Optimizing Glass Coating Lines: MIP Models and Valid Inequalities

Céline Gicquel, Nicolas Miègeville, Michel Minoux, Yves Dallery

Optimizing glass coating lines: MIP model and valid inequalities

C. Gicquel [a,*], N. Mièvegeville [b], M. Minoux [c], Y. Dallery [a]

[a] *Laboratoire de Génie Industriel, Grande Voie des Vignes 92290 Chatenay-Malabry, France*

[b] *Saint-Gobain Recherche, 39 quai Lucien-Lefranc, 93303 Aubervilliers, France*

[c] *Laboratoire d'Informatique de Paris 6, 4 place Jussieu, 75005 Paris, France*

Abstract

Glass coating is a specific transformation aiming at improving glass performance. The work presented in this paper deals with the determination of the optimal configuration of the production lines used to perform this operation. We propose a first MIP formulation of the problem and then discuss several types of valid inequalities for improving it. The main idea is to exploit explicit or implicit binary exclusion constraints to derive stronger valid inequalities: the maximal clique constraints. Efficient (polynomial time) separation algorithms exploiting special structure of the problem are described, giving rise to a cutting-plane generation procedure for strengthening the initial formulation. The computational study carried out shows that, with the enhanced formulation, good solutions can be obtained within reasonable computation times using currently available integer programming software.

Keywords: Integer programming, Branch & Bound, Valid inequalities, Production line design, Glass coating.

*Corresponding author.
Tel.: +33 141 13 13 88; fax: +33 141 13 12 72.
E-mail address: celine.gicquel@ecp.fr

1 Introduction

This work is motivated by an industrial problem arising in the glass industry in connection with a specific transformation of flat glass called *glass coating*. Glass coating consists of depositing in vacuum thin layers of metal on the surface of glass sheets. As a general rule, the process involves several layers of distinct metals. This aims at giving the glass additional properties such as a better thermal insulation: see e.g. Arnaud (1997) for an overview on the applications of coated glass. According to the sequence and thickness of the layers, the property obtained is different: hence production managers have to cope with some product diversity.

Glass coating can be done on specific production lines called "soft-coating lines" using a process called "cathodic sputtering" (Suzuki (1999)). Basically, these lines are made of a number of metallic cathodes, each being used to spray or "sputter" a specific metal on the glass sheets. Each sheet can go only once through the production line: during this single passage, all the metal layers to be deposited on the sheet must be sputtered following the sequence imposed by the product specifications.

The cathodes are ordered along the line: a configuration of the line corresponds to a sequence of cathodes. A cathode contains a finite volume of a single metal. Once the metal of a cathode has been used up, the cathode must be changed. But, due to technical reasons, this requires a line shutdown during several days. Because of these time-consuming changeovers, soft-coating lines are operated according to the following organization. All cathodes on the line are changed together during a line shutdown. After this, production takes place continuously with this configuration during the next production run, the duration of which is typically about one month. When the run is over, all cathodes are changed and a new configuration is set up.

The problem addressed in the present paper concerns the determination of the optimal configuration to be set up between two line shutdowns. This decision can be based on reliable future demand forecasts: the requested products and the anticipated surface to be coated are assumed to be perfectly

known. The configuration set up at the beginning of a production run should be able to process all needed products in the quantity requested until the next production shutdown. In this context, determining the configuration to be set up consists of selecting among a set of available cathodes the ones to be placed on the line, ordering them along the production line and deciding how to use them to process the requested products. Because of its limited capacity, a cathode may not be sufficient to sputter the entire volume needed to process a given layer. Thus we have to consider the situation where a layer is sputtered by several cathodes placed at different positions on the line. The objective is to minimize the number of cathodes to be placed on the line. Indeed, the larger the number of cathodes to be placed on the line during a setup, the greater the changeover operations will be and the more time will be lost for useful production. Investigation of models and algorithms for solving the resulting discrete optimization problem is the subject addressed in the present paper.

The problem under study shares some common features with a string processing problem called the Shortest Common Supersequence problem (see e.g. Maier (1978)). It is however significantly different due to various extra constraints which must be taken into account, one of the most significant being the limitations imposed on cathode capacity, which frequently result in the use of a significant number of additional positions.

The problem of optimizing glass coating lines can also be related to the "Assembly Line Design Problem" (ALDP). In the ALDP, a production line is described as a series of workstations, each being responsible for performing a specific set of assembly tasks. The problem consists of selecting a piece of equipment for each workstation and deciding which tasks should be performed by which workstation. Recent overviews on the literature on the ALDP can be found in Becker and Scholl (2006) and Boysen et al. (2007). Nevertheless, the glass coating line problem is different from the ALDP studied in most papers (see e.g. Pinnoi and Wilhelm (1998) and Bukchin and Tzur (2000)). The main reason is that in the ALDP, each assembly task is

performed exactly once, i.e is assigned to a single workstation on the line, whereas on a glass-coating line, a layer can be sputtered by several cathodes placed at different positions on the line. The problem of optimizing glass coating lines can thus be seen as an extension of the ALDP to the case where a task can be assigned to more than one workstation ("parallelized" in the terminology of Boysen et al. (2007)). To the best of our knowledge, the only solution approach already available to deal with this particular extension of the "Assembly Line Design Problem" can be found in Boysen and Fliedner (2008) who propose a flexible heuristic search procedure that can be modified to solve various extensions of the "Assembly Line Balancing Problem". In their paper, the authors assume that the processing time of a parallelized task is equally allocated to the chosen workstations. On the contrary, on a glass coating line, the volume of a layer sputtered by several cathodes can be unequally divided among the various cathodes so that we have to decide about the allocation of the metal volume to be sputtered among the chosen cathodes. Moreover, their solution approach is purely heuristic whereas ours being based on a mixed integer linear programming (MIP) model is intended to provide exact optimal solutions.

The paper is organized as follows. In section 2, we introduce an initial mathematical formulation of this problem as a mixed integer linear program. In section 3, we consider several ways to strengthen this initial formulation by adding valid inequalities of various types. In section 4, we discuss the results of some computational experiments showing the practical usefulness of the proposed valid inequalities at improving the efficiency of a Branch & Bound type procedure. Conclusions and perspectives for future work are presented in section 5.

2 Problem formulation

We wish to determine the optimal configuration of a glass coating line to be set up between two production shutdowns. In this section, we introduce

an initial formulation for this optimization problem as a mixed integer linear program. To describe the problem precisely we introduce the following notation.

The set of anticipated requirements is supposed to involve M metals and P distinct final products. Each metal type is indexed by m: $m = 1, 2, ..., M$. Each product, indexed $p = 1, 2, ..., P$, is made of a glass sheet on which O_p layers are to be sputtered. For a given product p, a layer $o = 1, 2, ..., O_p$ is made of a specific metal denoted m_{po} and its thickness is given by e_{po}. The anticipated surface of product p to be processed during the production run, S_p, being known, the volume of metal m_{po} needed to sputter the o^{th} layer of product p can be deduced as: $V_{po} = e_{po} * S_p$.

Possible positions of cathodes on the production line are indexed by $i = 1, 2, ..., N$. These positions are ordered according to the orientation of production flow. The cathode $i + 1$ is located immediately after the cathode i along the line.

Available cathodes correspond to C types of cathodes. For each type of cathode $c = 1, 2..., C$, we assume that we know m_c, the corresponding metal, V_c, the volume of available metal in each cathode and ν_c, the number of cathodes belonging to this type. We agree to use an additional type $c = 0$ (the empty cathode) to represent free positions on the line. For each metal m, we denote $C(m)$ the subset of cathode types c such that $m_c = m$. The complementary subset is denoted $\overline{C}(m) = \{c = 1, 2, ..., C \text{ st } m_c \neq m\}$.

2.1 First formulation of the problem as a MIP

Here we first provide a mathematical statement of the problem involving the following decision variables:

- $z_c^i = 1$ if a cathode of type c is placed in line position i, $z_c^i = 0$ otherwise.
- $y_{po}^i = 1$ if the cathode placed in position i is used to sputter the o^{th} layer of product p, $y_{po}^i = 0$ otherwise.
- x_{po}^i gives the proportion of V_{po} sputtered by the cathode placed in the i^{th} position. Thus all the x_{po}^i are continuous variables in $[0; 1]$.

Considering the criterion of minimizing the total number of positions used, the formulation proposed is:

$$min \sum_{i=1}^{N} \sum_{c=1}^{C} z_c^i \tag{1}$$

$$\forall i, \sum_{c=0}^{C} z_c^i = 1 \tag{2}$$

$$\forall c, \sum_{i=1}^{N} z_c^i \leq \nu_c \tag{3}$$

$$\forall i, \forall p, \forall o, y_{po}^i + \sum_{c \in \overline{C}(m_{po})} z_c^i \leq 1 \tag{4}$$

$$\forall p, \forall o, \sum_{i=1}^{N} x_{po}^i = 1 \tag{5}$$

$$\forall i, \forall p, \forall o, x_{po}^i \leq y_{po}^i \tag{6}$$

$$\forall p, \forall (o, o') \text{ st } o > o', \forall (i, i') \text{ st } i < i', y_{po'}^{i'} + y_{po}^i \leq 1 \tag{7}$$

$$\forall i, \forall m, \sum_{c \in C(m)} V_c z_c^i - \sum_{(p,o) \text{ st } m_{po}=m} x_{po}^i V_{po} \geq 0 \tag{8}$$

$$\forall i \in [1; N-1], z_0^i \leq z_0^{i+1} \tag{9}$$

$$\forall i, \forall p, \forall o, y_{po}^i \in [0; 1], x_{po}^i \in [0; 1] \text{ and } \forall i, \forall c, z_c^i \in [0; 1] \tag{10}$$

$$\forall i, \forall p, \forall o, y_{po}^i \in \{0; 1\} \text{ and } \forall i, \forall c, z_c^i \in \{0; 1\} \tag{11}$$

The objective expressed by (1) is to minimize the total number of cathodes placed on the line. Constraints (2) ensure that at most one cathode is placed in position i. $z_0^i = 1$ means that the i^{th} position on the line is free. Constraints (3) ensure that no more than the number of available cathodes of type c, ν_c, are placed on the line. Constraints (4) guarantee the compatibility between the metal m_{po} for layer o of product p and the metal of the cathode placed in position i: we cannot open the connection $y_{po}^i = 1$ if the cathode in position i contains a metal other than m_{po}. Equalities (5) ensure that the demand is

perfectly met: all the volume of each layer should be sputtered. Constraints (6) link the continuous variables x^i_{po} with the binary variables y^i_{po}: some volume of the o^{th} layer of product p can be sputtered in position i only if the connection is open. Precedence constraints (7) force compliance with the order according to which the layers of a product should be sputtered: for a given product p, the layer o which is above the layer o' should not be processed with a cathode i placed before the cathode i' if the latter is used to sputter o'. Inequalities (8) guarantee that the limited capacity of the cathodes is not exceeded: for each type of metal, the volume remaining at the end of the production run in the cathode placed in position i should be non-negative. Constraints (9) are used to enforce consecutive empty positions at the end of the line in case all positions are not used.

2.2 A small illustrative example

Problem P0 is a small instance we use in order to illustrate the problem and its resolution. P0 involves $M = 4$ metals, $P = 3$ products made of 3 or 4 layers and $N = 12$ positions on the line. Table 1 gives the numerical data relative to this example. The optimal configuration in this case is a sequence of $Z^* = 7$ cathodes. Table 2 gives this sequence as well as the optimal use of cathodes to process the 3 products. We may notice that the first layer of product $p = 3$ is sputtered by two cathodes made of gold (placed at positions 2 and 4). This is due to the fact that the volume of metal needed to sputter this layer exceeds the capacity of a single cathode made of gold. In the sequel, P0 is used to illustrate various features of the proposed resolution method.

3 Valid inequalities

The formulation introduced in section 2 enables us to solve exactly only small instances: computation times for industrial problems of larger size using one of the best currently available commercial MIP solver are prohibitively long as can be seen from table 4. A possible explanation for this lies in the ob-

Table 1: Problem P0: data on products and available cathodes

Product $p = 1$	o	1	2	3	4
	Metal m_{po}	Ag	Au	Ti	Ag
	Volume V_{po}	210	400	100	100
Product $p = 2$	o	1	2	3	
	Metal m_{po}	Ag	Ti	Au	
	Volume V_{po}	410	800	200	
Product $p = 3$	o	1	2	3	
	Metal m_{po}	Au	Pt	Ti	
	Volume V_{po}	4000	1000	1000	

Cathode c	1	2	3	4
Number ν_c	5	5	5	5
Metal m_c	Ag	Ti	Au	Pt
Volume V_c	1000	2000	3000	2000

Table 2: Problem P0: optimal solution

Line		Volume sputtered for the layer (p, o)		
i	Cathode	$p = 1$	$p = 2$	$p = 3$
1	Ag, 1000	(1,1) 210	(2,1) 410	
2	Au, 3000	(1,2) 400		(3,1) 1200
3	Ti, 2000	(1,3) 100	(2,2) 800	
4	Au, 3000		(2,3) 200	(3,1) 2800
5	Pt, 2000			(3,2) 1000
6	Ti, 2000			(3,3) 1000
7	Ag, 1000	(1,4) 100		

servation that the linear relaxation of the problem (1)-(11) only provides a poor approximation to the exact optimal integer solution values. In order to address this issue, we investigate below several ways of strengthening the initial formulation (i.e. of reducing the integrality gap). The enhancements discussed here focus on various aspects of the problem under study, namely:

- available cathodes have a limited capacity,
- only one metal can be assigned to each position on the line,
- precedence constraints between layers of a given product must be respected.

In section 4, computational experiments will be reported showing that, thanks to these enhancements, the linear relaxation is tightened and instances of significantly larger size can be solved exactly with standard integer linear programming tools.

3.1 Valid inequalities from limited capacity of available cathodes

For each metal, we can compute a lower bound on the number of cathodes containing this metal to be placed on the line. This gives M valid inequalities (12) that can be added to the formulation.

$$\forall m, \sum_{i=1}^{N} \sum_{c \in C(m)} z_c^i \geq \left\lceil \frac{\sum_{p=1}^{P} \sum_{o=1 \ldots O_p \text{ st } m_{po}=m} V_{po}}{Max\{V_c, c \in C(m)\}} \right\rceil \tag{12}$$

Namely, for each metal m, dividing the global volume of metal needed to process all final products by the volume contained in the maximum capacity cathode containing metal m and rounding up gives the minimal number of cathodes of type $c \in C(m)$ to be placed on the production line.

3.2 Valid inequalities from metal compatibility constraints

In this subsection, we discuss another family of valid inequalities to further strengthen the formulation. In a first step, we derive a series of binary exclusion constraints. These constraints are logical consequences of the formulation (1)-(11). In a second step, we exploit the special structure of these constraints to derive stronger valid inequalities which correspond to *maximal clique constraints* in the underlying graph. (See e.g. Nemhauser and Wolsey (1988)). We note here that similar approaches have been used on other optimization problems such as assembly line design (Pinnoi and Wilhelm (1998)), harvest scheduling (Goycoolea et al. (2005)), cellular telecommuni-

cations networks design (Kalvenes et al. (2005)) or air line crew scheduling (Zeghal and Minoux (2006)). We observe however that in all the above-mentioned references the structures of the underlying constraint graphs were significantly different from those studied in the present paper, leading to clearly distinct separation algorithms. In particular, in Pinnoi and Wilhelm (1998), the separation of clique constraints is carried out using either complete enumeration or a greedy heuristic whereas our separation algorithms are exact and polynomial.

We first state various families of binary exclusion constraints. These constraints are implied by the constraints (2)-(11) of the initial formulation but their explicit statement turns out to be useful with respect to strengthening. They link pairs of binary variables related to the same position i on the production line, but to different products, layers or types of cathodes:

$$\forall i, \forall c, \forall p, \forall o \text{ st } m_c \neq m_{po}, z_c^i + y_{po}^i \leq 1 \tag{13}$$

$$\forall i, \forall p, \forall o, \forall p', \forall o' \text{ st } m_{po} \neq m_{p'o'}, y_{po}^i + y_{p'o'}^i \leq 1 \tag{14}$$

$$\forall i, \forall p, \forall (o, o') \text{ st } o \neq o', y_{po}^i + y_{po'}^i \leq 1 \tag{15}$$

$$\forall i, \forall c, \forall c' \text{ st } c \neq c', z_c^i + z_{c'}^i \leq 1 \tag{16}$$

Constraints (13) state that for a given position, there is an incompatibility between a cathode and a given layer if the corresponding metals are different. Similarly, constraints (14) state that two layers made of distinct metals cannot be sputtered at the same position. Constraints (15) are a consequence of the precedence constraints: they guarantee that two layers belonging to a given product will not be sputtered at the same position on the production line. Constraints (16) ensure that two distinct cathodes will not be placed at the same position on the production line.

We next investigate a strengthened formulation for the constraints (13)-(16) based on the analysis of the associated constraint graph and the use of valid inequalities deduced from maximal cliques.

In the constraint graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, a node $v \in \mathcal{V}$ represents either a

type of cathode placed in a given position (i.e a variable z_c^i) or a metal layer sputtered in a given position (i.e a variable y_{po}^i). There is an edge $a \in \mathcal{A}$ between two pairs of nodes if the corresponding variables are linked by one of the binary exclusion constraints (13)-(16). Constraints (13)-(16) all deal with variables related to a same position i on the production line. Therefore, there is no edge in graph $\mathcal{G}$ between two nodes corresponding to different positions on the line. In addition, for a given position, the binary variables involved as well as the exclusion relations linking them are identical. $\mathcal{G}$ is thus seen to decompose into N independent subgraphs with identical structure: $\mathcal{G}^i = (\mathcal{V}^i, \mathcal{A}^i)$ with $\mathcal{V}^i$ the subset of nodes related to position i and $\mathcal{A}^i$ the subset of edges linking these nodes. In the remainder of this subsection, we will study one of these graphs $\mathcal{G}^i$ for an arbitrary choice of i.

A set $C \subset \mathcal{V}^i$ is called a *clique* if each pair of nodes in C is connected by an edge. A *maximal clique* is a clique which is not properly contained in another clique. Each maximal clique in $\mathcal{G}^i$ thus gives rise to a valid inequality called a *maximal clique constraint* stating that the sum of corresponding binary variables should be less than or equal to 1. In the following, those are referred to as *type I valid inequalities*. We observe that constraints (2) and (4) of the initial formulation are among the clique constraints we can obtain thanks to the study of one of the subgraphs $\mathcal{G}^i$. But not all of them are *maximal* clique constraints. In the sequel, we show how to exploit the special structure of the graphs $\mathcal{G}^i$ to strengthen these constraints and find other maximal clique constraints, in particular constraints linking variables related to various distinct products.

The structure of a constraint graph $\mathcal{G}^i$ is close to that of a complete multipartite graph, i.e a graph with node set partitioned into clusters such that any two nodes belonging to different clusters have an edge connecting them and that there is no connection between nodes within a single cluster. Here, the set of nodes $\mathcal{V}^i$ can be divided into $M+1$ subsets $\mathcal{V}_m^i$ which will be referred to as clusters. The cluster $\mathcal{V}_m^i \subset \mathcal{V}^i$ is the subset of nodes in $\mathcal{G}^i$ related to metal m. There is an additional cluster, denoted $\mathcal{V}_0^i$, made of a

single node, namely the node related to the variable z_0^i. Thanks to constraints (13), (14) and (15), there is an edge between any two nodes belonging to different clusters. But, because of the constraints (15) and (16), there are some additional edges between nodes belonging to the same cluster, namely in the cases where the corresponding variables are related to two layers of the same product or to two distinct types of cathodes.

The clusters $\mathcal{V}_m^i$ can be seen to have a special structure. Namely let $\pi(m) \subset \{1, ..P\}$ be the set of indices of products p using metal m in at least one layer. The cluster $\mathcal{V}_m^i$ is made of $1 + |\pi(m)|$ disjoint cliques (possibly containing a single node):
- K_m^0, the clique containing the nodes related to cathode types $c \in C(m)$,
- for each $p \in \pi(m)$, K_m^p, the clique containing the nodes related to the layers of product p made of metal m.
Proposition 1 is a direct consequence of this particular graph structure.

Proposition 1. *A maximal clique in $\mathcal{G}^i$ is the union of $M+1$ cliques, each clique belonging to a different cluster $\mathcal{V}_m^i$ of the graph.*

Figure 1 illustrates the structure of a graph $\mathcal{G}^i$ for problem P0 introduced in section 2.2. We show a maximal clique containing 6 nodes and yielding the valid inequality: $z_0^i + z_2^i + y_{1,1}^i + y_{1,4}^i + y_{2,3}^i + y_{3,2}^i \leq 1$. For the sake of simplicity, only the edges linking the nodes of this maximal clique are displayed.

Using proposition 1, we can compute the number of maximal cliques in a graph $\mathcal{G}^i$ as the number of possible combinations obtained by choosing in each cluster $\mathcal{V}_m^i$ one clique out of $(1 + |\pi(m)|)$ cliques:

Proposition 2. *The number of maximal cliques in a graph $\mathcal{G}^i$ is given by:* $\prod_{m=1..M}(1 + |\pi(m)|)$.

This number can be quite high: e.g for the industrial problem P20 (see appendix), $\mathcal{G}^i$ has 2880 maximal cliques so that $2880 * 30 = 86400$ type I valid inequalities should be added to the formulation. Due to this large number, all maximal cliques constraints cannot be added *a priori* to the model. They

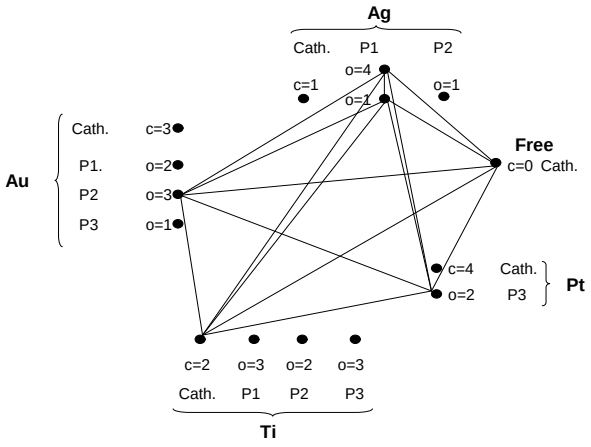

Figure 1: Constraint graph $\mathcal{G}^i$ for problem P0

can, however, be generated as needed according to a cutting-plane strategy. In order to do this, we need to address the so-called *separation problem*.

The separation problem here can be stated as follows: "given (z^*, y^*, x^*) the optimal solution of the linear relaxation of the problem, find a violated type I valid inequality or decide that (z^*, y^*, x^*) satisfies all type I valid inequalities ". To solve this problem, we use the following separation algorithm:

(SEP1) Given (z^*, y^*, x^*) the optimal solution of (1)-(10), for $i = 1...N$,

1. assign to each node in $\mathcal{G}^i$ a weight equal to the value of the corresponding variable,

2. for $m = 0...M$,

- compute the weight of each clique in the cluster $\mathcal{V}_m^i$: this weight is defined as the sum of the weight of all clique nodes.

- select the clique K_m^{max} of maximal weight w_m^{max}.

3. compute $W = \sum_{m=0...M} w_m^{max}$.

- if $W > 1$, the valid inequality given by the maximal clique $C = \bigcup_{m=0...M}(K_m^{max})$ is violated ,

- else all valid inequalities corresponding to position i are satisfied.

If, for each position $i = 1...N$, $W \leq 1$, then (z^*, y^*, x^*) satisfies all type I valid inequalities, otherwise at least one violated valid inequality has been

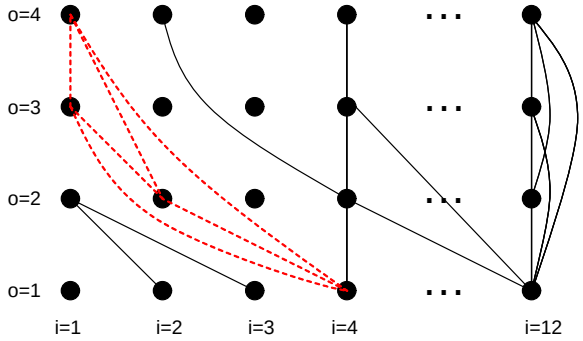

Figure 2: Constraint graph $\mathcal{G}_1$ for product $p = 1$ of problem P0: the edges drawn as dotted lines connect nodes belonging to a maximal clique.

found. In the sequel, algorithm (SEP1) is used to generate type I violated inequalities in order to strengthen the initial formulation.

3.3 Valid inequalities from precedence constraints between layers

We now focus on another subset of constraints in our problem: the precedence constraints between layers of a given product. As in subsection 3.2, we exploit the special structure of these binary exclusion constraints to derive a family of stronger valid inequalities corresponding to maximal clique constraints and to further strengthen the formulation.

We first explain how this family of stronger valid inequalities is derived. We have two families of binary exclusion constraints related to a single product p: constraints (7) of the original model stated in section 2 and valid inequalities (15) stated in the previous subsection. We define the corresponding constraint graph $\mathcal{G}_p = (\mathcal{V}_p, \mathcal{A}_p)$. A node $v \in \mathcal{V}_p$ refers to a binary variable y^i_{po} and can thus be indexed by (i, o). There is an edge $a \in \mathcal{A}_p$ between two nodes of $\mathcal{V}_p$ if there is a binary exclusion constraint (7) or (15) linking the corresponding variables. Figure 2 shows the graph $\mathcal{G}_1$ obtained for product $p = 1$ of problem P0. Only a fraction of edges is presented, the edges drawn as dotted lines connect the nodes belonging to a maximal clique.

Maximal cliques in graphs $\mathcal{G}_p$ have special features that can be exploited as shown by the following result:

Proposition 3. *A maximal clique in $\mathcal{G}_p$ consists of exactly O_p nodes, each node related to a different layer of product* p.

Proof. No two nodes in $\mathcal{V}_p$ related to the same layer are connected so that the cardinality of a clique in $\mathcal{G}_p$ cannot be greater than O_p, the number of layers of product p.

In addition, a clique in $\mathcal{G}_p$ cannot be maximal if it does not include a node related to each layer $1...O_p$. Namely, suppose K is a clique containing $O_p - 1$ nodes. All nodes relate to a different layer so that all layers $1, 2, ...O_p$ except layer o are present. K is thus the subset of nodes $K = \{(i_1, 1), (i_2, 2), ..., (i_{o-1}, o-1), (i_{o+1}, o+1), ..., (i_{O_p}, O_p)\}$ with $i_1 \geq i_2 \geq ... \geq i_{o-1} \geq i_{o+1} \geq ... \geq i_{O_p}$.

We now show that K cannot be a maximal clique. Consider a node (i_o, o) such that $i_{o-1} \geq i_o \geq i_{o+1}$. This node is connected to each node in K. We have namely:
- $\forall \omega = 1...o-1, i_\omega \geq i_o$ and $\omega < o$. Hence there is a precedence constraint linking $y_{p\omega}^{i_\omega}$ and $y_{po}^{i_o}$ and (i_o, o) is connected to (i_ω, ω).
- similarly, $\forall \omega = o+1...O_p, i_\omega \leq i_o$ and $\omega > o$. Hence there is a precedence constraint linking $y_{p\omega}^{i_\omega}$ and $y_{po}^{i_o}$ and (i_o, o) is connected to (i_ω, ω).
So $K \bigcup \{(i_o, o)\}$ is a clique containing K: K is not a maximal clique of $\mathcal{G}_p$. □

Each maximal clique in $\mathcal{G}_p$ provides a valid inequality for our problem. These valid inequalities will be referred to as *type II valid inequalities*.

We can compute the number of maximal cliques in a graph $\mathcal{G}_p$ by induction, as stated below:

Proposition 4. *Let $\mathcal{G}_p(N, L)$ be the graph for a product* p *made of* L *layers and a production line with* N *positions. We denote by $\mu(N, L)$ the number of maximal cliques in $\mathcal{G}_p(N, L)$. We have:*
(i) $\forall N, \mu(N, 1) = N$

(ii) $\forall N, \forall L \geq 2, \mu(N,L) = \sum_{i=1}^{N} \mu(N-i+1, L-1)$
(The proof is left to the reader).

The number of maximal cliques in $\mathcal{G}_p$ grows very fast with the problem size, in particular with the number N of positions. With the recurrence given above, the reader can easily check that e.g. for the product $p = 5$ in problem P20, there are more than 38 billion type II valid inequalities. Hence it is not possible to include directly all type II valid inequalities in the formulation. This is why we propose two ways of using them to strengthen the formulation.

First, we remove constraints (7) from the formulation and replace them by a much smaller number of type II valid inequalities. This involves finding a minimal subset of type II valid inequalities such that every binary exclusion constraint of type (7) is implied by at least one valid inequality belonging to this subset, i.e. to find a minimal subset of maximal cliques in $\mathcal{G}_p$ such that each edge $a \in \mathcal{A}_p$ is covered by at least one maximal clique belonging to this subset. The following heuristic procedure **(REP)** was devised in order to achieve this. It is based on the idea that the edges of graph $\mathcal{G}_p$ can be covered in a systematic way by relying on the angle they make with the horizontal axis. More precisely, for each node (i, O_p) in $\mathcal{V}_p$, we generate the maximal cliques made up by the edges forming an angle α with the horizontal axis such that $tan(\alpha) = \frac{b}{a}$ where $a = 1...N-i$ and $b = 1...O_p - 1$.

(REP)

1. For $p = 1...P$, for $i = 1...N$, for $a = 1...N-i$, for $b = 1...O_p - 1$, generate the following type II valid inequality:

$$\sum_{k=1}^{\lceil \frac{O_p}{b} \rceil} \sum_{o=O_p-b(k-1)}^{O(O_p,k,b)} y_{p,o}^{I(i,k,a)} \leq 1$$

with $I(i,k,a) = max(i + a(k-1); N)$ and $O(O_p,k,b) = min(O_p - bk + 1; 1)$.

2. Check wether each binary exclusion constraint of type (7) is covered by at least one generated type II valid inequality. If an uncovered binary exclusion constraint is found corresponding to layers o and $o' < o$ and positions i and $i' > i$, generate the following type II valid inequality: $\sum_{\omega=1}^{o'} y_{p,\omega}^{i'} + \sum_{\omega=o'+1}^{O_p} y_{p,\omega}^{i} \leq 1$
3. For each generated type II valid inequalities, check wether all binary exclusion constraints it replaces are covered by more than one type II valid inequalities. If so, eliminate the corresponding type II valid inequality.

As shown by the computational experiments to be presented in section 4, the use of procedure (REP) results in a substantial reduction on the total number of constraints in the model as well as in an enhancement of the formulation.

Second, in order to further strengthen the formulation, we generate additional type II valid inequalities according to a cutting-plane strategy. This involves solving the following separation problem for type II valid inequalities: "given (z^*, y^*, x^*) the optimal solution of the linear relaxation of the problem, find a type II violated valid inequality or decide that (z^*, y^*, x^*) satisfies all type II valid inequalities". In order to solve it, we will make use of proposition 5 below. We first build the oriented graph $\tilde{\mathcal{G}}_p = (\tilde{\mathcal{V}}_p, \tilde{\mathcal{A}}_p)$ in which nodes in $\tilde{\mathcal{V}}_p$ correspond to binary variables y_{po}^i and are indexed by (i, o). There is an oriented arc from node (i', o') to node (i, o) if $i \leq i'$ and $o = o' + 1$. We define a path as a sequence of nodes linked by arcs directed from a node to the following one. A maximal path is a path which is not contained in another path. Figure 3 shows the graph $\tilde{\mathcal{G}}_1$ obtained for product 1 in problem P0. Only a fraction of all arcs is presented.

Proposition 5. *There is an 1-1 correspondence between the maximal cliques of $\mathcal{G}_p$ and the maximal paths of the associated oriented acyclic graph $\tilde{\mathcal{G}}_p$.*

Proof. The graph $\tilde{\mathcal{G}}_p$ is an oriented acyclic graph. Due its special structure, a maximal path P in $\tilde{\mathcal{G}}_p$ contains O_p nodes, each one corresponding to a

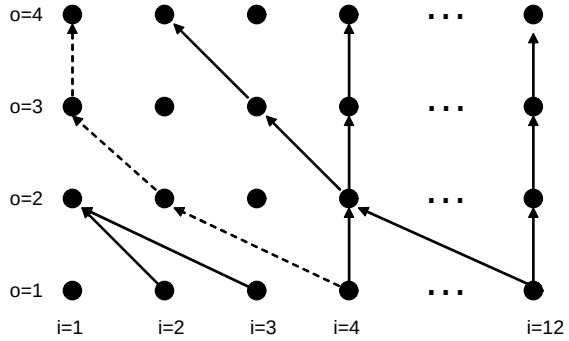

Figure 3: Oriented graph $\mathcal{G}_1$ for product $p = 1$ of problem P0: the arcs drawn as dotted lines are the arcs connecting nodes belonging to the maximal path corresponding to the maximal clique shown in figure 2.

different layer. P is a subset of nodes in $\tilde{\mathcal{V}}_p$: $P = \{(i_1, 1)..., (i_o, o)..., (i_{O_p}, O_p)\}$ with $i_1 \geq ... \geq i_o \geq ... \geq i_{O_p}$. Thanks to proposition 3, we know that the corresponding subset of nodes in $\mathcal{V}_p$ is a maximal clique of $\mathcal{G}_p$. Thus a maximal path of $\tilde{\mathcal{G}}_p$ corresponds to a maximal clique in $\mathcal{G}_p$. The converse is straightforward. □

Thanks to proposition 5, solving the separation problem for type II valid inequalities reduces to the solution of a number of longest path problems in an acyclic graph, leading to the following separation algorithm: **(SEP2)** Given (z^*, y^*, x^*) the optimal solution of (1)-(10), for $p = 1...P$:

1. Assign to each node (i, o) in $\tilde{\mathcal{G}}_p$ a weight equal to the value of the corresponding variable y^i_{po}.
2. Find the maximal weight path $\tilde{P}^{max}$ in the acyclic oriented graph $\tilde{\mathcal{G}}^p$ with a standard longest path algorithm.
3. Let W^{max} be the weight of $\tilde{P}^{max}$.

- if $W^{max} > 1$, the valid inequality given by the maximal clique K^{max} corresponding to the path $\tilde{P}^{max}$ is violated,
- else all type II valid inequalities are satisfied for product p.

If $W^{max} \leq 1$ for all products $p = 1...P$, then all type II valid inequalities are satisfied, otherwise we have found at least one violated inequality. In the

sequel, algorithm (SEP2) is used to generate type II violated inequalities in order to strengthen the initial formulation.

4 Computational results

In this section, we discuss the results of the computational experiments carried out to evaluate the impact of the formulation enhancements presented in section 3. We also present an empirical study carried out to show the impact of some problem parameters on the algorithmic performance.

4.1 Comparison of the initial and enhanced formulations

In order to evaluate the impact of the proposed formulation enhancements, we solved the problem with a standard MIP software (CPLEX 8.1.0) using either the initial formulation described in section 2 or the enhanced formulation. More precisely, the strengthened formulation is obtained thanks to the following procedure:

1. We use procedure (REP) to replace precedence constraints of type (7) by a subset of type II valid inequalities and we add the M valid inequalities (12) to the formulation. We solve the linear relaxation of the problem.
3. We use the separation algorithm (SEP1) to add type I violated valid inequalities.
4. When no more type I violated valid inequalities can be found, we look for type II violated valid inequalities using the separation algorithm (SEP2).
5. When no more type II violated valid inequalities can be found, we go back to step 3 and repeat until no more violated valid inequalities (whether of type I or type II) can be generated.

All the tests were run on a Pentium 4 (2.8 GHz) with 504 Mo of RAM, running under Windows XP. We used the default settings of CPLEX solver. This means that some cutting planes, among which are clique cuts, cover cuts and Gomory fractional cuts, are added automatically to the model (see

ILOG (2002) for more details).

We used an industrial data set available in Miegeville (2005) to build 20 test problems. P20 is the industrial problem presented in Miegeville (2005) and described in table 7 (see Appendix). P0 is the simple example introduced in section 2, P1 to P19 are simpler versions of P20. These instances were obtained by using one or several of the following simplifications: removal of possible positions along the production line; removal of some products; removal of some layers; removal of a metal; removal of some cathode types. Table 3 displays the following information for each problem tested: the number N of possible positions along the line; the total number $\sum_p O_p$ of layers to be sputtered; the number C of cathode types; the number *Var* of binary variables and the number *Const* of constraints in the initial formulation.

Table 3: Test problems

	N	$\sum_p O_p$	C	*Var*	*Const*	Remarks
P0	12	10	4	180	1117	small example in section 2.2
P1	20	17	10	560	8826	P20 without products 4 and 5
P2	20	19	10	600	11378	P20 without products 2 and 3
P3	20	20	10	620	11799	P20 without products 1 and 5
P4	20	22	10	660	14351	P20 without products 1 and 3
P5	20	21	10	640	13170	P20 without products 1 and 2
P6	25	24	5	750	20333	P20 without product 5
P7	25	25	5	775	22484	P20 without product 4
P8	25	26	5	800	24335	P20 without product 3
P9	25	25	5	775	22484	P20 without product 2
P10	25	28	5	850	27137	P20 without product 1
P11	25	26	8	875	20108	P20 without metal Ag
P12	25	22	8	775	13004	P20 without metal Ti
P13	25	25	8	750	17957	P20 without metal Au
P14	25	28	8	925	22310	P20 without metal Pt
P15	25	25	8	850	17032	P20 without metal Steel
P16	25	15	5	525	5449	P20: at most 3 layers per product
P17	25	20	5	650	10204	P20: at most 4 layers per product
P18	25	24	5	750	15208	P20: at most 5 layers per product
P19	25	28	5	850	21437	P20: at most 6 layers per product
P20	30	32	26	1770	41772	see Appendix

The computational results obtained with the initial and enhanced formulations are displayed in table 4. For both series of results, we provide:
- $Const$: the number of constraints in the formulation. For the enhanced formulation, this is the value obtained after applying the procedure (REP).
- Gap_0: the initial gap, i.e. the relative difference between the lower bound provided by the linear relaxation of the problem and the best integer solution found after at most 8 hours of computation. For the enhanced formulation, we use the value obtained after the strengthening procedure has stopped.
- $Nodes$: the number of nodes of the search tree explored before the optimal solution is found or the computation time limit of 8 hours is reached.
- CPU_{IP}: the time in seconds required to find the optimal integer solution when it has been found.
- Gap: the gap obtained after at most 8 hours of computation between the best integer solution found and the best lower bound found.
For the enhanced formulation, we also provide:
- $CutsI$ and $CutsII$: the number of type I and type II cuts added to the formulation during the strengthening procedure,
- CPU_{cuts}: the time in seconds spent to find the type I and type II violated inequalities.

As can been seen from table 4 (columns 2-6), using the initial formulation, only 7 of the 21 problems can be solved exactly within the computational limits. Despite long computation times (8 hours), non-optimal integer solutions are found for 13 problems and in these cases, the remaining gaps obtained remain quite large (16% on average). In addition, no feasible integer solution can be found for problem P20.

We compare these results with the ones obtained while using the enhanced formulation to solve the problem. The results from table 4 (columns 7-14) show that computation times for small instances are decreased and that more instances (11 out of 21 problems) are solved exactly. In addition, using the enhanced formulation, a feasible integer solution is found for all test problems and, in case the optimal integer solution could not be found after 8 hours of

computation, the remaining gap is smaller (9.6 % on average).

Comparison between the results obtained with the two formulations thus shows that the enhanced formulation improves the efficiency of the Branch & Bound procedure. The main explanatory factor for this is that the lower bounds provided by the linear relaxation of the enhanced formulation (table 4 column 11) appear to be stronger than the ones provided by the linear relaxation of the initial formulation (table 4 column 3). Indeed, the integrality gap (i.e the relative difference between Z_{LP} and Z_{IP}) is reduced on average from around 22% with the initial formulation to about 7.1% with the enhanced formulation. Moreover, it is worth pointing out here that the results provided in table 4 strongly suggest that the automatic cutting-plane generation procedures embedded in the CPLEX software do not seem able to identify the type I and type II valid inequalities exhibited and discussed in section 3.

4.2 Influence of cathode capacity

We carried out some additional numerical tests to evaluate the influence of cathode capacity on the algorithmic performance. We considered problems P1 to P5 described in table 3 and we modified the data relative to the cathodes. More precisely, we considered only one type of cathodes per metal and we built instances with various cathode capacity values:
- infinite capacity,
- large capacity: for each metal, the available cathode is the cathode with the largest volume among those described in table 7,
- medium capacity: for each metal, the available cathode is the cathode with the second largest volume among those described in table 7,
- small capacity: for each metal, the available cathode is the cathode with the third largest volume among those described in table 7.

We used the enhanced formulation to solve these instances. Table 5 displays the computational results. We provide Gap_0, Gap, CPU_{IP} and *Nodes* as defined in subsection 4.1 and *Opt* the number of instances that could be

Table 4: Comparison of the initial and the enhanced formulations: results

	Initial formulation					Enhanced formulation							
	Const	Gap_0	*Nodes*	CPU_{IP} (s)	*Gap*	*Const*	*CutsI*	*CutsII*	CPU_{cuts} (s)	Gap_0	*Nodes*	CPU_{IP} (s)	*Gap*
P0	1117	35.2	351	1.8	0	777	12	16	0.2	14.5	46	1.2	0
P1	8826	23.9	10547	688	0	3100	129	604	72	0	933	393	0
P2	11378	30.3	143329	10501	0	3465	114	174	30	13.3	53131	8334	0
P3	11799	31.6	163415	13706	0	3634	116	461	122	7.1	42141	10027	0
P4	14351	30.8	240214	#	14.5	3999	157	292	108	6.7	112190	25344	0
P5	13170	21.5	133718	17456	0	3846	131	59	48	0	3942	1500	0
P6	20333	25.6	119944	#	22.2	6342	187	1110	716	5.9	39543	#	5.9
P7	22484	22.9	89052	#	21.5	6660	226	700	686	5.9	38352	#	5.9
P8	24335	23.3	92031	#	21.2	6884	184	632	660	16.6	30136	#	16.6
P9	22484	22.2	98344	#	19.4	6660	189	74	179	5.3	26011	#	5.3
P10	27137	26.7	73411	#	25.6	7454	214	264	644	10.5	24531	#	9.6
P11	20108	13.8	105293	#	13.8	6817	227	299	275	0	12355	5604	0
P12	13004	11.1	266188	#	5.88	5700	193	364	143	5.5	50360	#	5.5
P13	17957	11.2	123220	#	11.2	6526	246	469	250	5.5	27850	#	5.5
P14	22310	11.1	82000	#	11.1	7340	268	250	427	5.5	39710	#	5.5
P15	17032	16.5	51626	12104	0	6566	190	403	424	0	20915	9208	0
P16	5449	21.9	5316	279	0	3739	13	23	0.5	10.0	3537	259	0
P17	10204	21.8	247543	#	7.1	5184	1	109	0.3	0	393	82	0
P18	15208	19.8	143136	#	19.6	6260	91	426	125	0	16880	7916	0
P19	21437	15.2	87485	#	15.2	7489	177	183	600	11.1	56047	#	11.1
P20	41772	_	51762	_	_	11754	279	70	1294	25.0	3254	#	25.0

The symbol " #" indicates that a guaranteed optimal solution could not be found within 8 hours of computation.

Table 5: Influence of cathodes capacity

Cathode capacity	Gap_0	*Opt*	*Gap*	CPU_{IP} (s)	*#Nodes*
infinite	11.4	5	0	2617	8781
large	5.4	5	0	10930	30983
medium	11.9	1	8.8	> 28800	105480
small	2.6	1	2.6	> 28800	39088

solved to optimality within the computation limit. These results suggest that instances with medium or small capacity cathodes are more difficult to solve than instances with infinite or large capacity cathodes. Namely, all instances using infinite or large capacity could be solved to optimality within 2 hours of computation whereas only 2 out of the 10 instances using medium or small capacity cathodes could be solved to optimality within 8 hours of computation. Moreover no feasible solution could be found for 2 out of the 5 instances using small capacity cathodes.

4.3 Influence of product composition

We finally discuss the results of some experiments carried out to evaluate the influence of product composition, i.e. of the sequence of metal layers to be deposited on the glass sheets. We built 15 instances involving $M = 5$ metals, $P = 5$ products made of 6 layers, $N = 20$ positions on the line, $C = 5$ infinite capacity cathodes. They differ only with respect to the sequence of metal layers:

- In E1 to E5, there is a basic sequence of metal defined by product 1. Products 2 to 5 are obtained by a simple modification of this sequence (switch between two consecutive layers or modification of the metal for one layer).
- In R1 to R5, the sequences of metal layers are randomly generated from a discrete uniform $DU(1, 5)$ distribution. If two consecutive layers are made of the same metal, we repeat the random generation until a product is obtained without any identical consecutive layers.
- In H1 to H5, the sequence of layers for each product are chosen in order to

obtain supposedly difficult instances (products made of reverse sequences of metal, products made of sequences with no common metal...).

In order to compare the generated instances, we introduce a measure aiming at evaluating the difference between the products of a given instance with respect to the sequence of metal layers. This difference denoted d is defined as: $d = \sum_{p_1=1}^{P} \sum_{p_2=p_1+1}^{P} d(p_1, p_2)$ where $d(p_1, p_2) = SCS(p_1, p_2) - LCS(p_1, p_2)$. $SCS(p_1, p_2)$ is defined as the minimum number of cathodes needed to sputter products p_1 and p_2 and $LCS(p_1, p_2)$ is the maximum number of cathodes that can be used to sputter layers from both p_1 and p_2. $SCS(p_1, p_2)$ and $LCS(p_1, p_2)$ can be computed by a dynamic programming algorithm as respectively the Shortest Common Supersequence containing p_1 and p_2 and the Longest Common Subsequence contained in p_1 and p_2.

We used the enhanced formulation to solve these instances. The computational results are displayed in table 6. These results suggest that instances with a large value of d are more difficult to solve than instances with a small value of d. Namely, all instances E1-E5 could be solved to optimality within one hour of computation whereas the mean computation time for the instances R1-R5 and H1-H5 is above 4.5 hours. Moreover no feasible solution could be found for 2 out of the 5 instances H1-H5. It is worth pointing out that for the instance P20 presented in Appendix $d = 5.8$. This seems to indicate that in a industrial situation, the products to be made on the glass coating line are quite different with respect to the sequence of metal layers to be deposited, leading to an additional difficulty to solve the problem.

Table 6: Influence of product composition

Instances	d	Gap_0	*#Opt*	*Gap*	CPU_{IP} (s)	*#Nodes*
E1-E5	2.8	22.4	5	0	2184	7405
R1-R5	5.5	13.1	4	6.5	14642	32325
H1-H5	6.4	11.1	3	0	19075	29792

5 Conclusion and perspectives

In this paper, we studied an optimization problem arising in the context of the glass industry in connection with a specific transformation of flat glass called glass coating. In order to improve an initial MIP formulation, three families of valid inequalities have been discussed: valid inequalities from limited capacity constraints; valid inequalities from metal compatibility constraints (type I valid inequalities); valid inequalities from precedence constraints between layers of a given product (type II valid inequalities). The results of our computational experiments confirm the positive impact of the proposed enhancements on the computation times and solution quality.

Among the possible research directions suggested by the present work, it might be worth exploring other optimization criteria such as minimizing the volume of unused metal remaining in the cathodes at the end of the production run. Indeed, partially consumed cathodes at the end of a production run represent a cost, either as a direct loss because of the unused metal or as additional constraints for the forthcoming production run because they will impose the use of a set of initial reduced capacity cathodes. Looking for other families of valid inequalities in order to further improve the formulation might also be an interesting research direction.

References

Arnaud, A. (1997). Industrial production of coated glass: future trends for expanding needs. *Journal of Non-crystalline Solids*, 218:12–18.

Becker, C. and Scholl, A. (2006). A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, 168:694–715.

Boysen, N. and Fliedner, M. (2008). A versatile algorithm for assembly line balancing. *European Journal of Operational Research*, 184:39–56.

Boysen, N., Fliedner, M., and Scholl, A. (2007). A classification of assembly line balancing problems. *European Journal of Operational Research*, 183:674–693.

Bukchin, J. and Tzur, M. (2000). Design of flexible assembly line to minimize equipment cost. *IIE Transactions*, 32:585–598.

Goycoolea, M., Murray, A., Barohona, F., Epstein, R., and Weintraub, A. (2005). Harvest scheduling subject to maximum area restrictions: exploring exact approaches. *Operations Research*, 53(3):490–500.

ILOG (2002). *ILOG CPLEX 8.0 : User's manual*. ILOG, Gentilly, France.

Kalvenes, J., Kennington, J., and Olinick, E. (2005). Hierarchical cellular network design with channel allocation. *European Journal of Operational Research*, 160:3–18.

Maier, D. (1978). The complexity of some problems on subsequences and supersequences. *Journal of the Association for Computing Machinery*, 25(2):322–336.

Miegeville, N. (2005). *Supply Chain Optimization in the process industry. Methods and Case Study of the Glass Industry*. PhD thesis, Ecole Centrale Paris, Paris, France.

Nemhauser, G. and Wolsey, L. (1988). *Integer and Combinatorial Optimization*. John Wiley and Sons, USA.

Pinnoi, A. and Wilhelm, W. (1998). Assembly system design: a branch and cut approach. *Management Science*, 44(1):103–118.

Suzuki, K. (1999). State of the art in large area vacuum coatings on glass. *Thin Solid Films*, 351:8–14.

Zeghal, F. and Minoux, M. (2006). Modeling and solving a crew assignment problem in air transportation. *European Journal of Operational Research*, 175:187–209.

Appendix

Table 7: Problem P20: data on products and cathodes

$p = 1$	o	1	2	3	4				
	m_{po}	Ag	Au	Ti	Ag				
	V_{po}	2200	2400	2000	2000				
$p = 2$	o	1	2	3	4	5	6	7	
	m_{po}	Ag	Au	Ti	St	Au	Ti	Ag	
	V_{po}	2100	1300	1000	1000	700	2000	4000	
$p = 3$	o	1	2	3	4	5	6		
	m_{po}	Au	Pt	Ti	St	Au	Pt		
	V_{po}	2000	1000	1000	2400	1000	2000		
$p = 4$	o	1	2	3	4	5	6	7	
	m_{po}	St	Ti	St	Au	Pt	Ti	St	
	V_{po}	1500	1000	2400	1000	750	500	1000	
$p = 5$	o	1	2	3	4	5	6	7	8
	m_{po}	Au	Pt	Ti	St	Ag	St	Ti	St
	V_{po}	1000	750	500	1000	2000	1500	1000	2400

c	1	2	3	4	5	6	7	8	9	10
ν_c	10	10	10	10	10	10	10	10	10	10
m_c	Ag	Ag	Ag	Ag	Ag	Ag	Ti	Ti	Ti	Ti
V_c	300	500	1000	2000	3000	4000	4000	3000	2500	1000
c	11	12	13	14	15	16	17	18	19	20
ν_c	10	10	10	10	10	10	10	10	10	10
m_c	Ti	Au	Au	Au	Au	Au	St	St	St	St
V_c	400	100	500	1000	2500	3500	500	750	1000	1500
c	21	22	23	24	25	26				
ν_c	10	10	10	10	10	10				
m_c	St	Pt	Pt	Pt	Pt	Pt				
V_c	2000	500	750	1000	1500	2000				