	8.54	6:00:00
AnHuMa2019_30_2_50_set2	1190182.0	10.18	6:00:00	1403038.0	6.20	6:00:00	1313143.0	9.59	6:00:00
AnHuMa2019_30_2_50_set3	1382982.5	13.84	6:00:00	1594822.5	6.69	6:00:00	1427897.5	6.12	6:00:00
AnHuMa2019_30_2_50_set4	796984.5	14.71	6:00:00	-	-	6:00:00	845343.5	12.09	6:00:00
AnHuMa2019_30_2_50_set5	1264840.0	5.23	6:00:00	1605363.0	6.58	6:00:00	1440629.0	5.69	6:00:00
AnHuMa2019_40_2_50_set1	1797395.0	17.39	6:00:00	-	-	6:00:00	-	-	6:00:00
AnHuMa2019_40_2_50_set2	1908035.5	19.17	6:00:00	-	-	6:00:00	-	-	6:00:00
AnHuMa2019_40_2_50_set3	2579535.5	20.71	6:00:00	-	-	6:00:00	2834119.5	16.37	6:00:00
AnHuMa2019_40_2_50_set4	2263493.5	19.27	6:00:00	-	-	6:00:00	2514638.5	15.45	6:00:00
AnHuMa2019_40_2_50_set5	1925755.0	14.29	6:00:00	-	-	6:00:00	2089446.0	13.76	6:00:00

Table 7.3: Results obtained by our ASP I formulation for instances with 10 to 40 machines that are allocated to 2 cells, each containing the same amount of machines. The optimality gap $\left(\frac{\text{Best Layout}-\text{Optimum ILP}}{\text{Best Layout}}\right)$ is given in percent and denotes the relative gap between the best objective found by the ASP and the optimal value found by the ILP. We set a time limit of 6 hours. - indicates that no feasible solution was found during the given time limit. The running times are given in sec.msec or h:min:sec respectively.

Instance	(2,2,0)			(2,0,2)			(2,1,1)		
	Best Layout	Gap [%]	Time	Best Layout	Gap [%]	Time Time	Best Layout	Gap [%]	Time
AnHuMa2019_10_2_50_set1	228626.5	0.00	0.17	252576.5	0.00	0.86	237970.5	0.00	0.33
AnHuMa2019_10_2_50_set2	160229.0	0.00	0.20	184876.0	0.00	0.86	167952.0	0.00	0.41
AnHuMa2019_10_2_50_set3	209208.0	0.00	0.15	225035.0	0.00	0.74	216690.0	0.00	0.29
AnHuMa2019_10_2_50_set4	160683.0	0.00	0.12	185014.0	0.00	0.51	171587.0	0.00	0.29
AnHuMa2019_10_2_50_set5	160291.0	0.00	0.18	185056.0	0.00	0.85	170466.0	0.00	0.44
AnHuMa2019_20_2_50_set1	397895.0	1.42	6:00:00	472062.0	2.84	6:00:00	436699.0	2.96	6:00:00
AnHuMa2019_20_2_50_set2	542192.5	1.19	6:00:00	649012.5	2.01	6:00:00	586622.5	1.93	6:00:00
AnHuMa2019_20_2_50_set3	630113.0	2.70	6:00:00	729804.0	1.28	6:00:00	683351.0	2.04	6:00:00
AnHuMa2019_20_2_50_set4	459830.0	2.74	6:00:00	587594.0	2.21	6:00:00	513863.0	2.23	6:00:00
AnHuMa2019_20_2_50_set5	391643.0	2.53	6:00:00	460063.0	0.61	6:00:00	427601.0	2.28	6:00:00
AnHuMa2019_30_2_50_set1	1175888.0	7.10	6:00:00	1421656.0	8.65	6:00:00	1276585.0	5.98	6:00:00
AnHuMa2019_30_2_50_set2	1149979.0	7.03	6:00:00	1419779.0	7.31	6:00:00	1245763.0	4.70	6:00:00
AnHuMa2019_30_2_50_set3	1282973.5	7.13	6:00:00	1623539.5	8.34	6:00:00	1434597.5	6.56	6:00:00
AnHuMa2019_30_2_50_set4	736454.5	7.70	6:00:00	953156.5	11.34	6:00:00	808049.5	8.03	6:00:00
AnHuMa2019_30_2_50_set5	1264551.0	5.20	6:00:00	1633202.0	8.18	6:00:00	1420843.0	4.38	6:00:00
AnHuMa2019_40_2_50_set1	1741842.0	14.75	6:00:00	1968828.0	7.10	6:00:00	1848080.0	9.88	6:00:00
AnHuMa2019_40_2_50_set2	1795139.5	14.08	6:00:00	2038841.5	5.95	6:00:00	1953988.5	13.48	6:00:00
AnHuMa2019_40_2_50_set3	2374642.5	13.87	6:00:00	2828313.5	6.09	6:00:00	2667149.5	11.13	6:00:00
AnHuMa2019_40_2_50_set4	2219938.5	17.69	6:00:00	2613536.5	7.96	6:00:00	2497062.5	14.86	6:00:00
AnHuMa2019_40_2_50_set5	1899463.0	13.10	6:00:00	2202105.0	9.54	6:00:00	1964862.0	8.29	6:00:00

Table 7.4: Results obtained by our `ASP` Π formulation for instances with 10 to 40 machines that are allocated to 2 cells, each containing the same amount of machines. The optimality gap $\left(\frac{\text{Best Layout}-\text{Optimum } \texttt{ILP}}{\text{Best Layout}}\right)$ is given in percent and denotes the relative gap between the best objective found by the `ASP` and the optimal value found by the `ILP`. We set a time limit of 6 hours. - indicates that no feasible solution was found during the given time limit. The running times are given in sec.msec or h:min:sec respectively.

Instance	n	(5,5,0)		(5,0,5)	
		Optimum	Time	Optimum	Time
AnHuMa2019_20_5_50_set1	20	350066.5	0.01	362305.5	0.01
AnHuMa2019_20_5_50_set2	20	454713.0	0.01	486194.0	0.01
AnHuMa2019_20_5_50_set3	20	443896.0	0.01	474242.0	0.01
AnHuMa2019_20_5_50_set4	20	524559.5	0.01	561419.5	0.01
AnHuMa2019_20_5_50_set5	20	493828.0	0.01	509371.0	0.01
AnHuMa2019_30_5_50_set1	30	810465.0	0.03	886666.0	0.01
AnHuMa2019_30_5_50_set2	30	895152.0	0.02	998065.0	0.01
AnHuMa2019_30_5_50_set3	30	704484.0	0.02	802975.0	0.01
AnHuMa2019_30_5_50_set4	30	573619.5	0.03	651785.5	0.01
AnHuMa2019_30_5_50_set5	30	959752.0	0.02	1074452.0	0.01
AnHuMa2019_40_5_50_set1	40	1158597.5	0.11	1286066.5	0.01
AnHuMa2019_40_5_50_set2	40	1147391.5	0.09	1319784.5	0.01
AnHuMa2019_40_5_50_set3	40	1247887.5	0.07	1464152.5	0.01
AnHuMa2019_40_5_50_set4	40	1367239.5	0.06	1604725.5	0.01
AnHuMa2019_40_5_50_set5	40	1299926.5	0.07	1486657.5	0.01

Table 7.5: Results obtained by our `ILP` approach for instances with 20 to 40 machines that are allocated to 5 cells, each containing the same amount of machines. All instances could be solved to global optimality. The running times are given in sec.msec.

Instance	n	(5,3,2)		(5,2,3)	
		Optimum	Time	Optimum	Time
AnHuMa2019_20_5_50_set1	20	355574.5	0.01	356940.5	0.01
AnHuMa2019_20_5_50_set2	20	464325.0	0.01	473079.0	0.01
AnHuMa2019_20_5_50_set3	20	457249.0	0.01	466993.0	0.01
AnHuMa2019_20_5_50_set4	20	535584.5	0.01	546102.5	0.01
AnHuMa2019_20_5_50_set5	20	501271.0	0.01	502468.0	0.01
AnHuMa2019_30_5_50_set1	30	836331.0	0.01	851642.0	0.01
AnHuMa2019_30_5_50_set2	30	929186.0	0.01	952323.0	0.01
AnHuMa2019_30_5_50_set3	30	740868.0	0.01	761890.0	0.01
AnHuMa2019_30_5_50_set4	30	598289.5	0.02	613398.5	0.02
AnHuMa2019_30_5_50_set5	30	1009554.0	0.02	1019115.0	0.01
AnHuMa2019_40_5_50_set1	40	1203730.5	0.07	1231724.5	0.06
AnHuMa2019_40_5_50_set2	40	1204103.5	0.14	1229580.5	0.03
AnHuMa2019_40_5_50_set3	40	1335860.5	0.06	1368371.5	0.11
AnHuMa2019_40_5_50_set4	40	1453556.5	0.05	1506373.5	0.03
AnHuMa2019_40_5_50_set5	40	1369533.5	0.05	1415588.5	0.05

Table 7.6: Results obtained by our `ILP` approach for instances with 20 to 40 machines that are allocated to 5 cells, each containing the same amount of machines. All instances could be solved to global optimality. The running times are given in sec.msec.

Instance	n	(5,5,0)			(5,0,5)		
		Best Layout	Gap [%]	Time	Best Layout	Gap [%]	Time
AnHuMa2019_20_5_50_set1	20	350066.5	0.00	0.35	362305.5	0.00	0.94
AnHuMa2019_20_5_50_set2	20	454713.0	0.00	0.28	486194.0	0.00	1.19
AnHuMa2019_20_5_50_set3	20	443896.0	0.00	0.23	474242.0	0.00	1.11
AnHuMa2019_20_5_50_set4	20	524559.5	0.00	0.20	561419.5	0.00	1.14
AnHuMa2019_20_5_50_set5	20	493828.0	0.00	0.39	509371.0	0.00	1.06
AnHuMa2019_30_5_50_set1	30	818682.0	1.00	6:00:00	889665.0	0.00	6:00:00
AnHuMa2019_30_5_50_set2	30	902613.0	0.83	6:00:00	1003300.0	0.01	6:00:00
AnHuMa2019_30_5_50_set3	30	709184.0	0.66	6:00:00	807008.0	0.00	6:00:00
AnHuMa2019_30_5_50_set4	30	576622.5	0.52	6:00:00	655661.5	0.01	6:00:00
AnHuMa2019_30_5_50_set5	30	967459.0	0.80	6:00:00	1077261.0	0.00	6:00:00
AnHuMa2019_40_5_50_set1	40	1218711.5	4.93	6:00:00	1317664.5	0.02	6:00:00
AnHuMa2019_40_5_50_set2	40	1224720.5	6.31	6:00:00	1332565.5	0.01	6:00:00
AnHuMa2019_40_5_50_set3	40	1377900.5	9.44	6:00:00	1478593.5	0.01	6:00:00
AnHuMa2019_40_5_50_set4	40	1442379.5	5.21	6:00:00	1619749.5	0.01	6:00:00
AnHuMa2019_40_5_50_set5	40	1379634.5	5.78	6:00:00	1504288.5	0.01	6:00:00

Table 7.7: Results obtained by our `ASP` I formulation for instances with 20 to 40 machines that are allocated to 5 cells, each containing the same amount of machines. The optimality gap $\left(\frac{\text{Best Layout}-\text{Optimum } \texttt{ILP}}{\text{Best Layout}}\right)$ is given in percent and denotes the relative gap between the best objective found by the `ASP` and the optimal value found by the `ILP`. We set a time limit of 6 hours. The running times are given in sec.msec or h:min:sec respectively.

Instance	n	(5,3,2)			(5,2,3)		
		Best Layout	Gap [%]	Time	Best Layout	Gap [%]	Time
AnHuMa2019_20_5_50_set1	20	355574.5	0.00	0.39	356940.5	0.00	0.56
AnHuMa2019_20_5_50_set2	20	464325.0	0.00	0.37	473079.0	0.00	0.62
AnHuMa2019_20_5_50_set3	20	457249.0	0.00	0.40	466993.0	0.00	0.69
AnHuMa2019_20_5_50_set4	20	535584.5	0.00	0.28	546102.5	0.00	0.57
AnHuMa2019_20_5_50_set5	20	501271.0	0.00	0.54	502468.0	0.00	0.67
AnHuMa2019_30_5_50_set1	30	844208.0	0.93	6:00:00	853483.0	0.22	6:00:00
AnHuMa2019_30_5_50_set2	30	933641.0	0.48	6:00:00	957707.0	0.56	6:00:00
AnHuMa2019_30_5_50_set3	30	745089.0	0.57	6:00:00	766444.0	0.59	6:00:00
AnHuMa2019_30_5_50_set4	30	605938.5	1.26	6:00:00	616213.5	0.46	6:00:00
AnHuMa2019_30_5_50_set5	30	1014653.0	0.50	6:00:00	1021774.0	0.26	6:00:00
AnHuMa2019_40_5_50_set1	40	1247580.5	3.51	6:00:00	1259575.5	2.21	6:00:00
AnHuMa2019_40_5_50_set2	40	1242797.5	3.11	6:00:00	1258641.5	2.31	6:00:00
AnHuMa2019_40_5_50_set3	40	1360922.5	1.84	6:00:00	1391524.5	1.66	6:00:00
AnHuMa2019_40_5_50_set4	40	1487515.5	2.28	6:00:00	1538202.5	2.07	6:00:00
AnHuMa2019_40_5_50_set5	40	1400053.5	2.18	6:00:00	1441484.5	1.80	6:00:00

Table 7.8: Results obtained by our `ASP I` formulation for instances with 20 to 40 machines that are allocated to 5 cells, each containing the same amount of machines. The optimality gap $\left(\frac{\text{Best Layout}-\text{Optimum }\texttt{ILP}}{\text{Best Layout}}\right)$ is given in percent and denotes the relative gap between the best objective found by the `ASP` and the optimal value found by the `ILP`. We set a time limit of 6 hours. The running times are given in sec.msec or h:min:sec respectively.

Instance	n	(5,5,0)			(5,0,5)		
		Best Layout	Gap [%]	Time	Best Layout	Gap [%]	Time
AnHuMa2019_20_5_50_set1	20	350066.5	0.00	1.53	362305.5	0.00	8.37
AnHuMa2019_20_5_50_set2	20	454713.0	0.00	1.50	486194.0	0.00	7.75
AnHuMa2019_20_5_50_set3	20	443896.0	0.00	1.05	474242.0	0.00	7.43
AnHuMa2019_20_5_50_set4	20	524559.5	0.00	0.65	561419.5	0.00	7.56
AnHuMa2019_20_5_50_set5	20	493828.0	0.00	2.06	509371.0	0.00	8.38
AnHuMa2019_30_5_50_set1	30	823820.0	1.62	6:00:00	892422.0	0.01	6:00:00
AnHuMa2019_30_5_50_set2	30	898489.0	0.37	6:00:00	1006801.0	0.01	6:00:00
AnHuMa2019_30_5_50_set3	30	705837.0	0.19	6:00:00	808190.0	0.01	6:00:00
AnHuMa2019_30_5_50_set4	30	576786.5	0.55	6:00:00	657782.5	0.01	6:00:00
AnHuMa2019_30_5_50_set5	30	962880.0	0.32	6:00:00	1081189.0	0.01	6:00:00
AnHuMa2019_40_5_50_set1	40	1272257.5	8.93	6:00:00	1306188.5	0.02	6:00:00
AnHuMa2019_40_5_50_set2	40	1334142.5	14.00	6:00:00	1338441.5	0.01	6:00:00
AnHuMa2019_40_5_50_set3	40	1297236.5	3.80	6:00:00	1481146.5	0.01	6:00:00
AnHuMa2019_40_5_50_set4	40	1487555.5	8.09	6:00:00	1622168.5	0.01	6:00:00
AnHuMa2019_40_5_50_set5	40	1377550.5	5.63	6:00:00	1507751.5	0.01	6:00:00

Table 7.9: Results obtained by our `ASP` Π formulation for instances with 20 to 40 machines that are allocated to 5 cells, each containing the same amount of machines. The optimality gap $\left(\frac{\text{Best Layout}-\text{Optimum } \texttt{ILP}}{\text{Best Layout}}\right)$ is given in percent and denotes the relative gap between the best objective found by the `ASP` and the optimal value found by the `ILP`. We set a time limit of 6 hours. The running times are given in sec.msec or h:min:sec respectively.

Instance	n	(5,3,2)			(5,2,3)		
		Best Layout	Gap [%]	Time	Best Layout	Gap [%]	Time
AnHuMa2019_20_5_50_set1	20	355574.5	0.00	4.05	356940.5	0.00	4.91
AnHuMa2019_20_5_50_set2	20	464325.0	0.00	3.44	473079.0	0.00	4.90
AnHuMa2019_20_5_50_set3	20	457249.0	0.00	3.52	466993.0	0.00	4.94
AnHuMa2019_20_5_50_set4	20	535584.5	0.00	2.36	546102.5	0.00	3.90
AnHuMa2019_20_5_50_set5	20	501271.0	0.00	4.66	502468.0	0.00	5.23
AnHuMa2019_30_5_50_set1	30	840182.0	0.46	6:00:00	867263.0	1.80	6:00:00
AnHuMa2019_30_5_50_set2	30	936539.0	0.79	6:00:00	960815.0	0.88	6:00:00
AnHuMa2019_30_5_50_set3	30	743419.0	0.34	6:00:00	765267.0	0.44	6:00:00
AnHuMa2019_30_5_50_set4	30	610830.5	2.05	6:00:00	618678.5	0.85	6:00:00
AnHuMa2019_30_5_50_set5	30	1014247.0	0.46	6:00:00	1032561.0	1.30	6:00:00
AnHuMa2019_40_5_50_set1	40	1250320.5	3.73	6:00:00	1263684.5	2.53	6:00:00
AnHuMa2019_40_5_50_set2	40	1250700.5	3.73	6:00:00	1251457.5	1.75	6:00:00
AnHuMa2019_40_5_50_set3	40	1373352.5	2.73	6:00:00	1389259.5	1.50	6:00:00
AnHuMa2019_40_5_50_set4	40	1490045.5	2.45	6:00:00	1536966.5	1.99	6:00:00
AnHuMa2019_40_5_50_set5	40	1477311.5	7.30	6:00:00	1434162.5	1.30	6:00:00

Table 7.10: Results obtained by our ASP Π formulation for instances with 20 to 40 machines that are allocated to 5 cells, each containing the same amount of machines. The optimality gap $\left(\frac{\text{Best Layout}-\text{Optimum ILP}}{\text{Best Layout}}\right)$ is given in percent and denotes the relative gap between the best objective found by the ASP and the optimal value found by the ILP. We set a time limit of 6 hours. The running times are given in sec.msec or h:min:sec respectively.

Instance	n	(10,10,0)		(10,0,10)		(10,5,5)	
		Optimum	Time	Optimum	Time	Optimum	Time
AnHuMa2019_30_10_50_set1	30	687703.5	0.01	715743.5	0.01	704594.5	0.01
AnHuMa2019_30_10_50_set2	30	652972.0	0.01	666199.0	0.01	661217.0	0.01
AnHuMa2019_30_10_50_set3	30	747556.5	0.01	770674.5	0.01	758671.5	0.01
AnHuMa2019_30_10_50_set4	30	705758.5	0.01	723755.5	0.01	710639.5	0.01
AnHuMa2019_30_10_50_set5	30	724494.0	0.01	741343.0	0.01	734538.0	0.01
AnHuMa2019_40_10_50_set1	40	955612.0	0.01	1004716.0	0.01	983285.0	0.01
AnHuMa2019_40_10_50_set2	40	870451.5	0.01	923380.5	0.01	890223.5	0.01
AnHuMa2019_40_10_50_set3	40	1139077.5	0.01	1223709.5	0.01	1182417.5	0.01
AnHuMa2019_40_10_50_set4	40	1189946.0	0.01	1257222.5	0.01	1219487.0	0.01
AnHuMa2019_40_10_50_set5	40	1087075.5	0.01	1136673.5	0.01	1109605.5	0.01

Table 7.11: Results obtained by our `ILP` approach for instances with 30 or 40 machines that are allocated to 10 cells, each containing the same amount of machines. All instances could be solved to global optimality. The running times are given in sec.msec.

Instance	(10,10,0)			(10,0,10)			(10,5,5)		
	Best Layout	Gap [%]	Time	Best Layout	Gap [%]	Time Time	Best Layout	Gap [%]	Time
AnHuMa2019_30_10_50_set1	687703.5	0.00	0.14	715743.5	0.00	0.11	704594.5	0.00	0.14
AnHuMa2019_30_10_50_set2	652972.0	0.00	0.14	666199.0	0.00	0.12	661217.0	0.00	0.14
AnHuMa2019_30_10_50_set3	747556.5	0.00	0.14	770674.5	0.00	0.12	758671.5	0.00	0.13
AnHuMa2019_30_10_50_set4	705758.5	0.00	0.13	723755.5	0.00	0.11	710639.5	0.00	0.12
AnHuMa2019_30_10_50_set5	724494.0	0.00	0.14	741343.0	0.00	0.12	734538.0	0.00	0.12
AnHuMa2019_40_10_50_set1	955612.0	0.00	6:00:00	1004716.0	0.00	6:00:00	983285.0	0.00	6:00:00
AnHuMa2019_40_10_50_set2	870451.5	0.00	6:00:00	923825.5	0.00	6:00:00	890223.5	0.00	6:00:00
AnHuMa2019_40_10_50_set3	1139077.5	0.00	6:00:00	1223725.5	0.00	6:00:00	1182417.5	0.00	6:00:00
AnHuMa2019_40_10_50_set4	1189946.0	0.00	6:00:00	1257222.0	0.00	6:00:00	1219487.0	0.00	6:00:00
AnHuMa2019_40_10_50_set5	1087075.5	0.00	6:00:00	1136673.5	0.00	6:00:00	1109605.5	0.00	6:00:00

Table 7.12: Results obtained by our `ASP I` formulation for instances with 30 or 40 machines that are allocated to 10 cells, each containing the same amount of machines. The optimality gap $\left(\frac{\text{Best Layout}-\text{Optimum ILP}}{\text{Best Layout}}\right)$ is given in percent and denotes the relative gap between the best objective found by the `ASP` and the optimal value found by the `ILP`. We set a time limit of 6 hours. The running times are given in h:min:sec.

Instance	(10,10,0)			(10,0,10)			(10,5,5)		
	Best Layout	Gap [%]	Time	Best Layout	Gap [%]	Time Time	Best Layout	Gap [%]	Time
AnHuMa2019_30_10_50_set1	687703.5	0.00	0.11	715743.5	0.00	1.18	704594.5	0.00	0.44
AnHuMa2019_30_10_50_set2	652972.0	0.00	0.13	666199.0	0.00	1.22	661217.0	0.00	0.48
AnHuMa2019_30_10_50_set3	747556.5	0.00	0.12	770674.5	0.00	1.10	758671.5	0.00	0.44
AnHuMa2019_30_10_50_set4	705758.5	0.00	0.13	723755.5	0.00	1.22	710639.5	0.00	0.46
AnHuMa2019_30_10_50_set5	724494.0	0.00	0.12	741343.0	0.00	1.14	734538.0	0.00	0.44
AnHuMa2019_40_10_50_set1	955612.0	0.00	6:00:00	1004967.0	0.00	6:00:00	984533.0	0.13	6:00:00
AnHuMa2019_40_10_50_set2	870451.5	0.00	6:00:00	923493.5	0.00	6:00:00	890223.5	0.00	6:00:00
AnHuMa2019_40_10_50_set3	1139077.5	0.00	6:00:00	1224397.5	0.00	6:00:00	1182417.5	0.00	6:00:00
AnHuMa2019_40_10_50_set4	1189946.0	0.00	6:00:00	1257464.0	0.00	6:00:00	1219628.0	0.01	6:00:00
AnHuMa2019_40_10_50_set5	1087075.5	0.00	6:00:00	1137396.5	0.00	6:00:00	1110118.5	0.05	6:00:00

Table 7.13: Results obtained by our ASP Π formulation for instances with 30 or 40 machines that are allocated to 10 cells, each containing the same amount of machines. The optimality gap $\left(\frac{\text{Best Layout}-\text{Optimum ILP}}{\text{Best Layout}}\right)$ is given in percent and denotes the relative gap between the best objective found by the ASP and the optimal value found by the ILP. We set a time limit of 6 hours. The running times are given in h:min:sec.

Instance	n	(10,10,0)		(10,0,10)		(10,5,5)	
		Optimum	Time	Optimum	Time	Optimum	Time
AnHuMa2019_50_10_50_set1	50	1552582.0	0:00:01	1717778.0	0:00:03	1625784.0	0:00:02
AnHuMa2019_50_10_50_set2	50	1472168.5	0:00:01	1617938.5	0:00:01	1548033.5	0:00:01
AnHuMa2019_50_10_50_set3	50	1286752.5	0:00:01	1398806.5	0:00:02	1349438.5	0:00:01
AnHuMa2019_50_10_50_set4	50	1599467.5	0:00:01	1717724.5	0:00:04	1655439.5	0:00:02
AnHuMa2019_50_10_50_set5	50	1479895.0	0:00:01	1575147.0	0:00:02	1509695.0	0:00:02
AnHuMa2019_60_10_50_set1	60	1679375.5	0:00:01	1861688.5	0:00:29	1762452.5	0:00:16
AnHuMa2019_60_10_50_set2	60	2092249.0	0:00:01	2321738.0	0:01:31	2199065.0	0:00:02
AnHuMa2019_60_10_50_set3	60	1426658.0	0:00:01	1592971.0	0:01:21	1505029.0	0:00:07
AnHuMa2019_60_10_50_set4	60	1934101.0	0:00:01	2143448.0	0:00:03	2026326.0	0:00:02
AnHuMa2019_60_10_50_set5	60	2035162.5	0:00:01	2218497.5	0:01:50	2105900.5	0:00:02
AnHuMa2019_70_10_50_set1	70	2436289.5	0:00:01	2720053.5	0:01:54	2560335.5	0:00:07
AnHuMa2019_70_10_50_set2	70	2162891.0	0:00:02	2450271.0	0:01:19	2311495.0	0:01:18
AnHuMa2019_70_10_50_set3	70	2131691.0	0:00:01	2435519.0	0:00:04	2264726.0	0:00:03
AnHuMa2019_70_10_50_set4	70	2045096.5	0:00:02	2338862.5	0:01:43	2172892.5	0:00:07
AnHuMa2019_70_10_50_set5	70	2086338.5	0:00:02	2331158.5	0:02:18	2207610.5	0:00:04
AnHuMa2019_160_10_50_set1	160	7913309.0	0:10:44	9838495.0	4:16:28	8912902.0	2:45:18
AnHuMa2019_160_10_50_set2	160	8218452.5	0:28:15	10336515.5	4:50:24	9220920.5	3:49:51
AnHuMa2019_160_10_50_set3	160	8986090.5	0:13:04	11492506.5	4:40:28	10298190.5	3:25:34
AnHuMa2019_160_10_50_set4	160	7807915.0	0:23:35	9653194.0	3:48:08	8658369.0	1:21:53
AnHuMa2019_160_10_50_set5	160	8349981.0	0:10:57	10281281.0	3:20:04	9301331.0	2:16:59

Table 7.14: Results obtained by the SDP approach for instances with 50 to 160 machines that are allocated to 10 cells, each containing the same amount of machines. We set a time limit of 6 hours. The running times are given in h:min:sec.

Instance	n	(10,10,0)		(10,0,10)		(10,5,5)	
		Optimum	Time	Optimum	Time	Optimum	Time
AnHuMa2019_50_10_50_set1	50	1552582.0	0.03	1717778.0	0.02	1625784.0	0.01
AnHuMa2019_50_10_50_set2	50	1472168.5	0.01	1617938.5	0.01	1548033.5	0.01
AnHuMa2019_50_10_50_set3	50	1286752.5	0.02	1398806.5	0.01	1349438.5	0.01
AnHuMa2019_50_10_50_set4	50	1599467.5	0.02	1717724.5	0.01	1655439.5	0.01
AnHuMa2019_50_10_50_set5	50	1479895.0	0.01	1575147.0	0.01	1509695.0	0.01
AnHuMa2019_60_10_50_set1	60	1679375.5	0.03	1861688.5	0.01	1762452.5	0.02
AnHuMa2019_60_10_50_set2	60	2092249.0	0.03	2321738.0	0.01	2199065.0	0.02
AnHuMa2019_60_10_50_set3	60	1426658.0	0.03	1592971.0	0.01	1505029.0	0.02
AnHuMa2019_60_10_50_set4	60	1934101.0	0.05	2143448.0	0.01	2026326.0	0.04
AnHuMa2019_60_10_50_set5	60	2035162.5	0.03	2218497.5	0.01	2105900.5	0.02
AnHuMa2019_70_10_50_set1	70	2436289.5	0.06	2720053.5	0.01	2560335.5	0.03
AnHuMa2019_70_10_50_set2	70	2162891.0	0.06	2450271.0	0.01	2311495.0	0.04
AnHuMa2019_70_10_50_set3	70	2131691.0	0.05	2435519.0	0.01	2264726.0	0.04
AnHuMa2019_70_10_50_set4	70	2045096.5	0.06	2338862.5	0.01	2172892.5	0.04
AnHuMa2019_70_10_50_set5	70	2086338.5	0.06	2331158.5	0.01	2207610.5	0.03
AnHuMa2019_160_10_50_set1	160	7913309.0	5.06	9838495.0	0.09	8912902.0	2.10
AnHuMa2019_160_10_50_set2	160	8218452.5	5.42	10336515.5	0.09	9220920.5	1.96
AnHuMa2019_160_10_50_set3	160	8986090.5	5.54	11492506.5	0.09	10298190.5	2.12
AnHuMa2019_160_10_50_set4	160	7807915.0	5.49	9653194.0	0.08	8658369.0	2.47
AnHuMa2019_160_10_50_set5	160	8349981.0	5.61	10281281.0	0.10	9301331.0	1.95

Table 7.15: Results obtained by our `ILP` approach for instances with 50 to 160 machines that are allocated to 10 cells, each containing the same amount of machines. All instances could be solved to global optimality. The running times are given in sec.msec.

Instance	(10,10,0)			(10,0,10)			(10,5,5)		
	Best Layout	Gap [%]	Time	Best Layout	Gap [%]	Time Time	Best Layout	Gap [%]	Time
AnHuMa2019_180_10_50_set1	11847086.0	0.00	1:15:15	14876406.0	0.00	6:00:00	13326950.0	0.00	6:00:00
AnHuMa2019_180_10_50_set2	13691880.0	0.00	0:44:21	17448829.0	0.00	6:00:00	15661433.0	0.00	6:00:00
AnHuMa2019_180_10_50_set3	9659151.5	0.00	1:11:32	12356163.5	0.00	6:00:00	11086298.5	0.00	3:20:38
AnHuMa2019_180_10_50_set4	9990825.5	0.00	1:12:17	12524010.5	0.00	6:00:00	11301164.5	0.00	6:00:00
AnHuMa2019_180_10_50_set5	11296590.0	0.00	1:13:20	14114312.0	0.00	6:00:00	12842777.0	0.00	4:47:50
AnHuMa2019_200_10_50_set1	11541375.0	0.00	6:00:00	14698636.0	0.00	6:00:00	13195055.0	0.00	6:00:00
AnHuMa2019_200_10_50_set2	13844361.0	0.00	6:00:00	17527259.0	0.00	6:00:00	15822453.0	0.00	6:00:00
AnHuMa2019_200_10_50_set3	10435531.5	0.00	6:00:00	13546225.5	0.00	6:00:00	12052728.5	0.00	6:00:00
AnHuMa2019_200_10_50_set4	10866279.5	0.00	6:00:00	13760529.5	0.00	6:00:00	12337457.5	0.00	6:00:00
AnHuMa2019_200_10_50_set5	13034866.5	0.00	6:00:00	16380776.5	0.00	6:00:00	14752058.5	0.00	6:00:00
AnHuMa2019_220_10_50_set1	15024578.0	0.00	6:00:00	19278993.0	0.00	6:00:00	17004335.0	0.00	6:00:00
AnHuMa2019_220_10_50_set2	16963680.0	0.00	6:00:00	22058072.0	0.00	6:00:00	19566235.0	0.00	6:00:00
AnHuMa2019_220_10_50_set3	13332956.5	0.00	6:00:00	17108604.5	0.00	6:00:00	15203931.5	0.00	6:00:00
AnHuMa2019_220_10_50_set4	12536751.5	0.00	6:00:00	16490966.5	0.00	6:00:00	14412927.5	0.00	6:00:00
AnHuMa2019_220_10_50_set5	16488186.0	0.00	6:00:00	20718107.0	0.00	6:00:00	18670904.0	0.00	6:00:00
AnHuMa2019_240_10_50_set1	16359203.5	0.02	6:00:00	21081860.0	0.77	6:00:00	18642671.5	0.00	6:00:00
AnHuMa2019_240_10_50_set2	15327623.0	0.01	6:00:00	20138071.0	0.98	6:00:00	17512293.0	0.00	6:00:00
AnHuMa2019_240_10_50_set3	17961278.5	0.04	6:00:00	23709077.0	1.54	6:00:00	20659606.0	0.00	6:00:00
AnHuMa2019_240_10_50_set4	16880738.0	0.03	6:00:00	21762166.0	0.58	6:00:00	19365839.5	0.00	6:00:00
AnHuMa2019_240_10_50_set5	16971269.0	0.03	6:00:00	22466517.0	0.91	6:00:00	19813487.5	0.00	6:00:00

Table 7.16: Results obtained by the SDP approach for instances with 180 to 240 machines that are allocated to 10 cells, each containing the same amount of machines. The optimality gap $\left(\frac{\text{Best Layout}-\text{Optimum ILP}}{\text{Best Layout}}\right)$ is given in percent and denotes the relative gap between the best objective found by the SDP and the optimal value found by the ILP. We set a time limit of 6 hours. The running times are given in h:min:sec.

Instance	n	(10,10,0)		(10,0,10)		(10,5,5)	
		Optimum	Time	Optimum	Time	Optimum	Time
AnHuMa2019_180_10_50_set1	180	11847086.0	9.52	14876406.0	0.12	13326950.0	4.32
AnHuMa2019_180_10_50_set2	180	13691880.0	11.08	17448829.0	0.24	15661433.0	4.33
AnHuMa2019_180_10_50_set3	180	9659151.5	11.11	12356163.5	0.14	11086298.5	4.31
AnHuMa2019_180_10_50_set4	180	9990825.5	11.04	12524010.5	0.13	11301164.5	4.26
AnHuMa2019_180_10_50_set5	180	11296590.0	9.50	14114312.0	0.14	12842777.0	4.15
AnHuMa2019_200_10_50_set1	200	11541375.0	20.13	14698636.0	0.17	13195055.0	8.17
AnHuMa2019_200_10_50_set2	200	13844361.0	20.12	17527259.0	0.20	15822453.0	8.19
AnHuMa2019_200_10_50_set3	200	10435531.5	17.89	13546225.5	0.21	12052728.5	8.19
AnHuMa2019_200_10_50_set4	200	10866279.5	16.75	13760529.5	0.19	12337457.5	8.19
AnHuMa2019_200_10_50_set5	200	13034866.5	16.46	16380776.5	0.19	14752058.5	8.23
AnHuMa2019_220_10_50_set1	220	15024578.0	31.88	19278993.0	0.25	17004335.0	12.20
AnHuMa2019_220_10_50_set2	220	16963680.0	31.71	22058072.0	0.21	19566235.0	13.22
AnHuMa2019_220_10_50_set3	220	13332956.5	30.82	17108604.5	0.24	15203931.5	14.30
AnHuMa2019_220_10_50_set4	220	12536751.5	33.08	16490966.5	0.27	14412927.5	12.25
AnHuMa2019_220_10_50_set5	220	16488186.0	33.05	20718107.0	0.27	18670904.0	13.43
AnHuMa2019_240_10_50_set1	240	16355931.5	47.79	20919529.5	0.38	18642671.5	23.37
AnHuMa2019_240_10_50_set2	240	15326090.0	51.00	19940718.0	0.31	17512293.0	24.02
AnHuMa2019_240_10_50_set3	240	17954094.0	50.51	23343957.0	0.30	20659606.0	22.89
AnHuMa2019_240_10_50_set4	240	16875673.5	47.70	21635945.5	0.33	19365839.5	19.00
AnHuMa2019_240_10_50_set5	240	16966177.5	52.71	22262071.5	0.31	19813487.5	21.90

Table 7.17: Results obtained by our ILP approach for instances with 180 to 240 machines that are allocated to 10 cells, each containing the same amount of machines. All instances could be solved to global optimality. The running times are given in sec.msec.

Chapter 8

Conclusion

In this thesis we proposed an Integer Linear Programming (ILP) formulation as well as two Answer Set Programming formulations (ASP) for solving the Combined Cell Layout Problem (CCLP). The CCLP minimizes the material handling costs in cellular manufacturing systems with at least two cells where some pieces need to be processed in more than one cell.

Our ILP approach outperformed the Semidefinite Programming (SDP) approach by Hungerländer and Anjos (2013), which was the only exact approach for solving the CCLP in the literature. While the ILP solved all considered instances with up to 240 machines arranged in 10 cells within one minute, the SDP approach took up to six hours.

It turned out that it is not beneficial to use an ASP approach for solving layout problems as both considered formulations were not able to solve instances with only 30 machines arranged in 5 cells to optimality within a given time limit of six hours.

Chapter 9

Outlook

In this thesis we considered only cells with single row and circular layouts. However, the method can in principle be extended to any existing layout type. Hence, it would be a worthwhile future research direction to extend the `ILP` approach to handle cells with multi-row layouts. In addition, it would be interesting to develop heuristic approaches for producing high-quality `CCLP` solutions significantly faster than our exact approaches.

Bibliography

[1] P. Afentakis. A loop layout design problem for flexible manufacturing systems. *International Journal of Flexible Manufacturing Systems*, 1:175–196, 1989.

[2] I. K. Altınel and T. Öncan. Design of unidirectional cyclic layouts. *International Journal of Production Research*, 43(19):3983–4008, 2005.

[3] A. R. S. Amaral. A new lower bound for the single row facility layout problem. *Discrete Applied Mathematics*, 157(1):183–190, 2009.

[4] A. R. S. Amaral. The Corridor Allocation Problem. *Computers & Operations Research*, 39(12):3325–3330, 2012.

[5] M. F. Anjos, A. Kennings, and A. Vannelli. A semidefinite optimization approach for the single-row layout problem with unequal dimensions. *Discrete Optimization*, 2(2):113 – 122, 2005.

[6] M. F. Anjos, P. Hungerländer, and K. Maier. An Integer Linear Programming Approach for the Combined Cell Layout Problem. In *2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 705–709, Dec 2018. doi: 10.1109/IEEM.2018.8607272.

[7] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[8] Y. Bozer and S. Rim. Exact solution procedures for the circular layout problem. Technical report, University of Michigan, USA, Report No. 89-33, 1989.

[9] C. Buchheim, A. Wiegele, and L. Zheng. Exact algorithms for the quadratic linear ordering problem. *INFORMS Journal on Computing*, 22(1):168–177, 2010.

[10] M. Chimani and P. Hungerländer. Exact approaches to multilevel vertical orderings. *INFORMS Journal on Computing*, 25(4):611–624, 2013.

[11] M. Chimani, P. Hungerländer, M. Jünger, and P. Mutzel. An SDP approach to multi-level crossing minimization. *Journal of Experimental Algorithmics*, 17: 3.3:3.1–3.3:3.26, 2012.

[12] C.-H. Chu. Recent advances in mathematical programming for cell formation. In *Planning, Design, and Analysis of Cellular Manufacturing Systems*, number 24 in Manufacturing Research and Technology, pages 3–46. Elsevier Science B.V., 1995.

[13] J. Chung and J. Tanchoco. The double row layout problem. *International Journal of Production Research*, 48(3):709–727, 2010.

[14] D. Datta, A. R. S. Amaral, and J. R. Figueira. Single row facility layout problem using a permutation-based genetic algorithm. *European Journal of Operational Research*, 213(2):388–394, 2011.

[15] J. DeGaspari. Cell Culture. *Mechanical Engineering*, 123(3):56–59, 2001.

[16] A. Fischer, F. Fischer, and P. Hungerländer. New exact approaches to row layout problems. Technical report, Alpen-Adria Universität Klagenfurt, Mathematics, Optimization Group, TR-ARUK-M-O-15-05, 2015.

[17] I. Fischer, G. Gruber, F. Rendl, and R. Sotirov. Computational experience with a bundle method for semidefinite cutten plane relaxations of max-cut and equipartition. *Mathematical Programming*, 105:451–469, 2006.

[18] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145, 1995.

[19] C. Helmberg, F. Rendl, R. Vanderbei, and H. Wolkowicz. An interior-point method for semidefinite programming. *SIAM Journal on Optimization*, 6:342–361, 1996.

[20] S. S. Heragu and A. Kusiak. Machine Layout Problem in Flexible Manufacturing Systems. *Operations Research*, 36(2):258–268, 1988.

[21] P. Hungerländer. A semidefinite optimization approach to the directed circular facility layout problem. *IFAC Proceedings Volumes*, 46(9):2033 – 2038, 2013. 7th IFAC Conference on Manufacturing Modelling, Management, and Control.

[22] P. Hungerländer and M. F. Anjos. An exact approach for the combined cell layout problem. In *Operations Research Proceedings 2012*, pages 275–281, 2013.

[23] P. Hungerländer and M. F. Anjos. A semidefinite optimization-based approach for global optimization of multi-row facility layout. *European Journal of Operational Research*, 245(1):46–61, 2015.

[24] P. Hungerländer and F. Rendl. A computational study and survey of methods for the single-row facility layout problem. *Computational Optimization and Applications*, 55(1):1–20, 2013.

[25] P. Hungerländer, K. Maier, J. Pöcher, and C. Truden. On a New Modelling Approach for Circular Layouts and Its Practical Advantages. In *2017 IEEE International Conference on Industrial Engineering and Engineering Managment (IEEM)*, pages 1581–1585, 2017.

[26] P. Hungerländer, K. Maier, J. Pöcher, and C. Truden. Exact and Heuristic Approaches for a New Circular Layout Problem. Technical report, Alpen-Adria Universität Klagenfurt, Mathematics, Optimization Group, TR-AAUK-M-O-18-06-09, 2018.

[27] R. Kothari and D. Ghosh. Tabu search for the single row facility layout problem using exhaustive 2-opt and insertion neighborhoods. *European Journal of Operational Research*, 224(1):93–100, 2013.

[28] P. Kouvelis and M. W. Kim. Unidirectional loop network layout problem in automated manufacturing systems. *Operations Research*, 40:533–550, 1992.

[29] P. Kouvelis, W.-C. Chiang, and A. Kiran. A survey of layout issues in flexible manufacturing systems. *Omega*, 20(3):375–390, 1992.

[30] V. Liberatore. Circular arrangements. In P. Widmayer, S. Eidenbenz, F. Triguero, R. Morales, R. Conejo, and M. Hennessy, editors, *Automata, Languages and Programming*, volume 2380 of *Lecture Notes in Computer Science*, pages 782–783. Springer Berlin / Heidelberg, 2002.

[31] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1:166–190, 1991.

[32] K. Maier. Exact and Heuristic Approaches for the Directed Circular Facility Layout Problem. Master's thesis, Alpen-Adria-Universität Klagenfurt, Austria, 2018.

[33] T. Öncan and I. K. Altınel. Exact solution procedures for the balanced unidirectional cyclic layout problem. *European Journal of Operational Research*, 189 (3):609–623, 2008.

[34] F. Ozcelik and A. Islier. Unidirectional loop layout problem with balanced flow. In M. Ali and R. Dapoigny, editors, *Advances in Applied Artificial Intelligence*, volume 4031 of *Lecture Notes in Computer Science*, pages 741–749. Springer Berlin / Heidelberg, 2006.

[35] J.-C. Picard and M. Queyranne. On the One-Dimensional Space Allocation Problem. *Operations Research*, 29(2):371–391, 1981.

[36] J. Qin, Y. Liu, and R. Grosvenor. A categorical framework of manufacturing for industry 4.0 and beyond. *Procedia CIRP*, 52:173 – 178, 2016.

[37] H. Samarghandi and K. Eshghi. An efficient tabu algorithm for the single row facility layout problem. *European Journal of Operational Research*, 205(1):98 – 105, 2010.

[38] D. M. Simmons. One-dimensional space allocation: An ordering algorithm. *Operations Research*, 17:812–826, 1969.

[39] J. Suryanarayanan, B. Golden, and Q. Wang. A new heuristic for the linear placement problem. *Computers & Operations Research*, 18(3):255–262, 1991.

[40] A. W. Tucker. On directed graphs and integer programs. Technical report, IBM Mathematical Research Project, 1960.

[41] U. Wemmerlov and D. Johnson. Cellular manufacturing at 46 user plants: Implementation experiences and performance improvements. *International Journal of Production Research*, 35(1):29–49, 1997.

[42] D. H. Younger. Minimum feedback arc sets for a directed graph. *IEEE Transactions on Circuit Theory*, 10(2):238–245, 1963.

[43] Z. Zhang and C. C. Murray. A corrected formulation for the double row layout problem. *International Journal of Production Research*, 50(15):4220–4223, 2012.