A system for online glass cutting optimization

May 18, 2007

Chapter 1

Introduction.

To be written.

Chapter 2

Description of the process

2.1 General float process description

The basic float glass manufacturing process was invented in the mid-1950's. A schematic overwiew of a float plant is shown in figure 2.1. Raw materials like sand, soda ash, limestone, dolomite, salt cake and others are weighted and mixed in the batch house. This mixture is layered with broken glass ("cullet") returned from the end of the process line and conveyed to the melting furnace where the raw materials are melted using natural gas. The glass level in the furnace is controlled by the operation of the batch charger. In the furnace the temperature can be as high as 1600°C. Once the batch material is melted into solution, the molten glass is gradually cooled in the refiner section of the furnace. By the time the glass reaches the end of the furnace it should be completely free of unmelted batch. This homogeneous blend of molten glass is now delivered to the tin bath in a constant pouring action through the canal.

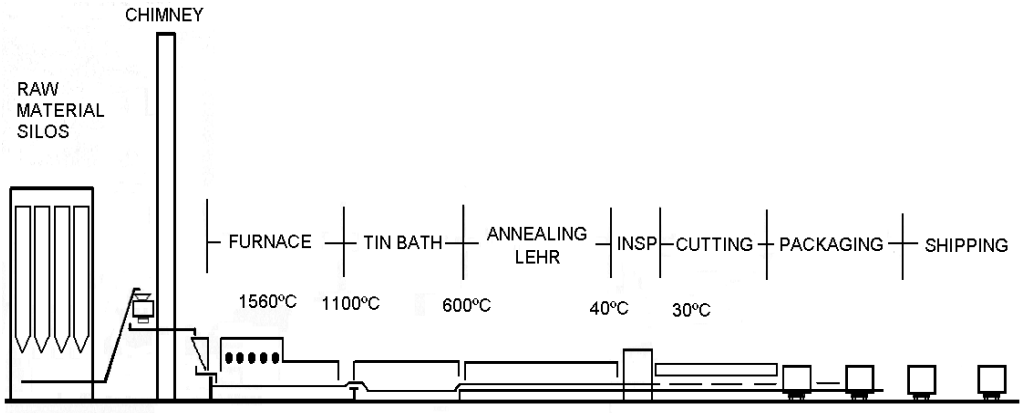

Figure 2.1: Schematic overview of a float line

In the tin bath the glass flows onto the surface of a pool of melted tin. The molten glass flowing on the surface of molten tin, forms a ribbon with perfectly flat parallel surfaces of 6 mm thick. The temperature of the glass in the tin bath is approximately between 1100 °C and 600 °C. At this temperature the glass is still elastic. The thickness of the ribbon can therefore be changed with mechanical top rolls. Given that the tonnage of glass is constant, the higher the draw speed the thinner the ribbon and visa versa. The glass produced in a float has a thickness ranging from 2 mm to 16 mm and more.

After the tin bath the glass enters the annealing lehr where it is cooled in preparation for cutting into sheets. The glass is cooled from 600 °C to approximately 30 °C in a precise and uniform manner to prevent temporary stresses that can cause ribbon fractures. The speed of the ribbon is maintained constant in the lehr and the beginning of the cold end until the cutting. All the conveyor rolls of these sections are driven by a common "king shaft".

On the cold end or capping line the ribbon is cut into sheets as dictated by custom orders.

The glass temperature is now approximately room temperature. The sheets are then either placed on racks, boxes or on dollies for storage or direct shipment. The capping line is discussed in more detail in the next section.

2.2 Description of a capping line

The function of the capping line is to score and snap the continuous glass ribbon into caps containing one or more sheets and to store them on predestined stackers on one of the side-legs, or to have them picked off by the personnel in the foreseen zones .

A cap is a sheet of glass between two full ribbonwidth X-cuts (see sub section 2.2.7). A cap can be composed of one or more sheets. The division of a cap into sheets is obtained by longitudinal or Y-cuts (see sub section 2.2.6) between the two bordering X-cuts. This is in the literature referred to as 2-stage guillotine cutting. An example of such cuts and the resulting caps and sheets is shown in figure 2.2.

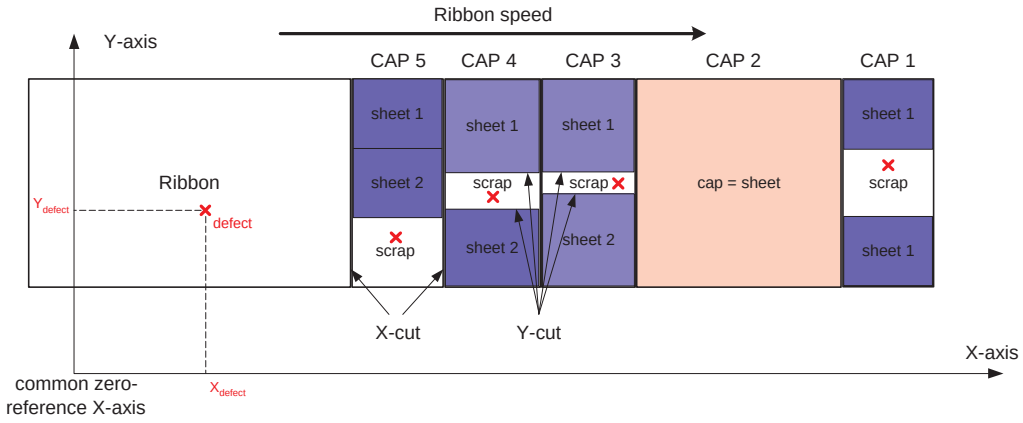

Figure 2.2: Example of caps and sheets on the ribbon

The choice of the caps/sheets to be cut is based on an optimization algorithm which generally takes in account the defects in the glass ribbon, a predefined order schedule, the availability of the line elements (conveyors, gates, sidelegs) and the demand of the stackers.

The glass produced in a float line is not always perfect. Sometimes undissolved batch or refractory stones can cause defects in the glass. Depending on the type and size of these defects and the destination of the glass (windshields for cars, mirrors, coated glass) the defect glass zones must be cut out. As shown in figure 2.2 this is done by adding scrap sheets in the caps.

Some terminology used in the glass sheet industry concerning the size of the sheets that will also be used in the rest of this document:

LES (Lehr End Size), EWG (EinWegGestell) or DLF (Dimension Largeur de Fabrication): A piece of glass that has the full net ribbon width.

JUMBO or PLF (Pleine longeur fini): The largest piece of glass that can be unloaded, usually 6 m or more.

END CAP SIZE: A small piece of glass. In this case two or more end cap size sheets (also called lites) compose the cap.

CAP: A full width part of the ribbon between two consecutive cross cuts. These caps can result in LES or Jumbo's without longitudinal cutting and in End Caps if they are longitudinally cut.

Typically the thickness of the transported glass is from 2 mm to 16 mm. The speed of the glass ribbon (before the snap roll) can be between 3 m/min and 30 m/min, depending on the glass

thickness. The glass transport speed on the main line of the cold end (after the snap roll) is maximum 90 m/min.

In the next sections the elements of the cold end shown in figure 2.3 are described in the order they appear in the line. In order to avoid confidentiality problems the enumeration of elements and their description are limited to the parts which are in our opinion essential to the understanding of the functioning of a capping line.

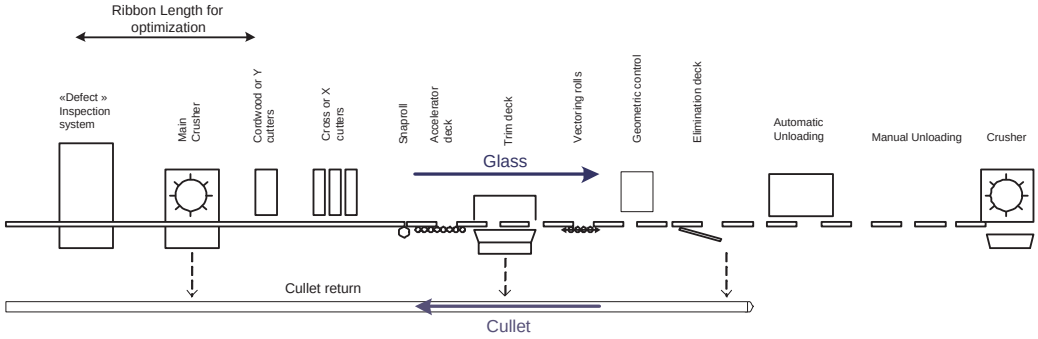

Figure 2.3: Symbolic ovrview of the capping line

2.2.1 King-shaft.

The conveyor rolls starting from the lehr until the snap roll are driven by the "king-shaft". In this zone the glass on the conveyors is part of the continuous ribbon coming from the furnace and this means that the speed must be constant and uniform. This ribbon speed is determined by one of the lehr drive motors.

The conveyors of the capping line starting from the snap roll are actuated by electrical motors which are controlled by variable speed drives.

2.2.2 Glass Inspection

The inspection of the glass can either be done manually: the operator selects by push button the corresponding zone where he detects a defect in the glass, either automaticly by an automatic glass defect detection system (normal operation). The automatic glass detection system is placed just before the manual inspection booth.

2.2.3 Main crusher

The line is equipped with a main breaker capable of crushing all the glass coming from the furnace. The glass is crushed in cullet and falls on the cullet return belt under the capping line.

2.2.4 Ribbon speed measurement

Two measuring wheel photoelectric pulse generator (encoder) assemblies are used to measure the ribbon speed. They are located immediately upstream from the longitudinal cutters. These assemblies generate milimeter and meter pulses that are used as common reference for all the devices from inspection system until the snaproll.

2.2.5 Common X-Reference

The inspection system(s), the longitudinal cutters and the cross cutters use the same reference for absolute zero.

2.2.6 Longitudinal or Y cutters

The Y cutters determine the Y-dimensions of the sheets and they cut the ribbon in the X-direction.

Two sets of independent solenoid operated longitudinal cutters are installed on two bridges. The cutters are numbered starting from the operator side. For example if the number of cutters is 13 then cutter 1 and 13 are trim cutters (see). Cutters 2 and 12 can be either trim cutters or pattern cutters. The longitudinal cutting bridges are equipped with a linear sensor which give the position of each cutter. An extra position is foreseen for the drift of the bridge. The two bridges can be used at the same time. For instance it must be possible to inside trim with bridge one and outside trim with bridge two on the same cap (needed when the two trims are very close).

2.2.7 Cross or X cutters

The X cutters determine the X-dimension of the caps (and the sheets) and they cut the ribbon in the Y-direction. There can be three to five cross-cutters. Each cross-cutter is mounted on a carriage that can move along a high precision rail. This rail is mounted on a bridge over the conveyor. The cross cutter is placed with an angle of 8° with the line. The X-component of the speed vector of the cross-cutter must match the ribbon speed in real time in order to obtain a perfect orthogonal score.

2.2.8 Marking Bridge

The glass defects detected by the automatic inspection system can be marked.

2.2.9 Snap roll

After the cross cutters the snap roll breaks the glass at the scores made by the cross cutters. From this point the glass ribbon is transformed into caps. As a rule we say that at this point the leading edge of the next cap is formed.

2.2.10 Acceleration deck

Once a cap is snapped it needs to be accelerated for separation from the ribbon. The acceleration deck is composed of several sections which can run respectively at lehr or at line speed, via clutches and servomotors. As long as a cap is not scored it is still part of the ribbon, thus the speed of the rolls on which it is laying must be the ribbon speed. As soon as the cap is snapped it can be accelerated to line speed. The line speed is at least three times the lehr or ribbon speed.

2.2.11 Trim deck

The top rolls (also called attenuators) leave marks on the border of the ribbon. These need to be cut and trimmed. In order to obtain the net glass width (from 2438 mm to 3660 mm) , the glass is trimmed at the trim deck. The knocked off excess glass falls immediately in the edge trim crusher and then on the cullet return belt under the line.

2.2.12 Vectoring rolls

At this stage the glass is broken in the longitudinal sense (on the score lines from the Y-cutters). The breaking of the glass is done by means of bars (called bump slitters) or small rolls (called score wheels). The sheets are then further separated from each other by the vectoring rolls.

2.2.13 Camera system

This system verifies the dimensions and the quality of the cuts of the sheets (broken edges or coners, markings, ...).

This information permits the elimination of the defect sheets in the elimination deck.

2.2.14 Elimination deck

Scrap sheets are eliminated from the line individualy.

2.2.15 Unload areas.

The unload areas are divided in manual unload sections, where operators take the sheets off the line, and automatic unloaders (or stackers).

Automatic unloaders have their own control system and communicate with the other (line-) control systems. All the stackers are installed on the sidelegs. The stackers have specific unload capabilities in the size of the sheets they can pick up.

For instance:

- DLF unloaders
- Jumbo unloaders
- End cap size unloaders

2.2.16 Drop sections

These are gates which can be lowered or raised (automatically or manually) to take the bad pieces of glass (scrap) off the line or to send the glass to the side-legs of the capping line.

2.2.17 End of the line breaker

Sheets which are not taken by the operators in the manual take off zone are eliminated in the end of the line breaker.

2.2.18 The ribbon length for optimization

The X- and Y-cutters need some time to arm and prepare for scoring. This means that the decision for scoring a cap, in fact injecting the cap in the queue of caps to score tracked by respectively the Y- and X-cutters, needs to be made before the cap injection point. The injection point usually corresponds to the location of the ribbon speed measurement wheels, approximatly one meter before the longitudinal cutters. As shown in figure 2.3 there is a certain distance between the detection of the defects and the Y-cutters. This distance corresponds to the length of the ribbon that is used for optimization. In the next chapters we will refer to this distance as the ribbon length (RLength).

For practical reasons we will convert the X-values of the defects and the cap boundaries such that the zero value corresponds to the trailing edge of the previous injected cap (= leading edge of the next not yet injected cap = first cap in the optimal cap sequence). In practice this means that each time a cap is injected in the queue, all the X-values of the defects and the boundaries of the caps in the current optimal sequence, are reduced with the length of this cap and at the end of the ribbon the new defects on an area of the same length are added.

Chapter 3

The problem statement

3.1 Defining the context

The context of the presented MAI end work is a simplified version of the "real-world" float glass cutting optimization problem. It is our opinion that this simplification does not reduce the relevance and the industrial applicability of the results of the reported study.

Out of a ribbon of glass of a given length a sequence of caps needs to be cut. The caps in this cap sequence are chosen from a list of caps containing the desired caps (containing the sheets) that are to be stacked on the cold end. We use the term cap sequence because the ribbon is "covered" with caps in the order they appear in the sequence (see 2.2.18). The first cap starts at position zero and all the next caps start where the previous caps end. The cap list is predetermined by the operator based on the list of orders, the mechanical restrictions of the cold end and the need to maximize the yield of the process. Each cap in the cap list has a given length and a given value. The value corresponds to the surface of good glass. As described in the previous chapter a cap can contain one of more sheets. Some of these sheets are scrap and are used to cutout not allowed defects. Due to the guillotine-cut restrictions of glass, the sheets, including the scraps, must completely cover the cap in such a way that they can be cut and separated on the cold end.

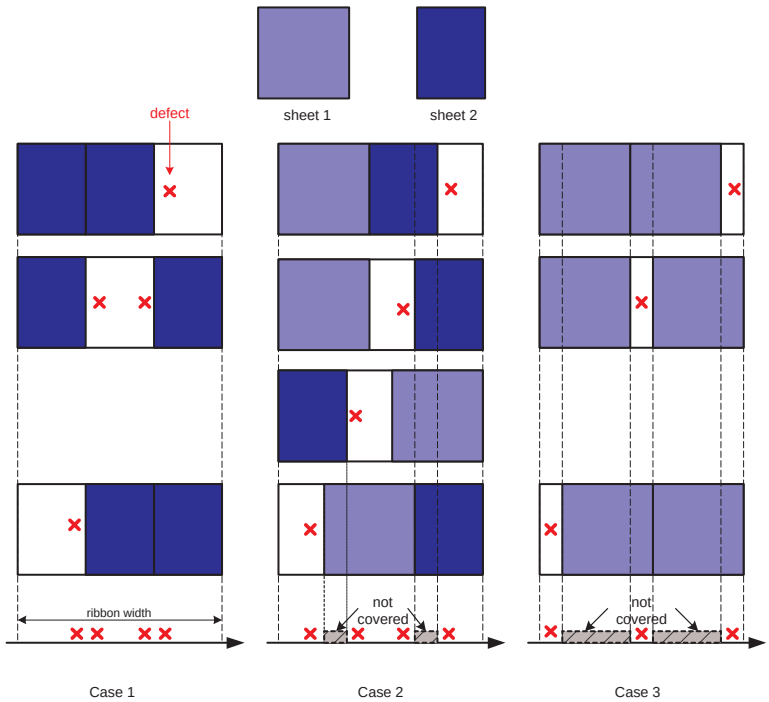

Figure 3.1: The coverage of defects with scrap using two sheets

The given ribbon of glass has been previously inspected with an online glass defect detection system. Therefore the types and relative coordinates in the ribbon of the glass defects are known. Glass defects have also a variable size, cutting to close to certain defects can cause glass breakage. To avoid these problems a safety zone before and after the defect must be foreseen.

In order to be able to recover as much glass as possible and allow optimization the cap list must contain caps with scrap sheets. These scrap sheets must be placed on various locations spread over the width of the ribbon. Depending on the position of the scrap sheet and the size of the good sheets, different caps can be defined. The more valuable caps that do not contain such scraps can obviously not be placed in zones containing proscribed defects. The coverage of defects on different positions in the width of the ribbon by the creation of different caps composed of two different sheets and a scrap (white area) is illustrated in figure 3.1. Depending on the Y-position of the defect some sheet combinations can not be used to create a fitted scrap. In case 1, using only sheet 2, any deffect position can be cut out. In case 2, using both sheets 1 and 2, some zones are not covered. Finally in case 3, using only sheet 1, the not covered zones become even larger. As a result one of the three caps of case 1 can always be used to cut out deffects and their combined covered zones cover the complete width of the ribbon.

However the width of the scrap is clearly the smallest in case 3 and largest in case 1. The presence of scrap sheets reduces the value of a cap proportionally. The value will be highest for caps of case 3, followed by case 2 and worst in case 1. This is what the optimization is about: reducing the losses by placing as much as possible caps with small or no scraps. It must also be noted that all the sheets in the cap have the same lenght. This means that shorter caps with scrap zones can also improve the total value. But the length of a cap is physically limited by the distance between the rolls of the conveyors and the order book must be respected.

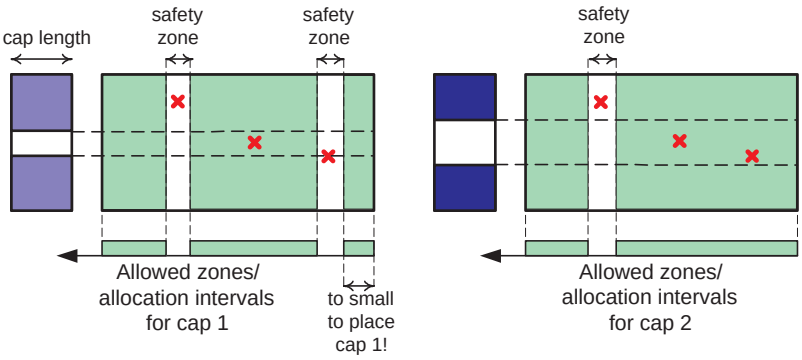

Figure 3.2: Two examples of allowed zones and corresponding allocation intervals

Once a cap is defined, with or without a scrap sheet, and the defect locations are known one can determine the zones of the ribbon where the cap is allowed or not. In figure 3.2 two examples show how the allowed zones are obtained. Cap1 has a smaller scrap sheet then cap 2 and has in function of the location of the defects smaller allowed zones.

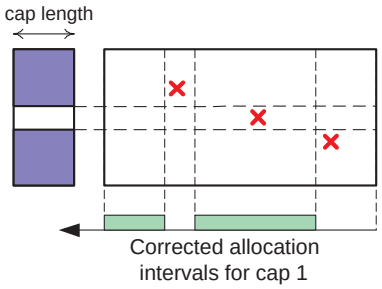

Figure 3.3: Corrected allocation intervals for cap 1

For the first example on the left an allowed zone is found that is smaller then the cap length. This makes no sense for optimization and needs to be corrected as shown in figure 3.3.

If several defects occur at approximatly the same X-position (length) but on different Y-positions (width) it can be that none of the "normal" caps can be placed. To resolve this kind of situations the cap list also contains short scrap caps of different lengths.

Depending on the optimization method it can be more interesting to define allocation restrictions (example for Constraint Logic Programming) but the conversion from one type to another is easily made and in the rest of this chapter we will use the former type.

The main advantage of this formulation is that it allows the reduction of the two-dimensional process (length and width) to a one-dimensional problem (length only). The allowed zones become allocation intervals and abstraction is made of the sheets within the caps. This two-stepped optimization process is presented in 3.4.

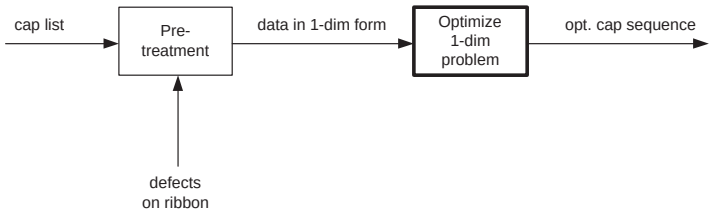

Figure 3.4: Pretraitment of data to obtain a one-dimensional problem

The pretreatment transforms the data from the cap list and the information on the defects in the ribbon into a dataset representing the problem in a one-dimensional formulation. The simplified original cap list of the operator contains the following information:

- for each sheet in the cap: its width, its relative position in the cap and a Boolean information scrap or not
- for the cap as a whole: its length and the glass quality of the non-scrap parts

The glass quality of the non scrap parts refers in fact to a matrix containing the number of allowed occurrences of each defect type (rows) and size (columns). However the pretreatment allows to convert also this infomation to allocation intervals in one dimension.

The idea for the use of allowed zones is indirectly based on the article published by G. Scheithauer in the journal Optimization [1]. In his paper G. Scheithauer uses the term "allocation constraints" in a comparable context. However this paper deals with stock packing and cutting problems and as will be shown in the next chapter this is fundamentally different from the problem treated in this work. Before we can compare this problem to other better known problems we need define and formulate the online glass cutting optimization problem.

3.2 Definition of the online glass cutting optimization problem

In order to define the optimization problem we need to specify the objectives of the optimization. A float glass line operates continuously during 15 years, 7 days per week, 24 hours per day. The global objective of the process is the continuous (online) optimization of the yield of the float line, which is the number of metric tons of glass stacked and ready for shipment divided by the number of metric tons of glass produced. The thickness of the glass is measured when it enters the lehr and the speed and the width of the ribbon are also known. The value of this yield, expressed in percentages, is one of the most important performance indicators of a float plant in general and of the shift supervisors in particular. This means that the optimization algorithm should not only maximize one ribbon length at the time, but should rather optimize an endless sequence of ribbon lengths.

Furthermore the decision time t_d for the online cutting process depends on the length of the previous injected cap, $t_d = capLength/ribbonSpeed$. For instance, suppose that $ribbonSpeed = 20\,m/min$ and $capLength = 1\,m$, then the decision time is 3 seconds! For campaigns with high quality glass the defect detection threshold is lowered and more defects are detected and tracked.

Therefore more short caps will be cut and as a result of this the decision time will be very low. In order to reduce the compution time it should not be necessary to recalculate the optimization completely each time a cap is injected and a new length is added to the ribbon, instead it should be possible to preserve the already optimized cap sequence as much as possible.

Based on these premises the online glass cutting optimization can be defined as follows:

> *Given a list of caps with associated lengths, values, an inspected ribbon of known length and corresponding allocation intervals for each cap in the cap list, what is the cap sequence that maximizes the yield of the cutting continuously?*

The above definition is a simplified version of the problem because it focuses only on the cutting and does not consider the optimization of the production of the capping line as a whole. In relation to this it must also be noted that there is no limitation on the number of each cap in the cap list. These and other possible extensions are discussed in chapter 8.

3.3 Formulation of the one-dimensional online glass cutting optimization problem

From hereon we will assume that the data used for the optimization is in a one-dimensional form.

The part of the ribbon that is to be optimized has a length $RLength$ and is represented by the interval $[0, RLength]$.

The K caps of the cap list are defined in 3 data sets:

- an indexed list of the cap lengths, $capLengths = [L_1, L_2, \cdots, L_k, \cdots, L_K]$
- an indexed list of the cap values, $capValues = [V_1, V_2, \cdots, V_k, \cdots, V_K]$
- an indexed list of allocation intervals lists, $capAlloc = [A_1, A_2, \cdots, A_k, \cdots, A_K]$

The first two lists only change when the operator/user changes the composition of the cap list. The list of allocation intervals however is updated each time a cap is injected.

The allocation intervals for a cap k are defined by start positions s_{kl} and end positions e_{kl}, where all $[s_{kl}e_{kl}] \subseteq [0, RLength]$.

The number of allocation intervals N_k per cap k depend on the corresponding cap and the defect list.

We define the elements A_k of the $capAlloc$ as a a list (or a matrix):

$$A_k = [[s_{k1}, e_{k1}], [s_{k2}, e_{k2}], \ldots, [s_{kN_k}, e_{kN_k}]]$$

The cap sequence T resulting from the optimization is a list of indexes t^i refering to the cap list(s).

If $t^i = k$ then the coresppponding cap has a length $L^i = L_k = L_k^i$, a value $V^i = V_k = V_k^i$ and allocation intervals $A^i = A_k$.

Each cap t^i of the cap sequence T has a start position s^i end an end position e^i such that $\left[s^i, e^i\right] \subseteq [0, RLength]$ and when $t^i = k$ then $e_k^i = s_k^i + L_k^i$.

The start and end positions of a cap must be part of an allocation interval of the associated cap in the cap list : $\left[s_k^i, e_k^i\right] \subset A_k$.

The guillotine cutting constraint of glass implies that the start position of a cap is equal to the end position of the previous cap: $s^i = e^{i-1} = s^{i-1} + L^{i-1}$ except for the first cap who starts always at position 0, $s^1 = 0$.

If the cap sequence contains N_c caps then the total length L of the cap sequence is:

$$L = \sum_{i=1}^{N_c} L^i$$

and the corresponding total value becomes:

$$V = \sum_{i=1}^{N_c} V^i$$

To determine the optimized sequence T^* a first, but wrong, approach would be to constrain the total length to $RLength$ and to maximize the total value of the caps. In equation form this gives:

$$\sum_{i=1}^{N_c} L^i = RLength$$

and

$$T^* = argmax_{N_c,(k=1,\ldots,K)} \sum_{i=1}^{N_c} V_k^i$$

This can lead to two undesired effects. The first effect is that the caps are not only choosen for their value but also to fill the complete length. For instance a depth-first based method with a by length sorted cap list would place at the end of the sequence smaller caps in order to satisfy the total length constraint. As discussed in the next chapter this situation corresponds in fact to a bin packing problem.

The second undesired effect is a result of the first. Shorter caps have often a lower value and therefore possibly also the total value of the solution. As a result a sub-optimal solution will be found. An example of such a situation is given in figure 3.5.

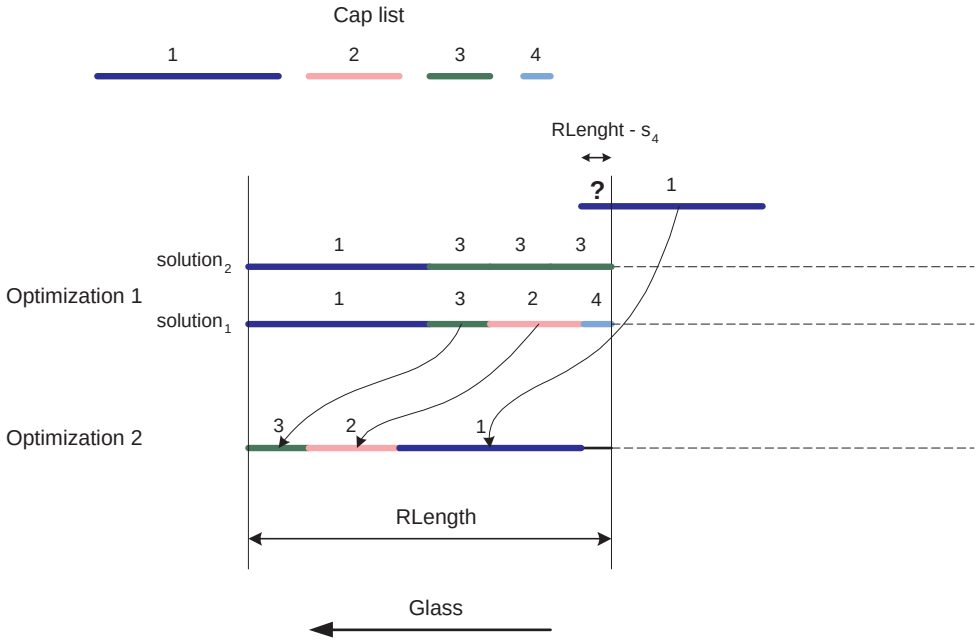

Figure 3.5: A wrong formulation of the constraints leads to undesired effects

Given a cap list with 4 caps and a ribbon of length $RLength$. Suppose that the best first three caps are 1, 3 and 2. If these are chosen only cap 4 can fill in the remainder of the ribbon (solution 1). Due to the low value of cap 4, solution 2 is found as the best sequence.

Since the defects on the ribbon are only known for the length $RLength$, it could be that the next part of the ribbon is defect free and that a cap of high value (cap1) could be placed once this is known (optimization 2).

In short what is needed is that the sequence is still optimal until position $N_c - 1$ included and that the cap on position N_c does not penalize the solution. To achieve this we suppose that

the ribbon beyond $RLength$ is perfect so that the best cap can be placed on position N_c in the sequence. However for the calculation of the total value only the part of the the value of the last cap within the optimized interval is used.

This gives the following equations:

for the total length constraint we have

$$\sum_{i=1}^{N_c-1} L^i < RLength \ , \ \sum_{i=1}^{N_c} L^i \geq RLength$$

for the optimal sequence we have

$$T^* = argmax_{N_c,(k=1,\ldots,K)} \left[\sum_{i=1}^{N_c-1} V_k^i + V_k^{N_c} \cdot \frac{RLength - s^{N_c}}{L^{N_c}} \right]$$

Finally the constraint on the start and end positions needs to be relaxed: $\left[s^i, e^i\right] \subseteq [0, RLength]$ for $i = 1...N_c - 1$

The summarized formulation of the one-dimensional online glass cutting optimization problem that will be used from hereon is as follows:

Given a part of the ribbon $[0, RLength]$ and cap data:

- $capLengths = [L_1, L_2, \cdots, L_k, \cdots, L_K]$
- $capValues = [V_1, V_2, \cdots, V_k, \cdots, V_K]$
- $capAlloc = [A_1, A_2, \cdots, A_k, \cdots, A_K]$ with

$$A_k = [[s_{k1}, e_{k1}], [s_{k2}, e_{k2}], \ldots, [s_{kN_k}, e_{kN_k}]] \tag{3.1}$$

where $[s_{kl} e_{kl}] \subseteq [0, RLength]$ for $k = 1...K$ and $l = 1...N_k$

Find the optimal cap sequence $T^* = \left\{t^i\right\}$ with $i = 1...N_c$

$$T^* = argmax_{N_c,(k=1,\ldots,K)} \left[\sum_{i=1}^{N_c-1} V_k^i + V_k^{N_c} \cdot \frac{RLength - s^{N_c}}{L^{N_c}} \right] \tag{3.2}$$

subjected to:

$$\left[s^i, e^i\right] \subseteq [0, RLength] \text{ for } i = 1...N_c - 1 \tag{3.3}$$

$$\text{and for } t^i = k \text{ we have } e_k^i = s_k^i + L_k^i \text{ and } \left[s_k^i, e_k^i\right] \subset A_k \tag{3.4}$$

and

$$s^i = e^{i-1} = s^{i-1} + L^{i-1}; s^1 = 0, \ e^1 = L^1 \tag{3.5}$$

and

$$\sum_{i=1}^{N_c-1} L^i < RLength \ , \ \sum_{i=1}^{N_c} L^i \geq RLength \tag{3.6}$$

3.4 Remarks

As defined above only the first $N_{c1} - 1$ caps of the first optimization $T^{1*}$are optimal:

$$t^1, t^2, \cdots, t^{N_{c1}-1} \text{ over a length } = \sum_{i=1}^{N_{c1}-1} L^i$$

When the first cap t^1 of this sequence is injected a new optimization is initiated, resulting in $N_{c2} - 1$ optimal caps. Since the previous first $t^2, \cdots, t^{N_{c1}-1}$ left over caps where optimal they are kept and we have $(N_{c2} - 1) - (N_{c1} - 2) = N_{c2} - N_{c1} - 1 \geq 0$ new optimal caps.

If we look at the cap sequences in a global continuous way, at optimization 2 we have now injected 1 cap and optimized a total sequence of $(N_{c2} - 1) + (2 - 1) = N_{c2}$ caps:

$$t^1, t^2, \cdots, t^{N_{c2}} \text{ over a length } = \sum_{i=1}^{N_{c2}} L^i$$

At optimization 3 we have injected $3 - 1 = 2$ caps and optimized a total sequence of $(N_{c3} - 1) + (3 - 1) = N_{c3} + 1$ caps:

$$t^1, t^2, \cdots, t^{N_{c3}+1} \text{ over a length } = \sum_{i=1}^{N_{c3}+1} L^i$$

After M optimizations and $M - 1$ injected caps we have an optimized sequence of $(N_{cM} - 1) + (M - 1) = N_{cM} + M - 2$ caps:

$$t^1, t^2, \cdots, t^{N_{cM}+M-2} \text{ over a length } = \sum_{i=1}^{N_{cM}+M-2} L^i$$

We can conclude that if equation 3.2 holds for M optimization runs, the optimization will also hold over the total length of the injected caps and the $N_{cM} - 1$ cap lengths of the last run.

Chapter 4

Overview of related problems and approaches

4.1 Cutting and Packing problems

4.1.1 Introduction

The optimization problem as formulated in the previous chapter is a *Cutting & Packing* problem (abbreviated C&P in the following). C&P problems are the subject of a large number of publications in various disciplines: Computer Science, Logistics, Industrial Engineering, Operational Reasearch, Combinatorial Optimization, Manufacturing, Mathematics, Production and others. Of course also the A.I. community is involved in this research through Evolutionary Algorithms, Swarm Intelligence (Ant Algorithms), Constraint Logic Programming and Neural Networks. The "classic" approaches can also be combined with A.I. methods in Hybrid methods.

As often various names are used for the same type of problems in different disciplines. This has motiviated some authors to create a typology of C&P problems. The most cited are H. Dyckhoff's "*A typology of cutting and packing problems*" [2] and from G. Wascher et al. "*An improved typology of cutting and packing problems*" [3].

C&P problems are combinatorial optimization problems with a common structure which can according to [3] be summarized as follows:

Given are two sets of elements:

- a set of large elements
- a set of small items

The sets can be defined in one, two, three or an even larger number (n) of geometric dimensions. Select some or all small items, group them into one or more subsets and assign each of the resulting subsets to one of the large objects such that the geometric condition holds.
For instance the small items of each subset have to be laid out on the corresponding large object such that

- all small items of the subset lie entirely within the large object
- the small items do not overlap

And a given (single-dimensional or multi-dimensional) objective function is optimized.

In order to categorize the C&P problems the following basic criteria are used:

- Dimensionality
- Assortment of small items
- Assortment of large objects

- Shape of small items
- Kind of assignement

The first four criteria can easily be identified for the online cutting problem:

- one dimensional,
- a weakly heterogeneous assortment of small items, the items are grouped in classes of the same shape and size (one of the caps in the cap list) with an unlimited demand
- one large object (ribbon), dimension fixed
- the items are one dimensional, shape is irrelevant but the dimension is fixed.

Typically two basic kinds of assignment are considered:

Output value maximization: The set of large objects is not sufficient to accommodate the small items and all the large objects are to be used. A subset of small items of maximal value needs to be assigned to each large object.

Input value minimization: The set of large objects is sufficient to accommodate all the small items. A subset of small items needs to be assigned to a subset of large objects of minimal value.

The online cutting optimization problem is clearly an output value maximization problem with one large object.

It is not the purpose of this chapter to give a complete a thorough overview of the C&P problems. However in order to find and compare different possible approaches to the online glass cutting optimization problem we must be able to find an appropriate classification for it. In the following sections some typical C&P problems are characterized.

4.1.2 Output maximization types

Identical item packing: Assignment of the largest possible number of identical small items to a given, limited set of large objects. Examples: the classic manufacturer's pallet loading (packing) problem, the cylinder packing problem, the single-box-type container packing problem.

Placement problem: A weakly heterogeneous assortment of small items has to be assigned to a given, limited set of large objects. The value or the total size (as an auxiliary objective) of the accommodated small objects has to be maximised, or, alternatively, the corresponding waste has to be minimised.

Knapsack problem: A strongly heterogeneous assortment of small items which have to be allocated to a given set of large objects. Again, the availability of the large objects is limited such that not all small items can be accommodated. The value of the accommodated items is to be maximised.

4.1.3 Input minimization types

For completeness also some well known input minimization problems are defined.

Open dimension problem

Cutting Stock Problem

Bin Packing Problem

When dealing with non-symbolic items and objects, a mathematical formulation can be used

4.2 Mixed integer optimization problems

In the book [4] integer and combinatorial optimization are described as problems of maximizing or minimizing a function of many variables subject to:

- Inequality and equality constraints.
- Integrality constraints on all or some of the variables.

Since minimizing a function is equivalent to maximizing the negative of the same function we will descible henceforth only the maximizing problem. A linear mixed-integer programming problem can be written as

$$max\left\{cx + hy : Ax + Gy \leq b, x \in Z_+^n, y \in \mathbb{R}_+^p\right\} \tag{4.1}$$

Where $z = cx + hy$ is referred to as the *objective function*, $Ax + Gy \leq b$ as the *constraints*, $\mathbb{Z}_+^n$ is the set of nonnegative integer-valued n-dimensional vectors and $\mathbb{R}_+^p$ the set of non negative real-valued n-dimensional vectors, and $x = (x_1, x_2, \cdots, x_n)$ and $y = (y_1, y_2, \cdots, y_p)$ are the variables or unknowns. An instance of the problem is specified by the data (c, h, A, G, b). The set $S = \left\{x \in \mathbb{Z}_+^n,\ y \in \mathbb{R}_+^p,\ Ax + Gy \leq b\right\}$ is called the *feasible region.* A solution $\left(x^0, y^0\right) \in S$ is optimal if the object function is as large as possible, or

$$cx^0 + hy^0 \geq cx + hy\ \forall\,(x, y) \in S \tag{4.2}$$

Sometimes an instance (c, h, A, G, b) of the MIP is unbounded and has no optimal solution. Thus to solve an instance of MIP with rational data means to produce an optimal solution or to show that it unbounded.

The Integer Programming problem (IP) and the Linear Programming problem (LP) are special cases of the MIP problem.

For IP we have:

$$max\left\{cx : Ax \leq b, x \in Z_+^n\right\} \tag{4.3}$$

and for LP we have:

$$max\left\{hy : Gy \leq b, y \in \mathbb{R}_+^p\right\} \tag{4.4}$$

There is no generally agreed-upon definition of a combinatorial optimization problem (CP). Most CP's can be defined as a 0-1 IP in which $x \in \mathbb{Z}_+^n$is replaced by $x \in B^n$ and where B^n is the set of n-dimensional binary vectors. A more generic definition could be as follows: let $I = \{1, 2, \ldots, n\}$ be a finite set and let $c = (c_1, c_2; \ldots, c_n)$ be a n-vector. For $F \subseteq I$ define $c\,(F) = \sum_{j \in F} c_j$.

Supposing we are given a collection of subsets $\Im$ of N we have for CP:

$$max\left\{c\,(F) : F \in \Im\right\} \tag{4.5}$$

In the next section some examples related to industrial applications are given.

4.3 Formalisation of the cutting problem

In [1], packing problems with pieces of variable length and additional allocation contraints are examend. This one-dimensional problem is formulated as follows:

> Pieces T_i, $(i = 1, \ldots, m)$ are to be packed on non-homogeneous stock material of length L in such a way that they are non-overlapping and that the total value of the packing pattern is maximal. Additionally the placement of a packed piece is restricted by further constraints. It is allowed that a piece is packed several times.

In this paper the lengths of the pieces are variable and the value of the pieces depends only on its length. This is a packing problem since some parts of the length may not be used in an optimal solution (scrap). If in addition if it would be required to cover the complete stock material then we would have a partionning (or covering) problem.

In compairison we have pieces of predefined length except for the scrap part and the value of the caps depends on a combination of the the good surface and/or the priority. Lets define the problem more preciselly, the interval $I = [0, L]$ is considered which represents the part of the ribbon for which the defects are known. The packing of piece T_i with length l_i starting by the allocation point x covers the interval $[x + l_i]$ and will be denoted $T_i\,(x + l_i)$.

The placement conditions for T_i are described by allocation intervals A_{ik}, $k = 1, \ldots, k_i$. These intervals correspond to the zones of the ribbon where the cap can be placed in function of the defects and the quality of the cap. It is assumed that the allocation intervals are given in the form

$$A_{ik} = [b_{ik}, e_{ik}] \subset I \text{ and } e_{ik} - b_{ik} \geq l_i \tag{4.6}$$

Once the allocation constraints are determined for each piece based on the desired glass quality and the defects present on the glass ribbon the cutting optimization problem is reduced to a one-dimensionnal problem .

[5]

Chapter 5

Search Algorithms

Example cap list for depth first search, the caps are sorted by length.

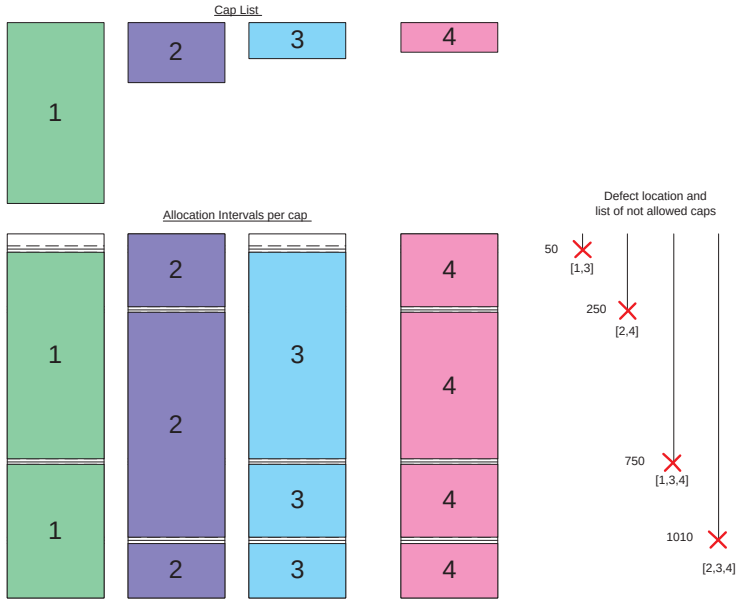

Figure 5.1: An example of a cap list and allocation intervals

Example of Depth first Search Tree

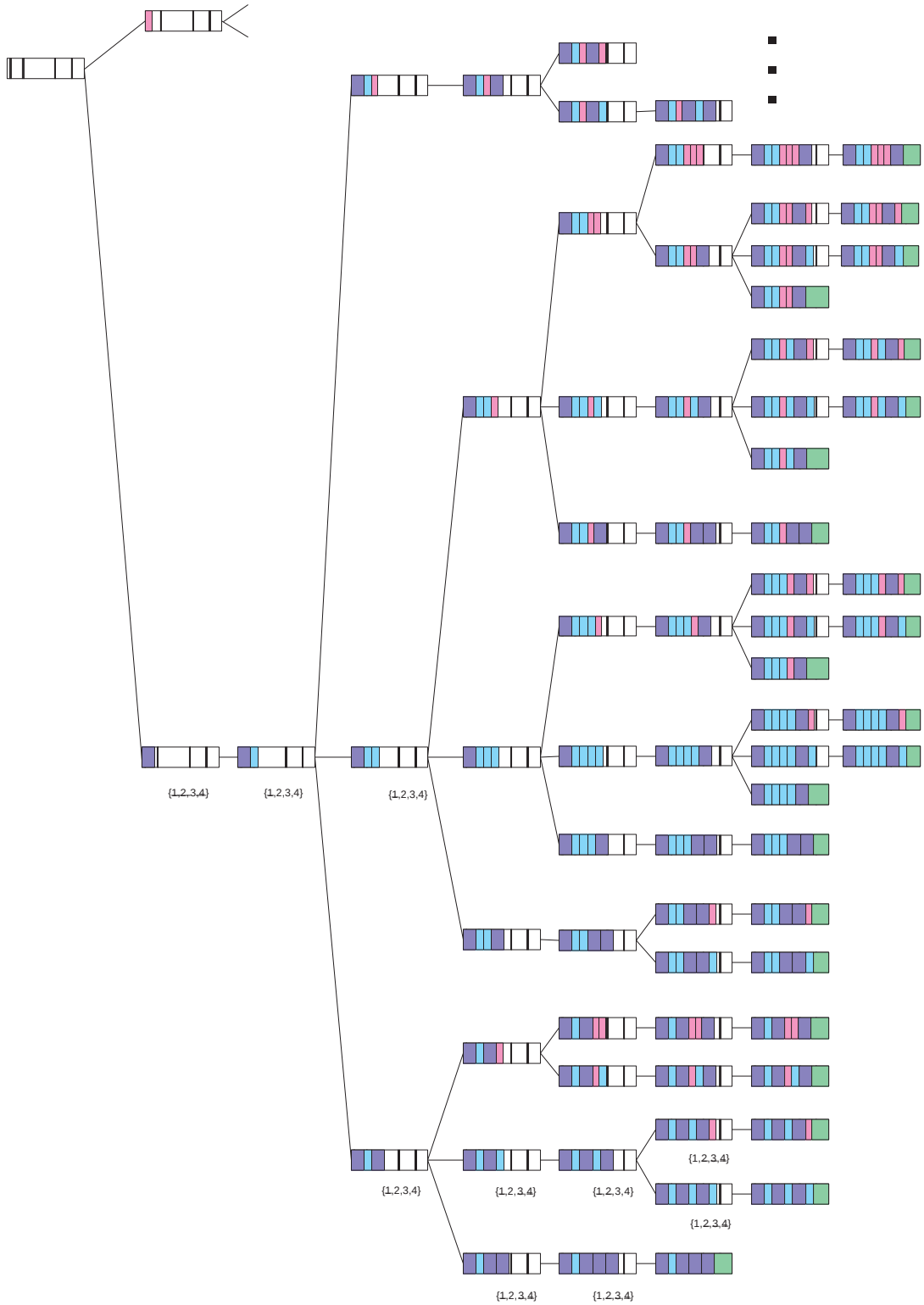

Figure 5.2: Depth first search tree example

Chapter 6

Constrainted Processing.

To be written

Chapter 7

Conclusions

Original contributions of this work:

make the connection between real world industrial problem and theory

practical transformation of 2 dim to 1 dim.

"Real" Optimization in stead of sequence and first fit.

Study of possibilities, to be placed in global approach to improve design and control of cold end.

Chapter 8

Future work

Include priorities, traveling time, scheduling: global cold end optimization: using MAS

Extend optimization to the level above, choice of caplist per shift, take into account number of deffects, foresee alternative list in case changes or better still make also dynamic.

Make simulation of cold end (example using Java and Java Beans) flexible composition of cold end, test out different strategies for design and control.

Implement optimization in PLC using Function blocks with STL (structured text language = Pascal like)

implement MAS in typical industrial process environment : PLC + SCADA PC's (with agents), can operate with or without MAS !!!!

Bibliography

[1] G. Scheithauer, "The solution of packing problems with pieces of variable length and additional allocation constraints," *Optimization*, vol. 34, pp. 81–96, 1995.

[2] H. Dyckhoff, "A typology of cutting and packing problems," *European Journal of Operational Research*, vol. 44, pp. 145–159, 1990.

[3] S. H. Wäscher Gerhard, Haussner Heike, "An improved typology of cutting and packing problems," doi:10.1016/j.ejor.2005.12.047, 2005, european Journal of Operational Research, Article in Press, Corrected Proof.

[4] G. L. NemHauser and L. A. Wolsey, *Integer and Combinatorial Optimization*, ser. Discrete Mathematics and Optimization. Wiley-Interscience, 1999.

[5] J. Puchinger, G. R. Raidl, and G. Koller, "Solving a Real-World glass cutting problem," in *Evolutionary Computation in Combinatorial Optimization EvoCOP 2004*, J. Gottlieb and G. R. Raidl, Eds. Springer-Verlag, April 2004, vol. 3004 of LNCS, pp. 162–173.