ARTUR GIL DE VASCONCELLOS

TWO-DIMENSIONAL CUTTING AND PACKING PROBLEMS WITH TETRIS-LIKE ITEMS

São Paulo
2022

ARTUR GIL DE VASCONCELLOS

TWO-DIMENSIONAL CUTTING AND PACKING PROBLEMS WITH TETRIS-LIKE ITEMS

Graduation Thesis presented to the Polytechnic School of the University of São Paulo to obtain the Degree of Production Engineering.

São Paulo
2022

ARTUR GIL DE VASCONCELLOS

TWO-DIMENSIONAL CUTTING AND PACKING PROBLEMS WITH TETRIS-LIKE ITEMS

Graduation Thesis presented to the Polytechnic School of the University of São Paulo to obtain the Degree of Production Engineering.

Advisor:
Leonardo Junqueira

São Paulo
2022

To my grandparents, who could not witness my graduation and will always be remembered with love

ACKNOWLEDGMENTS

I would like to thank the São Paulo Research Foundation (FAPESP, grant 2018/11877-4) for assisting the research that was used as the basis of this thesis with a generous scholarship.

I thank my supervisor Professor Leonardo Junqueira, who has supported me since years before this thesis, even with matters unrelated to this research, and with whom I could always speak openly no matter the circumstances.

I thank all the Professors and assistants that have shown me again and again the joy in the quest for knowledge, and that were essential for my graduation.

I thank my family, that has supported me in every imaginable way in my pursuit for my dreams, and that has made possible everything that I have ever achieved.

I thank all the friends that I've made during this journey for being by my side in the best and worst moments of my life, and for teaching me things about myself that I could never have known otherwise, making me a better person.

From the bottom of my heart, thank you.

"Knowledge is power. Information is liberating. Education is the premise of progress, in every society, in every family."

-- Kofi Annan

ABSTRACT

Cutting and packing problems is an area in Operations Research that has been studied extensively over the years. These problems have many real-world applications and, therefore, can have many different defining characteristics. One such characteristic is the shape of the items involved in a cutting and packing problem, with both regular items (e.g., rectangles and circles) and irregular items (e.g., concave polygons) being contemplated in the literature. This graduation thesis concerns the cutting/packing of tetris-like items, popularized from the famous game "Tetris". These items are modelled through two rectangles with flexible dimensions connected by their edges, and they are packed in a large rectangular board with either two fixed dimensions, or one fixed and the other variable. Furthermore, two different kinds of assignments were contemplated: maximizing output value and minimizing input value, with one mixed integer programming model being developed for each of the two. As far as the performed research indicates, the developed models are a novelty in the cutting and packing problems literature. In order to provide a firm basis for the understanding of this thesis and to present the cutting-edge researches in cutting and packing problems, a literature review is performed that analyzes different variants of cutting and packing problems. After that, the developed models for the maximization and minimization versions of the problem are presented and explained in detail, and an adaptation of the normal patterns variable generation is presented and explained as well. The maximization version concerns a situation where the goal is to optimize the placement of the tetris-like pieces in the board, so that the sum of the values of the placed items is maximized. The minimization version concerns a situation where there is a given number of pieces to be placed and the goal is to minimize one of the dimensions of the board while satisfying this constraint. One problem with mixed integer programming models is that they tend to be slow for these types of problems, which are generally NP-complete, and, therefore, two heuristics with a similar logic are also developed and presented: one relax-and-fix and one fix-and-optimize heuristic. The models and the heuristics were then implemented and tested using Python and Gurobi to reach solutions for each of the generated pseudo-random instances. The results show that the models are effective and feasible for representing tetris-like items and that the adapted normal patterns have a significant positive impact in the solver's performance. Furthermore, the developed relax-and-fix heuristic is effective and efficient for the maximization problem, and is able to provide sub-optimal initial solutions for the minimization problem.

Keywords – Operations Research, Cutting and Packing Problems, Tetris-like Items, Rectangular Items.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

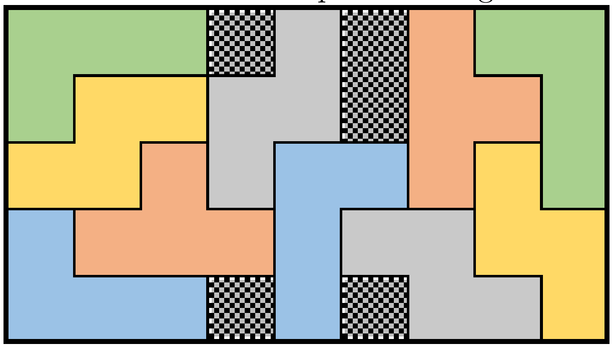
Cutting and packing problems can be broadly divided into cutting problems, which consist of cutting larger units into smaller units, and packing problems, which consist of packing smaller units into larger units. The larger units are typically called objects, and the smaller units are typically called items. Both problems aim at optimizing certain goals (WÄSCHER; HAUSSNER; SCHUMANN, 2007). Examples of cutting problems applications appear in cutting steel bars, paper reels, cardboard boxes, metal or wood sheets, fabric rolls, glass plates, etc. Examples of packing problems applications appear in loading boxes onto pallets or into containers and trucks.

Although in practice they are two very distinct problems, from a mathematical point of view it does not matter whether the pattern obtained for a given set of units is interpreted as being a cutting pattern or a packing pattern, which implies the existence of a "duality" between cutting problems and packing problems, that is, the duality between cutting material/packing space and cutting space/packing material. Since the early 1990s, in addition to these two classes of problems, other related or integrated problems with similar logical structure have been treated as cutting and packing problems (JUNQUEIRA, 2009).

The object of study of this thesis consists of a variant of two-dimensional cutting and packing problems involving tetris-like items, for example, with "L", "J", "T", "S" and "Z" shapes, among other. In these problems, items of these types must be arranged without overlapping and inside a rectangular board, and the objective may consist of maximizing the occupied area of the board (i.e., where not necessarily all available items are used) or minimizing one of the dimensions of the board (i.e., where necessarily all available items are used).

The following Figure 1 illustrates a solution (cutting/packing pattern) for an example involving tetris-like items. In this figure, the hatched areas represent unoccupied parts of the board. It is important to state that the items addressed in this thesis are not restricted to tetrominoes, which consist of four identical squares connected by their edges, as originally considered in the game Tetris (TETRIS, 2022), but to more general items that have the shapes of tetrominoes, but not necessarily their proportions.

Figure 1: Solution to an Example Involving Tetris-like Items



Source: The Author

According to the typology of Wäscher, Haußner & Schumann (2007), these problems could be classified as 2D-R-IIPP/SLOPP/SKP, based on the first objective, and as 2D-R-ODP, based on the second objective, and assuming that there is a certain regularity in the form of the items. If it is understood that this regularity does not exist, then these problems could be classified as 2D-I-IIPP/SLOPP/SKP and as 2D-I-ODP, respectively (BENNELL; OLIVEIRA, 2008). In other words, the consideration of tetris-like items makes the problems addressed in this thesis not fit perfectly into the typology proposed by Wäscher, Haußner & Schumann (2007), so they would be better classified as a variant of two-dimensional cutting and packing problems.

Two-dimensional cutting and packing problems with tetris-like items can occur in many specific sub-sectors of the industry (e.g., textiles, apparel, footwear, metalworking, coatings, etc.) where the items to be cut or packed have shapes that can be approximated by shapes like those in Figure 1. Some real-world applications might include cutting fabrics and leather, stamping metal sheets, cutting ceramic coatings, designing printed circuit boards, and layout of magazines, newspapers, and web pages.

As a variant of two-dimensional cutting and packing problems, the problems considered in this thesis are combinatorial optimization problems that are difficult to solve exactly (LODI; MARTELLO; MONACI, 2002; SILVA; OLIVEIRA; WÄSCHER, 2014), and, to the best of our knowledge, there are no works in the literature that have treated these problems and proposed mixed integer linear programming models to describe them.

1.1 Objectives and Contributions of this Thesis

The objectives of this graduation thesis are the following:

I To study the broad class of Cutting and Packing Problems, focusing on a specific combination of sub-problems and variants;

II To build two mathematical models with different objective functions capable of contemplating the cutting/packing of tetris-like items in a large rectangular board;

III To implement the models in a computational language in order to solve the problem and analyze the performance of the models for different instances with different characteristics;

IV To implement heuristics based on mathematical programming models in order to solve the problem and analyze the performance of the heuristics compared to the monolithic models.

As for the main contributions of this thesis, they are the following:

I Building two novel and original models for the cutting/packing of tetris-like items modeled by a pair of connected rectangles, to be placed on a large rectangular board with either both dimensions fixed or one variable dimension;

II Providing faster heuristic alternatives to the monolithic models capable of reaching solutions with similar qualities.

1.2 Structure of this Thesis

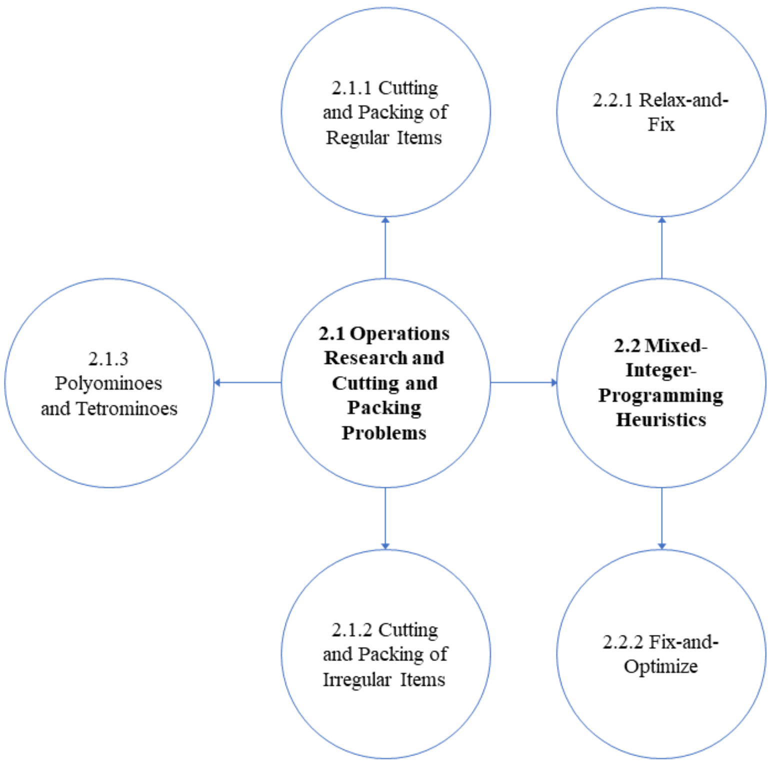
- **Chapter 1 - Introduction:** Presentation of the thesis' theme, its motivations, objectives, contributions and structure;
- **Chapter 2 - Literature Review:** Review of studies which served as the basis for the mathematical model, covering concepts such as regular and irregular items in cutting and packing problems, tetrominoes and mixed integer programming-based heuristics;
- **Chapter 3 - Mathematical Modeling:** Explanation of how the tetris-like items were modelled, presenting the used notation and the models themselves, along with explanatory comments to the models;
- **Chapter 4 - Solution Approach:** Explanation of how the heuristics for the developed models were designed, along with explanatory comments to the heuristics;

- **Chapter 5 - Computational Test:** Tests to evaluate the models and heuristics performance in the generated instances, with results and discussions;
- **Chapter 6 - Future Prospects:** Closure of the thesis, with a summary of the work, its limitations and future research, and final considerations.

2 LITERATURE REVIEW

In this chapter, the most relevant academic subjects explored in this thesis are introduced to provide the knowledge-basis necessary for understanding the conducted research, and to exhibit the state-of-the-art research on these subjects. First, the topics of Operations Research and, more specifically, cutting and packing problems are introduced, followed by two subsets of these problems, one with regular items and the other with irregular items. Then, the subject of tetrominoes and their study in cutting and packing problems is discussed, and, finally, some heuristic methods based on mathematical programming that can be used to generate solutions in cutting and packing problems are analyzed. Figure 2 shows the structure of this chapter and its topics:

Figure 2: Structure for the Literature Review


2.1.1 Cutting and Packing of Regular Items
2.2.1 Relax-and-Fix
2.1.3 Polyominoes and Tetrominoes
2.1 Operations Research and Cutting and Packing Problems
2.2 Mixed-Integer-Programming Heuristics
2.1.2 Cutting and Packing of Irregular Items
2.2.2 Fix-and-Optimize

Source: The Author

2.1 Operations Research and the Class of Cutting and Packing Problems

According to Murthy (2007), Operations Research is a field of mathematics that is useful as a scientific base for management to optimize decisions to their problems. This area of study was developed during the Second World War as time-setting problem for dropping bombs from airplanes onto submarines. The area of focus went on to expand into resource optimization for the military, and, since then, Operations Research has expanded to address many other subjects, such as the industry, economic growth planning, agriculture and traffic control.

This research utilizes mathematical models to describe systems, which are defined as an organization of interdependent components that work together to accomplish a goal (WINSTON, 1997). These models possess three main parts, the objective function, the decision variables, and the constrains. The goal of an Operations Research model is to either maximize or minimize the objective function, which is affected by the decision variables. The values of these variables can be determined by the user of the model, whose main task is choosing these values so that the objective function is either maximized or minimized. The decision variables are described by the model's constrains, which restrict the values of the variables.

Furthermore, the optimization methods for these problems can be classified according to different characteristic of the problem and the model (DUTTA, 2016). One of the possible classifications is based on the structure of the equations and subdivides models into ones that require linear programming (LP) and ones that require non-linear programming (NLP). Simply put, a model in which all the equations are linear, including both the objective function and constrains, requires linear programming. Another possible classification is based on the nature of the decision variables and subdivides models into ones that require continuous optimization and ones that require discrete optimization. If the variables are contained in the real numbers domain and the objective function and constrains are continuous, then the model requires continuous optimization, but if the variables can only possess integer values or are binary, then the model requires discrete optimization. The latter requires integer programming (IP), and in a case where a mix of continuous and discrete decision variables are used, then it requires mixed integer programming (MIP).

The field of Operations Research has different types of problems, and this thesis concerns cutting and packing problems, which have a common structure (WÄSCHER;

HAUSSNER; SCHUMANN, 2007). These problems have two sets of elements: one of large objects; one of small items. Both of them are defined exhaustively in a certain number of geometric dimensions, and subsets of the small items must lie entirely within its designated large object without overlaps. Furthermore, a single- or multi-dimensional objective function is optimized and a solution to the problem may result in using either all or a portion of the large objects and small items. In order to achieve a global optimum, five sub-problems must be simultaneously solved:

1. Selection problem regarding the large objects;
2. Selection problem regarding the small items;
3. Grouping problem regarding the selected small items;
4. Allocation problem regarding the assignment of the subsets of small items to the large objects;
5. Layout problems regarding the arrangement of the small items on each of the selected large objects with respect to the geometric condition.

Although cutting and packing are two very distinct problems in practice, from a mathematical point of view it does not matter whether the pattern obtained for a given set of units is interpreted as being a cutting pattern or a packing pattern, which implies the existence of a "duality" between cutting problems and packing problems, that is, the duality between cutting material/packing space and cutting space/packing material. Since the early 1990s, in addition to these two classes of problems, other related or integrated problems with similar logical structure have been treated as cutting and packing problems (JUNQUEIRA, 2009).

Although the structure is the same for all cutting and packing problems, these models can be further subdivided into categories. Wäscher, Haußner & Schumann (2007) propose a new categorization rooted on a previous one developed by Dyckhoff (1990). In this new typology, five criteria are used to describe the problems and categorize them, and all problems where the assumptions in the criteria are broken are treated as variants. In the following are the five criteria explained:

1. **Dimensionality:** problems are distinguished between one-, two-, and three-dimensional ones. In the literature, there also cutting and packing problems with four geometric dimensions or more (LINS; LINS; MORABITO, 2002), however, in this typology they are considered variants of lower-dimensional problems;

2. **Kind of assignment:** two basic situations are proposed, one where the output value is maximized, and another where the input value is minimized. In both assignments, a set of small items must be assigned to a set of large objects. In the maximization version, the former is large enough that the latter is not sufficient to contain all of the small items, removing the sub-problem of selecting large objects. In this version, a subset of small items with maximal output value has to be selected and assigned to the large objects. In the second kind of assignment, the set of large objects is sufficient to accommodate all of the small items, removing the sub-problem of selecting them. In this version, a subset of large objects with minimal input value has to be selected, and every small item must be assigned to them;

3. **Assortment of small items:** three cases are considered, the first with identical small items, the second with a weakly heterogeneous assortment of small items, and the third with a strongly heterogeneous assortment of small items. In the first case, the shape and size of the items is the same, possessing the same length, width and height depending on the dimensionality, and the demand for the items is infinite. In the second case, there only a few classes of small items in relation to the total number of items for which the components have the same shape and size. Furthermore, the demand for each class of items is either unlimited or relatively large in this case. Finally, the third case contains only very few elements with the same shape and size, and the demand for each type of item is one. In this typology, items with the same shape and size but different orientations are considered different between one another, and it is assumed that the demands for the items are uniform;

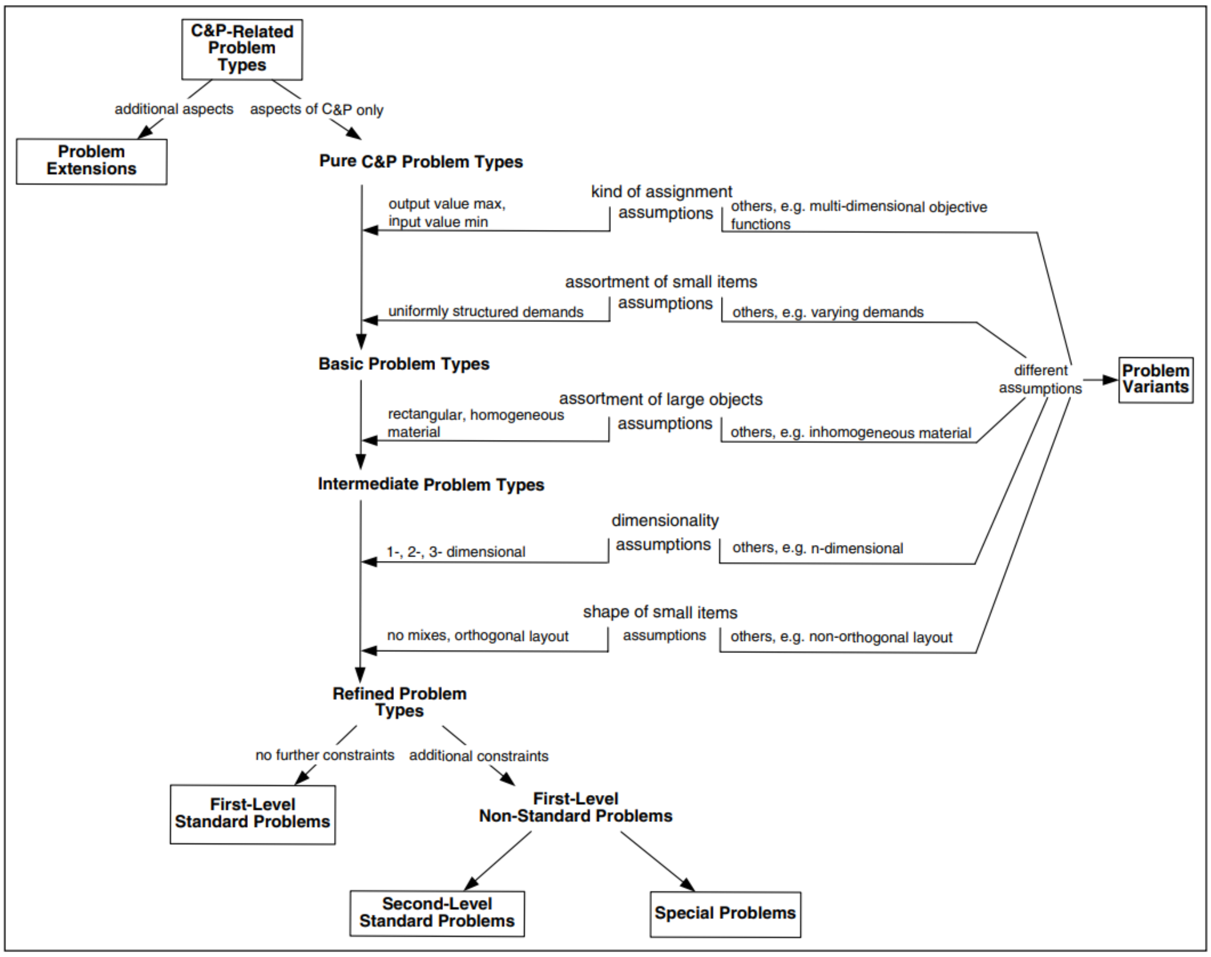
4. **Assortment of large objects:** for this criteria, only two possibilities are considered, one with only one large object, and another with several large objects. For the first possibility, the singular large object can have its dimensions fixed, possessing a predetermined shape and size, or one or more of its dimensions as variable. For the second possibility, it is assumed that all large objects dimensions are fixed, and the criteria is further divided into identical large objects, weakly heterogeneous assortment of large objects and strongly heterogeneous assortment of large objects. Furthermore, this typology only considers rectangular large objects, treating problems with non-rectangular ones as variants;

5. **Shape of small items:** this criteria is only sensible in a two- or three-dimensional problem, and small items can be distinguished between regular (rectangles, circles, boxes, cylinders, balls, etc.) and irregular. The precise definition of regular and irregular is missing from Wäscher, Haußner & Schumann (2007), but a definition

can be found in Bennell & Oliveira (2009), where it is stated that a piece is irregular if it requires a minimum of three parameters to identify it. For example, a rectangle and a circle are regular since they can be described only by length and width and by radius, respectively, whereas a trapezoid cannot be described so simply. It is assumed that the rectangular items are only laid out orthogonally in this typology.

An overview of the categorization structure can be seen in Figure 3, in which the combination of criteria relevant to the problem determines its level of complexity as either "basic", "intermediate" or "refined". Moreover, problems variants are also considered on the right side of the figure, and they emerge when a assumption in a criteria is broken, such as the one mentioned for the dimensionality criteria, or when a uniform distribution of demands is not the case such as in Riehme, Scheithauer & Terno (1996).

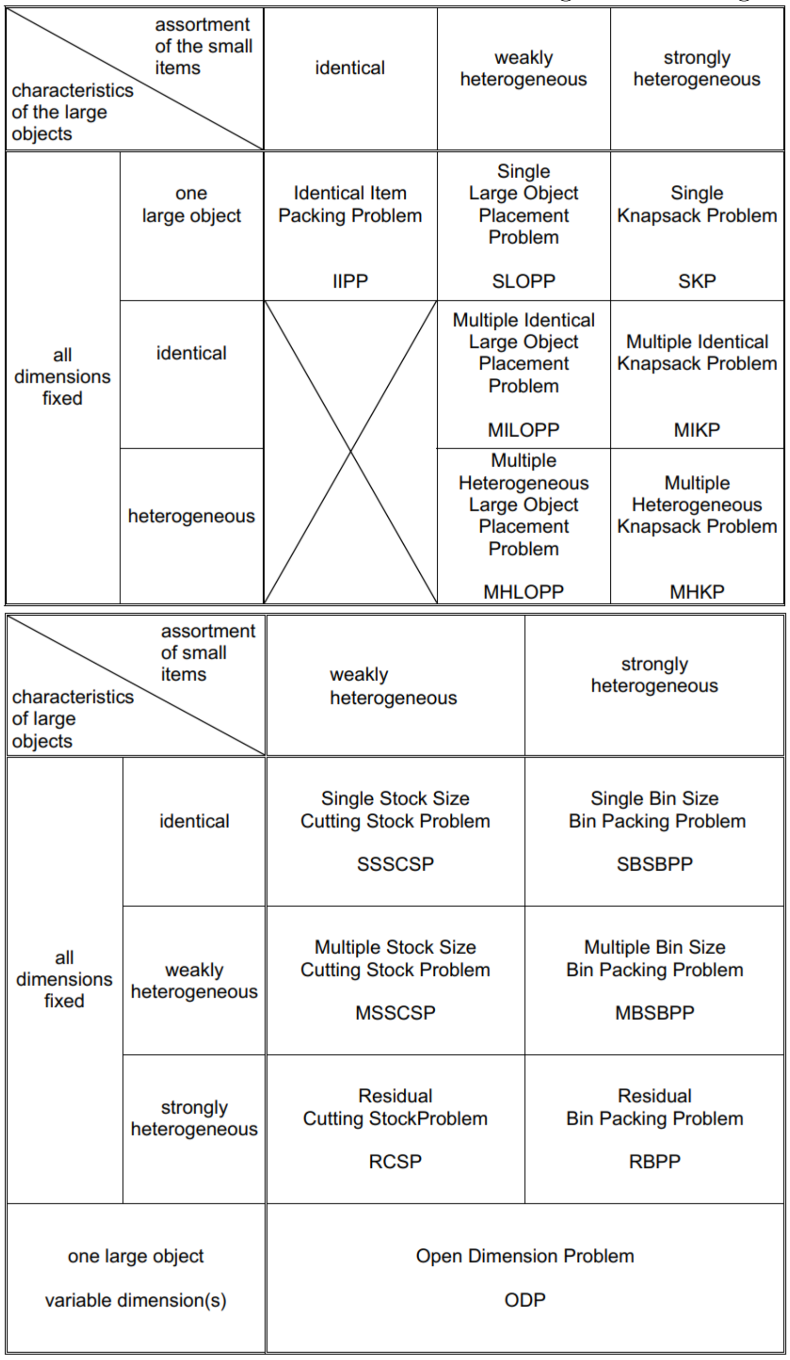
Figure 3: Categorization of Cutting and Packing Problems according to the Five Criteria


C&P-Related Problem Types
additional aspects
aspects of C&P only
Problem Extensions
Pure C&P Problem Types
kind of assignment
output value max, input value min
assumptions
others, e.g. multi-dimensional objective functions
assortment of small items
uniformly structured demands
assumptions
others, e.g. varying demands
Basic Problem Types
different assumptions
Problem Variants
assortment of large objects
rectangular, homogeneous material
assumptions
others, e.g. inhomogeneous material
Intermediate Problem Types
dimensionality
1-, 2-, 3- dimensional
assumptions
others, e.g. n-dimensional
shape of small items
no mixes, orthogonal layout
assumptions
others, e.g. non-orthogonal layout
Refined Problem Types
no further constraints
additional constraints
First-Level Standard Problems
First-Level Non-Standard Problems
Second-Level Standard Problems
Special Problems

Source: Wäscher, Haußner & Schumann (2007)

With the five criteria it is possible to create subcategories of cutting and packing problems that have different combinations of criteria characteristics. Figure 4 contains the different nomenclatures given for intermediate problems, which can be further differentiated based on dimensionality and shape of small items to categorize the refined problems. These names are taken from the usual nomenclature in the literature, and in Figure 4 the upper image contains the problems with output maximization, and the lower image contains ones with input minimization.

Figure 4: Nomenclature of Intermediate Cutting and Packing Problems



characteristics of the large objects \ assortment of the small items		identical	weakly heterogeneous	strongly heterogeneous
all dimensions fixed	one large object	Identical Item Packing Problem IIPP	Single Large Object Placement Problem SLOPP	Single Knapsack Problem SKP
	identical		Multiple Identical Large Object Placement Problem MILOPP	Multiple Identical Knapsack Problem MIKP
	heterogeneous		Multiple Heterogeneous Large Object Placement Problem MHLOPP	Multiple Heterogeneous Knapsack Problem MHKP

characteristics of large objects \ assortment of small items		weakly heterogeneous	strongly heterogeneous
all dimensions fixed	identical	Single Stock Size Cutting Stock Problem SSSCSP	Single Bin Size Bin Packing Problem SBSBPP
	weakly heterogeneous	Multiple Stock Size Cutting Stock Problem MSSCSP	Multiple Bin Size Bin Packing Problem MBSBPP
	strongly heterogeneous	Residual Cutting StockProblem RCSP	Residual Bin Packing Problem RBPP
one large object variable dimension(s)		Open Dimension Problem ODP	

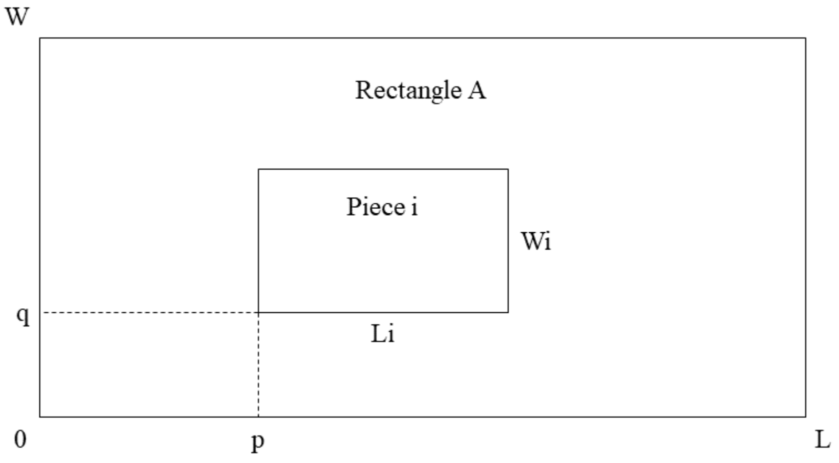
Source: Wäscher, Haußner & Schumann (2007)

2.1.1 Cutting and Packing of Regular Items

In the cutting and packing literature, the most common assessed objects are rectangles, and this research has many applications that have been useful in a number of industry fields, such as in chip design, integration of circuits, aerospace, apparel, platform layout and chemicals (WU et al., 2021). Furthermore, these problems have been proven to be NP-complete (FOWLER; PATERSON; TANIMOTO, 1981), which means that they cannot be solved by polynomial-time algorithms.

One of the most traditional models for this type of problems comes from Beasley (1985). In this research, a mixed integer programming model for placing small rectangles inside a large one is developed. This problem can be classified as a two-dimensional rectangular problem, where the large object has fixed length and width and the objective is output value maximization. In this model, the large rectangle's positions in the x- and y-axes are discretized in order to avoid the continuous nature of distances and formulate a model with finite coordinates. With these integer coordinates, binary variables that indicate whether a piece is placed with its lower left vertex in a certain position are formulated. Furthermore, in the original paper the author also discussed possible extensions of the model, contemplating multiple connected large rectangles and the so-called defects, which prevent certain coordinates of having pieces placed in them. Figure 5 shows an example of a piece placed according to this model, where L and W represent the length and width of the large rectangle/object, respectively, L_i and W_i represent the length and width of the small placed rectangle/item, respectively, and p and q represent the integer coordinates of the bottom left corner of the placed rectangle:

Figure 5: Example of Beasley's Model



Source: Adapted from Beasley (1985)

The original model by Beasley (1985) has been further extended by Junqueira (2009), a research in which the author contemplates three-dimensional problems and explores real-world restrictions that required modifications to the model to be contemplated. The most important contemplated extension for this thesis was the multiple orientations of the placed boxes, since it is assumed in the original model that the items are orthogonally placed. For this to work, the author proposes that either each orientation of the boxes should be considered a different item, or an additional index should be added to the original binary variables with all the possible orientations of the boxes.

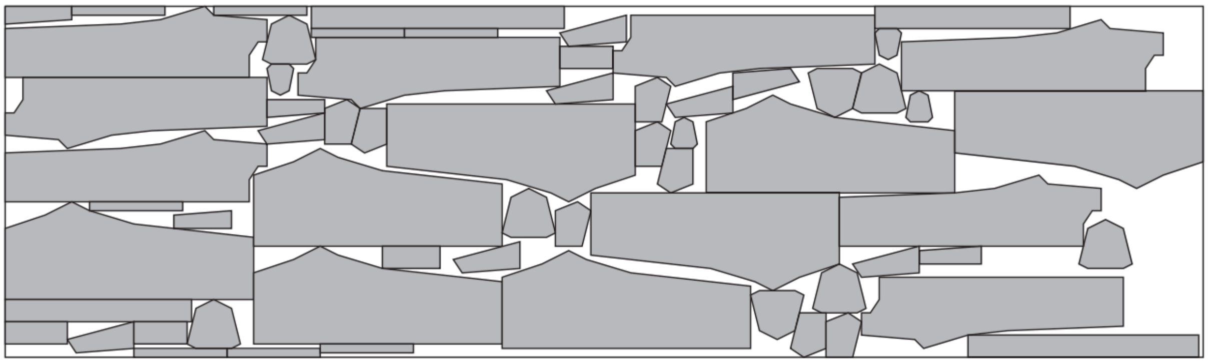
These extensions for real-world conditions are common in the rectangular packing problems, and the main ones are, according to Júnior et al. (2022): weight distribution (QUEIROZ; MIYAZAWA, 2014); load bearing (QUEIROZ; MIYAZAWA, 2013); level packing (BETTINELLI; CESELLI; RIGHINI, 2008; BEKRAR et al., 2010; BEZERRA et al., 2020); positioning (SILVEIRA; MIYAZAWA; XAVIER, 2013; SILVEIRA; XAVIER; MIYAZAWA, 2014; KENMOCHI et al., 2009); packing (FEKETE; KAMPHANS; SCHWEER, 2014); complexity (KIERKOSZ; LUCZAK, 2014; SALTO; ALBA; MOLINA, 2008; SUGI et al., 2020); cutting process (HAWA; LEWIS; THOMPSON, 2018; RINALDI; FRANZ, 2007).

Besides rectangles, there is also a smaller literature for other regular items such as circles, which can be useful in fiber optic cable manufacturing, container loading, cylinder packing, dashboard layout (LAI et al., 2022) and in the woodworking industry (HINOSTROZA; PRADENAS; PARADA, 2013; SILVA et al., 2022).

2.1.2 Cutting and Packing of Irregular Items

The history of cutting and packing problems with irregular items extends over forty years, and is one of the variants of the original cutting and packing problems to be widely researched in the Operations Research field (BENNELL; OLIVEIRA, 2009). These variants are also called "nesting problem", and have been proven to be NP-complete (FOWLER; PATERSON; TANIMOTO, 1981), similar to the version with regular items (see Section 2.1.1). The applicability of the nesting problems literature can be seen in the clothing industry, furniture, leather, glass, or sheet metal cutting (LEAO et al., 2020). Figure 6 shows an example of a nesting problem in the garment industry, in which a set of sixty four pieces with regular and irregular items is feasibly packed into a rectangular board.

Figure 6: An Example of a Nesting Problem

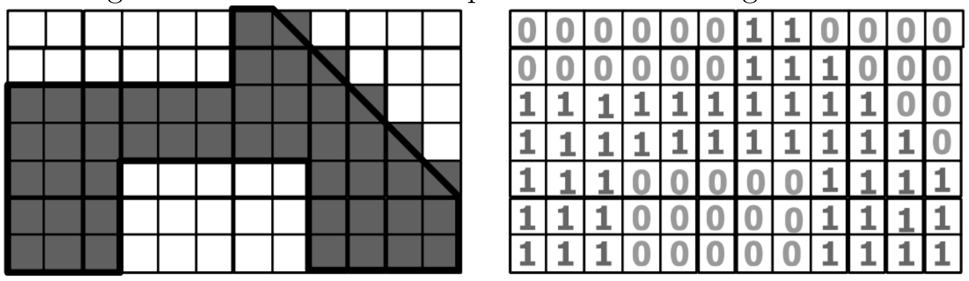


Source: Leao et al. (2020)

Dealing with the geometry of irregularly-shaped items and guaranteeing that they are correctly placed in the large object without overlaps is a much harder task than doing the same for regularly-shaped, and, as a consequence, the results obtained so far in the literature are much more limited regarding the size of the tackled problem (LEAO et al., 2020).

There are several different methods used to contemplate the geometry of irregular items, and a popular and simple one is described by Oliveira & Ferreira (1993), in which a raster method that discretizes the continuous nature of the large object is used. In this model, the large object is a rectangle, and for each discretized integer coordinate of this rectangle a binary variable denotes whether there is a piece in this position or not. Figure 7 shows an example of how the irregular items were represented in the model by Oliveira & Ferreira (1993).

Figure 7: The 0-1 Raster Representation for Irregular Pieces



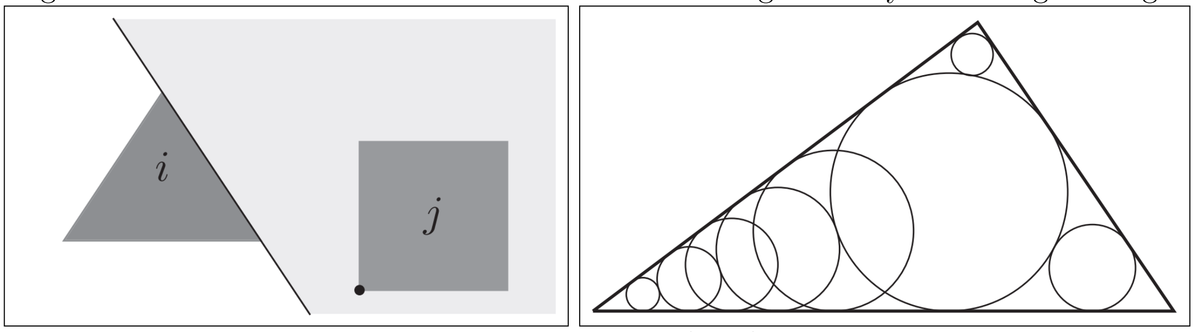
0	0	0	0	0	0	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	0	0	0	0	0	1	1	1	1
1	1	1	0	0	0	0	0	1	1	1	1
1	1	1	0	0	0	0	0	1	1	1	1

Source: Oliveira & Ferreira (1993)

Other raster representations of irregular items were developed in the literature (SEGENREICH; BRAGA, 1986; BABU; BABU, 2001), but this method is not able to accurately represent irregular items with non-orthogonal edges due to the nature of the raster discretization. There are other methods that can represent irregular items with non-orthogonal edges, such as direct trigonometry, no-fit polygon and phi functions, which have been effectively used in nesting problems (LEAO et al., 2020).

The direct trigonometry method has its own set of modelling strategies to describe irregular items, and one of the most common ones is the D-function (LEAO et al., 2020), that uses continuous variables to position the items (SCHEITHAUER; TERNO, 1993; CHERRI et al., 2016; CHERRI; CHERRI; SOLER, 2018). This strategy determines the relative position between a piece's vertex and another piece's edge to avoid overlaps between them, and was derived from the equation of the distance between a point and a line (LEAO et al., 2020). Another direct trigonometry strategy is the inner circle one, where items are defined by a set of inscribing circles (JONES, 2014; ROCHA et al., 2016). Figure 8 shows a visual representation of the two strategies, with the D-function to the left, and the inner circle to the right.

Figure 8: The D-Function and Inner Circles Direct Trigonometry Modelling Strategies



Source: Leao et al. (2020)

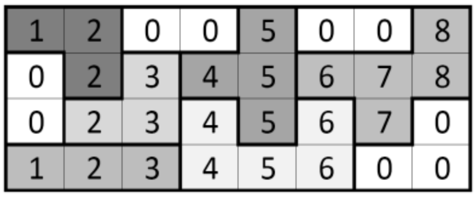
2.1.3 Polyominoes and Tetrominoes

The irregular items that concern this thesis are called "tetrominoes", which were first introduced by Golomb (1954) as a subset of the so-called "polyominoes". A polyomino can be defined as a simply-connected set of n squares which are "rook-wise connected", which means that a rook placed in any square of the n-omino must be able to get to any other square in a finite number of moves. Cutting and packing problems involving polyominoes have real-life applications in different industries. These include sheet metal stamping, design of printed circuits boards, timber cutting and layout of newspaper pages, where the geometries of the shapes involved in these processes are similar to polyominoes (KASHKOUSH; SHALABY; ABDELHAFIEZ, 2012).

The polyominoes study has been historically made by mathematicians in the "tiling" field of study (GOLOMB, 1966; BODINI, 2003; KLARNER, 1969; REID, 1997, 1997), which is tied to Operations Research's cutting and packing problems. It is possible to describe some tiling problems through cutting and packing problems, and some authors, although not many, recently brought the ”polyominoes tiling” topic into the Operations Research literature, where the problems are mainly classified as two-dimensional problems with one large object and irregular small items, and the polyominoes are described through the constrains of the model.

The first author to formalize a Operations Research model for the cutting and packing of polyominoes was Kashkoush, Shalaby & Abdelhafiez (2012). In this research, a mixed integer linear programming model was introduced. The large object was a rectangle with fixed width and variable length that could be subdivided into unit squares. The objective of the model was minimizing the large object’s variable length, though indirectly, by penalizing the placement of polyominoes at higher lengths. A solution produced by this model can be seen in Figure 9, in which the numbers represent the penalties that are summed in the objective function and minimized:

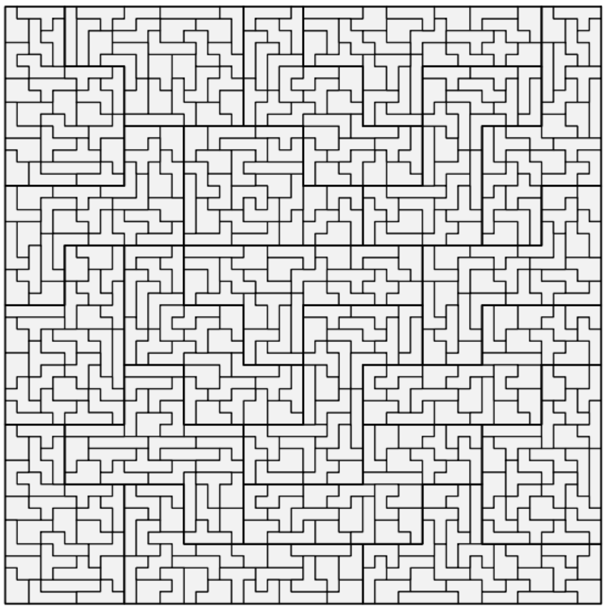
Figure 9: Solution for the Model proposed by Kashkoush, Shalaby & Abdelhafiez (2012)



Source: Kashkoush, Shalaby & Abdelhafiez (2012)

Another author of polyominoes cutting and packing problems was motivated by a new application of polyominoes tiling, the design of phased array antennas (KARADEMIR; PROKOPYEV; MAILLOUX, 2016). In this problem, the large object is a rectangle with fixed length and width, and this author modeled the polyominoes through a completely different logic, using a mixed integer non-linear programming model, which was then linearized. A solution produced by this model can be seen in Figure 9:

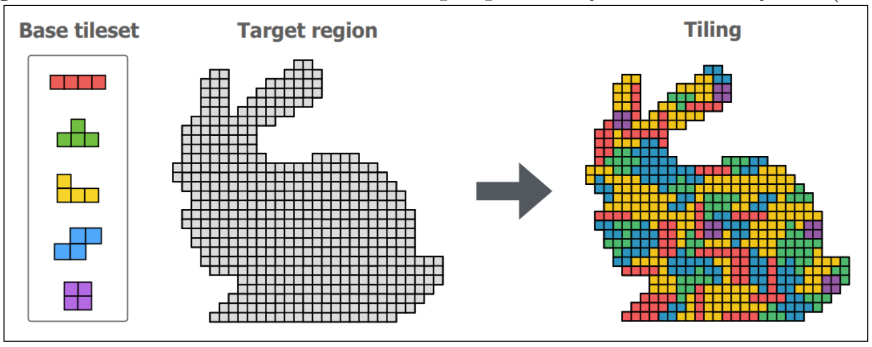
Figure 10: Solution for the Model proposed by Karademir, Prokopyev & Mailloux (2016)



Source: Karademir, Prokopyev & Mailloux (2016)

Another author researches the tiling of polyominoes in an irregular large object that is composed of "rook-connected" squares, maximizing the large object occupation by assigning "weights" for each polyomino and maximizing their sum (KITA; MIYATA, 2021). The utilized model uses mixed integer programming, and the purpose of this research was generating polyominoes puzzles, which is a theme seeing in other literature for these objects (LO; FU; LI, 2009). A solution produced by this model can be seen in Figure 9:

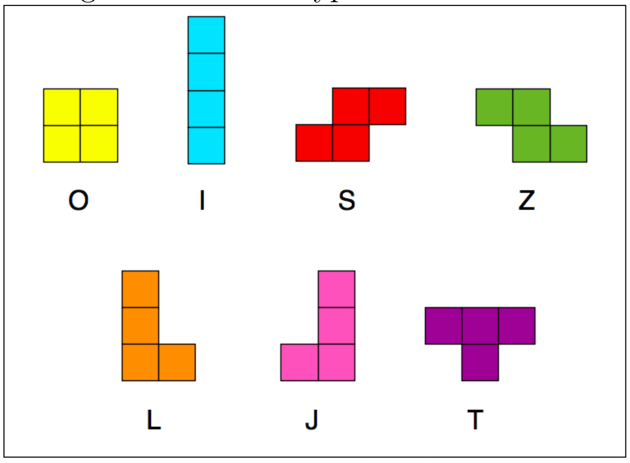
Figure 11: Solution for the Model proposed by Kita & Miyata (2021)



Source: Kita & Miyata (2021)

The polyominoes with four squares are defined as "tetrominoes", and they can come in seven different shapes considering the possible reflections of a tetromino as different shapes, and the possible rotations as the same shape. These seven types of tetrominoes can be seen in Figure 12:

Figure 12: Seven Types of Tetrominoes

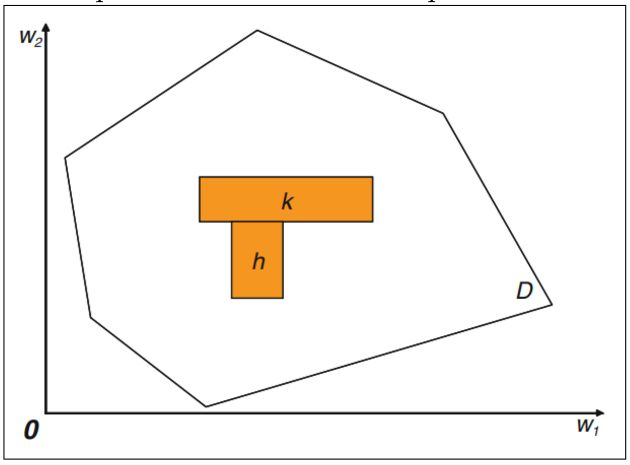


Source: Liu (2017)

These items are famous due to the popular game "Tetris" (TETRIS, 2022), and each shape has a different coined name due to the letters that they resemble: "O", "I", "S", "Z", "L", "J" and "T". These items are the main focus of this thesis, and the study of two-dimensional cutting and packing problems with tetris-like items can be useful in the arrangement of cargo in vehicles and space modules (FASANO, 2013), and for dealing with particular cases in the related area of project scheduling problems in which activities may consume different amounts of a given resource throughout their execution (HARTMANN, 2000). Furthermore, every application that pertains polyominoes also pertains tetrominoes, except for instances where specific n-ominoes with $n \neq 4$ are at issue.

The literature regarding tetrominoes specifically is even smaller than the one pertaining polyominoes, and, similarly to the polyominoes one, the problems are mainly classified as two-dimensional problems with one large object and irregular small items, and the tetrominoes are described through the constrains of the model. The most relevant research in the subject was made by Fasano (2013), where the author defines "tetris-like" items as a cluster of mutually orthogonal parallelepipeds. This definition of a tetris-like item makes the classification of the items as regular or irregular difficult, and this in turn allows the classification of problems containing these items as either extensions, or variations of the problems described in Wäscher, Haußner & Schumann (2007). In the research that coined this term, the placement of tetris-like items in a convex large object was modeled through a mixed integer programming model. One example of a tetris-like item placed in a convex large object can be seen in Figure 13, in which k and h represent the item's parallelepipeds:

Figure 13: Example of a Tetris-like Item placed in a Convex Space



Source: Fasano (2013)

2.2 MIP Heuristics

An heuristic can be defined as a procedure with a collection of rules or steps that guides one to a solution that may or may not be the optimal one (LAGUNA; MARTÍ, 2013). The solution set of the majority of real world optimization problems is often large or infinite, and a heuristic can assist one in finding acceptable solutions to a problem.

This thesis concerns heuristics used in Operations Research models that require mixed integer programming to be solved (see Section 2.1). These types of problems belong to the set of NP-complete problems, since they are combinatorial in nature, and this means that the solution time scales exponentially as the problem size increases in the worst-case scenario (FLOUDAS; LIN, 2005).

According to Wolsey (1998), there are many reasons that might cause one to employ an MIP heuristic. The first, and perhaps the most important one, is that a solution is required rapidly, what might be even more relevant if the instance is so large that it cannot be formulated as a MIP model of reasonable size. Furthermore, there are certain combinatorial problems that, such as vehicle routing and machine scheduling, for which it is easy to find feasible solutions by analyzing the model's structure.

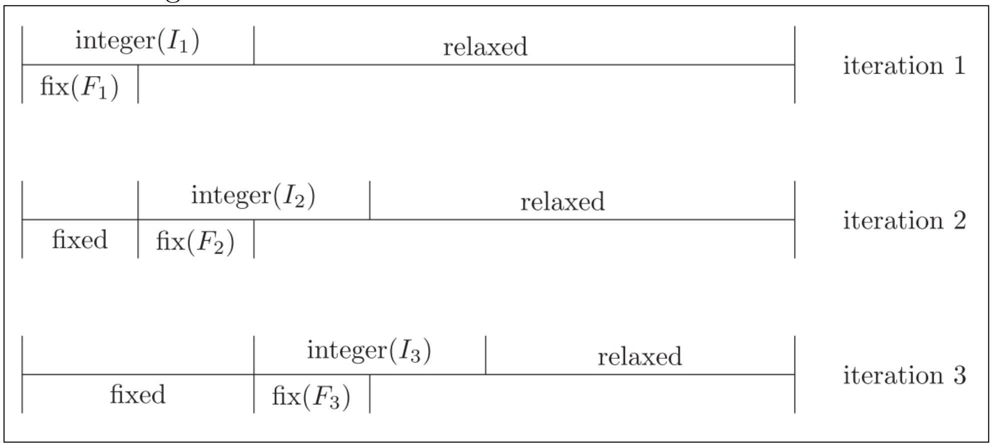
Given the wide range of applications of MIP in real-world problems, it is only natural that heuristics that are faster than the optimization procedure would emerge. There are many of these procedures in the MIP literature (BALAS et al., 2001; SALTZMAN; HILLIER, 1992; GLOVER; LØKKETANGEN; WOODRUFF, 2000; BALAS; MARTIN, 1980), but this thesis concerns only the relax-and-fix heuristic, introduced and detailed in the next section.

2.2.1 Relax-and-Fix

The MIP relax-and-fix heuristic was first introduced by Dillenberger et al. (1994) for a lot-sizing problem, where the goal was to optimize the production volumes of various part types across different machine groups and in different time periods. As stated in the aforementioned research, the relax-and-fix algorithm does not consider the integrality of all binary decision variables at once, but successively. This heuristic consists basically in considering iteratively the integrality of some subset of binary variables and fixing their values at the end of an iteration.

Figure 14 can be used to better visualize the procedure of the heuristic. In this image, the horizontal line represent the set of binary variables present in a MIP model, and this set is divided into four others. The "integer" subset is represented by I_k in iteration k and contains the originally binary variables which will have their integrality considered, and, therefore, will be set as binary during the iteration. The "fix" subset is represented by F_k and contains the variables which will have their values fixed at the end of iteration k with the results generated from the iteration. The "fixed" subset contains the values of binary variables which were fixed in previous iterations and will remain so until the last iteration of the heuristic. Finally, the "relaxed" subset contains the originally binary variables which will not have their integrality considered, and, therefore, will be set as continuous during the iteration. At the end of the heuristic, the subset of "fixed" variables coincides with the complete set of originally binary variables, and a solution for the problem is obtained (ABSI; HEUVEL, 2019).

Figure 14: Illustration of the Relax-and-Fix Heuristic



Source: Absi & Heuvel (2019)

This heuristic is used very frequently in variants of lot-sizing problems (ABSI; Kedad-Sidhoum, 2007; ARAUJO; ARENALES; CLARK, 2007; CLARK, 2003; FEDERGRUEN;

MEISSNER; TZUR, 2007; MERCÉ; FONTAN, 2003; STADTLER, 2003; AKARTUNALI; MILLER, 2009), where the topic concerns production planning across time periods and the subsets of binary variables are usually based on these different timeframes. It is always seen in the literature for other problems that also concern scheduling, time-dependent tasks, such as the traveling umpire problem (OLIVEIRA; SOUZA; YUNES, 2014), where the topic concerns the minimization of distances traveled by umpires in tournaments with several rounds and the subsets of binary variables are divided according to the rounds numbers, and also the maritime inventory routing problem (FRISKE; BURIOL; CAMPONOGARA, 2022), where the topic concerns the optimization of ship deliveries across different ports and the subsets of binary variables are divided according to time, since the schedule for the deliveries is also of concern.

However, the relax-and-fix is also used effectively in other Operations Research problems that do not necessarily take time into account, such as in the grid-based location problem (Noor-E-Alam; DOUCETTE, 2012), where the topic concerns resource allocation across different coordinates in a grid and the subsets of binary variables are divided according to coordinates, and also in the controlled tabular adjustment problem (BAENA; CASTRO; GONZÁLEZ, 2015), where the topic concerns tabular data protection, and the subsets of binary variables are divided according to cells in the table.

Another type of problems in which relax-and-fix heuristics have been applied is the one this thesis concerns, cutting and packing problems. Keeping to the main trend of the heuristic of subdividing the variables based on time periods, Oliveira et al. (2020) describes the utilization of a relax-and-fix heuristic for an extended version of an one-dimensional Single Stock Size Cutting Stock Problem (see Figure 4), where the delivery date is also considered in its formulation.

More interestingly for this thesis, Cherri et al. (2016) describes the utilization of this heuristic for a two-dimensional Open Dimension Problem with irregular items, where the large object is a rectangle with a fixed width and an infinite length, and the model's objective is to minimize this length. In this research, the developed heuristic was a hybrid of well-know heuristics, and relax-and-fix was the first step of the complete developed heuristic, being used to provide an initial solution to the problem. In this version of the relax-and-fix heuristic, the binary variables were subdivided according to the positions of the small items in the large object, and in order to determine the sizes of the generated subsets the authors developed an arbitrary method that depended on the characteristics of the instance.

Finally, no researches were found where a relax-and-fix algorithm was used in a polyomino-related cutting and packing problem.

2.2.2 Fix-and-Optimize

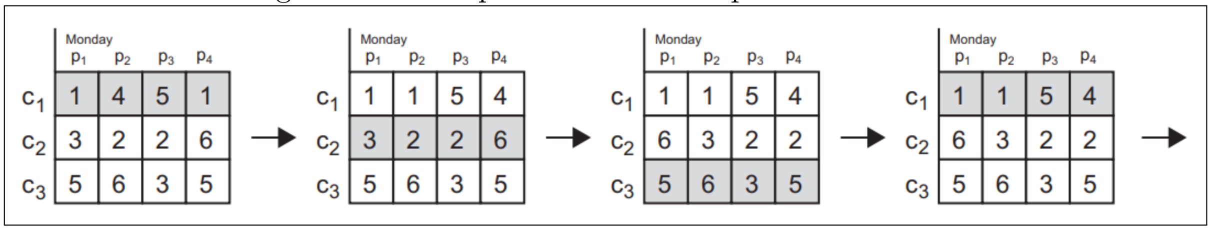
The fix-and-optimize heuristics, also called “exchange” heuristics, are described as an improvement or a variation of the relax-and-fix heuristics (ABSI; HEUVEL, 2019). This heuristic, similarly to the relax-and-fix, consists of an iterative procedure where the set of binary variables is subdivided and each iteration considers the integrality of only a subset of the variables. However, in the fix-and-optimize heuristic, there are only two subsets of variables in each iteration: one that contains the originally binary variables which will have their integrality considered, corresponding to the “integer” subset in the relax-and-fix heuristic (see Figure 14), and another one that contains the values of binary variables which were previously fixed, which is similar to the “fixed” subset in the relax-and-fix heuristic (POCHET; WOLSEY, 2006).

The main difference between the two algorithms is that the fix-and-optimize heuristic requires a previously obtained solution that will be used for the first iterations of the algorithm as the “fixed” subset. This initial solution is obtained through other heuristics, such as the relax-and-fix heuristic, and each iteration of the fix-and-optimize heuristic will remove a subset of the fixed variables and treat them as binary again, transforming them into the “integer” subset. Originally, the iterations stopped once one of them could not improve the result of the objective function, but it is possible to set other termination criteria for the heuristic. Furthermore, the “fix” subset of the relax-and-fix heuristic coincides with the “integer” subset in this case, since all of the variables with their integrality should be fixed at the end of an iteration (POCHET; WOLSEY, 2006).

The fix-and-optimize heuristic is also used in lot-sizing problems, and the criteria for dividing the variables is also dependent on the configuration of the algorithm. Helber & Sahling (2010) describes an application of the fix-and-optimize heuristic for a multi-level capacitated lot-sizing problem, where the authors experiment with three criteria for subdividing the binary variables: product, resource/machine and time period.

This heuristic is also seen in other scheduling-related problems, such as the fleet sizing and replenishment planning problem (DASTJERD; ERTOGRAL, 2019), which concerns routine supply runs for various customers and the subsets of binary variables are divided according to the various customers. It is also seen in the high school timetabling problem (DORNELES; ARAÚJO; BURIOL, 2014), where the topic is the allocation of teachers to classes in different timeframes and the subsets of binary variables are divided according to classes, teachers or days. Figure 15 shows the initial solution and three iterations of the fix-and-optimize heuristic from left to right used in the aforementioned high school timetabling problem research, with the class as the criteria for subdividing variables:

Figure 15: Example of a Fix-and-Optimize Heuristic



Source: Dorneles, Araújo & Buriol (2014)

Finally, no researches were found where a fix-and-optimize algorithm was used in a polyomino-related, or in any sort of cutting and packing problem.

3 MATHEMATICAL MODELING

In this chapter, the developed mathematical models that encompass all characteristics of the tetris-like items packing maximization and minimization problems are discussed. To describe them, two Operations Research cutting and packing formulations (see Section 2.1), each with one objective function and various constraints, were developed. These models are inspired in Beasley (1985), Fasano (2013) and Junqueira (2009) (see Chapter 2).

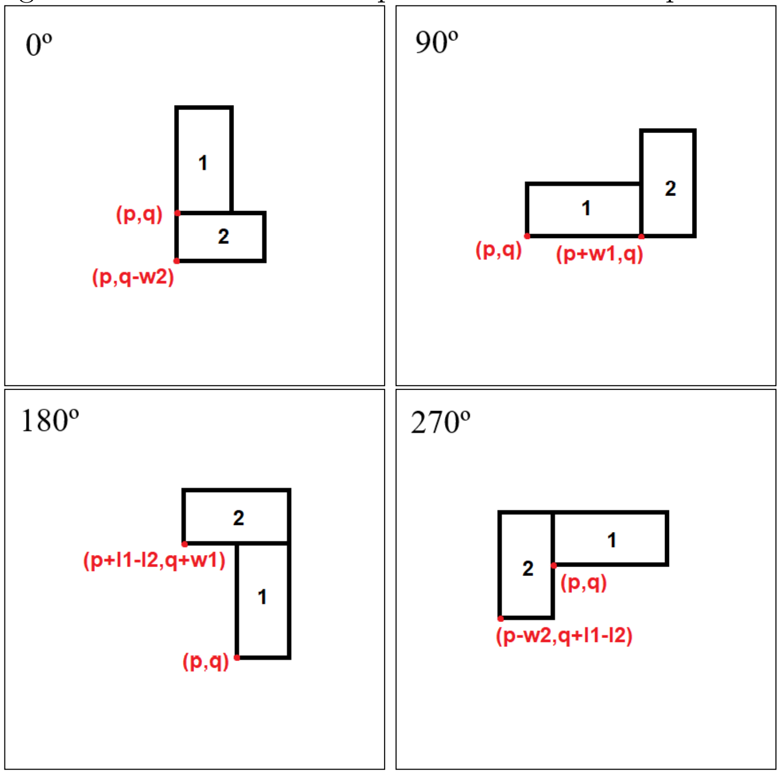
3.1 Mathematical Representation of Tetrominoes

In order to be able to come up with a model that could describe a tetris-like item, firstly it was decided that these items would be modeled with two rectangles. After this, the next logical step is evaluating the positional relationship between the rectangles, which must be determined to create tetris-like items. Furthermore, since the intention is to describe a mathematical model that can rotate the pieces, this evaluation must be performed in every possible rotation. To do this, each item "L", "J", "T", "S" and "Z" (see Figure 12) was analyzed in four counter-clockwise rotations: 0, 90, 180 and 270 degrees. Furthermore, the model should contemplate not only tetrominoes, but all shapes that might resemble the original tetrominoes shapes, the tetris-like items (see Section 2.1.3), in order to be more general and useful in more situations.

This analysis resulted in images for each shape and rotation, where the position of the lower-left corners of the rectangles was analyzed, which was inspired in Beasley (1985), where the formulation of the rectangles follows this logic. Furthermore, for the 0 degrees rotation, the upper rectangle was arbitrarily defined as rectangle 1, and the lower rectangle as 2, with the other rotations following this logic, but rotated. In these images, p describes the position of the lower-left corner of the first rectangle in the x-axis, q describes the position of the lower-left corner of the first rectangle in the y-axis, $l1$ and $w1$ describe the length and width of the first rectangle, and $l2$ and $w2$ of the second rectangle. This was needed to provide a visual basis for the formulation of the model, and the numeration of the rectangles could have been inverted without the loss of generality, not affecting the model.

Figure 16 in the following displays the positional relationship of the L-shaped items in all four rotations in order (0, 90, 180 and 270 degrees), from the upper-left image to the lower-right image. The formula for the position of the lower-left corner of the second rectangle was derived by defining the L-shaped items as items in which the left sides of both rectangles are aligned in the 0 degrees rotation, with the remaining rotations following suit.

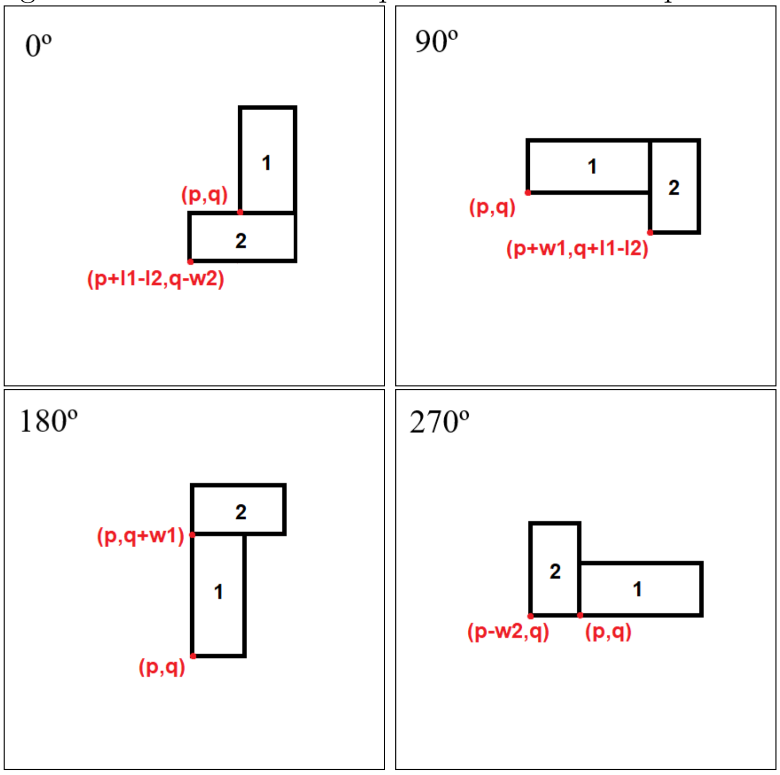
Figure 16: Mathematical Representation of L-shaped Items



Source: The Author

Figure 17 in the following displays the positional relationship of the J-shaped items in all four rotations in order (0, 90, 180 and 270 degrees), from the upper-left image to the lower-right image. The formula for the position of the lower-left corner of the second rectangle was derived by defining the J-shaped items as items in which the right sides of both rectangles are aligned in the 0 degrees rotation, with the remaining rotations following suit.

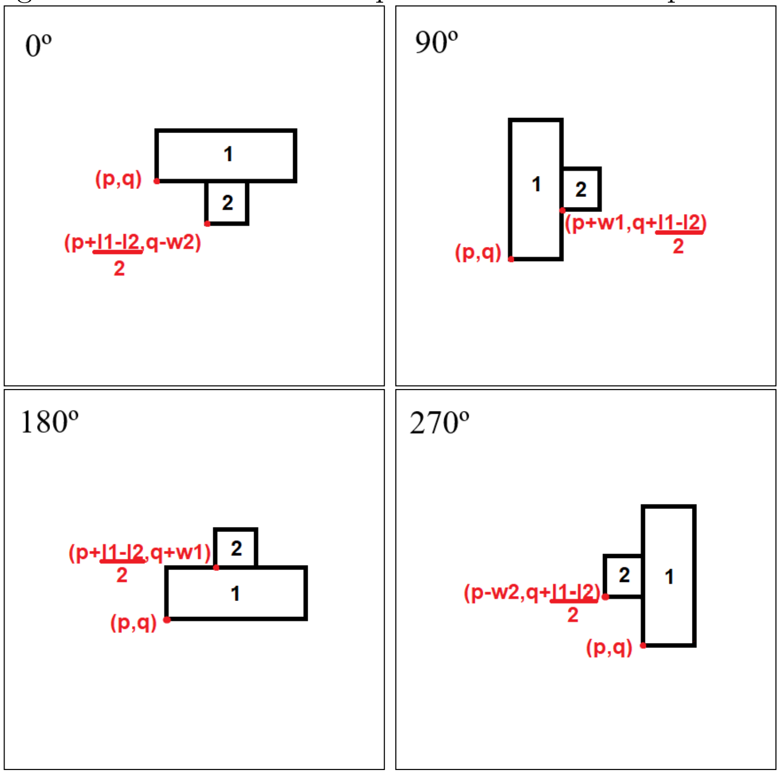
Figure 17: Mathematical Representation of J-shaped Items



Source: The Author

Figure 18 in the following displays the positional relationship of the T-shaped items in all four rotations in order (0, 90, 180 and 270 degrees), from the upper-left image to the lower-right image. The formula for the position of the lower-left corner of the second rectangle was derived by defining the T-shaped items as items in which both rectangles are vertically aligned in the 0 degrees rotation, with the remaining rotations following suit.

Figure 18: Mathematical Representation of T-shaped Items

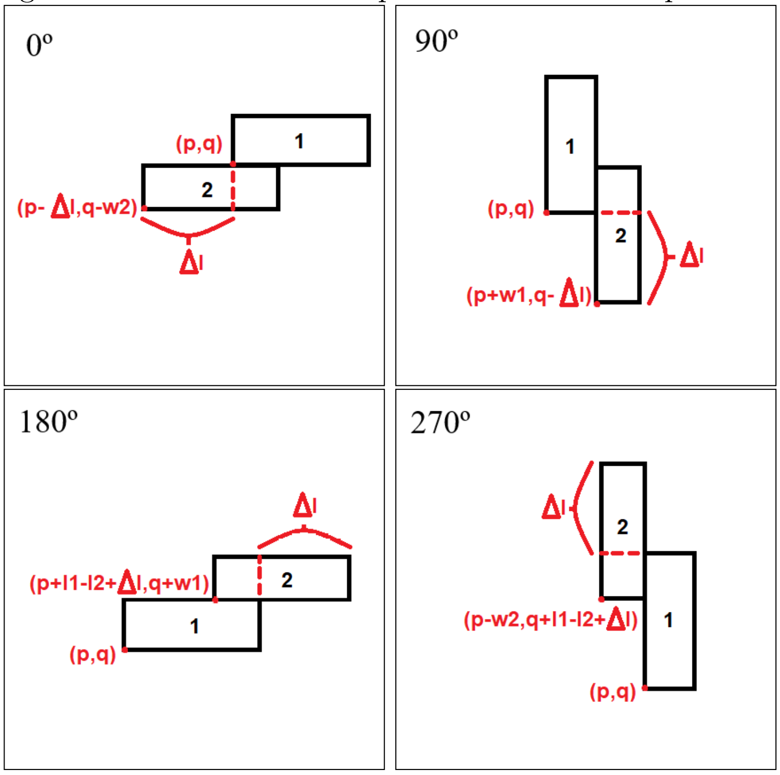


Source: The Author

For the S- and Z-shaped items, another parameter must be analyzed since it is not possible to create formulas that relate the positions of the lower-left corner of the rectangles with only the given parameters. How far the second rectangle is from the first must also be measured, what is described by the parameter Δl. This parameter can be defined as the distance between the left sides of the first and second rectangles in S-shaped items, and between the right sides in Z-shaped items.

Figure 19 in the following displays the positional relationship of the S-shaped items in all four rotations in order (0, 90, 180 and 270 degrees), from the upper-left image to the lower-right image. The formula for the position of the lower-left corner of the second rectangle was derived by defining the S-shaped items as items in which the left sides of both rectangles are distant Δl units in the 0 degrees rotation, with the remaining rotations following suit.

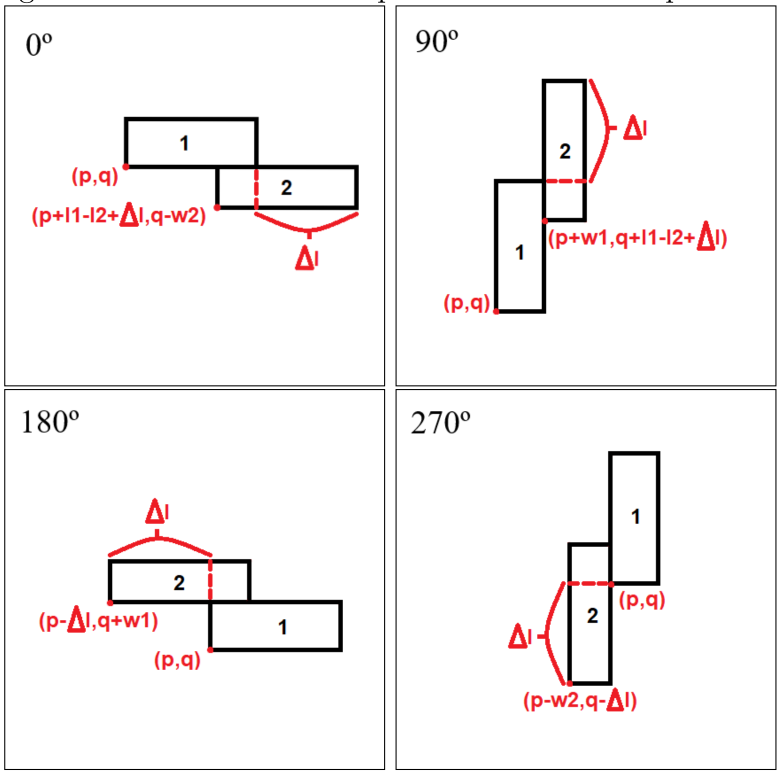
Figure 19: Mathematical Representation of S-shaped Items



Source: The Author

Figure 20 in the following displays the positional relationship of the Z-shaped items in all four rotations in order (0, 90, 180 and 270 degrees), from the upper-left image to the lower-right image. The formula for the position of the lower-left corner of the second rectangle was derived by defining the Z-shaped items as items in which the right sides of both rectangles are distant Δl units in the 0 degrees rotation, with the remaining rotations following suit.

Figure 20: Mathematical Representation of Z-shaped Items



Source: The Author

It is important to notice that the presented positional relationships between the rectangles for each item type are not enough to maintain the intended shape of the item. For example, if the length and width of both rectangles are equal, they would result in rectangles 1 and 2 being squares, and by using the described positional relationships it would not be possible to create a L-, J- or T-shaped item since the result would be a rectangle with the larger side twice the size of the smaller one. Therefore, these relationships are only valid if the dimensions of the items also make sense. One method to guarantee that the items will have the intended shapes is used in this thesis for the generation of the instances (see Section 5.1).

3.2 Developed Models

Following the typology of Wäscher, Haußner & Schumann (2007) (see Section 2.1), the developed models for describing the placement of tetris-like items are two-dimensional. Furthermore, the tetris-like items can not be easily classified as regular/rectangular or irregular in this model due to the chosen modelling logic, which uses two rectangles to form the different tetris shapes. The developed model also allows for generating instances with different assortments of small items, from identical ones, if the instance has only one item with replicas, to strongly heterogeneous items, if it has various. Regarding the assortment of large objects, this model contains one large object, denominated "board", and two different formulations with non-overlapping constraints were formulated to contemplate both kinds of assignments: output maximization and input minimization. In the maximization problem, the board has both dimensions fixed and the objective is to maximize the sum of the value of each item when arranging the tetris-like items (i.e., not necessarily all available items are used). In the minimization problem, the length of the board is fixed (x-axis) and the width is variable (y-axis), and the objective is to minimize the variable dimension of the board when arranging the tetris-like items (i.e., necessarily all available items are used). Given these characteristics, the maximization model can be classified as a variation/extension of a 2D-R-IIPP/SLOPP/SKP problem, depending on the assortment of small items, and the minimization model as a variation/extension of a 2D-R-ODP problem.

The indexes, parameters and variables used in this formulation are shown in the following:

Indexes:

i : index for the item;

f : index for the rectangle number in tetris type items (1 or 2);

g : index for the rotation of the item (1 for 0º, 2 for 90º, 3 for 180º or 4 for 270º);

p, s : indexes for the position in the board of the rectangle's lower-left corner relative to the x-axis;

q, r : indexes for the position in the board of the rectangle's lower-left corner relative to the y-axis.

Sets:

I : set of all items;

I_R : set of rectangular items;

I_L, I_J, I_T, I_S, I_Z : sets of items of types "L", "J", "T", "S", "Z", respectively;

X : set of all possible integer positions on the board length (x-axis);

Y : set of all possible integer positions on the board width (y-axis);

$X_{i,f,g}$: set of possible positions on the board length of rectangle f of item i with orientation g;

$Y_{i,f,g}$: set of possible positions on the board width of rectangle f of item i with orientation g.

Parameters:

v_i : value of item i;

b_i : maximum number of replicas of item i that can be placed inside the board (only in the maximization problem);

B_i : exact number of replicas of item i that must be placed inside the board (only in the minimization problem);

$(l_{i,f}, w_{i,f})$: length and width, respectively, of rectangle f of item i;

$\mathit{\Delta} l_i$: distance in length between the left sides of rectangles 1 and 2 for items of type “S”, and between the right sides for items of type “Z”;

(L, W) : length and width of the board, respectively.

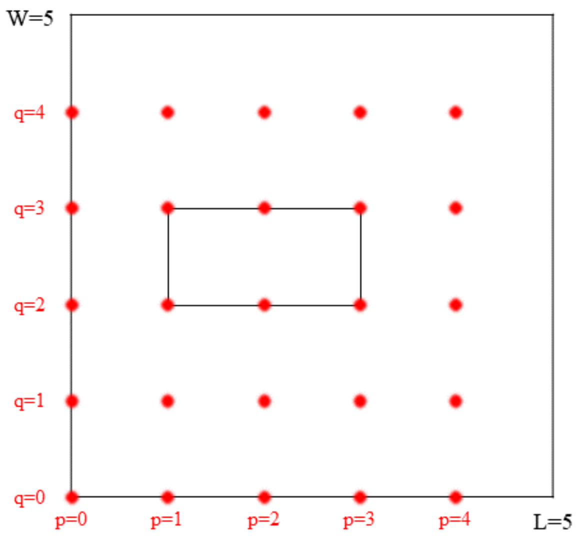
Decision variables:

$x_{i,f,g,p,q}$: binary variable, which equals 1 if rectangle f of item i with orientation g is placed with its lower-left corner at position (p, q), and otherwise equals 0;

$\overline{W}$: non-negative real variable, which corresponds to the variable width of the board (only in the minimization problem).

Additionally, following the inspiration in the models from Beasley (1985), all the dimensions of the small and large items in the model ($l_{i,f}$, $w_{i,f}$, $\mathit{\Delta} l_i$, L and W) have an integer value, as well as the possible positions p and q of the rectangles in the board. This limits the number of p and q values, which would be continuous and infinite otherwise. In the next section, the exact discretization of the board is described, and sets X, Y, $X_{i,f,g}$ and $Y_{i,f,g}$ are defined. Assuming that all possible integer points in the board are used as p and q positions in a board with length $L = 5$ and width $W = 5$, Figure 21 shows an example of a rectangle with no rotation and length 2 and width 1 placed in position $p = 1$ and $q = 2$:

Figure 21: Example of a 5x5 Board with a 2x1 Rectangle



Source: The Author

The constraints to which the models for both the maximization and minimization problems are subject are shown in the following:

$$\begin{aligned}
&\sum_{i\in I}\sum_{f=1}^{2}\sum_{\{p\in X_{i,f,1}|s-l_{i,f}+1\le p\le s\}}\sum_{\{q\in Y_{i,f,1}|t-w_{i,f}+1\le q\le t\}} x_{i,f,1,p,q}+\\
&\sum_{i\in I}\sum_{f=1}^{2}\sum_{\{p\in X_{i,f,2}|s-w_{i,f}+1\le p\le s\}}\sum_{\{q\in Y_{i,f,2}|t-l_{i,f}+1\le q\le t\}} x_{i,f,2,p,q}+\\
&\sum_{i\in I}\sum_{f=1}^{2}\sum_{\{p\in X_{i,f,3}|s-l_{i,f}+1\le p\le s\}}\sum_{\{q\in Y_{i,f,3}|t-w_{i,f}+1\le q\le t\}} x_{i,f,3,p,q}+\\
&\sum_{i\in I}\sum_{f=1}^{2}\sum_{\{p\in X_{i,f,4}|s-w_{i,f}+1\le p\le s\}}\sum_{\{q\in Y_{i,f,4}|t-l_{i,f}+1\le q\le t\}} x_{i,f,4,p,q} \le 1, \quad \forall s\in X, \forall t\in Y
\end{aligned} \tag{3.1}$$

$$\begin{aligned}
x_{i,2,1,p,q-w_{i,2}} &= x_{i,1,1,p,q}, & \forall i\in I_L, \forall p\in X_{i,f,1}, \forall q\in Y_{i,f,1}\\
x_{i,2,2,p+w_{i,1},q} &= x_{i,1,2,p,q}, & \forall i\in I_L, \forall p\in X_{i,f,2}, \forall q\in Y_{i,f,2}\\
x_{i,2,3,p+l_{i,1}-l_{i,2},q+w_{i,1}} &= x_{i,1,3,p,q}, & \forall i\in I_L, \forall p\in X_{i,f,3}, \forall q\in Y_{i,f,3}\\
x_{i,2,4,p-w_{i,2},q+l_{i,1}-l_{i,2}} &= x_{i,1,4,p,q}, & \forall i\in I_L, \forall p\in X_{i,f,4}, \forall q\in Y_{i,f,4}
\end{aligned} \tag{3.2}$$

$$\begin{aligned}
x_{i,2,1,p+l_{i,1}-l_{i,2},q-w_{i,2}} &= x_{i,1,1,p,q}, & \forall i\in I_J, \forall p\in X_{i,f,1}, \forall q\in Y_{i,f,1}\\
x_{i,2,2,p+w_{i,1},q+l_{i,1}-l_{i,2}} &= x_{i,1,2,p,q}, & \forall i\in I_J, \forall p\in X_{i,f,2}, \forall q\in Y_{i,f,2}\\
x_{i,2,3,p,q+w_{i,1}} &= x_{i,1,3,p,q}, & \forall i\in I_J, \forall p\in X_{i,f,3}, \forall q\in Y_{i,f,3}\\
x_{i,2,4,p-w_{i,2},q} &= x_{i,1,4,p,q}, & \forall i\in I_J, \forall p\in X_{i,f,4}, \forall q\in Y_{i,f,4}
\end{aligned} \tag{3.3}$$

$$\begin{aligned}
x_{i,2,1,p+\frac{l_{i,1}-l_{i,2}}{2},q-w_{i,2}} &= x_{i,1,1,p,q}, \quad \forall i \in I_T, \forall p \in X_{i,f,1}, \forall q \in Y_{i,f,1} \\
x_{i,2,2,p+w_{i,1},q+\frac{l_{i,1}-l_{i,2}}{2}} &= x_{i,1,2,p,q}, \quad \forall i \in I_T, \forall p \in X_{i,f,2}, \forall q \in Y_{i,f,2} \\
x_{i,2,3,p+\frac{l_{i,1}-l_{i,2}}{2},q+w_{i,1}} &= x_{i,1,3,p,q}, \quad \forall i \in I_T, \forall p \in X_{i,f,3}, \forall q \in Y_{i,f,3} \\
x_{i,2,4,p-w_{i,2},q+\frac{l_{i,1}-l_{i,2}}{2}} &= x_{i,1,4,p,q}, \quad \forall i \in I_T, \forall p \in X_{i,f,4}, \forall q \in Y_{i,f,4}
\end{aligned} \tag{3.4}$$

$$\begin{aligned}
x_{i,2,1,p-\Delta l_i,q-w_{i,2}} &= x_{i,1,1,p,q}, \quad \forall i \in I_S, \forall p \in X_{i,f,1}, \forall q \in Y_{i,f,1} \\
x_{i,2,2,p+w_{i,1},q-\Delta l_i} &= x_{i,1,2,p,q}, \quad \forall i \in I_S, \forall p \in X_{i,f,2}, \forall q \in Y_{i,f,2} \\
x_{i,2,3,p+l_{i,1}-l_{i,2}+\Delta l_i,q+w_{i,1}} &= x_{i,1,3,p,q}, \quad \forall i \in I_S, \forall p \in X_{i,f,3}, \forall q \in Y_{i,f,3} \\
x_{i,2,4,p-w_{i,2},q+l_{i,1}-l_{i,2}+\Delta l_i} &= x_{i,1,4,p,q}, \quad \forall i \in I_S, \forall p \in X_{i,f,4}, \forall q \in Y_{i,f,4}
\end{aligned} \tag{3.5}$$

$$\begin{aligned}
x_{i,2,1,p+l_{i,1}-l_{i,2}+\Delta l_i,q-w_{i,2}} &= x_{i,1,1,p,q}, \quad \forall i \in I_Z, \forall p \in X_{i,f,1}, \forall q \in Y_{i,f,1} \\
x_{i,2,2,p+w_{i,1},q+l_{i,1}-l_{i,2}+\Delta l_i} &= x_{i,1,2,p,q}, \quad \forall i \in I_Z, \forall p \in X_{i,f,2}, \forall q \in Y_{i,f,2} \\
x_{i,2,3,p-\Delta l_i,q+w_{i,1}} &= x_{i,1,3,p,q}, \quad \forall i \in I_Z, \forall p \in X_{i,f,3}, \forall q \in Y_{i,f,3} \\
x_{i,2,4,p-w_{i,2},q-\Delta l_i} &= x_{i,1,4,p,q}, \quad \forall i \in I_Z, \forall p \in X_{i,f,4}, \forall q \in Y_{i,f,4}
\end{aligned} \tag{3.6}$$

$$x_{i,f,g,p,q} \in \{0,1\}, \quad \forall i \in I, f = 1,2, g = 1,2,3,4, \forall p \in X_{i,f,g}, \forall q \in Y_{i,f,g} \tag{3.7}$$

In formulation (3.1) - (3.7), constraints (3.1) guarantee that at most one rectangle positioned with its lower-left corner at some point (p, q) will contain the point (s, t). Constraints (3.2) - (3.6) relate the position of the lower-left corners of rectangles 1 and 2 of each tetris-like item to create the desired shape of items of types “L”, “J”, “T”, “S” and “Z”, respectively, for all four possible orientations. Finally, constraints (3.7) define the domain of the decision variables for the items, which must be binary.

In order to complete the formulation of the maximization model, the constraints (3.1) - (3.7) that pertain both models must be added to the objective function and constraints in the following:

$$\max \sum_{i \in I} \sum_{g=1}^{4} \sum_{p \in X_{i,1,g}} \sum_{q \in Y_{i,1,g}} v_i * x_{i,1,g,p,q} \tag{3.8}$$

$$\sum_{g=1}^{4} \sum_{p \in X_{i,1,g}} \sum_{q \in Y_{i,1,g}} x_{i,f,g,p,q} \leq b_i, \quad \forall i \in I \tag{3.9}$$

In the formulation (3.8) - (3.9) with (3.1) - (3.7), the objective function (3.8) aims at maximizing the sum of the values of item replicas placed and arranged within the board. It is important to notice that if the value v_i equals the area of the item ($v_i = \sum_{f=1}^{2} l_{i,f} * w_{i,f}$), then this objective function corresponds to maximizing the total area of item replicas placed within the board. Constraints (3.9) ensure that the number of replicas of each item within the board is not above the stipulated limit. For both these constraints, it was decided that only the variables pertaining rectangle 1 are used, ignoring the ones pertaining rectangle 2, but it is possible to use the second rectangle instead of the first one in both these equations with no effects on the model whatsoever. The decision to use the first rectangle came from the fact that the position of the second rectangle was derived from the position of the first one, and, therefore, continuing to use rectangle 1 as the "main" rectangle follows logically.

In order to complete the formulation of the minimization model, the constraints (3.1) - (3.7) that pertain both models must be added to the objective function and constraints in the following:

$$\min \overline{W} \tag{3.10}$$

$$\sum_{g=1}^{4} \sum_{p \in X_{i,1,g}} \sum_{q \in Y_{i,1,g}} x_{i,1,g,p,q} = B_i, \quad \forall i \in I \tag{3.11}$$

$$\begin{aligned}
&\sum_{i \in I} \sum_{f=1}^{2} x_{i,f,1,p,q} * (q + w_{i,f}) + \\
&\sum_{i \in I} \sum_{f=1}^{2} x_{i,f,2,p,q} * (q + l_{i,f}) + \\
&\sum_{i \in I} \sum_{f=1}^{2} x_{i,f,3,p,q} * (q + w_{i,f}) + \\
&\sum_{i \in I} \sum_{f=1}^{2} x_{i,f,4,p,q} * (q + l_{i,f}) \leq \overline{W}, \quad \forall p \in X, \forall q \in Y
\end{aligned} \tag{3.12}$$

$$\overline{W} \in \mathbb{R}^{+} \tag{3.13}$$

In the formulation (3.8) - (3.13) with (3.1) - (3.7), the objective function (3.8) aims at minimizing the value of the variable $\overline{W}$, which represents the final board width required to arrange all items as compactly as possible. Constraints (3.11) are similar to constraints (3.9) in the maximization problem, but instead of ensuring that the number of replicas of each item within the board is not above a stipulated limit, they ensure that the number of replicas is equal to the stipulated value B_i. Constraints (3.12) ensure that the replicas of the items are arranged below the variable width $\overline{W}$. Finally, constraints (3.13) define the domain of the decision variable for the board's width, which is the positive real numbers, although in the end the variable will have an integer value due to the nature of the board's discretization.

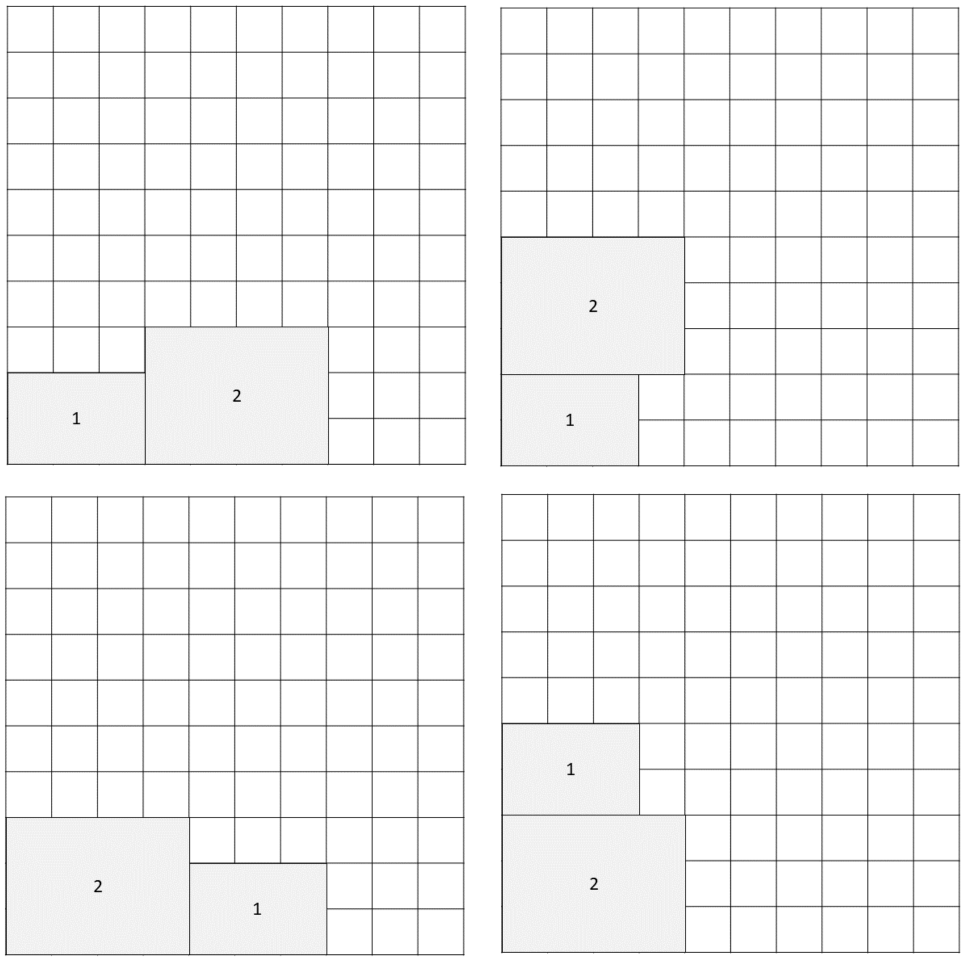
Furthermore, for the minimization problem a lower bound for $\overline{W}$ is given to the model and it is addressed in this thesis as *LowBo*. The value of this bound is obtained by calculating the total area of the items in I for all their replicas B_i using the length $l_{i,f}$ and width $w_{i,f}$ of each item, then dividing the resulting value by the length of the board L, and, finally, rounding the result to the closest integer above it. The reasoning for this calculation comes from the fact that, in a best-case scenario, the placed replicas of the items would form a rectangle with length L, and $\overline{W}$ would consequently be equal to the width of this rectangle. Since $\overline{W}$ is an integer, rounding this width to the closest integer above it provides a slightly better lower bound. One can better imagine this ideal situation by supposing that the items could assume a “liquid” form and occupy the board from bottom to top, resulting in the minimum possible board width.

3.3 Grid Discretization

The first step in generating the variables $x_{i,f,g,p,q}$, is discretizing the board and generating sets X and Y. In the last section, an example of a board where all possible integer positions were used can be seen in Figure 21, a method of discretization called “full sets”. One interesting question to ask is: could the discretization of the board’s positions be made in such a way that less points are generated, and no significant solutions are lost in the process? This is relevant due to the fact that less generated points implies less generated variables, which in turn results in less work for the solver.

This question has been answered for models that only depict rectangles, and indeed there are methods for generating variables that produce less points. In Herz (1972) and Christofides & Whitlock (1977) the “normal patterns” or “conic combinations” methods are discussed. These authors mathematically proved that, without losing relevant solutions, one can move a rectangle downwards and to the left, until the left and bottom sides of the rectangle either touch the left and bottom side of the board, or touch the right and upper side of another rectangle. This means that only the points which are a combination of the dimensions of the rectangles must be generated for this board. For example, suppose one has only two rectangles to place on a 10x10 board, rectangle 1 with a length of 3 and width of 2, and rectangle 2 with a length of 4 and width of 3, and no rotations are allowed. Following the normal patterns logic, there are only four significant solutions, which are presented in Figure 22 in the following. In this example, the discretization of the board on the x-axis only has to contemplate the combinations of the lengths of the two rectangles (3 and 4), and only the points $p = 0, 3, 4, 7$ would have to be generated, instead of $p = 0, 1, ..., 9$. On the y-axis, the discretization would only have to contemplate the combinations of the widths of the two rectangles (2 and 3), and only the points $q = 0, 2, 3, 5$ would have to be generated, instead of $q = 0, 1, ..., 9$. Furthering the example, if there was one more rectangle with length 1 and width 1, the generated p values would be $p = 0, 1, 3, 4, 5, 7, 8$, with $p = 5$ deriving from the combination 1+4 and $p = 8$ from 3+4+1, while the generated q values would be $q = 0, 1, 2, 3, 4, 5, 6$, with $q = 4$ deriving from the combination 1+3 and $q = 6$ from 2+3+1.

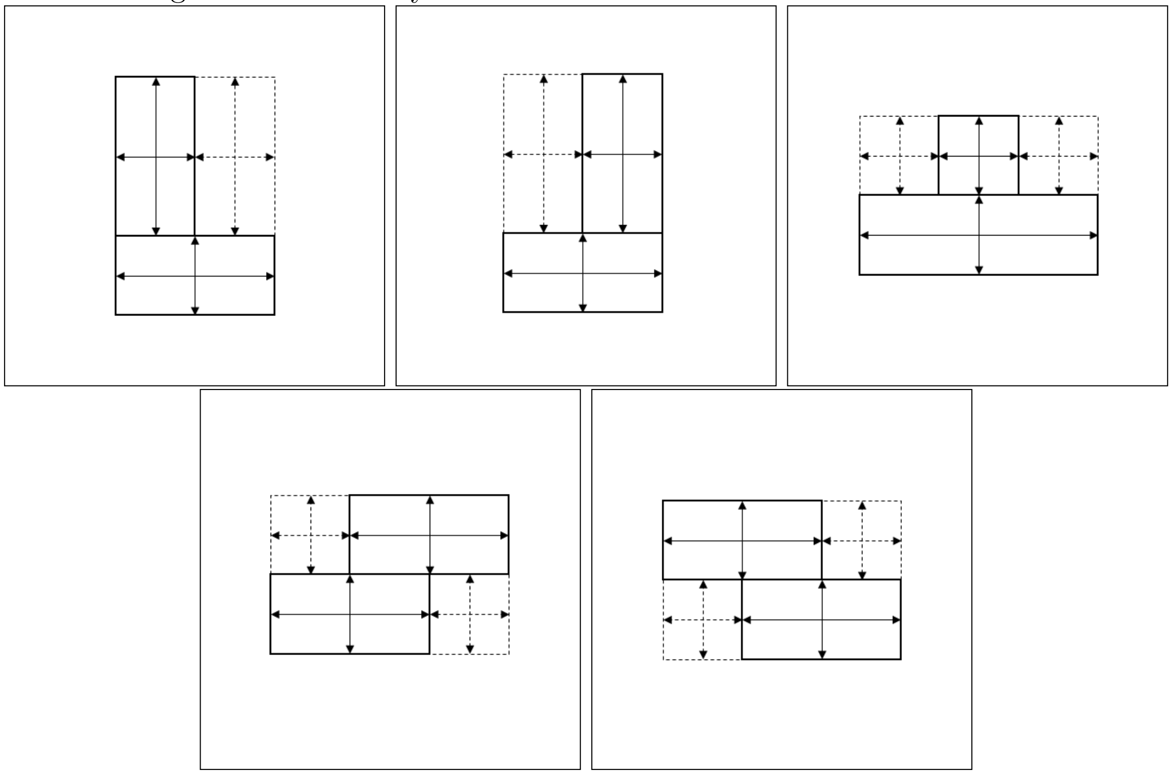
Figure 22: Possible Solutions for the Normal Patterns Example



Source: The Author

This method of board discretization can be extremely effective in reducing model complexity. In the mentioned example with two rectangles, the number of (p, q) combinations was reduced by 84%, and by adding the third rectangle this reduction comes down to 51%. However, in order to apply normal patterns in a model with tetris-like items, some adaptations are necessary. Firstly, it is not possible to only use the lengths $l_{i,f}$ and widths $w_{i,f}$ to generate the combinations, since the intricate positional relationship between the two rectangles that compose a tetris-like piece creates other dimensions that must be considered in order to correctly generate the positions along the axes. A simple example comes from the S-shaped items, which require the position $p = \Delta l_i$ to be generated with the 0 degrees rotation, in order to place the first rectangle of the item. To obtain all the dimensions necessary for a normal patterns discretization, it is necessary to calculate the lengths and widths of all the rectangles generated by superimposing a large rectangular "hull" on a tetris-like item that extends from its lower-left corner to its upper-right corner. This is shown in Figure 23 in the following, where the relevant dimensions are represented by double-headed arrows, and the items are displayed in the "L","J","T","S","Z" order, from the upper-left image to the bottom-right one.

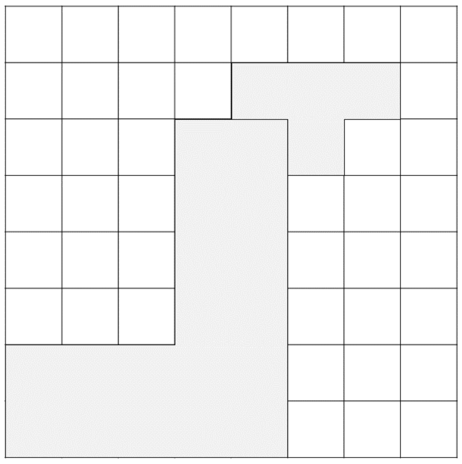
Figure 23: Necessary Dimensions for Normal Patterns Generation



Source: The Author

Even with all the extra dimensions calculated, which can be performed with the parameters provided to the model, there are still modifications that need to be made in the normal patterns generation to adapt it to the tetris-like items. For rectangles, the number of values that need to be combined is always equal to the total number of replicas, meaning that if one has only one rectangle, the only relevant coordinates are $(0, 0)$ and the coordinate for the length of the rectangle in the x-axis and for the width of the rectangle in the y-axis. If one has two rectangles, the combination of the lengths and widths of both of them needs to be added to the set of coordinates. If one has three, all combinations of three dimensions must be added for each axis, and so on. For tetris-like items, however, each replica contains multiple dimensions that have to be combined in order to generate the essential coordinates for that piece, what makes this process very complex using the usual conic combinations method. Moreover, due to the concave nature of the items, it is possible that positions where one replica "fills another's gap" are favorable for the solution, such as the one in Figure 24 in the following, and some of the necessary values along the axes might not be generated by this method.

Figure 24: Example of a Position where Two Tetris-like Pieces "Fit"



Source: The Author

One possible solution to these challenges, which is used in this thesis, is generating all possible combinations for all dimensions in Figure 23 along both axes to adapt to the dimensional requirements of tetris-like items, and also adding the reflection of these combinations in regards to the opposite side of the board ($L-p$ or $W-q$) to incorporate the positions where tetris-like pieces "fit" into the method. Take for instance one rectangle of length 4 and width 5 in a board with length 10 and width 10: applying this adapted methodology, the length and width would have to be used in the combinations for both axes if rotation was allowed, and all combinations of these values that are esmaller than the dimensions of the board would be part of sets X and Y. The result would be that $p = q = 0, 4, 5, 8, 9$ without considering the reflection on the opposite sides of the board, and $p = q = 0, 1, 2, 4, 5, 6, 8, 9$ after adding the reflections, only a 20% reduction from the full set. This solution is not as impactful as the original normal pattern generation, especially as the number of items increases and dimensions are added to the set of values to be combined, however, it can still significantly reduce the number of variables for the solver. Furthermore, unlike the original conic combinations for rectangles, there is no analytical proof that this adapted version does not cause information loss. This is very unlikely, since the adaptations to the original normal patterns generations were carefully developed precisely to avoid this. However, for this reason the effectiveness of this method is tested and compared to a full set board discretization in Chapter 5.

After discretizing the board, the variable generation is a simple process. It is only necessary to take the generated X and Y sets, and for each item i and rectangle f and orientation g remove the values that are above the related board dimension minus the related item dimension to create sets $X_{i,f,g}$ and $Y_{i,f,g}$ ($L - l_{i,f}$ and $W - w_{i,f}$ for $g = 1$ and 3, and $L - w_{i,f}$ and $W - l_{i,f}$ for $g = 2$ and 4). This sets are in turn used to generate the variables $x_{i,f,g,p,q}$ for $p \in X_{i,f,g}$ and $q \in Y_{i,f,g}$. As an example, take a rectangle of length 5 and width 4 and two given sets X and Y. To generate $X_{i,f,g}$, one would take all the values in X that are equal to or smaller than $L - 5$ for $g = 1$ and 3, and equal to or smaller than $L - 4$ for $g = 2$ and 4, whereas to generate $Y_{i,f,g}$ one would take all the values in Y that are equal to or smaller than $W - 4$ for $g = 1$ and 3, and equal to or smaller than $W - 5$ for $g = 2$ and 4. It is important to observe that rectangular items, that belong to I_R, do not have a second rectangle and only have two rotations, which means that $x_{i,f,g,p,q}$ variables for these items are only generated for $f = 1$ and $g = 1, 2$.

With the grid discretization methods described in this section it is possible to run the model with a solver.

4 SOLUTION APPROACH

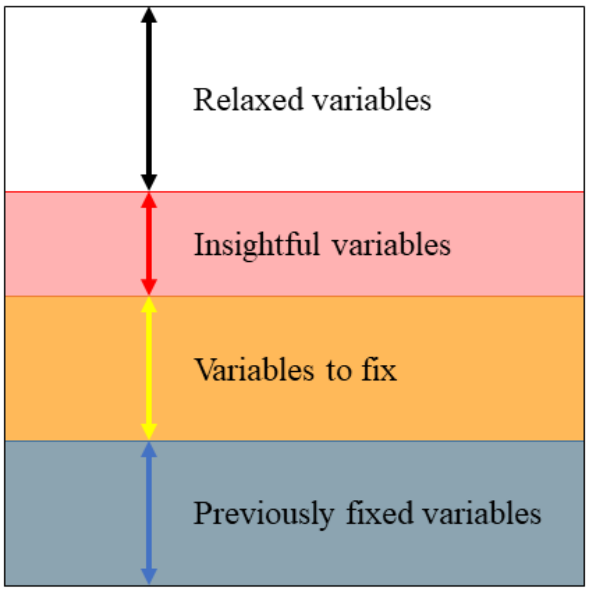
In this chapter, the heuristic algorithms developed for both the maximization and minimization models are discussed in detail, in order to expose the intricacies of the heuristic algorithms that can affect the obtained results.

4.1 Relax-and-Fix Heuristic

As mentioned in Section 2.2.1, a relax-and-fix heuristic divides a MIP problem into sub-problems by determining subsets of variables that, in an iteration of the heuristic, are either fixed, continuous, or binary.

However, this definition allows several parameters to be different between relax-and-fix algorithms, and all the relevant ones are discussed here. First of all, the developed algorithm does not directly divide the variables into subsets, but rather divides the board width (or initial board width in the minimization case) into horizons. In a given iteration, the board can be divided into four horizons, with the first one placed in the bottom of the board, and the last one at the top of the board, what can be seen in Figure 25:

Figure 25: One Iteration of the Relax-and-Fix Heuristic



Source: The Author

Each horizon can be defined as follows:

1. **Previously fixed variables:** contains $x_{i,f,g,p,q}$ variables that have been fixed in previous iterations if their value was decided to be 1, and $x_{i,f,g,p,q}$ binary variables that were not fixed in previous iterations since their value was deemed as 0. Not fixing variables, whose value was deemed as 0 in previous iterations is intended to give the chance for following iterations to use empty spaces to place new pieces that might require it.

2. **Variables to fix:** contains binary $x_{i,f,g,p,q}$ variables that will either be fixed as 1 at the end of the iteration if decided by the heuristic, or remain as binary if the algorithm sets their value as 0.

3. **Insightful variables:** contains $x_{i,f,g,p,q}$ binary variables that will not be fixed, independently of the algorithm's decision. This horizon is optional, since it is not necessary for a relax-and-fix heuristic, and also increases the number of binary variables in an iteration, slowing the algorithm down. However, if used, this horizon allows the heuristic to choose better placements of the pieces by giving the algorithm the ability to accommodate for pieces that will be placed and fixed in future iterations.

4. **Relaxed variables:** contains $x_{i,f,g,p,q}$ continuous variables that will not be fixed, independently of the algorithm's decision.

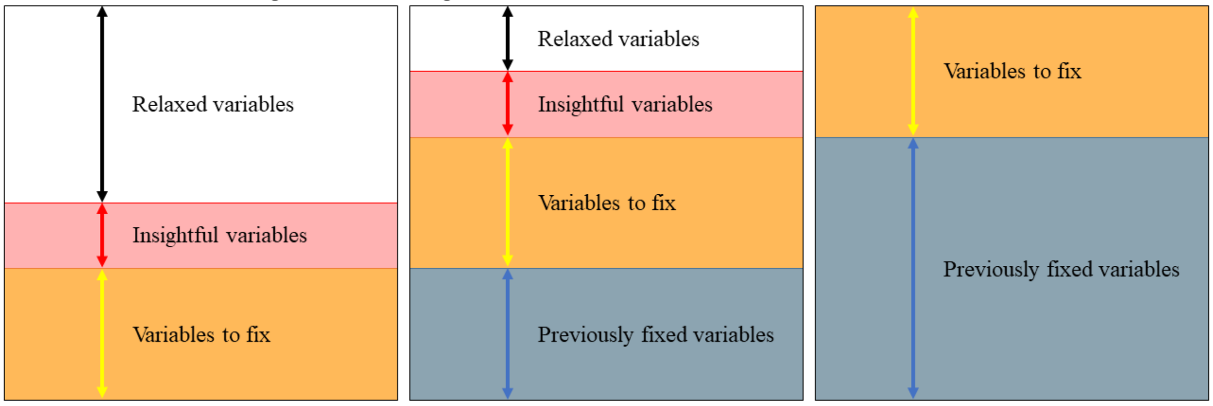
Before running the heuristic, the width of horizon 2 and 3 must be decided. These parameters are shown in the following, and can be visualized as the yellow and red double-headed arrows in Figure 25, respectively:

Heuristic Parameters:
fix : width of the "variables to fix" horizon;
ins : width of the "insightful variables" horizon;

The most important parameter is fix, which must be larger than or equal to 1 and smaller than the board width (W), since using the board width as fix would simply result in the monolithic model. Conversely, ins can have a value of 0, what would cause it to be nonexistent in the heuristic, and must be smaller than or equal to $W - fix$. Logically, both parameters must have a positive value since there are no negative q coordinates.

The procedure of the heuristic starts with horizon 2 (variables to fix) placed at the bottom of the board, where all variables that have a q value from 0 to $fix - 1$ will be fixed at the end of the iteration if their value is deemed as 1. If ins is larger than 0, then above horizon 2 is horizon 3 (insightful variables), where all variables that have a q value from fix to $fix + ins - 1$ will be binary during the iteration. Above either horizon 2 or 3, depending on whether ins is 0, is horizon 4, where all variables that have a q value from $fix + ins$ to $W - 1$ will be continuous during the iteration. This relaxed model is then optimized, ending the iteration. After the first iteration, each horizon will move up by fix and horizon 1 will appear bellow horizon 2, containing the saved information about previous iterations. The process of moving up the horizons will repeat itself until the last iteration, where only horizon 1 and 2 will be present, since the last variables are being fixed. At the end of the last iteration, the algorithm results in the solution for the heuristic. The progression of the algorithm can be seen in Figure 26, in an example where fix is one third of the board and ins is one sixth of the board. The images are in order of first iteration on the left, to third and last iteration on the right:

Figure 26: Progression of the Relax-and-Fix Heuristic



Source: The Author

Without the use of normal patterns (see Section 3.3), this approach of dividing the board into horizons has the same effect as directly dividing the variables uniformly based on their q value, with each iteration having the same number of binary variables, excluding the ones in the fixed horizon. The exception are the last iterations due to the proximity to the board edge and the way variables are generated, which causes the last iterations to have less binary variables in the second horizon than the others. However, with the use of normal patterns, it is possible that different iterations will have different numbers of binary variables in horizon 2, since the $x_{i,f,g,p,q}$ variables might not exist for certain q values. The design of the algorithm was intended to capture different variable generation methods with the same logic, and, therefore, the choice of dividing the variables as described is appropriate.

Furthermore, each iteration has a time limit to ensure that the heuristic is feasible for use in a adequate amount of time. This time limit for each iteration is equal to 30 minutes divided by the number of total iterations that will be run, what is consistent for the time limit set in the monolithic model (see Section 5.1). If the time runs out for an iteration, the algorithm returns the information that no solution was found in the timeframe.

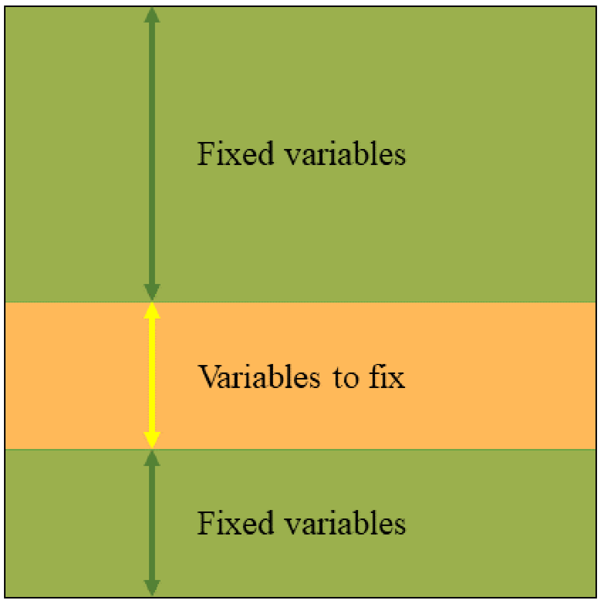
Finally, for the minimization problem, the lower bound ($LowBo$) used in the monolithic model was removed in order to encourage the heuristic to place pieces below this bound, since these problems only take the final width of the board ($\overline{W}$) into consideration, and setting a lower bound can make the heuristic indifferent to the placement of pieces below this threshold.

4.2 Fix-and-Optimize Heuristic

As mentioned in Section 2.2.2, a fix-and-optimize heuristic divides a MIP problem into sub-problems by determining subsets of variables that, in an iteration of the heuristic, are either fixed, or binary.

However, this definition allows several parameters to be different between fix-and-optimize algorithms, and all the relevant ones are discussed here. First of all, similarly to the last heuristic, the developed algorithm does not directly divide the variables into subsets, but rather divides the board width (or initial board width in the minimization case) into horizons. In a given iteration, the board can be divided into two horizons, what can be seen in Figure 27:

Figure 27: One Iteration of the Fix-and-Optimize Heuristic



Source: The Author

Each horizon can be defined as follows:

1. **Fixed variables:** contains $x_{i,f,g,p,q}$ variables that have been fixed in previous iterations independently of their assigned value. There is an exception for the first iterations, since the board has not been completely swept and fixed by this heuristic yet. For these iterations, this horizon contains $x_{i,f,g,p,q}$ variables that have been fixed in the previous heuristic, which, in this thesis, is the relax-and-fix heuristic. This means that, logically, the fix-and-optimize heuristic cannot work without an initial solution.

2. **Variables to fix:** contains binary $x_{i,f,g,p,q}$ variables that will be fixed at the end of the iteration independently of their assigned value.

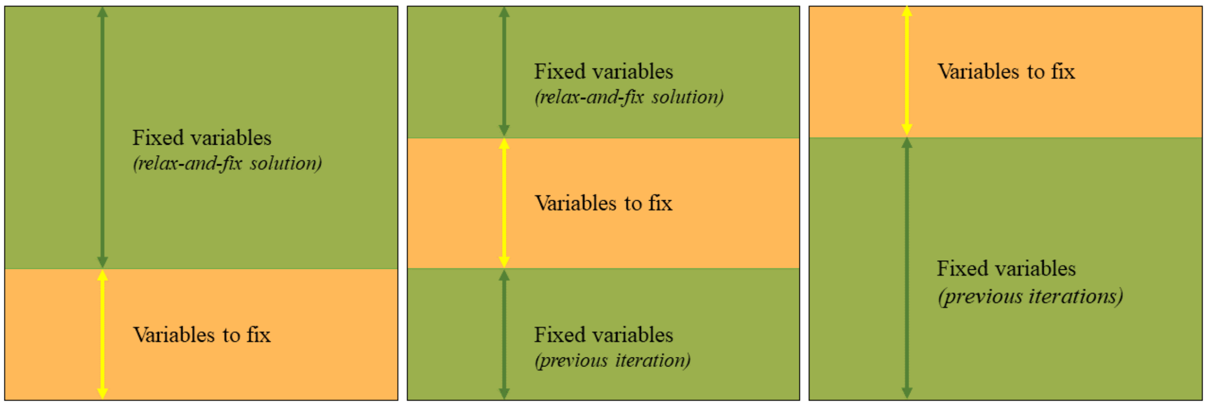
Before running the heuristic, the width of horizon 2 must be decided. This parameter is shown in the following, and can be visualized as the yellow arrow in Figure 27:

Heuristic Parameter:
opt : width of the "variables to fix" horizon;

This parameter must be larger than or equal to 1 and smaller than the board width (W), since using the board width as *opt* would simply result in the monolithic model. Logically, this parameter must have a positive value since there are no negative q coordinates.

The procedure of the heuristic starts with horizon 2 (variables to fix) placed at the bottom of the board, where all variables that have a q value from 0 to $opt-1$ will be fixed at the end of the iteration. Above horizon 2 is horizon 1, where all variables that have a q value from *opt* to $W-1$ will have fixed values obtained from the previous solution. This relaxed model is then optimized, ending the iteration. After the first iteration, horizon 2 will move up by *opt* and horizon 1 will appear bellow horizon 2, containing the saved information about previous iterations. The process of moving up the horizons will repeat itself until an iteration where horizon 2 will touch the upper edge of the board, where all variables that have a q value from $W-opt$ to $W-1$ will be fixed at the end of the iteration. At this point, if the termination criteria is not reached, the process will restart with horizon 2 placed at the bottom of the board, until the termination criteria is indeed reached. Interestingly, since all variables have a fixed value at the end of each iteration, each iteration provides a complete solution to the problem. The progression of the algorithm can be seen in Figure 28, in an example where *opt* is one third of the board. The images are in order of first iteration on the left, to third iteration on the right. At the third iteration, the process would either end, if the termination criteria was reached, or start over with the fourth iteration possessing a similar configuration as the first one, but with fixed variable values from the three previous fix-and-optimize iterations, and not from the relax-and-fix heuristic.

Figure 28: Progression of the Fix-and-Optimize Heuristic



Source: The Author

The developed termination criteria checks at every iteration in which horizon 2 touches the upper edge of the board, such as the right-most image in Figure 28, whether the objective function has improved at any point during this so-called “board sweep”. If so, the heuristic continues to run, and horizon 2 is again placed at the bottom of the board. If there is not an improvement, then the heuristic stops and the last generated solution is returned. Furthermore, the heuristic has a smart trigger for stopping in the maximization problem if the values v_i of the items are equivalent to their areas. In such cases, the algorithm checks whether the value of the objective function for the solution of every iteration reached the area of the board, since this implies that an optimal solution was reached. If so, the algorithm stops and returns this solution.

Furthermore, each iteration has a time limit to ensure that the heuristic is feasible for use in an adequate amount of time. This time limit for each iteration is equal to 30 minutes divided by the number of total iterations necessary for horizon 2 to sweep the board, what is consistent for the time limit set in the monolithic model (see Section 5.1). If the time runs out for an iteration, the algorithm returns the information that no solution was found in the timeframe. It is important to notice that it is theoretically possible that the heuristic ends up running for an excessively long time if the termination criteria is not reached in any sweep, what could happen with small increments in the objective function at every turning point in the heuristic. However, although theoretically possible, this issue is of no concern due to its unlikeliness. Furthermore, the lower bound ($LowBo$) was also removed for this heuristic.

5 COMPUTATIONAL TESTS

In this chapter, the generation of the instances for the maximization and minimization problem used for testing the models and heuristics will be detailed, and the results for both will be presented and discussed. For the monolithic models, the results of the version with normal patterns are compared to the ones with full sets (see Section 3.3). For the heuristics, the results are designed to find the best heuristic parameters combination (see Chapter 4). Furthermore, the results between the monolithic models and the heuristics are also compared in order to validate the efficiency of the heuristics.

5.1 Generated Test Instances

The main test instances used a pseudo-random generation of parameters for the tetris-like items, which were drawn from certain values that varied for each parameter and for each item type. Each dimension of each tetris-like item type was sorted into one of two categories, large dimension or small dimension, and this was done so that the item types would retain the shapes of the original tetris-like pieces. Thus, small dimensions were drawn from the numbers {4, 6, 8}, and large dimensions were drawn from the numbers {10, 12, 14}. In Table 1 are the classifications of the dimensions by item type (i.e., "L", "J", "T", "S", "Z"):

Table 1: Dimensions Classification by Type of Piece

L		**J**		**T**		**S**		**Z**	
$l_{i,1}$	small	$l_{i,1}$	small	$l_{i,1}$	large	$l_{i,1}$	large	$l_{i,1}$	large
$w_{i,1}$	large	$w_{i,1}$	large	$w_{i,1}$	small	$w_{i,1}$	small	$w_{i,1}$	small
$l_{i,2}$	large	$l_{i,2}$	large	$l_{i,2}$	small	$l_{i,2}$	large	$l_{i,2}$	large
$w_{i,2}$	small	$w_{i,2}$	small	$w_{i,2}$	small	$w_{i,2}$	small	$w_{i,2}$	small
Δl_i	-	Δl_i	-	Δl_i	-	Δl_i	small	Δl_i	small

Source: The Author

The sizes of the numbers from which the dimensions were drawn were selected in order to create items with areas that did not occupy neither too much nor too little of the are of the board. For the maximization problem, the dimensions of the board for all the generated instances were set as $L = 40$ and $W = 40$, resulting in a square board. For the minimization problems, the dimensions were set as $L = 40$ and $W = 100$, resulting in a rectangular initial board.

Twelve instances were generated containing the following items: 1 item type “L”; 5 items type “L”; 1 item type “J”; 5 items type “J”; 1 item type “T”; 5 items type “T”; 1 item type “S”; 5 items type “S”; 1 item type “Z”; 5 items type “Z”; 1 item for each type “L”, “J”, “T”, “S”, “Z”; 5 items for each type “L”, “J”, “T”, “S”, “Z” resulting in 25 items in total. These instances are represented in the results, respectively, as: 1L; 5Ls; 1J; 5Js; 1T; 5Ts; 1S; 5Ss; 1Z; 5Zs; 1Each; 5Each. The numbers of items in each instance were purposefully chosen to compare the results for instances with more items than the others. For each instance, the dimensions of each item were generated iteratively until all items of the same type had different sizes. Table 2 presents the parameters of each item of each generated instance, where the columns represent the instances and each block of 8 rows, marked by the colors white and grey, represent the parameters of an item:

Table 2: Generated Instances

	1L	**5Ls**	**1J**	**5Js**	**1T**	**5Ts**	**1S**	**5Ss**	**1Z**	**5Zs**	**1Each**	**5Each**				
$l_{i,1}$	6	6	4	6	14	12	12	12	14	10	6	4	4	12	12	10
$w_{i,1}$	14	10	10	12	8	4	6	4	6	8	10	14	14	8	4	6
$l_{i,2}$	10	14	10	12	6	4	10	12	12	10	14	12	10	8	12	12
$w_{i,2}$	8	6	6	4	6	6	8	6	8	4	6	6	6	8	6	6
Δl_i	-	-	-	-	-	-	6	8	6	4	-	-	-	-	8	8
b_i	9	6	15	5	10	12	10	13	8	11	4	10	11	7	10	3
B_i	10	3	16	3	11	5	11	3	9	3	3	1	1	1	1	1
Type	L	L	J	J	T	T	S	S	Z	Z	L	L	J	T	S	Z
$l_{i,1}$		4		6		12		10		10	8	8	8	10	12	10
$w_{i,1}$		10		10		6		4		8	14	10	10	4	4	6
$l_{i,2}$		10		12		8		14		12	12	10	14	6	10	14
$w_{i,2}$		8		8		6		6		8	6	8	4	8	4	6
Δl_i		-		-		-		6		8	-	-	-	-	4	6
b_i		5		8		2		7		1	2	4	4	1	10	8
B_i		3		3		3		3		2	2	1	1	1	1	1
Type		L		J		T		S		Z	J	L	J	T	S	Z
$l_{i,1}$		6		8		14		14		12	12	8	8	14	14	14
$w_{i,1}$		12		10		6		4		6	8	10	12	6	4	8
$l_{i,2}$		12		12		6		12		12	8	12	12	6	12	12
$w_{i,2}$		4		6		8		4		8	6	6	4	4	4	8
Δl_i		-		-		-		4		8	-	-	-	-	6	4
b_i		13		10		6		2		8	8	1	11	10	1	6
B_i		3		3		3		4		2	3	1	1	1	1	1
Type		L		J		T		S		Z	T	L	J	T	S	Z
$l_{i,1}$		4		4		10		14		12	12	4	6	10	14	10
$w_{i,1}$		12		10		8		6		8	8	14	10	4	4	4
$l_{i,2}$		10		14		6		10		10	14	12	10	6	12	14
$w_{i,2}$		8		8		8		8		4	8	4	4	6	8	4
Δl_i		-		-		-		6		4	4	-	-	-	8	6
b_i		2		7		4		2		4	1	10	6	11	2	4
B_i		3		3		3		2		3	2	1	1	1	1	1
Type		L		J		T		S		Z	S	L	J	T	S	Z
$l_{i,1}$		6		6		12		10		14	14	6	4	12	10	10
$w_{i,1}$		14		10		6		4		8	6	14	14	8	8	4
$l_{i,2}$		14		14		8		12		14	10	14	14	4	10	10
$w_{i,2}$		4		6		8		6		4	8	4	8	4	8	6
Δl_i		-		-		-		4		4	4	-	-	-	4	8
b_i		1		11		11		8		4	9	8	2	5	6	7
B_i		3		3		3		3		2	2	1	1	1	1	1
Type		L		J		T		S		Z	Z	L	J	T	S	Z

Source: The Author

Besides the dimension for each item in each instance, this table also contains the number of replicas of each item (see Section 3.2). For the maximization problem, this value is represented by b_i and was generated through two methods. For the instances with only one item, this value was chosen through a deterministic formula so that the 40x40 board could be occupied as much as possible, while for the instances with more than one item, this value was randomly generated iteratively using integers from 1 to 14, until the sum of the areas of the items resulted in at least the board area. For the minimization problem, this value is represented by B_i and was generated through one method only. The intention behind this method is generating a $\overline{W}$ that is close to 40, similar to the maximization problem. The method work as follows: take the area of the intended final board of 1600, divide by the number of items in the instance, divide this number by the area of the item and round the number up to the closest integer.

It is important to notice that the value of each item is missing from Table 2 for the maximization problem. This is the case, because the value of each item was set as the value of their areas, which is $l_{i,1} * w_{i,1} + l_{i,2} * w_{i,2}$. This way, the model and heuristics will aim at maximizing the occupied area of the 40x40 board instead of an item value with no meaning for our computational tests.

Another important topic to discuss are the results generated from the optimization procedures and the heuristics. For the monolithic models of both problems the following data was analyzed:

1. **Objective:** the value of the objective function at the end of the optimization procedures. This is the most important result from the models since the objective function is the object of optimization in all Operations Research problems. For the maximization model, this value equals the sum of the values of the items that were placed in the board, and for the minimization problem, the final width of the board.

2. **Run Time:** measured in seconds, it is either the amount of time it took the algorithm to produce the optimal solution, or the time limit for the procedure (see Section 3.2). This information gives an insight into the difficulty of running an instance, since the more it takes to run it, the more difficult it is for the algorithm to find the optimal solution. For the monolithic models there was a given run time limit of thirty minutes for each instance which was used to make solving an instance time-realistic, and to better compare the monolithic results with the heuristic ones.

3. **Absolute GAP:** the difference between the value of the objective function for the solution and the theoretical calculated best possible value for the problem. This data shows how close the algorithm was to finding the best possible solutions in case the time limit was reached, and has the value of 0 if the optimal solution was found.

4. **Relative GAP:** the absolute GAP divided by the the value of the objective function for the solution. This data also indicates how close the algorithm was to finding the best possible solutions in case the time limit was reached, however, it is measured in relative terms, generalizing better for instances with very different parameters, what is not the case in this thesis.

5. **Iterations:** the number of iterations that the solver took to find the solution. This information also gives an insight into the difficulty of running an instance, since the more take it takes to run it, the more work it took the solver to find the solution.

6. **Equations:** the number of constraints equations generated by the instance. This value is independent of the solution since it depends on the parameters of the instance, but also gives an insight into the difficulty of running an instance, since the more take it takes to run it, the more work it probably will take the solver to find the solution.

7. **Variables:** the number of $x_{i,f,g,p,q}$ variables generated by the instance. This value is independent of the solution, since it only depends on the parameters of the instance, but also gives an insight into the difficulty of running an instance, since the more take it takes to run it, the more work it probably will take the solver to find the solution.

8. **Images:** the images of the boards with the placed items from the found solution. Although some are presented in this section as examples, they can all be seen in Appendix A for the maximization problem with normal patterns, and Appendix B for the minimization problem with normal patterns.

For the heuristics, the following data was analyzed:

1. **Objective:** the value of the objective function at the end of the heuristic procedures. Similar to the monolithic models, this is the most important result from the heuristic.

2. **Run Time:** measured in seconds, it is either the amount of time it took the algorithm to produce the solution, or the time limit for the procedure (see Chapter 4). Similar to the monolithic models, this information gives an insight into the difficulty of running an instance. The time limitation for the heuristics is explained in Chapter 4.

3. **Feasibility:** the feasibility of the instance at any point in the heuristic. This data was analyzed since it is possible that the placement of the pieces in the boards can done in such a way by the heuristics, that it becomes infeasible to comply with the constraints at later iterations of the algorithm. It is possible that an instance is infeasible even for the monolithic models, but their parameters were generated to avoid such an occasion.

4. **Images:** the images of the boards with the placed items from the found solution. Some are presented in this section as examples.

The results from the monolithic models that are not included here do not apply for the heuristics, since the iterative nature of the heuristics take the meaning out of the GAP function, and also out of the number of equations and number of variables, given that each iteration will have a different ones.

5.2 Results for the Maximization Problem

In this section, the results from both the monolithic model and the heuristics for the maximization problem are discussed. As described in Section 3.2, the aim of this model is maximizing the value of the placed pieces, and since the value of the items in the twelve instances correspond to the item's area, this problems aims at maximizing the occupation of the board.

5.2.1 Monolithic Model

In Table 3, the results for the monolithic model of the maximization problem can be seen. Two scenarios were compared: one where the full sets for grid discretization were used, and another where the adapted normal patterns were used (see Section 3.3). The values in bold indicate that the time limit was reached for an instance.

Table 3: Results for the Monolithic Model Maximization Problem

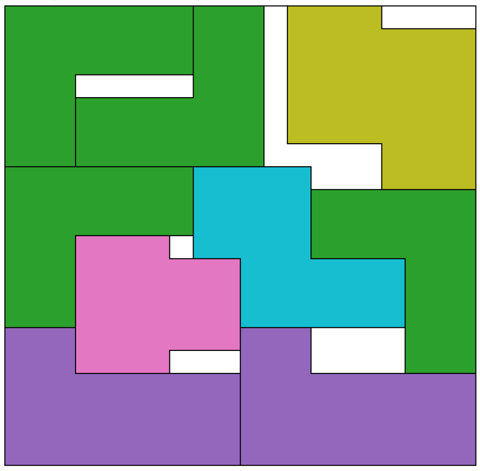
		1L	**5Ls**	**1J**	**5Js**	**1T**	**5Ts**
Full Sets	**Objective**	1312	1592	1600	1560	1184	1504
	Run Time	3,67	1005,76	1,92	1714,96	3,47	996,70
	Abs GAP	0,00	0,00	0,00	0,00	0,00	0,00
	Rel GAP	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
	Iterations	4352	1045184	1988	945998	1327	755933
	Equations	6582	29066	7158	29066	6582	29066
	Variables	7872	41312	8928	40944	8464	43680
With NP	**Objective**	1312	1592	1600	1560	1184	1504
	Run Time	0,31	19,27	0,16	48,22	0,15	37,43
	Abs GAP	0,00	0,00	0,00	0,00	0,00	0,00
	Rel GAP	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
	Iterations	424	34785	514	388708	354	112395
	Equations	2978	8906	3126	8906	2978	8906
	Variables	2096	10984	2368	10888	2248	11592

		1S	**5Ss**	**1Z**	**5Zs**	**1AllShapes**	**5AllShapes**
Full Sets	**Objective**	1216	1420	1080	1432	1460	1560
	Run Time	2,77	**1800,08**	9,42	33,57	1013,66	**1802,45**
	Abs GAP	0,00	24,00	0,00	0,00	0,00	40,00
	Rel GAP	0,00%	1,69%	0,00%	0,00%	0,00%	2,56%
	Iterations	1875	2572216	5882	4446	1137073	197349
	Equations	6582	29066	6582	29066	26186	138606
	Variables	8152	41528	7608	40464	39456	209584
With NP	**Objective**	1216	1420	1080	1432	1460	1592
	Run Time	0,40	167,74	0,19	1,64	17,64	**1800,33**
	Abs GAP	0,00	0,00	0,00	0,00	0,00	8,00
	Rel GAP	0,00%	0,00%	0,00%	0,00%	0,00%	0,50%
	Iterations	622	1039120	28	1446	142265	965233
	Equations	2978	8906	2978	8906	8166	37806
	Variables	2168	11040	2028	10764	10504	55696

Source: The Author

All the values of the objective functions have physical interpretations, such as items placed on a board or cuts made on a large piece of cloth (see Section 2.1.3), and in order to visualize the physical manifestations of the results, Figure 29 shows the resulting image for one of the instances:

Figure 29: Resulting Figure for the “1Each” Instance with Normal Patterns



Source: The Author

In order to analyze the influence of the normal patterns in the model, four indicators are analyzed: run time, iterations, equations and variables. To analyze them, the ratio between the results of each instance without and with normal patterns was calculated and averaged. The resulting ratios are, respectively: 25.7, 23.7, 2.9, 3.8. These ratios show that the normal patterns have a great influence in the maximization model, since their use reduces greatly the run time and the iterations that the solver uses, and have a smaller, but significant impact in the number of equations and variables. Furthermore, we can see in Table 3 that for the instance “5Ss” the model with full sets was not able to find the optimal solution, while the one with normal patterns was. Furthermore, for the instance “5Each”, although neither version drove the GAP to 0, the one with normal patterns found a better solution.

In order to analyze the influence of the number of items in the model, the same four indicators are analyzed: run time, iterations, equations and variables. To analyze them, the ratio between the results of each instance with five times more items and the one with 5 times less items was calculated and averaged only for the version with normal patterns. The resulting ratios are, respectively: 193.1, 480.8, 3.2, 5.1. The ratios show that the number of items has a great influence in the maximization model, even larger than the normal patterns, in all indicators, since their use reduces greatly the run time and the iterations that the solver uses, and has a smaller, but significant impact in the number of equations and variables.

In conclusion, the developed model is able to contemplate the maximization cutting and packing problem with tetris-like items, and the adapted normal patterns are useful and provide great results.

5.2.2 Relax-and-Fix Heuristic

For the maximization relax-and-fix heuristic, 9 combinations of the relax-and-fix heuristic parameters were compared (see Section 4.1). The tested values of the fix parameter where {4,5,8}, representing, respectively, one tenth, one eighth and one fifth of the board width ($W = 40$), while the ones of the ins parameter where {0,4,5}. Furthermore, all the instances used normal patterns. The results for the relax-and-fix heuristic of the maximization problem can be seen in Table 4 and Table 5. The first presents the values for the objective function and the second for the run time. The values in bold indicate that the value corresponds to the best for the instance. Furthermore, "R" represents the value of the fix parameter, and "F" of the ins parameter.

Table 4: Results for the Relax-and-Fix Maximization Problem - Objective

	R4,F0	**R4,F4**	**R4,F5**	**R5,F0**	**R5,F4**	**R5,F5**	**R8,F0**	**R8,F4**	**R8,F5**
1L	1312	1312	1312	1312	1312	1312	1312	1312	1312
5Ls	1592	1592	1592	1592	1592	1592	1592	1592	1592
1J	1600	1600	1600	1600	1600	1600	1600	1600	1600
5Js	1548	1548	1548	1548	1548	1548	1548	1548	1548
1T	1184	1184	1184	1184	1184	1184	1184	1184	1184
5Ts	1416	**1456**	1448	1412	**1456**	**1456**	1416	**1456**	1448
1S	1216	1216	1216	1216	1216	1216	1216	1216	1216
5Ss	1392	1392	1392	1392	1392	1392	1392	1392	**1404**
1Z	1080	1080	1080	1080	1080	1080	1080	1080	1080
5Zs	1416	**1432**	**1432**	1416	**1432**	**1432**	**1432**	**1432**	**1432**
1Each	1436	**1460**	**1460**	1436	1396	1396	**1460**	1436	1436
5Each	1568	1584	1564	1560	1564	1564	1584	1544	**1592**

Source: The Author

Table 5: Results for the Relax-and-Fix Maximization Problem - Run Time

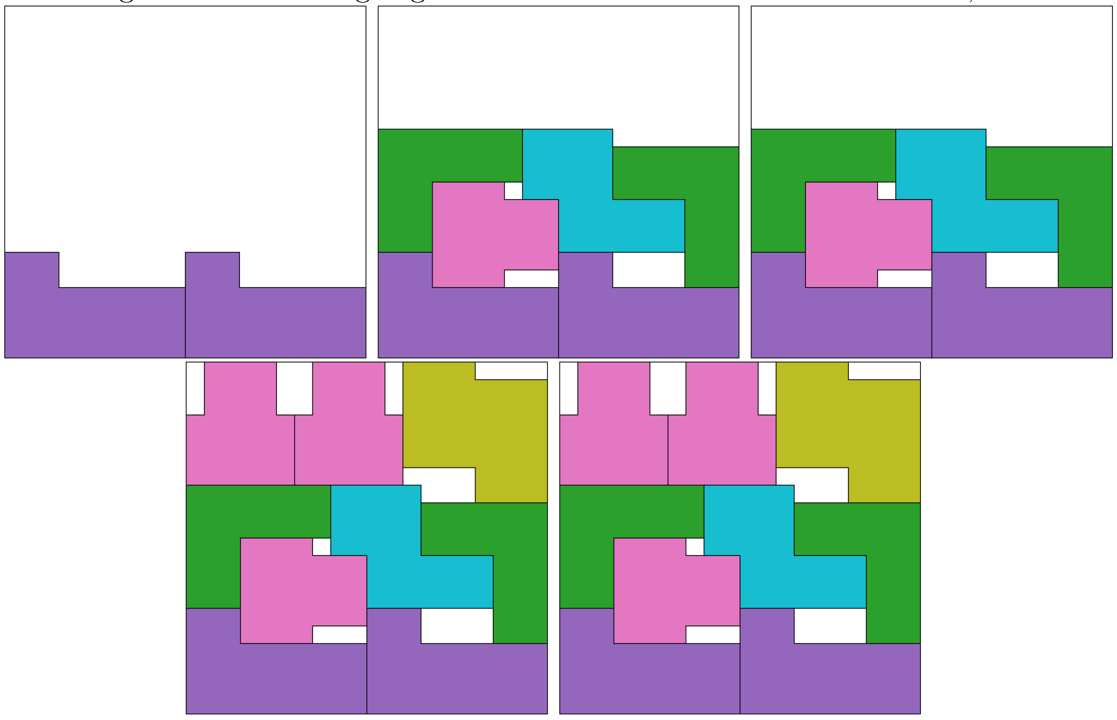
	R4,F0	**R4,F4**	**R4,F5**	**R5,F0**	**R5,F4**	**R5,F5**	**R8,F0**	**R8,F4**	**R8,F5**
1L	1,15	1,10	1,19	**0,99**	1,15	1,11	1,03	1,06	1,59
5Ls	29,71	26,57	30,67	11,24	21,19	21,83	17,49	**8,20**	93,29
1J	0,82	0,62	0,73	0,78	0,57	0,75	0,50	0,39	**0,37**
5Js	20,09	31,33	45,11	**19,67**	43,11	41,37	26,16	28,36	36,19
1T	0,74	0,72	0,80	0,80	0,64	0,66	0,52	0,60	**0,46**
5Ts	44,50	44,83	43,63	54,58	38,67	39,91	42,12	**36,74**	63,59
1S	0,86	0,88	0,88	0,79	0,91	0,77	0,64	0,60	**0,59**
5Ss	32,65	37,68	37,71	29,26	39,14	39,97	**29,04**	137,65	148,86
1Z	0,95	0,66	0,87	0,55	0,76	1,05	0,47	**0,42**	0,44
5Zs	5,57	6,51	6,75	5,58	5,49	5,56	4,40	5,81	**3,80**
1Each	11,30	14,08	17,92	**10,59**	16,96	16,93	11,13	16,26	15,42
5Each	**32,59**	110,96	250,34	92,74	193,45	211,26	92,05	80,43	322,11

Source: The Author

Regarding the feasibility of the heuristic, due to the nature of the maximization problem it is impossible for the heuristic to produce an infeasible result. This happens because in the worst-case scenario, no pieces would be placed on the board, and the objective function would result in 0 without breaking any constraints of the model. That is different for the minimization problem, since one of its constraints is stricter: the constraint of the number of replicas. For the minimization problem, this constraint requires the placement of the exact number of pieces B_i, while the maximization problem uses b_i only as an upper limit.

In order to better visualize the procedure of the heuristic, Figure 30 presents the graphical results at the end of each of the five iterations of the relax-and-fix heuristic for the "1Each" instance with $fix = 8$ and $ins = 0$ in sequence from the upper left image, to the lower right one:

Figure 30: Resulting Figures for the "1Each" Instance with "R8,F0"



Source: The Author

Which combination of the fix and ins parameters is best is a debatable topic, since there is not a clear winner in neither the run time, nor the objective results. Based on the objective value, the two most attractive options are "R4,F4" and "R8,F5", since they have the highest number of best values. Analyzing the run times, it is noticeable that the run times for "R8,F5" are erratic, since the combination of parameters possesses several worst and best values. Therefore, the chosen best combination of parameters is the more stable one, "R4,F4".

In order to evaluate the efficiency of the heuristic in comparison to the monolithic model, two indicators are analyzed: objective and run time. To analyze both of these results, the "R4,F4" relax-and-fix heuristic is compared to the monolithic model with normal patterns, since they were also used here. To analyze the objective, the number of instances where the heuristic produced either the same or better results than the monolithic model is calculated, and to analyze the run time, the ratio between the results of each instance from the “R4,F4” heuristic and the monolithic model is calculated. The resulting values are, respectively: 8 out of the 12 instances and 2.2 times. This means that in only 4 instances the heuristic did not produce the best results, and that the heuristic actually took more time than the monolithic model to run. The latter insight calls for a more intricate analysis, since it is expected that an heuristic runs faster than its monolithic model. Analyzing the data in more detail, it is possible to see that the heuristic performs better for more complex instances with more items. For example, the monolithic model could not prove an optimal result for the “5Each” instance in 1800 seconds, while the relax-and-fix heuristic reached a result in approximately 110 seconds, more than ten times faster, and the result is only worse by a value of 8. This observation is probably due to the fact that the simple instances of the maximization monolithic model ran so fast, that the iterative nature of heuristic ended up making it take longer to run these instances. Furthermore, the average run time of the instances for the monolithic model with normal patterns is 174.5 seconds, while for the relax-and-fix heuristic with normal patterns is 23.0 seconds, indicating that the heuristic is indeed faster.

As a conclusion, it is possible to state that the relax-and-fix heuristic for the maximization problem is effective, since it produces objective values similar to the monolithic model, but, although fast, the heuristic is faster for more complex instances. This is favorable for this heuristic in a practical setting, since it makes more sense to use heuristics for more complex problems that take too long for the monolithic model to run.

5.2.3 Fix-and-Optimize Heuristic

For the maximization fix-and-optimize heuristic, 3 different values for *opt* were compared, using the worst combination of the relax-and-fix maximization parameters $fix = 5$ and $ins = 0$ (or “R5,F0”) (see Section 4.2). The tested values of the *opt* parameter where {4,5,8}, representing, respectively, one tenth, one eighth and one fifth of the board width ($W = 40$). Furthermore, all the instances used normal patterns. The results for the fix-and-optimize heuristic of the maximization problem can be seen in Table 6, which presents the values for the objective function. Furthermore, “R” represents the value of the fix parameter, “F” of the *ins* parameter, and “O” of the *opt* parameter.

Table 6: Results for the Fix-and-Optimize Maximization Problem - Objective

	R5,F0,O4	R5,F0,O5	R5,F0,O8
1L	1312	1312	1312
5Ls	1592	1592	1592
1J	1600	1600	1600
5Js	1548	1548	1548
1T	1184	1184	1184
5Ts	1412	1412	1412
1S	1216	1216	1216
5Ss	1392	1392	1392
1Z	1080	1080	1080
5Zs	1416	1416	1416
1Each	1436	1436	1436
5Each	1560	1560	1560

Source: The Author

Determining which value of the *opt* parameter is best is not possible for this data set, since every solution for every instance.

In order to evaluate the efficiency of this heuristic in comparison to the relax-and-fix heuristic only the objective is analyzed. To analyze this result, any of the tested results is compared to the "R5,F0" relax-and-fix heuristic, since they all have the same results. To analyze the objective, the number of instances where this heuristic produced either the same or better results than the relax-and-fix heuristic is calculated. The resulting value is 0 out of the 12 instances. This result calls for a more intricate analysis, since it is expected that the results of the relax-and-fix heuristics are improved by the fix-and-optimize heuristic. Analyzing the data in more detail, it is possible to see that 6 out of the 12 instances already possessed the same objective value in "R5,F0" as the optimal solution from the monolithic model, which means that the fix-and-optimize heuristic could only possibly improve the results of the 6 remaining non-optimal solutions, half the number of tested instances. Furthermore, these 6 remaining solutions were, on average, 2.2% away from the solution obtained in the monolithic model, what gave the fix-and-optimize heuristic a very thin margin for optimization.

Unfortunately, no solid conclusion regarding the effectiveness of the fix-and-optimize heuristic for the maximization problem can be made due to the limitations of the dataset used for testing.

5.3 Results for the Minimization Problem

In this section, the results from both the monolithic model and the heuristic for the minimization problem are discussed. As described in Chapter 3, the aim of the model is minimizing the value of the variable height $\overline{W}$, compacting the pieces as much as possible.

5.3.1 Monolithic Model

In Table 7, the results for the monolithic model of the minimization problem can be seen. Two scenarios were compared, one where the full sets for grid discretization were used, and another where the adapted normal patterns were used (see Section 3.3). The values in bold indicate that the time limit was reached for these instances, and "NSF" indicates that no solutions that comply with the constraints were found in the given time limit.

Table 7: Results for the Monolithic Model Minimization Problem

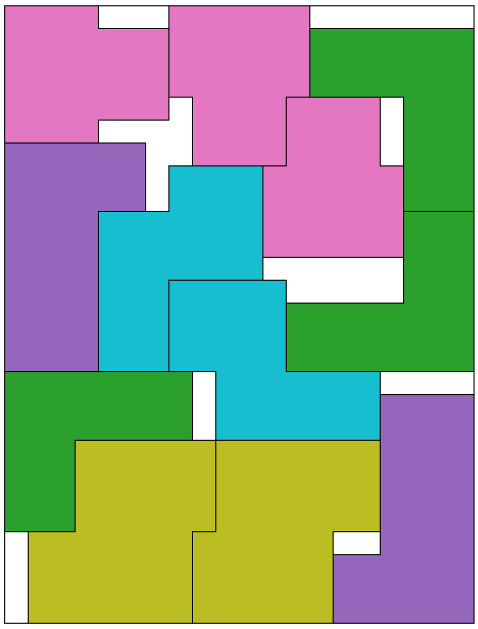
		1L	**5Ls**	**1J**	**5Js**	**1T**	**5Ts**
Full Sets	**Objective**	46	66	40	58	50	55
	Run Time	127,87	**1800,12**	5,86	**1800,19**	101,06	**1800,55**
	Abs GAP	0,00	-16,00	0,00	-2,00	0,00	-6,00
	Rel GAP	0,00%	24,24%	0,00%	3,45%	0,00%	10,91%
	Iterations	51269	598572	8233	416446	34679	760679
	Equations	20767	79515	22087	79515	20767	79515
	Variables	22993	118833	25009	117985	24065	123121
With NP	**Objective**	46	52	40	56	50	54
	Run Time	8,24	**1800,15**	1,35	235,33	8,14	**1800,16**
	Abs GAP	0,00	-2,00	0,00	0,00	0,00	-4,00
	Rel GAP	0,00%	3,85%	0,00%	0,00%	0,00%	7,41%
	Iterations	12594	6359406	3161	440889	9572	4201992
	Equations	8462	23697	8797	23697	8462	23697
	Variables	5997	30965	6509	30749	6269	32053

		1S	**5Ss**	**1Z**	**5Zs**	**1Each**	**5Each**
Full Sets	**Objective**	54	54	52	52	68	NSF
	Run Time	149,84	**1800,60**	124,13	**1800,36**	**1800,31**	NSF
	Abs GAP	0,00	-6,00	0,00	-2,00	-18,00	NSF
	Rel GAP	0,00%	11,11%	0,00%	3,85%	26,47%	NSF
	Iterations	65742	464031	37166	557817	313137	NSF
	Equations	20767	79515	20767	79515	73971	NSF
	Variables	23513	119289	22489	117025	115057	NSF
With NP	**Objective**	54	52	52	52	54	92
	Run Time	12,23	**1800,14**	3,70	150,49	782,76	**1800,30**
	Abs GAP	0,00	-3,00	0,00	0,00	0,00	-12,00
	Rel GAP	0,00%	5,77%	0,00%	0,00%	0,00%	13,04%
	Iterations	14482	4519570	5368	353766	1670203	587831
	Equations	8462	23697	8462	23697	22290	98197
	Variables	6129	31081	5869	30505	30005	156197

Source: The Author

All the values of the objective function have physical interpretations, and in order to visualize the physical manifestations of the results, Figure 31 shows the resulting image for one of the instances:

Figure 31: Resulting Figure for the “1Each” Instance with Normal Patterns



Source: The Author

Similarly to the maximization problem, in order to analyze the influence of the normal patterns in the minimization model, four indicators are analyzed: run time, iterations, equations and variables. To analyze them, the ratio between the results of each instance without and with normal patterns was calculated and averaged, with exception of the instances with all the item types, since the version with full sets could not produce results. The resulting ratios are, respectively: 9.4, 2.3, 2.9, 3.8. The ratios show that the normal patterns have a significant influence in the minimization model, since their use greatly reduces the run time and the iterations that the model needs, and has a smaller, but significant impact in the number of equations and variables. However, when compared to the maximization model (see section 5.2.1) the influence of the normal patterns is not as impactful. Furthermore, we can see in Table 7 that in multiple instances the model with full sets either was not able to find the optimal solution while the one with normal patterns was, or the one with normal patterns found a better solution, although neither version drove the GAP to 0. Furthermore, the impact of normal patterns was so great in the “5Each” instance, that with them, the model was able to find a feasible solution, what it was not able to do without them.

In order to analyze the influence of the number of items in the model, the same four indicators are analyzed: run time, iterations, equations and variables. To analyze them, the ratio between the results of each instance with five times more items and the one with 5 times less items was calculated and averaged only for the version with normal patterns.

The resulting ratios are, respectively: 134.0, 243.6, 3.1, 5.1. The ratios show that the number of items has a great influence in the minimization model, even larger than the normal patterns, in all indicators, since their use reduces greatly the run time and the iterations that the model needs, and has a smaller, but significant impact in the number of equations and variables. Compared to the maximization problem, the influence of the number of items is smaller but still very impactful.

In conclusion, the developed model is able to contemplate the minimization cutting and packing problem with tetris-like items, and the adapted normal patterns are useful and provide great results.

5.3.2 Relax-and-Fix Heuristic

For the minimization relax-and-fix heuristic, 9 combinations of the relax-and-fix heuristic parameters were compared (see Section 4.1). The tested values of the fix parameter where {10,15,20}, representing, respectively, one tenth, approximately one seventh and one fifth of the initial board width ($W = 100$), while the ones of the ins parameter where {0,10,15}. Furthermore, all the instances used normal patterns. The results for the relax-and-fix heuristic of the minimization problem can be seen in Table 8 and Table 9. The first presents the values for the objective function and the second for the run time. The values in bold indicate that the value corresponds to the best for the instance, and "NSF" indicates that no solutions that comply with the constraints were found in the given time limit in a given iteration. Furthermore, "R" represents the value of the fix parameter, and "F" of the ins parameter.

Table 8: Results for the Relax-and-Fix Minimization Problem - Objective

	R10,F0	**R10,F10**	**R10,F15**	**R15,F0**	**R15,F10**	**R15,F15**	**R20,F0**	**R20,F10**	**R20,F15**
1L	60	60	54	54	54	54	54	50	**48**
5Ls	62	62	60	60	56	60	56	58	**54**
1J	52	52	52	52	48	52	46	50	**40**
5Js	66	66	70	64	64	62	**60**	**60**	**60**
1T	56	56	64	**52**	62	**52**	**52**	**52**	58
5Ts	62	62	60	62	58	62	**56**	58	58
1S	56	56	62	**54**	62	**54**	56	56	**54**
5Ss	60	60	58	56	56	56	56	**54**	56
1Z	**54**	**54**	62	**54**	62	**54**	**54**	**54**	58
5Zs	60	60	66	58	58	58	**54**	**54**	58
1Each	60	60	68	60	62	60	**58**	**58**	**58**
5Each	NSF	NSF	NSF	NSF	NSF	NSF	NSF	NSF	NSF

Source: The Author

Table 9: Results for the Relax-and-Fix Minimization Problem - Run Time

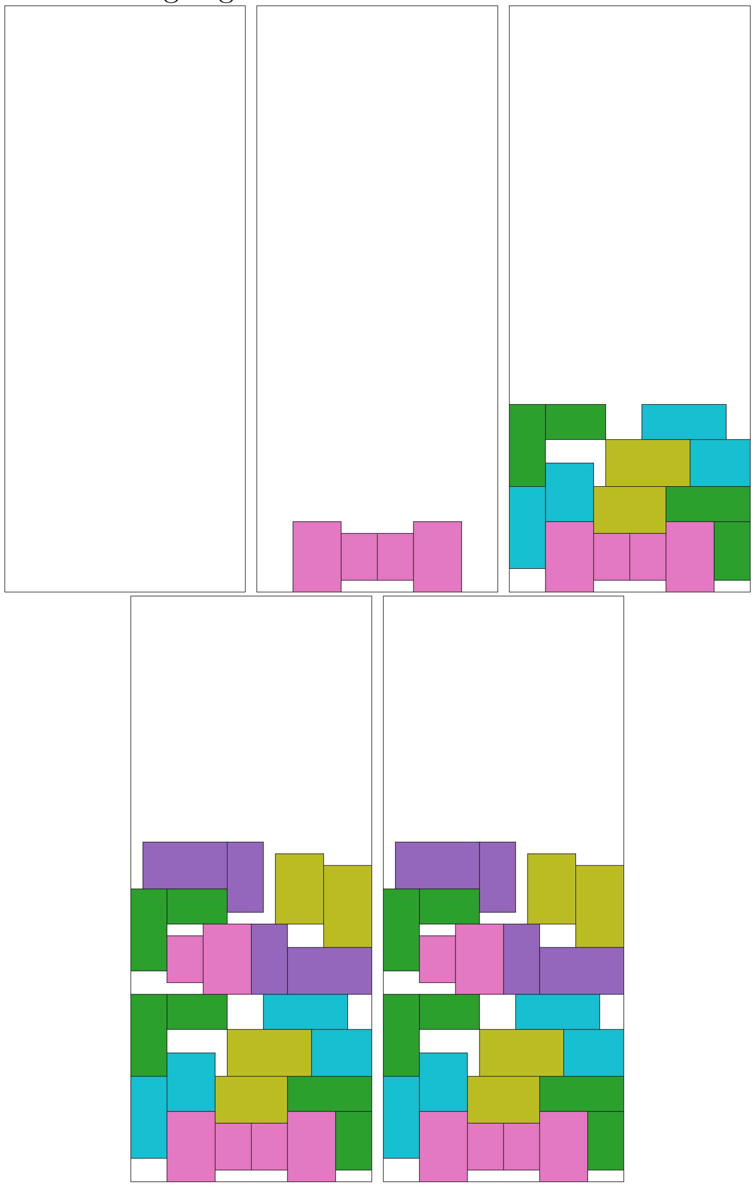
	R10,F0	**R10,F10**	**R10,F15**	**R15,F0**	**R15,F10**	**R15,F15**	**R20,F0**	**R20,F10**	**R20,F15**
1L	48,74	49,14	38,51	49,91	33,59	43,19	38,25	35,85	**26,33**
5Ls	430,83	**345,01**	351,03	857,47	677,02	732,86	725,55	626,61	455,84
1J	71,61	67,46	54,14	52,63	41,59	48,25	59,97	32,33	**27,26**
5Js	331,84	**294,27**	406,69	706,58	589,72	478,49	914,74	654,40	730,36
1T	43,41	40,80	29,73	30,98	26,37	26,32	24,36	26,23	**23,36**
5Ts	575,95	549,23	616,71	793,43	674,58	687,61	784,71	604,10	**526,03**
1S	37,92	30,98	36,73	32,01	24,74	27,83	27,61	**19,40**	22,11
5Ss	790,80	767,03	801,98	523,02	582,26	463,41	480,05	**435,68**	454,09
1Z	43,79	40,21	41,57	38,54	30,86	35,40	30,32	35,71	**22,62**
5Zs	449,93	492,26	555,77	442,14	323,43	442,85	**259,19**	373,51	367,58
1Each	438,81	**394,75**	554,56	510,29	633,49	601,45	482,70	627,74	514,10
5Each	NSF	NSF	NSF	NSF	NSF	NSF	NSF	NSF	NSF

Source: The Author

Regarding the feasibility of the heuristic, although it is possible due to the strict nature of the minimization problem that the relax-and-fix heuristic will run, at some point, into an infeasible solution, this did not occur with the generated instances.

In order to better visualize the procedure of the heuristic, Figure 32 presents the graphical results at the end of each of the five iterations of the relax-and-fix heuristic for the “1Each” instance with $fix = 20$ and $ins = 15$ in sequence:

Figure 32: Resulting Figures for the “1Each” Instance with “R20,F15”



Source: The Author

Unlike the maximization problem, the best combination of the *fix* and *ins* parameters is fairly clear. Based on the objective value, the most attractive options are the ones with “R20”, the largest *fix* value, since they have the highest number of best values. Analyzing the run times, the combination of parameters with the highest number of best values is “R20,F15”. Therefore, the chosen best combination of parameters is “R20,F15”.

Similarly to the maximization problem, in order to evaluate the efficiency of the heuristic in comparison to the monolithic model, two indicators are analyzed: objective and run time. To analyze both of these results, the same calculations as the ones in the maximization problem are used, however, for the run time ratio, the data from the “5Each” instance was excluded, since the heuristic could not find any solutions in time.

The resulting values are, respectively: 2 out of the 12 instances and 2.2 times. This means that in only 2 instances the heuristic produced the best results and that the heuristic actually took more time than the monolithic model to run. The first insight stands out since the results are significantly different than the ones for the maximization problem, which were much better. One reason might be able to explain this: a possible shortcoming of the heuristic. The minimization problem is marked by a constant conflict between two objectives: minimizing the board width W_{var} and placing the item replicas, and it is possible that, if the item placement was somehow more incentivized, the first iterations would be smarter and would place more items instead of focusing on minimizing the board width, what might help future iterations by diminishing the number of pieces that must still be placed in later iterations.

The run time ratio of 2.2 also calls for a more intricate analysis, and, similarly to the maximization problem, it is possible to see that the heuristic performs better for more complex instances with more items. For example, the monolithic model could not prove an optimal result for the "5Ts" instance in 1800 seconds, while the relax-and-fix heuristic reached a result in approximately 523.0 seconds, more than three times faster, and the result is only worse by a value of 4. Furthermore, the instances "5Ls" and "5Ss" present a similar story. The same explanation of the extremely fast run times for the simple instances and the iterative nature of the heuristic also work for the minimization problem. Furthermore, the average run time of the instances for the monolithic model with normal patterns is 600.1 seconds, while for the relax-and-fix heuristic with normal pattern it is 288.2 seconds, indicating that the heuristic is indeed faster.

As a conclusion, it is possible to state that the relax-and-fix heuristic for the minimization problem is not so effective at reaching close-to-optimal solutions, and, although fast, the heuristic is faster for more complex instances. The heuristic is useful, though, for generating initial solutions to be refined in later heuristics.

5.3.3 Fix-and-Optimize Heuristic

For the minimization fix-and-optimize heuristic, 3 different values for *opt* were compared using the worst combination of the relax-and-fix minimization parameters, $fix = 10$ and $ins = 15$ (or "R10,F15") (see Section 4.2). The tested values of the *opt* parameter where {10,15,20}, representing, respectively, one tenth, approximately one seventh and one fifth of the initial board width ($W = 100$). Furthermore, all the instances used normal patterns. The results for the relax-and-fix heuristic of the minimization problem can be seen in Table 10, which presents the values for the objective function and has the best value for an instance in bold:

Table 10: Results for the Fix-and-Optimize Minimization Problem - Objective

	R10,F15,O10	**R10,F15,O15**	**R10,F15,O20**
1L	54	**52**	54
5Ls	60	60	60
1J	52	52	52
5Js	70	70	70
1T	64	64	**62**
5Ts	60	60	60
1S	62	62	62
5Ss	58	58	58
1Z	62	62	62
5Zs	66	66	66
1Each	66	68	**64**
5Each	NSF	NSF	NSF

Source: The Author

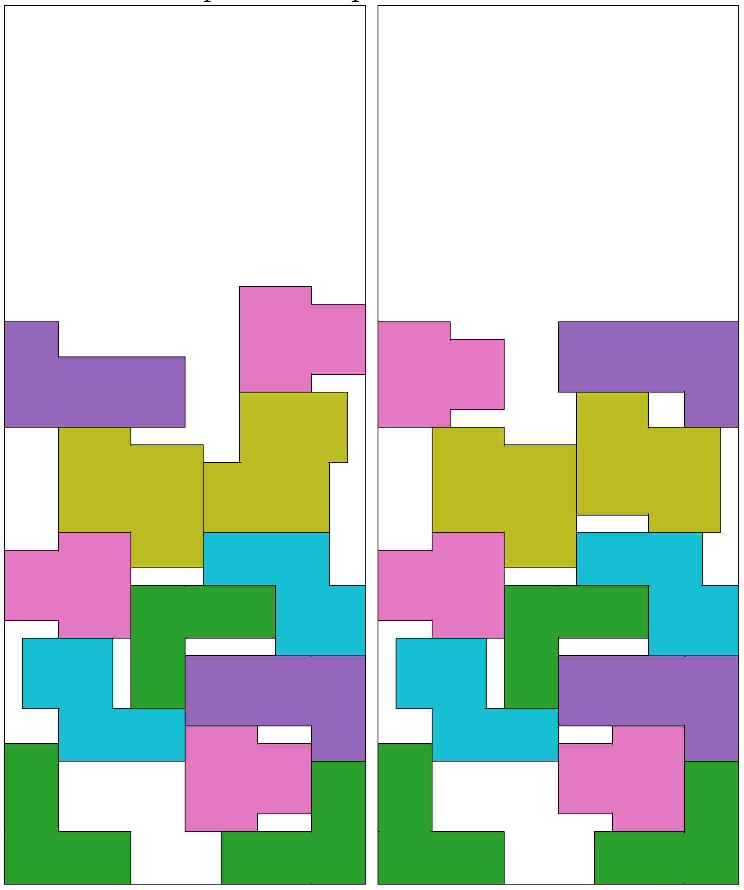
Regarding the feasibility of the heuristic, due to the nature of the fix-and-optimize heuristic and the fact that the initial solutions were feasible except for the “5Each” instance, it is impossible for the heuristic to produce an infeasible result. This is because, in the worst-case scenario of an iteration, the fixed values of the variables would simply remain the same, what cannot make the problem infeasible.

Unlike the maximization problem, the best value of the *opt* parameter is fairly clear. Based on the objective value, the most attractive option if *opt* = 20, since it was the only one to produce two improved results.

Similarly to the maximization problem, in order to evaluate the efficiency of the heuristic in comparison to the relax-and-fix heuristic, only the objective is analyzed. To analyze this result, the “O20” heuristic is compared to the “R10,F15” relax-and-fix heuristic. The same calculation as the one in the maximization problem is used. The resulting value is 2 out of the 11 instances, since the “5Each” instance had an infeasible solution and could not possibly be improved. This result calls for a more intricate analysis, since the result from the maximization problem was of low quality and this could have happened here too. Analyzing the data in more detail, it is possible to see that the fix-and-optimize heuristic could improve the results of all 11 feasible initial solutions, since none of them were optimal. Furthermore, these 11 solutions were, on average, 14.4% away from the solution obtained in the monolithic model, what gave the fix-and-optimize heuristic a good margin for optimization. This means that the data used in the minimization version was of good quality, contrary to the one in the maximization version.

The resulting value of 2 improvements out of the 11 possible ones indicates a low effectiveness of the heuristic, and also Figure 33 shows a phenomenon that was already expected: the fix-and-optimize heuristic only seems to be slightly effective at the top the board, where the variable width $\overline{W}$ can actually be optimized. In this figure, the image on the left shows the result obtained with the “R10,F15” relax-and-fix heuristic, while the image on the right shows the improved result obtained with the “R10,F15,O20” fix-and-optimize heuristic for the “1Each” instance:

Figure 33: Fix-and-Optimize Improvement for the “1Each” Instance



Source: The Author

As a conclusion, it is possible to state that the fix-and-optimize heuristic for the minimization problem is not effective, although it has caused more impact in the results than the heuristic for the maximization problem.

6 FUTURE PROSPECTS

In this chapter, the summary of the results of the monolithic models and the heuristics is discussed, as well as the limitations of the performed research, and possible future improvements to be made.

Regarding the monolithic models, it is possible to state that this graduation thesis was capable of describing two novelty cutting and packing models for extensions/variants of the 2D-I-IIPP/SLOPP/SKP and 2D-I-ODP problems. Furthermore, the effectiveness of the adapted normal patterns was tested and substantiated by the results for both kinds of assignments. This means that this thesis was able to provide a significant literature that expands across many real-world applications, considering the higher-level cutting and packing problems which the models concern, and that has direct applications in the design of phased array antennas, metal stamping, design of printed circuits boards, timber cutting and layout of newspaper pages, given that the polyomino literature is tied to these industries.

There are two identified possible limitations of the monolithic models for both problems: the mathematical modeling of the tetris-like items and the method for grid discretization. The first limitation derives from the fact that the developed mathematical models for these items create one constraint for each item type, for each orientation, and for each possible (p, q) combination, what causes a heavy burden on the model by greatly increasing the number of equations in it. To improve upon this, it is possible that the positional relation between the two rectangles in each item can be written differently with less constraints, or that the same problem can be modelled with a different logic, possibly non-linearly. In order to determine the difference in efficiency of this new mathematical description of the problems studied in this thesis, in a future research a battery of computational tests would have to be performed and the results would have to be compared with the ones obtained in this thesis.

Moreover, the developed adaptation of the normal patterns results in the generation of relatively many discretizations on the board, compared to the traditional method. Part of the reason for this, is that, due to the irregular and concave nature of the tetris-like items, the proof that the normal patterns generation does not loose relevant solutions is not valid in this case, and the adaptations were made to contemplate the nuances from these irregular pieces. This means that, although the adapted normal patterns did not seem to negatively affect the results, only improve them, there is no formal proof that no information is being lost with their use. Future research could aim at improving the grid discretization algorithm itself to reduce the number of discretizations, and could also prove the adequacy of the adapted normal patterns for the problems contemplated in this thesis.

Regarding the maximization and minimization relax-and-fix heuristics, it is possible to state that this graduation thesis was capable of developing an effective and efficient relax-and-fix heuristic that works on both kinds of assignment, although better in the maximization one. Furthermore, the optimal combinations of relax-and-fix parameters were reached for the generated instances, and could arguably be generalized by taking the proportion of the board to which the parameter is related, to be used in other instances or in future researches. This, however, is not certain to result in the best combination of parameters for new instances, since the determination of the best combination of parameters depends on many factors, which were not analyzed in this thesis, including: sizes of the tetris-like items, using values v_i that are not equal to an item's area and using rectangular items in the instances.

The main identified limitation of the developed relax-and-fix-heuristic is that the set of parameters combinations (9 for each kind of assignment) might not be comprehensive enough, and more data points could have been gathered to further consolidate the obtained results. This was not performed in this thesis due to deadline limitations, and the 9 studied pairs of parameters contemplated significantly different proportions of the board, generating solid results. This means that, although not so significant, future research could expand upon the developed test dataset with more instances and combination.

Regarding the maximization and minimization fix-and-optimize heuristics, it is possible to state that the developed algorithm was not effective at improving the relax-and-fix solution, although this statement can be challenged in future researches by increasing the number of tests in the maximization problem. These results show that the fix-and-optimize heuristics are the subjects of this thesis with the most impactful limitations. The first and most crucial limitation comes from the tests performed in the maximization version of the problem. The initial solutions given to this heuristic for the minimization problems were sub-optimal and numerous enough, that concluding that the developed heuristic is not effective in this case is logical and solid. However, this is not the case for the maximization problem, where the initial solutions given to the heuristic where close to optimal, with 6 of them already being optimal solutions. This decreased the quality of the test data significantly and made it difficult to conclude whether the maximization fix-and-optimize heuristic is effective in itself, since the given initial solutions were had such a small margin for improvement. This reformulation of the testing data could not be performed in this thesis due to deadlines, but it is encouraged that future researches test far from optimal initial solutions for the fix-and-optimize heuristic. These solutions could be generated by hand or with a new, weaker heuristic. With this methodology it would be possible to distinguish whether the fix-and-optimize heuristic in indeed inefficient for this cutting and packing problem, or it can be useful in situations where the initial solution is far from optimal.

Furthermore, a new heuristic tailored to the minimization problem could be developed, since the relax-and-fix heuristic was only effective at generating sub-optimal initial solutions for this assignment, and the fix-and-optimize heuristic was practically ineffective. Two ideas for this new heuristic are: incentivizing the placement of the pieces using a modification of the objective function, and minimizing a local variable height at each iteration that concerns the "variables to fix" horizon, instead of minimizing the "global" $\overline{W}$ variable height. For the second idea to work it would be necessary to also lower the y-axis positions of the pieces fixed above the "variables to fix" horizon if the local variable height was minimized.

In conclusion, this thesis provided valuable models, heuristics and insights for the cutting and packing problems literature, while also developing a basis for future researches and improvements upon this graduation thesis.

REFERENCES

ABSI, N.; HEUVEL, W. V. D. Worst case analysis of Relax and Fix heuristics for lot-sizing problems. *European Journal of Operational Research*, v. 279, n. 2, p. 449–458, dez. 2019. ISSN 03772217.

ABSI, N.; Kedad-Sidhoum, S. MIP-based heuristics for multi-item capacitated lot-sizing problem with setup times and shortage costs. *RAIRO - Operations Research*, v. 41, n. 2, p. 171–192, abr. 2007. ISSN 0399-0559, 1290-3868.

AKARTUNALI, K.; MILLER, A. J. A heuristic approach for big bucket multi-level production planning problems. *European Journal of Operational Research*, v. 193, n. 2, p. 396–411, mar. 2009. ISSN 03772217.

ARAUJO, S. A. D.; ARENALES, M. N.; CLARK, A. R. Joint rolling-horizon scheduling of materials processing and lot-sizing with sequence-dependent setups. *Journal of Heuristics*, v. 13, n. 4, p. 337–358, ago. 2007. ISSN 1381-1231, 1572-9397.

BABU, A. R.; BABU, N. R. A generic approach for nesting of 2-D parts in 2-D sheets using genetic and heuristic algorithms. *Computer-Aided Design*, v. 33, n. 12, p. 879–891, out. 2001. ISSN 00104485.

BAENA, D.; CASTRO, J.; GONZÁLEZ, J. A. Fix-and-relax approaches for controlled tabular adjustment. *Computers & Operations Research*, v. 58, p. 41–52, jun. 2015. ISSN 03050548.

BALAS, E. et al. Octane: A New Heuristic for Pure 0–1 Programs. *Operations Research*, v. 49, n. 2, p. 207–225, abr. 2001. ISSN 0030-364X, 1526-5463.

BALAS, E.; MARTIN, C. H. Pivot and Complement–A Heuristic for 0-1 Programming. *Management Science*, v. 26, n. 1, p. 86–96, jan. 1980. ISSN 0025-1909, 1526-5501.

BEASLEY, J. E. An Exact Two-Dimensional Non-Guillotine Cutting Tree Search Procedure. *Operations Research*, v. 33, n. 1, p. 49–64, fev. 1985. ISSN 0030-364X, 1526-5463.

BEKRAR, A. et al. An improved heuristic and an exact algorithm for the 2D strip and bin packing problem. *International Journal of Product Development*, v. 10, n. 1/2/3, p. 217, 2010. ISSN 1477-9056, 1741-8178.

BENNELL, J. A.; OLIVEIRA, J. F. The geometry of nesting problems: A tutorial. *European Journal of Operational Research*, v. 184, n. 2, p. 397–415, 2008. ISSN 0377-2217.

BENNELL, J. A.; OLIVEIRA, J. F. A tutorial in irregular shape packing problems. *Journal of the Operational Research Society*, v. 60, n. sup1, p. S93–S105, maio 2009. ISSN 0160-5682, 1476-9360.

BETTINELLI, A.; CESELLI, A.; RIGHINI, G. A branch-and-price algorithm for the two-dimensional level strip packing problem. *4OR*, v. 6, n. 4, p. 361–374, dez. 2008. ISSN 1619-4500, 1614-2411.

BEZERRA, V. M. R. et al. Models for the two-dimensional level strip packing problem – a review and a computational evaluation. *Journal of the Operational Research Society*, v. 71, n. 4, p. 606–627, abr. 2020. ISSN 0160-5682, 1476-9360.

BODINI, O. Tiling a rectangle with polyominoes. In: MORVAN, M.; RÉMILA, E. (Ed.). *Discrete Models for Complex Systems, DMCS'03, Lyon, France, June 16-19, 2003.* [S.l.]: DMTCS, 2003. (DMTCS Proceedings, AB), p. 81–88.

CHERRI, L. H. et al. A MODEL-BASED HEURISTIC FOR THE IRREGULAR STRIP PACKING PROBLEM. *Pesquisa Operacional*, v. 36, n. 3, p. 447–468, dez. 2016. ISSN 0101-7438.

CHERRI, L. H.; CHERRI, A. C.; SOLER, E. M. Mixed integer quadratically-constrained programming model to solve the irregular strip packing problem with continuous rotations. *Journal of Global Optimization*, v. 72, n. 1, p. 89–107, set. 2018. ISSN 0925-5001, 1573-2916.

CHERRI, L. H. et al. Robust mixed-integer linear programming models for the irregular strip packing problem. *European Journal of Operational Research*, v. 253, n. 3, p. 570–583, set. 2016. ISSN 03772217.

CHRISTOFIDES, N.; WHITLOCK, C. An Algorithm for Two-Dimensional Cutting Problems. *Operations Research*, v. 25, n. 1, p. 30–44, fev. 1977. ISSN 0030-364X, 1526-5463.

CLARK, A. R. Optimization approximations for capacity constrained material requirements planning. *International Journal of Production Economics*, v. 84, n. 2, p. 115–131, maio 2003. ISSN 09255273.

DASTJERD, N. K.; ERTOGRAL, K. A fix-and-optimize heuristic for the integrated fleet sizing and replenishment planning problem with predetermined delivery frequencies. *Computers & Industrial Engineering*, v. 127, p. 778–787, jan. 2019. ISSN 03608352.

DILLENBERGER, C. et al. On practical resource allocation for production planning and scheduling with period overlapping setups. *European Journal of Operational Research*, v. 75, n. 2, p. 275–286, jun. 1994. ISSN 03772217.

DORNELES, Á. P.; ARAÚJO, O. C. D.; BURIOL, L. S. A fix-and-optimize heuristic for the high school timetabling problem. *Computers & Operations Research*, v. 52, p. 29–38, dez. 2014. ISSN 03050548.

DUTTA, S. *Optimization in Chemical Engineering.* Daryaganj, Delhi, India: Cambridge University Press, is part of the University of Cambridge, 2016. ISBN 978-1-107-09123-8.

DYCKHOFF, H. A typology of cutting and packing problems. *European Journal of Operational Research*, v. 44, n. 2, p. 145–159, jan. 1990. ISSN 03772217.

FASANO, G. A global optimization point of view to handle non-standard object packing problems. *Journal of Global Optimization*, v. 55, n. 2, p. 279–299, fev. 2013. ISSN 0925-5001, 1573-2916.

FEDERGRUEN, A.; MEISSNER, J.; TZUR, M. Progressive Interval Heuristics for Multi-Item Capacitated Lot-Sizing Problems. *Operations Research*, v. 55, n. 3, p. 490–502, jun. 2007. ISSN 0030-364X, 1526-5463.

FEKETE, S. P.; KAMPHANS, T.; SCHWEER, N. Online Square Packing with Gravity. *Algorithmica*, v. 68, n. 4, p. 1019–1044, abr. 2014. ISSN 0178-4617, 1432-0541.

FLOUDAS, C. A.; LIN, X. Mixed Integer Linear Programming in Process Scheduling: Modeling, Algorithms, and Applications. *Annals of Operations Research*, v. 139, n. 1, p. 131–162, out. 2005. ISSN 0254-5330, 1572-9338.

FOWLER, R. J.; PATERSON, M. S.; TANIMOTO, S. L. Optimal packing and covering in the plane are NP-complete. *Information Processing Letters*, v. 12, n. 3, p. 133–137, jun. 1981. ISSN 00200190.

FRISKE, M. W.; BURIOL, L. S.; CAMPONOGARA, E. A relax-and-fix and fix-and-optimize algorithm for a Maritime Inventory Routing Problem. *Computers & Operations Research*, v. 137, p. 105520, jan. 2022. ISSN 03050548.

GLOVER, F.; LØKKETANGEN, A.; WOODRUFF, D. L. Scatter Search to Generate Diverse MIP Solutions. In: SHARDA, R. et al. (Ed.). *Computing Tools for Modeling, Optimization and Simulation*. Boston, MA: Springer US, 2000. v. 12, p. 299–317. ISBN 978-1-4613-7062-8 978-1-4615-4567-5.

GOLOMB, S. W. Checker Boards and Polyominoes. *The American Mathematical Monthly*, v. 61, n. 10, p. 675–682, dez. 1954. ISSN 0002-9890, 1930-0972.

GOLOMB, S. W. Tiling with polyominoes. *Journal of Combinatorial Theory*, v. 1, n. 2, p. 280–296, set. 1966. ISSN 00219800.

HARTMANN, S. Packing problems and project scheduling models: An integrating perspective. *Journal of the Operational Research Society*, v. 51, n. 9, p. 1083–1092, set. 2000. ISSN 0160-5682, 1476-9360.

HAWA, A. L.; LEWIS, R.; THOMPSON, J. M. Heuristics for the Score-Constrained Strip-Packing Problem. In: KIM, D.; UMA, R. N.; ZELIKOVSKY, A. (Ed.). *Combinatorial Optimization and Applications*. Cham: Springer International Publishing, 2018. v. 11346, p. 449–462. ISBN 978-3-030-04650-7 978-3-030-04651-4.

HELBER, S.; SAHLING, F. A fix-and-optimize approach for the multi-level capacitated lot sizing problem. *International Journal of Production Economics*, v. 123, n. 2, p. 247–256, fev. 2010. ISSN 09255273.

HERZ, J. C. Recursive Computational Procedure for Two-dimensional Stock Cutting. *IBM Journal of Research and Development*, v. 16, n. 5, p. 462–469, set. 1972. ISSN 0018-8646, 0018-8646.

HINOSTROZA, I.; PRADENAS, L.; PARADA, V. Board cutting from logs: Optimal and heuristic approaches for the problem of packing rectangles in a circle. *International Journal of Production Economics*, v. 145, n. 2, p. 541–546, out. 2013. ISSN 09255273.

JONES, D. R. A fully general, exact algorithm for nesting irregular shapes. *Journal of Global Optimization*, v. 59, n. 2-3, p. 367–404, jul. 2014. ISSN 0925-5001, 1573-2916.

JÚNIOR, A. N. et al. The rectangular two-dimensional strip packing problem real-life practical constraints: A bibliometric overview. *Computers & Operations Research*, v. 137, p. 105521, jan. 2022. ISSN 03050548.

JUNQUEIRA, L. *Modelos de programação matemática para problemas de carregamento de caixas dentro de contêineres*. Tese (Doutorado) — Universidade Federal de São Carlos, São Carlos, 2009.

KARADEMIR, S.; PROKOPYEV, O. A.; MAILLOUX, R. J. Irregular polyomino tiling via integer programming with application in phased array antenna design. *Journal of Global Optimization*, v. 65, n. 2, p. 137–173, jun. 2016. ISSN 0925-5001, 1573-2916.

KASHKOUSH, M. N.; SHALABY, M. A.; ABDELHAFIEZ, E. A. A mixed-integer model for two-dimensional polyominoes strip packing and tiling problems. *International Journal of Operational Research*, v. 15, n. 4, p. 391, 2012. ISSN 1745-7645, 1745-7653.

KENMOCHI, M. et al. Exact algorithms for the two-dimensional strip packing problem with and without rotations. *European Journal of Operational Research*, v. 198, n. 1, p. 73–83, out. 2009. ISSN 03772217.

KIERKOSZ, I.; LUCZAK, M. A hybrid evolutionary algorithm for the two-dimensional packing problem. *Central European Journal of Operations Research*, v. 22, n. 4, p. 729–753, dez. 2014. ISSN 1435-246X, 1613-9178.

KITA, N.; MIYATA, K. Computational design of polyomino puzzles. *The Visual Computer*, v. 37, n. 4, p. 777–787, abr. 2021. ISSN 0178-2789, 1432-2315.

KLARNER, D. A. Packing a rectangle with congruent N-ominoes. *Journal of Combinatorial Theory*, v. 7, n. 2, p. 107–115, set. 1969. ISSN 00219800.

LAGUNA, M.; MARTÍ, R. Heuristics. In: GASS, S. I.; FU, M. C. (Ed.). *Encyclopedia of Operations Research and Management Science*. Boston, MA: Springer US, 2013. p. 695–703. ISBN 978-1-4419-1137-7 978-1-4419-1153-7.

LAI, X. et al. Iterated dynamic thresholding search for packing equal circles into a circular container. *European Journal of Operational Research*, v. 299, n. 1, p. 137–153, maio 2022. ISSN 03772217.

LEAO, A. A. et al. Irregular packing problems: A review of mathematical models. *European Journal of Operational Research*, v. 282, n. 3, p. 803–822, maio 2020. ISSN 03772217.

LINS, L.; LINS, S.; MORABITO, R. An n-tet graph approach for non-guillotine packings of n-dimensional boxes into an n-container. *European Journal of Operational Research*, v. 141, n. 2, p. 421–439, set. 2002. ISSN 03772217.

LIU, M. M. *The Tetris Proof.* 2017.

LO, K.-Y.; FU, C.-W.; LI, H. 3D polyomino puzzle. *ACM Transactions on Graphics*, v. 28, n. 5, p. 1–8, dez. 2009. ISSN 0730-0301, 1557-7368.

LODI, A.; MARTELLO, S.; MONACI, M. Two-dimensional packing problems: A survey. *European Journal of Operational Research*, v. 141, n. 2, p. 241–252, 2002. ISSN 0377-2217.

MERCÉ, C.; FONTAN, G. MIP-based heuristics for capacitated lotsizing problems. *International Journal of Production Economics*, v. 85, n. 1, p. 97–111, jul. 2003. ISSN 09255273.

MURTHY, P. R. *Operations Research.* New Delhi: New Age International, 2007. ISBN 978-81-224-2069-2.

Noor-E-Alam, M.; DOUCETTE, J. Relax-and-fix decomposition technique for solving large scale grid-based location problems. *Computers & Industrial Engineering*, v. 63, n. 4, p. 1062–1073, dez. 2012. ISSN 03608352.

OLIVEIRA, E. D. A. S. et al. Relax-and-Fix Aplicado ao Problema de Corte de Estoque com Data de Entrega. In: *CNMAC 2019 - XXXIX Congresso Nacional de Matemática Aplicada e Computacional.* [S.l.: s.n.], 2020.

OLIVEIRA, J. F. C.; FERREIRA, J. A. S. Algorithms for Nesting Problems. In: FANDEL, G.; TROCKEL, W.; VIDAL, R. V. V. (Ed.). *Applied Simulated Annealing.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1993. v. 396, p. 255–273. ISBN 978-3-540-56229-0 978-3-642-46787-5.

OLIVEIRA, L. D.; SOUZA, C. C. D.; YUNES, T. Improved bounds for the traveling umpire problem: A stronger formulation and a relax-and-fix heuristic. *European Journal of Operational Research*, v. 236, n. 2, p. 592–600, jul. 2014. ISSN 03772217.

POCHET, Y.; WOLSEY, L. A. *Production Planning by Mixed Integer Programming.* New York Berlin: Springer, 2006. (Springer Series in Operations Research and Financial Engineering). ISBN 978-0-387-33477-6.

QUEIROZ, T. A. D.; MIYAZAWA, F. K. Two-dimensional strip packing problem with load balancing, load bearing and multi-drop constraints. *International Journal of Production Economics*, v. 145, n. 2, p. 511–530, out. 2013. ISSN 09255273.

QUEIROZ, T. A. D.; MIYAZAWA, F. K. Order and static stability into the strip packing problem. *Annals of Operations Research*, v. 223, n. 1, p. 137–154, dez. 2014. ISSN 0254-5330, 1572-9338.

REID, M. Tiling Rectangles and Half Strips with Congruent Polyominoes. *Journal of Combinatorial Theory, Series A*, v. 80, n. 1, p. 106–123, out. 1997. ISSN 00973165.

RIEHME, J.; SCHEITHAUER, G.; TERNO, J. The solution of two-stage guillotine cutting stock problems having extremely varying order demands. *European Journal of Operational Research*, v. 91, n. 3, p. 543–552, jun. 1996. ISSN 03772217.

RINALDI, F.; FRANZ, A. A two-dimensional strip cutting problem with sequencing constraint. *European Journal of Operational Research*, v. 183, n. 3, p. 1371–1384, dez. 2007. ISSN 03772217.

ROCHA, P. et al. Constraint Aggregation in Non-linear Programming Models for Nesting Problems. In: FONSECA, R. J.; WEBER, G.-W.; TELHADA, J. (Ed.). *Computational Management Science*. Cham: Springer International Publishing, 2016. v. 682, p. 175–180. ISBN 978-3-319-20429-1 978-3-319-20430-7.

SALTO, C.; ALBA, E.; MOLINA, J. M. Analysis of Distributed Genetic Algorithms for Solving a Strip Packing Problem. In: LIRKOV, I.; MARGENOV, S.; WAŚNIEWSKI, J. (Ed.). *Large-Scale Scientific Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. v. 4818, p. 609–617. ISBN 978-3-540-78825-6 978-3-540-78827-0.

SALTZMAN, R. M.; HILLIER, F. S. A Heuristic Ceiling Point Algorithm for General Integer Linear Programming. *Management Science*, v. 38, n. 2, p. 263–283, fev. 1992. ISSN 0025-1909, 1526-5501.

SCHEITHAUER, G.; TERNO, J. Modeling of packing problems. *Optimization*, v. 28, n. 1, p. 63–84, jan. 1993. ISSN 0233-1934, 1029-4945.

SEGENREICH, S. A.; BRAGA, L. M. P. F. Optimal nesting of general plane figures: A Monte Carlo heuristical approach. *Computers & Graphics*, v. 10, n. 3, p. 229–237, jan. 1986. ISSN 00978493.

SILVA, A. et al. A cutting plane method and a parallel algorithm for packing rectangles in a circular container. *European Journal of Operational Research*, v. 303, n. 1, p. 114–128, nov. 2022. ISSN 03772217.

SILVA, E.; OLIVEIRA, J. F.; WÄSCHER, G. 2DCPackGen: A problem generator for two-dimensional rectangular cutting and packing problems. *European Journal of Operational Research*, v. 237, n. 3, p. 846–856, set. 2014. ISSN 03772217.

SILVEIRA, J. L. D.; MIYAZAWA, F. K.; XAVIER, E. C. Heuristics for the strip packing problem with unloading constraints. *Computers & Operations Research*, v. 40, n. 4, p. 991–1003, abr. 2013. ISSN 03050548.

SILVEIRA, J. L. D.; XAVIER, E. C.; MIYAZAWA, F. K. Two-dimensional strip packing with unloading constraints. *Discrete Applied Mathematics*, v. 164, p. 512–521, fev. 2014. ISSN 0166218X.

STADTLER, H. Multilevel Lot Sizing with Setup Times and Multiple Constrained Resources: Internally Rolling Schedules with Lot-Sizing Windows. *Operations Research*, v. 51, n. 3, p. 487–502, jun. 2003. ISSN 0030-364X, 1526-5463.

SUGI, M. et al. Solution of the Rectangular Strip Packing Problem Considering a 3-Stage Guillotine Cutting Constraint with Finite Slitter Blades. *International Journal of Automation Technology*, v. 14, n. 3, p. 447–458, maio 2020. ISSN 1883-8022, 1881-7629.

TETRIS. *About Tetris®*. 2022.

WÄSCHER, G.; HAUSSNER, H.; SCHUMANN, H. An improved typology of cutting and packing problems. *European Journal of Operational Research*, v. 183, n. 3, p. 1109–1130, dez. 2007. ISSN 03772217.

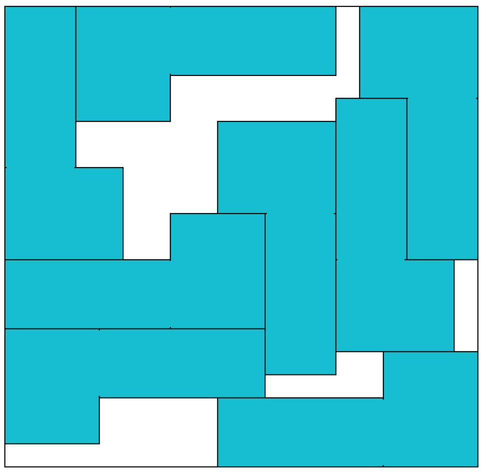
WINSTON, W. L. *Operations Research: Applications and Algorithms*. 3. ed.,[nachdr.]. ed. Belmont, Calif: Duxbury Press, 1997. ISBN 978-0-534-52020-5 978-0-534-20971-1.

WOLSEY, L. A. *Integer Programming*. New York: Wiley, 1998. (Wiley-Interscience Series in Discrete Mathematics and Optimization). ISBN 978-0-471-28366-9.

WU, L. et al. NHACR: A novel heuristic approach for 2D rectangle packing area minimization problem with central rectangle. *Engineering Applications of Artificial Intelligence*, v. 103, p. 104291, ago. 2021. ISSN 09521976.

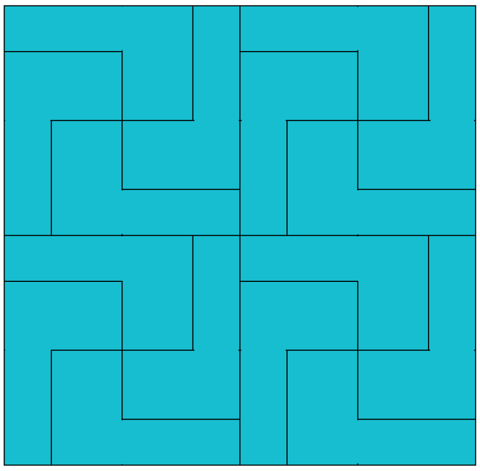
APPENDIX A – IMAGES GENERATED BY THE MAXIMIZATION MONOLITHIC MODEL WITH NORMAL PATTERNS

Solution for the “1L” Instance



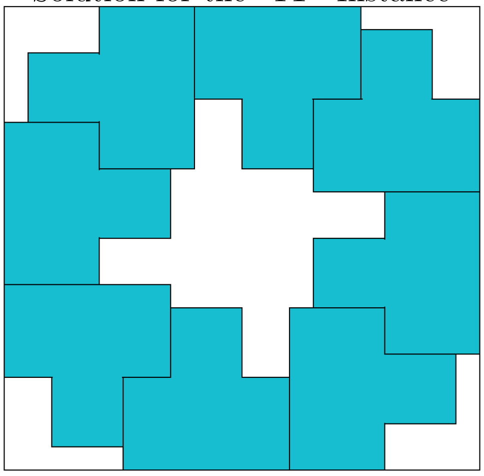
Source: The Author

Solution for the “1J” Instance



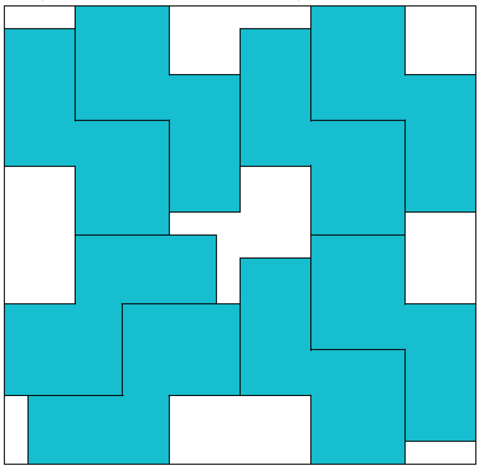
Source: The Author

Solution for the “1T” Instance



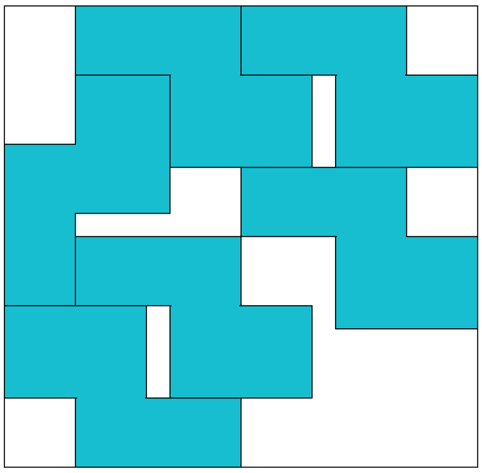
Source: The Author

Solution for the “1S” Instance



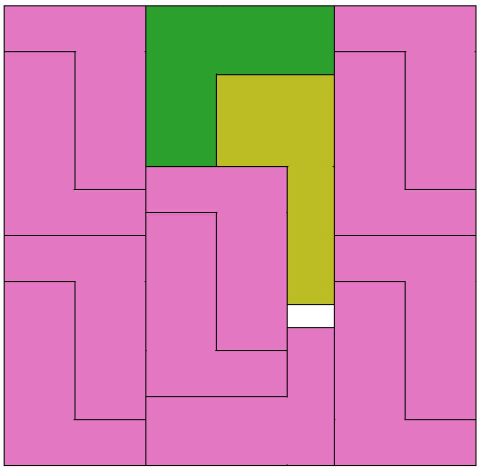
Source: The Author

Solution for the “1Z” Instance



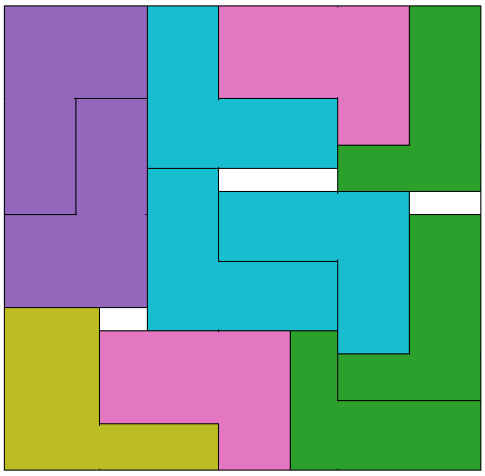
Source: The Author

Solution for the “5Ls” Instance



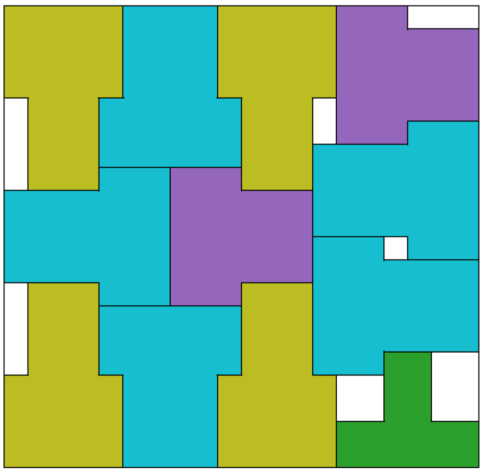
Source: The Author

Solution for the “5Js” Instance



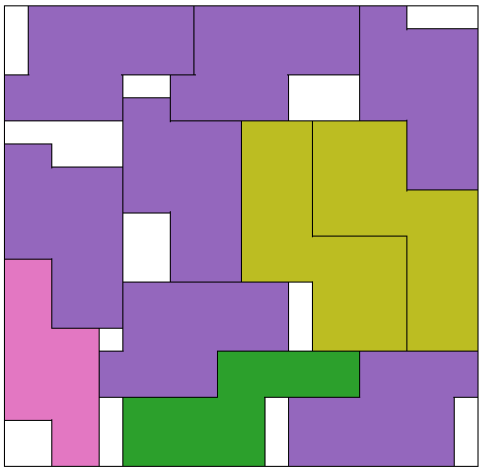
Source: The Author

Solution for the “5Ts” Instance



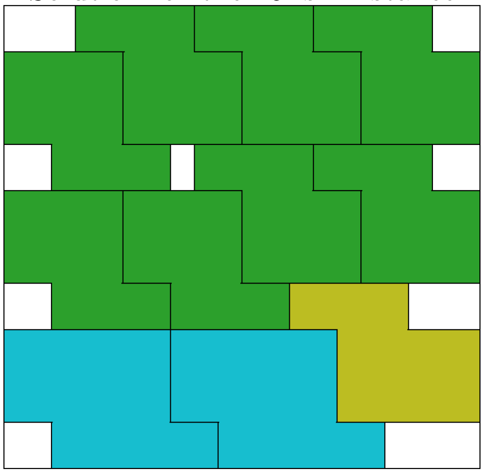
Source: The Author

Solution for the “5Ss” Instance



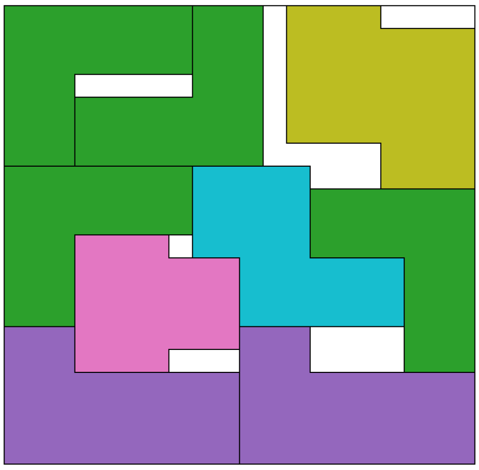
Source: The Author

Solution for the “5Zs” Instance



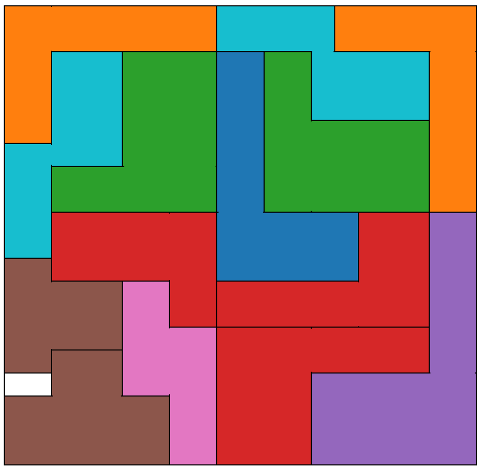
Source: The Author

Solution for the “1Each” Instance



Source: The Author

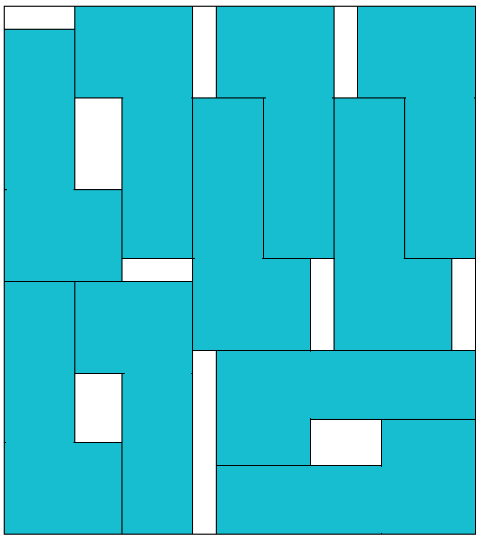
Solution for the “5Each” Instance



Source: The Author

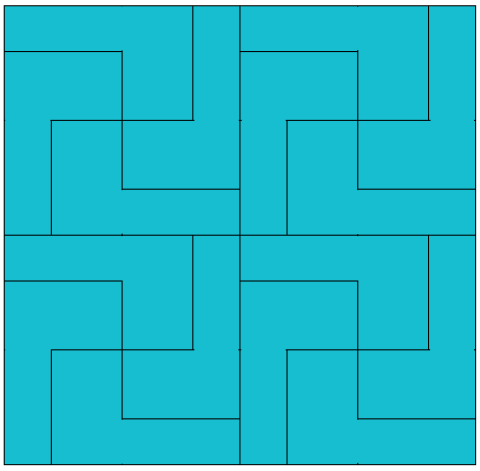
APPENDIX B – IMAGES GENERATED BY THE MINIMIZATION MONOLITHIC MODEL WITH NORMAL PATTERNS

Solution for the “1L” Instance



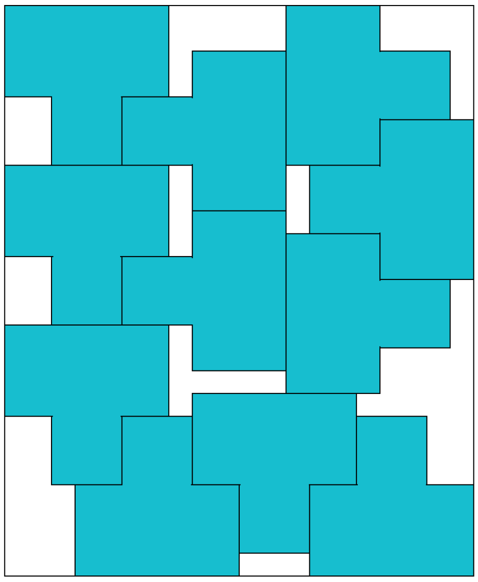
Source: The Author

Solution for the "1J" Instance



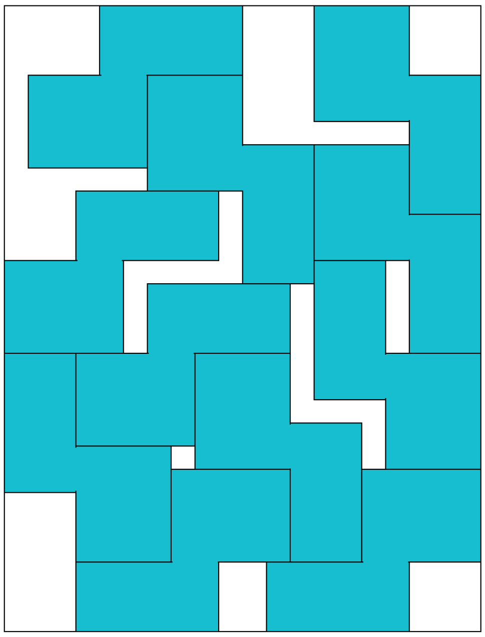
Source: The Author

Solution for the "1T" Instance



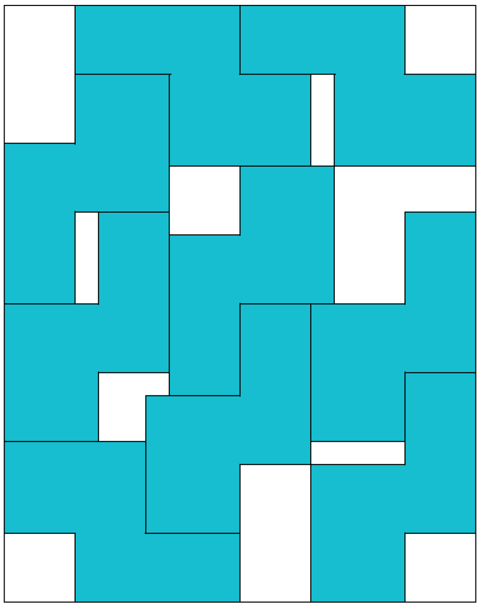
Source: The Author

Solution for the “1S” Instance



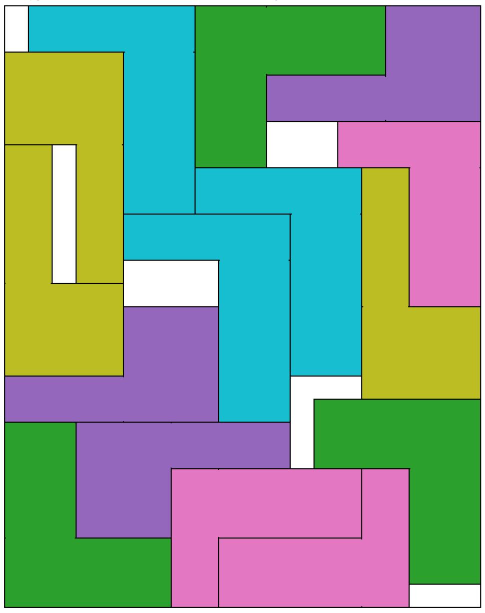
Source: The Author

Solution for the “1Z” Instance



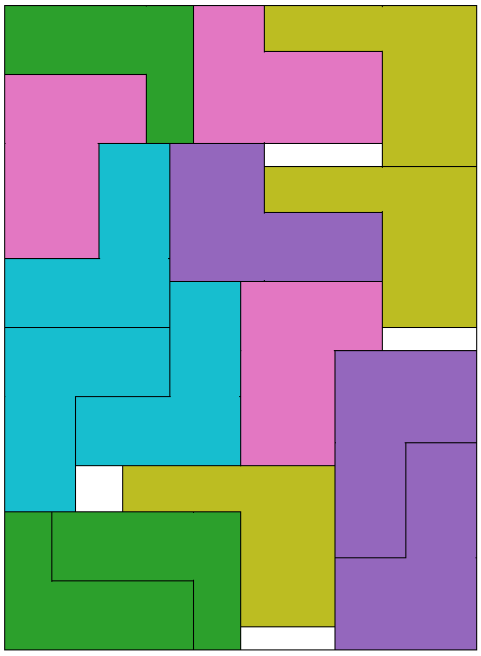
Source: The Author

Solution for the “5Ls” Instance



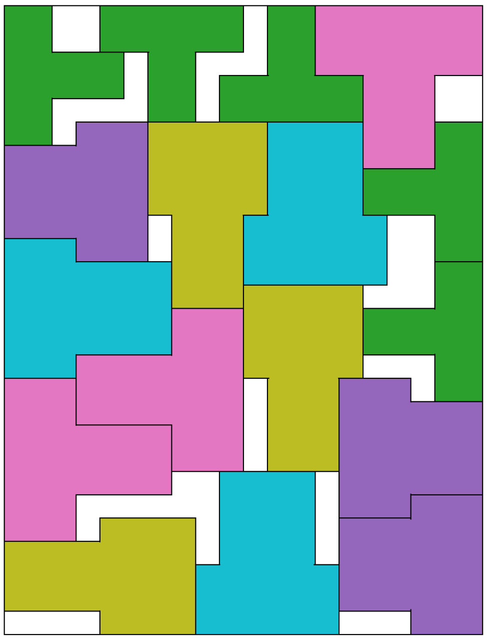
Source: The Author

Solution for the “5Js” Instance



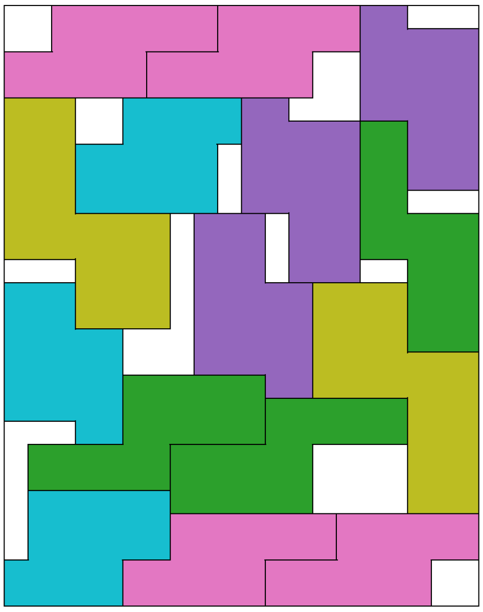
Source: The Author

Solution for the “5Ts” Instance



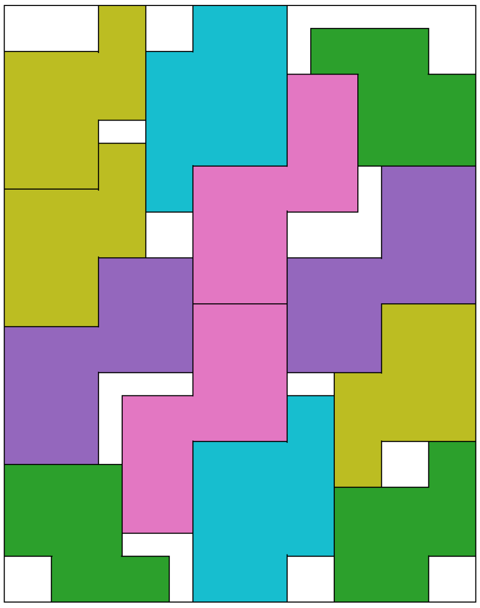
Source: The Author

Solution for the “5Ss” Instance



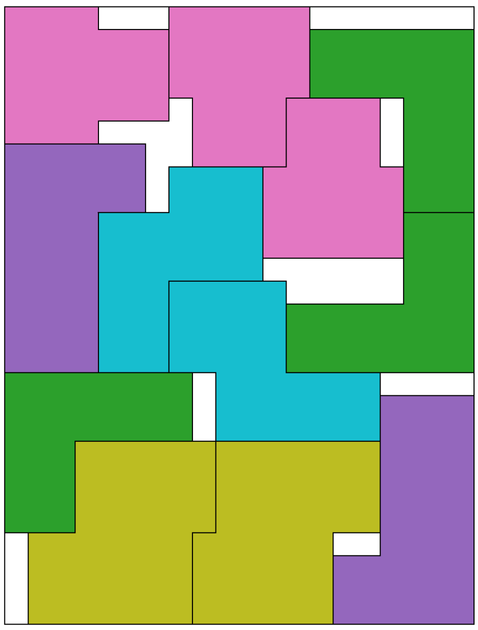
Source: The Author

Solution for the “5Zs” Instance



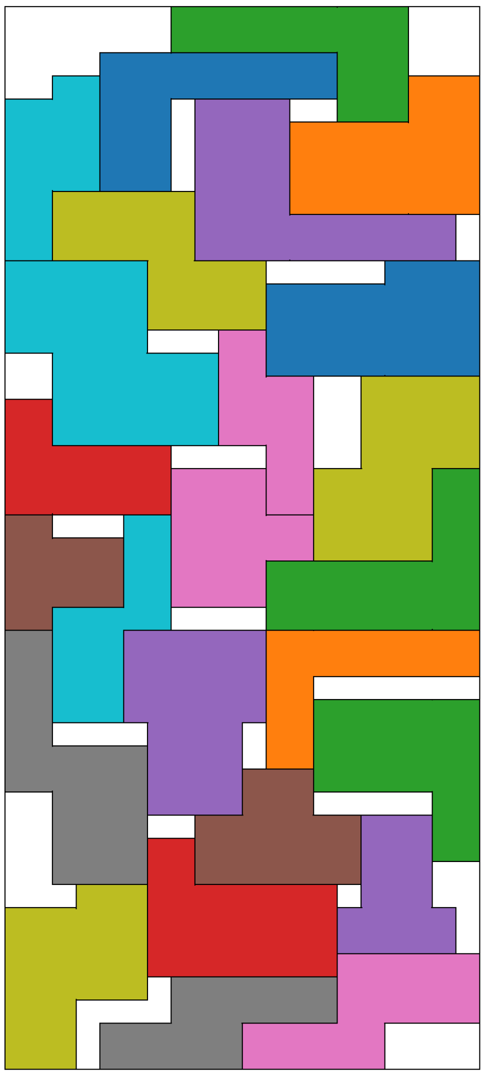
Source: The Author

Solution for the “1Each” Instance



Source: The Author

Solution for the “5Each” Instance



Source: The Author