A Decomposition Approach for the Piece-Mold-Machine Manufacturing Problem

Omar J. Ibarra-Rojas
Roger Z. Ríos-Mercado
Yasmin A. Rios-Solis[1]
Mario A. Saucedo-Espinosa

Graduate Program in Systems Engineering
Universidad Autónoma de Nuevo León, Mexico
e-mail: { *omar, roger, yasmin, mario* }*@yalma.fime.uanl.mx*

April 2011
Revised May 2011

[1]Corresponding author

Abstract

This study addresses a real manufacturing process of pieces that are produced with molds that are mounted on machines. The characteristics of the system include setup times between jobs, dedicated parallel machines, dedicated molds, and a different production rate for each piece-mold pair. There is a demand for each type of piece, and when the company fails to meet this demand, it is often forced to buy pieces from other companies to avoid loss of customers. We describe the system with a new integer quadratically constrained programming model. The proposed formulation improves others in literature as we do not force a mold to be mounted on a single machine, which is a more realistic description of the production process itself. To solve the problem we decompose the formulation into two subproblems: one that solves the lot-sizing of the products and another one that verifies if there is a feasible schedule for the solution of the first subproblem. This methodology is empirically tested, demonstrating its effectiveness on real size instances. Moreover, it reveals that the counterintuitive case where a mold visits more than one machine happens more often than expected.

Keywords: Manufacturing planning; assignment; lot-sizing; scheduling; integer linear programming; decomposition.

1 Introduction

A number of manufacturing companies require auxiliary equipment, such as molds, for their production processes. We base our study in a plastic injection system where molds are employed for shaping plastics into useful objects. These molds vary in shape, size, and mechanical properties as they are used for the production of different types of pieces. Generally, molds are extremely heavy and need to be moved with cranes, resulting in a large amount of time used for their installation, preparation, and removal (two hours on average for the plastic injection molds). The problem arises when different molds are able to process the same type of piece and when each mold can be installed on different machines, which represent assignment decisions. If we add the fact that for every piece-mold pair there is a different production rate, then the problem is evident: how to make an accurately schedule of the process to reduce wasted time due to installation and removal of molds. Moreover, the objective of minimizing the cost of the unfulfilled demand is interesting for the companies, since they are commonly forced to buy pieces from a third party to fulfill the demand, in an effort to retain customers. We name this problem as the Piece-Mold-Machine Problem (PMMP).

The characteristics of the PMMP system are the following (see Figure 1).

- Dedicated molds: Each piece of type i has a set of molds $J(i)$ that can be used to make it. In Figure 1 piece 1 can be produced with either mold 1 or mold 2.
- Different production rates: Each piece-mold compatible combination has its specific production rate due to technical differences of the molds.
- Dedicated machines: Due to technical differences, each mold j can be assigned to a certain number of machines $M(j)$. In Figure 1, mold 2 can only be placed on machine 1.
- The demand of the plastic pieces is seldom fulfilled totally by the company. Thus, the production capacity is expected to be fully used.

Therefore, it is necessary to make an accurate lot-sizing and scheduling of the pieces considering real system constraints. The system we base our study on has constraints such as scarce auxiliary equipment (molds), time-machine bounds, and presence of setup times. When the production of a specific type of piece on some machine is finished, there are two possible situations: to process the next type of piece with the same mold or to switch the mold. In the former case, a preparation time (named here as *piece-setup time*) for the next type of piece must be considered. For instance, this piece-setup time could represent a color or plastic change. In the latter case, if another mold is used, its installation and removal time incur in a *mold-setup time.* Indeed, setup times are dependent of both the last processed event and the event to be next. In general, mold-setup times are larger than piece-setup times.

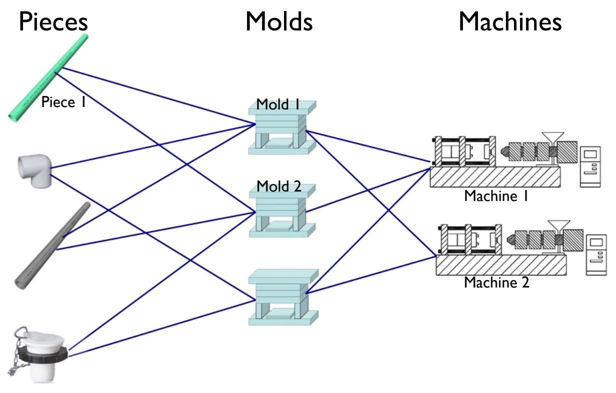


Figure 1: Illustration of a PMMP diagram.

In Figure 2 a feasible solution of the PMMP system we are focusing on is represented. On machine 2 there is a mold-setup time because mold 1 is placed after mold 4. Notice that mold 1 is also used on machine 1, so we must consider that a mold cannot be used on more than one machine at the same time. On machine 3 there are piece-setup times between the types of pieces 7 and 4 since mold 3 is used to produce pieces 7, 4, 5, and 9.

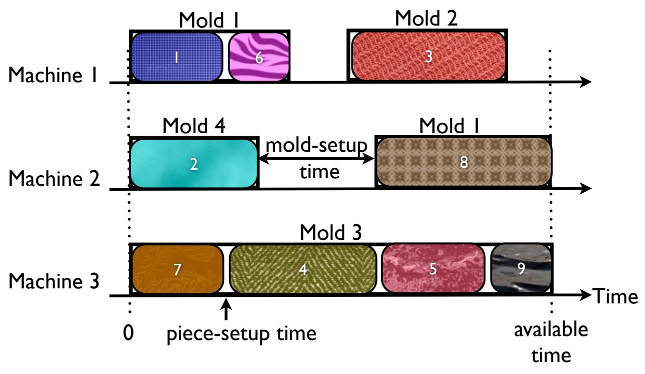


Figure 2: PMMP production planning Gantt chart.

To consider the auxiliary equipment requirements in parallel machines one needs to avoihe overlapping of processes that use the same mold, since most of the times this equipment is scarce. Usually, scheduling formulations make assumptions about the auxiliary equipment such as unique machine assignments. However, this implies sub-optimal solutions since it does not consider the possibility of processing pieces with a specific mold in different machines, which is a feasible solution and often occurs in real-life systems. Therefore, our main contribution is a new approach that gives us the quantities of pieces to produce and the piece-mold-machine assignments such that there is no overlapping of scarce equipment along the planning period.

The new model representing PMMP is a quadratically constrained integer linear program (QCILP) which is difficult to solve. Therefore, we propose a decomposition approach based on lot-sizing and scheduling mixed integer linear programming (MILP) formulations that gives high quality solutions in a reasonable time. A QCIQP is an optimization problem in which both the objective function and the constraints are quadratic functions (Boyd and Vandenberghe 2004). PMMP is a special case of QCIQP since the quadratic coefficients of its objective function are equal to 0, i.e., its objective function is linear but some of its restrictions are quadratic.

The rest of the paper is organized as follows. First, in Section 2 is presented some related literature from the scheduling and lot-sizing fields. The mathematical model of PMMP is described in Section 3. Since its computational time is large, in Section 4 we propose a decomposition approach by decomposing its QCILP formulation into two MILP subproblems. The first one is a formulation with the objective of minimizing the weighted unfulfilled demand that sets the lot-sizing of each piece and the piece-mold-machine assignments. The second one verifies if the solution from the first stage has a feasible mold to machine schedule within the planning period. In Section 5, we provide empirical evidence of the efficiency of our approach on real-word instances. We conclude with Section 6 with some final remarks and directions for future research.

2 Literature Review

There are several features present in our problem corresponding to scheduling and lot-sizing problems. Scheduling problems often consider sequence dependent setup times between the processing of different types of jobs. However, most of these works are focused on the single machine case (Eren 2007; Eren and Güner 2006; Sourd 2006; Stecco, Cordeau, and Moretti 2008). Studies that deal with the parallel machine case and sequence dependent setup times, for example Weng, Lu, and Ren (2001), do not consider the use of auxiliary equipment such as molds. Other formulations that present related characteristics to our problem are based on job grouping or scheduling families of jobs (Chen and Powell 2003; Haase and Kimms 2000; Li, Li, and Zhang 2005; Shim and Kim 2008; Vilím 2006). However, these types of problems, once again, do not consider additional equipment (e.g. molds).

Chen and Wu (2006) study a scheduling problem on parallel machines with dedicated requirements for the use of molds. Setup times occur when there is a job requiring a different mold than the previous one. Although the authors consider the requirement of molds, there is no possibility of producing a piece with a set of molds, i.e., the piece-mold assignment is not needed as it is in our problem. There are other formulations of scheduling problems that consider sequence dependent setup times and mold requirements, such as Lin, Wong, and Yeung (2002), where pieces have to be assigned to specific molds with different production rates. It is noteworthy that a mold is assigned to one machine only. Boctor, Renaud, and Rapp (2009) address a scheduling problem with the multiple-mold requirement, where processing a specific job requires two specific mold types and a setup implies stopping the entire production, since molds are unique and share common tools. This study has, as far as we know, the most similar characteristics of mold handling when compared to our problem, since the authors ensure molds that use common tools do not overlap for different machines and each instance of time. The authors design heuristic procedures to solve it.

The PMMP has some features present in lot-sizing models such as machine-time capacity, equipment assignment, and sequence dependent setup times (an excellent overview on lot-sizing problems can be found in Karimi, Ghomi, and Wilson (2003)). Chen, Ye, and Zhang (2006) address the problem of scheduling, where setups appear when different batch types (or job families) are contiguously processed. Even so, auxiliary equipment is not considered. Dastidar and Nagi (2005) consider lot-sizing and scheduling characteristics such as quantity production calculation, mold-machine assignment, and sequence-dependent setup times. Nevertheless, the authors suppose that a mold can be assigned to only one machine in each planning period. Ibarra-Rojas, Rios-Solis, and Chacon (2010) make the same simplification. This allows them to derive an integer linear program by eliminating the sequence-dependent setup times since they make the assumption of producing the pieces assigned to the same mold one after the other. They determine the piece-mold assignment and the lot-size for each piece-mold pair. They show that the problem is NP-hard and propose an iterated local search algorithm for finding high quality solutions.

3 Quadratically Constrained Linear Program Model for PMMP

In this section we present a model that integrates the lot-sizing and the scheduling characteristics of PMMP. We assume that a mold is installed at most once on each machine, but notice that it can visit many machines. Indeed, if we have a mold that is installed twice in the same machine then a solution that reduces the mold-setup times is the one that merges these two occurrences into one. On the left hand side of Figure 3, on machine 1 we have that mold 1 is installed twice. On the right hand side we have another solution that has merged these two occurrences. Notice that mold 1 on machine 2 had to be rescheduled to avoid mold overlapping. This grouping idea is consistent with the way that companies make their production and has been reported to yield positive results when implemented in solution algorithms (e.g., Chen and Wu (2006)). In fact, the company in which this study is based on, uses this assumption in practice.

For the formulation, let I be the set of pieces the company can produce, J the set of available molds for the production of the pieces, and M the set of available machines. We define $J(i)$ as the set of molds that can produce piece i, $M(j)$ as the set of machines compatible with mold j, $I(j)$ as the set of pieces that can be produced using mold j, and $Jm(k)$ as the set of molds that can

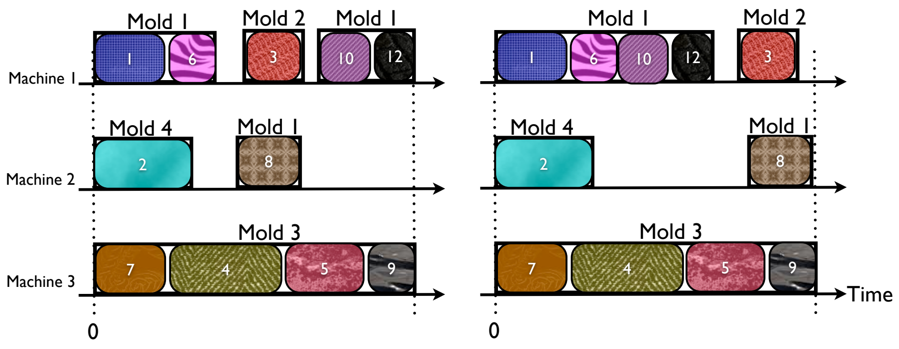


Figure 3: Illustration of the assumption that a mold is installed at most once on each machine.

be placed on machine k. The problem parameters required for our mathematical model are now described, and are summarized in Figure 5 of Section 4.

Demand of piece i is denoted as d_i. The parameter denoted by st_{ij} corresponds to the piece-setup time of i if produced with mold j, which is independent of the job sequence. Parameters it_{jk} and dt_{jk} represent the installation and removal time of mold j on machine k, respectively. There are molds that can produce the same piece type but at a different rate (more or less cavities) due to technological reasons. Hence the inverse production speed of piece i when using mold j is represented by v_{ij}. Each machine k has an available time for production: tm_k. These times can vary from one machine to another due to the company's preventive maintenance plans.

It is necessary to include a set of binary variables B_{ijk}, taking the value of 1 when there is at least one piece of type i produced with mold j in machine k. Similarly, we make use of another set of binary variables N_{jk}, which take the value of 1 if mold j is installed on machine k. Variables X_{ijk} represent the amount of pieces i to be produced with mold j on machine k.

A common problem in industry is minimizing the cost of the unfulfilled demand. As a consequence, we implicitly seek to maximize the cost of the weighted fulfilled demand. The weight w_i of each piece can be the cost of buying it from another company. Therefore, the objective function of the Piece-Mold-Machine Problem is then

$$\max \sum_{i \in I} w_i \sum_{j \in J(i)} \sum_{k \in M(J)} X_{ijk}. \tag{1}$$

System constraints for the lot-sizing of the pieces and for the piece-mold-machine assignments are described by the following expressions.

$$\sum_{j \in J(i)} \sum_{k \in M(j)} X_{ijk} \leq d_i \qquad i \in I \tag{2}$$

$$X_{ijk} \leq d_i B_{ijk} \qquad i \in I, j \in J(i), k \in M(j) \tag{3}$$

$$\sum_{j \in Jm(k)} [N_{jk}(it_{jk} + dt_{jk}) + \sum_{i \in I(j)} (X_{ijk} v_{ij} + B_{ijk} st_{ij})] \leq tm_k \qquad k \in M \tag{4}$$

$$\sum_{i \in I(j)} B_{ijk} \leq |I(j)| N_{jk} \qquad j \in J, k \in M(j) \tag{5}$$

$$X_{ijk} \in \mathbb{Z}^+, B_{ijk} \in \{0,1\}, N_{jk} \in \{0,1\} \qquad j \in J, i \in I(j), k \in M(j)$$

Constraints (2) limit the quantity of produced pieces of type i to the demanded amount, as demand must not be exceeded. Constraints (3) avoid producing pieces of type i with mold j mounted on machine k if the pieces of type i are not assigned to mold j or if mold j is not assigned to machine k. Constraints (4) limit the available time of each machine by taking into account the mold-setup times, the production speed of the pieces, and the pieces-setup times. Constraints (5) assign mold j to machine k if there is at least one piece produced with this mold-machine pair, regardless of the type of piece. Finally, last constraints define the domain of the decision variables. Notice that sets $I(j)$, $M(j)$, $J(i)$, and $Jm(k)$ represent the equipment compatibility.

It is important to recognize as an infeasible case the situation where a mold is assigned to different machines in such a way that this mold processes pieces simultaneously on these machines, i.e., the situation when we have overlapping molds. One may think that this situation is not very common, since the setup times tend to be minimized, but in Section 5 we empirically prove that it happens more often than imagined. However, by adding to the model the following set of constraints

$$\sum_{k \in M(j)} [N_{jk}(it_{jk} + dt_{jk}) + \sum_{i \in I(j)} (X_{ijk} v_{ij} + B_{ijk} st_{ij})] \quad \leq \quad \max_k \{tm_k\} \quad j \in J, \tag{6}$$

we considerably decrease the amount of overlapping molds as we are limiting the time used by mold j in all machines to the maximum available machine time. Nevertheless, the possibility of overlapping molds still exists. To ensure that we have a feasible solution we must add new sets of variables and restrictions that guarantee a feasible scheduling of the molds on the machines. First, one needs dependent variables T_{jk} to track the total time each mold j is used on machine k:

$$T_{jk} \quad = \quad N_{jk}(it_{jk} + dt_{jk}) + \sum_{i \in I(j)} (X_{ijk} v_{ij} + B_{ijk} st_{ij}), \quad j \in J, k \in M(j). \tag{7}$$

Then, we introduce the set of precedence binary variables $F_{j'jk}$, taking the value of 1 when mold j is installed after mold j' in machine k. In counterpart, the set of binary variables $G_{jk'k}$ take the value of 1 if mold j is installed on machine k' before it is installed on machine k. Moreover, we make use of binary variables Y_{jlk}, acquiring the value of 1 if mold j is the l-th event in machine k, i.e., each event of machine j is the installation of a new mold. Finally, variables Z_{jqk} take a value of 1 if machine k is the q-th machine visited by mold j. Event index l belongs to the set $L^k = \{1, \cdots, \sum_j N_{jk}\}$ for all k. Similarly, $q \in Q^j = \{1, \cdots, \sum_k N_{jk}\}$ for all j. The set of constraints that make a feasible scheduling of the molds on the machines are as follows.

$$\sum_{j' \neq j} T_{j'k} F_{j'jk} + T_{jk} \leq tm_k \qquad j \in J, k \in M(j) \tag{8}$$

$$\sum_{j' \neq j} T_{j'k'} F_{j'jk'} + T_{jk'} \leq \sum_{j' \neq j} T_{j'k} F_{j'jk} + \mathcal{M}(1 - G_{jk'k}) \qquad j \in J, k, k' \in M \tag{9}$$

$$\sum_{l \in L^k} Y_{jlk} = N_{jk} \qquad k \in M, j \in Jm(k) \tag{10}$$

$$\sum_{j \in Jm(k)} Y_{jlk} = 1 \qquad k \in M, l \in L^k \tag{11}$$

$$\sum_{l \in L^k} (lY_{jlk}) - N_{jk} = \sum_{j' \neq j} F_{j'jk} \qquad k \in M, j \in Jm(k) \tag{12}$$

$$\sum_{q \in Q^j} Z_{jqk} = N_{jk} \qquad j \in J, k \in M(j) \tag{13}$$

$$\sum_{k \in M(j)} Z_{jqk} = 1 \qquad j \in J, q \in Q^j \tag{14}$$

$$\sum_{q \in Q^j} (qZ_{jqk}) - N_{jk} = \sum_{k' \neq k} G_{jk'k} \qquad j \in J, k \in M(j) \tag{15}$$

$$F_{j'jk} \leq N_{jk} N_{j'k} \qquad j \in J, k \in M(j), j' \neq j \tag{16}$$

$$G_{jk'k} \leq N_{jk} N_{jk'} \qquad j \in J, k \in M(j), k' \neq k \tag{17}$$

$$G_{jk'k} + G_{jkk'} \leq 1 \qquad j \in J, k, k' \in M \tag{18}$$

$$F_{j'jk} + F_{jj'k} \leq 1 \qquad j, j' \in J, k \in M \tag{19}$$

$$F_{j'jk},\ G_{jk'k},\ Y_{ljk},\ Z_{qjk} \in \{0, 1\} \quad j, j' \in J, k, k' \in M, l \in L^k, q \in Q^j$$

Constraints (8) bound the completion time of mold j on machine k to the available machine time. Constraints (9), illustrated by Figure 4, imply that the time when mold j can be installed on machine k must be larger than its completion times on other machines k' where is was placed before. Thus, mold j must be placed on machine k after all its previous uses on different machines have been entirely finished. Value $\mathcal{M}$ corresponds to a very large number. Notice that (8) and (9) are non linear.

Assignment constrains (10) express that if mold j must be placed on machine k ($N_{jk} = 1$) then this event must be one of the l events that are going to take place on machine k, i.e., these equations assign one of the possible events to the task conformed by the use of mold j in machine

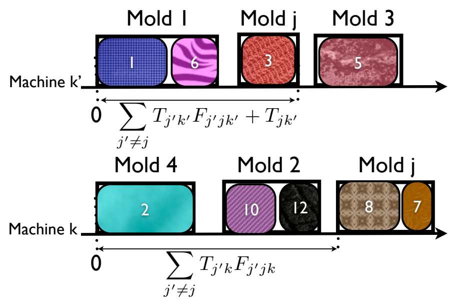

Mold 1
Mold j
Mold 3
1
6
3
5
Machine k'
0
$\sum_{j'\neq j} T_{j'k'}F_{j'jk'} + T_{jk'}$
Mold 4
Mold 2
Mold j
2
10
12
8
7
Machine k
0
$\sum_{j'\neq j} T_{j'k}F_{j'jk}$

Figure 4: Illustration of constraints (9) when $G_{jk'k} = 1$.

k. Constraints (11) say that each event in L^k of each machine k corresponds to only one mold-(machine k) assignment. Expressions (12) define that the number of events that take place before mold j is installed on machine k must be equal to the number of molds j' installed before j on machine k.

Constraints (13)-(15) are analogous to (10)-(12) but now from the point of view of the machines that every mold visits. For instance, constraints (13) assign the q-th machine where mold j is going to be installed. Constraints (14) are the assignments of the q visits of mold j. Constraints (15) sets the $q-1$ machines visited by mold j before being installed on machine k. Moreover, constraints (16) and (17), that can be easily linearized, set the relationship between variables $F_{j'jk}$ and $G_{jk'k}$ with N_{jk} and $N_{j'k}$. Constraints (18) and (19) avoid inconsistencies on the precedence variables. Finally, last constraints define the domain of the decision variables.

Summarizing, the QCILP for PMMP is as follows:

$$\begin{array}{ll}
\max & \sum_{i\in I} w_i \sum_{j\in J(i)} \sum_{k\in M(j)} X_{ijk} \\
\text{subject to} & (2)-(19) \\
X_{ijk}\in\mathbb{Z}^+, B_{ijk}\in\{0,1\}, N_{jk}\in\{0,1\} & j\in J, i\in I(j), k\in M(j) \\
F_{j'jk},\ G_{jk'k},\ Y_{ljk},\ Z_{qjk}\in\{0,1\} & j,j'\in J, k,k'\in M, l\in L^k, q\in Q^j
\end{array}$$

As mentioned in Section 2, in Ibarra-Rojas, Rios-Solis, and Chacon (2010) the authors propose a similar problem to PMMP but suppose that a mold can be assigned to only one machine. Their mathematical model substantially differs from ours since they do not have to guarantee a feasible schedule. Nevertheless, the same reduction they use to prove the NP-hardness of their problem can be adapted for PMMP.

Solving PMMP with a quadratic branch and bound (B&B) such as in CPLEX 11.2 did not yield feasible solutions in less than one hour. Indeed, constraints (8) and (9) are not convex, therefore the B&B algorithm tries to convexify them which is probably the reason for the inefficiency of this approach. The number of original variables is already considerable, therefore we do not try to linearize the quadratic constraints. Instead, we propose to decompose PMMP into two MILPs and solve them sequentially, as will be shown in Section 4.

4 A Decomposition Approach for PMMP

Given the complexity of PMMP, a decomposition approach is proposed. This method attempts to exploit the structure of two related subproblems. In the first level, the Lot-Sizing and Assignment subproblem (LSA) obtains a lot-sizing of the pieces together with a mold-machine assignment. Then, in a second level, a feasibility subproblem called Mold Overlapping Detection (MOD) is solved to verify if there is a scheduling of the molds on the machines for the solution obtained by LSA.

LSA is a MILP given by:

$$\begin{aligned} \max \quad & \sum_{i \in I} w_i \sum_{j \in J(i)} \sum_{k \in M(j)} X_{ijk} \\ \text{subject to} \quad & (2) - (6) \\ & X_{ijk} \in \mathbb{Z}^+, S_{ijk} \in \{0,1\}, N_{jk} \in \{0,1\} \quad j \in J, i \in I(j), k \in M(j) \end{aligned}$$

As mentioned before, LSA gives as a solution the number of pieces to produce of each type, the molds in which the pieces will be processed, and the machines where the molds are going to be placed. Nevertheless, this solution may be infeasible, since a single mold can be installed in two machines at the same time. Although the presence of constraints (6) considerably reduce the amount of overlapping molds, in Section 5 we provide evidence of the mold overlapping situation. Note LSA formulation is a relaxation of PMMP.

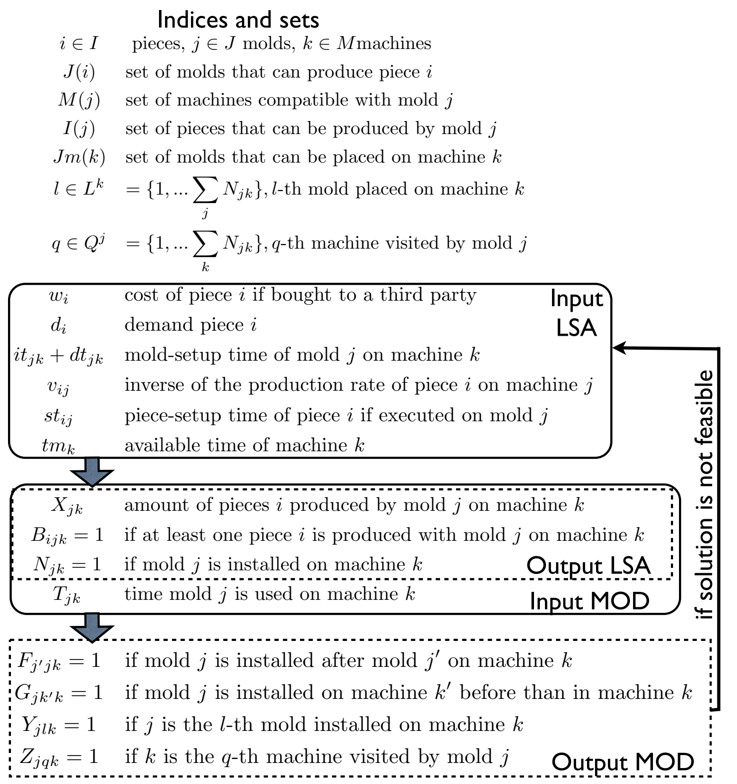
There is a simple way to verify if a solution given by LSA is feasible by looking at the values of the variables N_{jk} (mold j is installed or not on machine k). There is no risk of having overlapping molds if each of the molds is assigned to at most one machine, that is,

$$\sum_{k \in M(j)} N_{jk} \leq 1, j \in J. \tag{20}$$

This is a sufficient condition that guarantees that the solution given by LSA is optimal to PMMP. If conditions (20) are not satisfied, we have at least one mold assigned to two or more machines implying there could be mold overlapping in the schedule that must be detected. To this end, we solve MOD that is an integer linear constraint satisfaction problem where N_{jk}, S_{ijk}, and T_{jk} correspond now to input parameters.

MOD: Are there 0-1 values for variables $F_{j'jk}$, $G_{jk'k}$, Y_{ljk}, Z_{qjk} ($j, j' \in J, k, k' \in M, l \in L^k, q \in Q^j$) such that constraints (7)-(19) are satisfied?

Figure 5 compiles the notation we have used so far. It also schemes the input/output relations between LSA and MOD.



$i \in I$	pieces, $j \in J$ molds, $k \in M$machines
$J(i)$	set of molds that can produce piece i
$M(j)$	set of machines compatible with mold j
$I(j)$	set of pieces that can be produced by mold j
$Jm(k)$	set of molds that can be placed on machine k
$l \in L^k$	$= \{1, ... \sum_j N_{jk}\}$, l-th mold placed on machine k
$q \in Q^j$	$= \{1, ... \sum_k N_{jk}\}$, q-th machine visited by mold j
w_i	cost of piece i if bought to a third party
d_i	demand piece i
$it_{jk} + dt_{jk}$	mold-setup time of mold j on machine k
v_{ij}	inverse of the production rate of piece i on machine j
st_{ij}	piece-setup time of piece i if executed on mold j
tm_k	available time of machine k
X_{jk}	amount of pieces i produced by mold j on machine k
$B_{ijk} = 1$	if at least one piece i is produced with mold j on machine k
$N_{jk} = 1$	if mold j is installed on machine k
T_{jk}	time mold j is used on machine k
$F_{j'jk} = 1$	if mold j is installed after mold j' on machine k
$G_{jk'k} = 1$	if mold j is installed on machine k' before than in machine k
$Y_{jlk} = 1$	if j is the l-th mold installed on machine k
$Z_{jqk} = 1$	if k is the q-th machine visited by mold j

Figure 5: Input and output relations between LSA and MOD.

The decomposition method, named as DPMMP, is presented in Algorithm 1. First LSA is solved. If sufficient conditions (20) are verified, then the LSA solution is overlapping free, hence optimal. Otherwise, MOD is solved with the values of variables $\bar{X}_{ijk}$, $\bar{N}_{jk}$, and $\bar{B}_{ijk}$ as parameters. If there is a feasible mold to machine schedule, then the LSA solution is optimal for PMMP. If this is not met, then one must solve again the LSA model but this time the previously obtained solution $\bar{X}_{ijk}$ must be avoided (as described in the next subsection).

LSA and MOD can be solved by a B&B algorithm. Note that although Step 11 may yield an exponential number of iterations, in practice this seldom happens. In Section 5, experimental evidence shows that DPMMP procedure needs at most two iterations.

Cutting-off a Previous LSA Infeasible Solution for MOD

As mentioned before, when the problem MOD turns out to be infeasible it means that the last LSA solution must be avoided from the solution set of LSA. To this end, we propose the following cut.

Let $\bar{x} = (\bar{x}_1, \ldots, \bar{x}_n)$ be the values of the solution to be cutoff. Let $\mathcal{I} = \{i|\bar{x}_i = 1\}$ and $\bar{\mathcal{I}} = \{i|\bar{x}_i = 0\}$. Then the following constraint cuts-off this particular solution without cutting off

Algorithm 1 DPMMP: Decomposition algorithm for PMMP.

```
Input: Instance of PMMP
Output: $X_{ijk}$, $Y_{jlk}$ {Quantity-assignment variables and scheduling variables}
1: $FeasibleFlag \leftarrow \emptyset$
2: while ($FeasibleFlag = \emptyset$) do
3:     Solve LSA to obtain the values of the variables $\bar{X}_{ijk}$, $\bar{N}_{jk}$, and $\bar{B}_{ijk}$
4:     if ($\bar{N}_{jk} \leq 1, \forall j \in J, k \in M(j)$) then
5:         $FeasibleFlag \leftarrow 1$ {The LSA solution is mold overlapping free}
6:     else
7:         Solve MOD with values of $\bar{N}_{jk}$, $\bar{B}_{ijk}$ and $\bar{X}_{ijk}$ as input
8:         if (MOD is feasible) then
9:             $FeasibleFlag \leftarrow 1$ {The LSA solution is mold overlapping free}
10:        else
11:            Add to LSA cut (21) associated to the actual values of $\bar{X}_{ijk}$
12:        end if
13:    end if
14: end while
```

any other feasible solution for LSA:

$$\sum_{i \in \mathcal{I}} x_i + \sum_{i \in \bar{\mathcal{I}}} (1 - x_i) \leq n - 1. \tag{21}$$

Indeed, the only manner this inequality could be equal to n is by taking the values of $\bar{x}$. This cut is useful for binary variables but it is well known that any integer can be expressed in a binary form.

5 Experimental Results

In this section we test the efficiency of algorithm DPMMP to solve PMMP. To this end, we use the same instance generator proposed by Ibarra-Rojas, Rios-Solis, and Chacon (2010) since, except for the weight of pieces, it is based on information provided by a real manufacturer of plastic products. Parameters are integers uniformly generated within the following ranges: $d_i \in [1000, 18000]$, $v_{ij} \in [1/120, 1/1000]$, $w_i \in [1, 10]$. The piece-setup time is set to $st_{ij} = 0$, for all i and j. Indeed, these values do not affect the behavior of the algorithm, provided that $st_{ij} \ll it_{jk} + dt_{jk}$. The available time of all the machines is set to $t_k = 24$ hours.

On the one hand, the difficulty of the instances resides on the mold-setup times (in hours). Hence, we generate two set of instances. Mold-setup times of instance set E take integer values uniformly generated as follows: $it_{jk} + dt_{jk} \in [.75, 1.15]$. More difficult instances H are obtained by uniformly generating the installation setup time and the removal setup time independently from [.75, 1.15], i.e., $it_{jk} \in [.75, 1.15]$, and $dt_{jk} \in [.75, 1.15]$.

On the other hand, we consider (as in Ibarra-Rojas, Rios-Solis, and Chacon (2010)) two density levels (percentage of compatibilities) for the piece-mold compatibility, r_{ij}, and the mold-machine compatibility, m_{jk}. In general, a higher density makes a more difficult instance since there are more choices (therefore more variables). We use densities of 5% and 15% for the piece-mold compatibility together with a density of 60% for the mold-machine compatibility. Preliminary tests showed that the difficulty of an instance does not reside in the mold-machine density and 60% is close to the reality. Instances with 5% for the piece-mold compatibility correspond to a small enterprise or to old molds and machines. Realistic values are around 15%.

Note that we cannot compare our results to the ones of Chen and Powell (2003), Tsai and Tseng (2007), or Ibarra-Rojas, Rios-Solis, and Chacon (2010) because they assume that a mold can be only placed on at most one machine, so in essence we address a different problem.

For solving the instances we used GAMS/CPLEX 11.2 as the MILP optimizer (we kept the default settings except for the optimality tolerance that we set to 1×10^{-8}) on a Sun Fire V440 with 4 UltraSparc III processors at 1.062 GHz and 8 GB of RAM. We set a time limit of one hour for each iteration of LSA, and another hour limit for each iteration of MOD.

Table 1 presents the experimental results obtained by solving the instance set E with DPMMP, while Table 2 is for the instance set H. Each row of these tables is the average of 10 executions of different instances. The variance between each class of instances is not significant, so the average is a good indicator of the results. The tables present only the first iteration of the algorithm since only two out of 100 needed a second iteration (these two instances will be discussed later). Column "Instances" refers to the instance size: number of pieces, number of molds, and number of machines, $(|I|, |J|, |M|)$. Column "Densities" is about the densities of the piece-mold and the mold-machine compatibilities denoted as r_{ij} and m_{jk}, respectively. Column "LSA GAP %" is the relative gap obtained when solving LSA with CPLEX B&B. Column "LSA Time" is the average time in seconds that CPLEX took to solve LSA, limited to one hour. Column "Molds ($\sum_k N_{jk}$)" indicates how many molds are assigned to different machines in the most extreme single instance: a value of 5→2 indicates that there is an instance for which 5 molds visit, each of them, 2 machines ("-" indicates that all molds visit only one machine). Column labeled as "MOD %" shows the percentage of instances that needed MOD to check for a feasible scheduling. The "Overlap free %" column represents the percentage of instances that have a feasible scheduling for the LSA solution. Finally, the last column is the average time in seconds needed by CPLEX to solve MOD.

The real size of the instances we observed in a plastic manufacturing company were (120, 80, 20) with a density of (5, 60), from the instance set E. Indeed, the maximum size of the instances presented in Tables 1 and 2 is, to the best of our knowledge, the largest in literature.

Instances	Densities	LSA	LSA	Molds	MOD	Overlap	MOD
$(\lvert I\rvert, \lvert J\rvert, \lvert M\rvert)$	(r_{ij}, m_{jk})	GAP %	Time	$(\sum_k N_{jk})$	%	free %	Time
(50,30,5)	(15,60)	0.00	103.4	1→2	10	100	0.4
(120,80,20)	(5,60)	0.76	3600.0	5→2	100	100	14.3
(120,80,20)	(15,60)	1.00	3600.0	5→2	80	100	14.3
(200,120,25)	(5,60)	1.09	3600.0	7→2	100	100	39.9
(200,120,25)	(15,60)	2.49	3600.0	13→2 and 1→3	100	80	41.0

Table 1: Results obtained by solving PMMP with algorithm DPMMP on the instance set E (shorter mold-setup times).

Instances	Densities	LSA	LSA	Molds	MOD	Overlap	MOD
$(\lvert I\rvert, \lvert J\rvert, \lvert M\rvert)$	(r_{ij}, m_{jk})	GAP %	Time	$(\sum_k N_{jk})$	%	free %	Time
(50,30,5)	(15,60)	0.00	71.4	–	0	–	–
(120,80,20)	(5,60)	2.73	3600.0	3→2	80	100	14.2
(120,80,20)	(15,60)	4.17	3600.0	3→2	80	100	14.5
(200,120,25)	(5,60)	3.83	3600.0	6→2	100	100	39.9
(200,120,25)	(15,60)	6.20	3600.0	4→2	80	100	39.6

Table 2: Results obtained by solving PMMP with algorithm DPMMP on the instance set H (longer mold-setup times).

The density of the piece-mold compatibilities is a factor that makes an instance harder than another one, regardless of the mold-setup times. Indeed, when the piece-mold compatibility is only 5% there will be less options, therefore less variables.

Tables 1 and 2 show that the solutions given by LSA are close to the optimum since the gap is never above 7% (or is never above 2.5% under the real system characteristics denoted in set E), which is for the industry more than acceptable. Notice that the gaps for the harder instance set H are larger since longer mold-setup times make the assignments of the molds to the machines more difficult.

Except for the smallest instances, the computational times for the PMMP model reach the time limit for LSA. Nevertheless, letting LSA run for more than one hour did not improve the gap much. Figure 6 represents the relative gap of LSA for a single instance of set E along the time. Notice that the quality of the solution is already acceptable after one hour: Although the gap keeps decreasing beyond this limit, the drop in the gap gets shorter as the computational time increases.

We noted that the harder instances H required fewer times the use of MOD. Indeed, since the mold-setup times are larger, in the optimal solution there is less tendency to use a mold on several

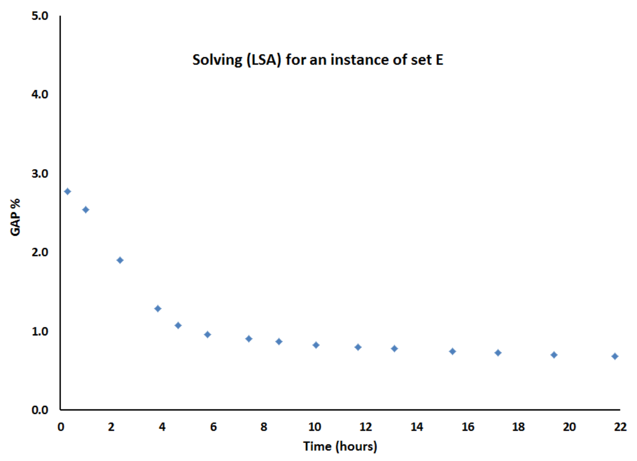


Figure 6: Relative gap of LSA for a single instance of size (200,120,80) with density (15,60) of set E along the time.

machines. Nevertheless, notice the case (200,120,25) from instance set H where six molds visit, each one of them, two machines.

The counterintuitive case of molds assigned to several machines is in fact real and happens frequently, making it necessary, in most of the cases, to use MOD. Moreover, MOD finds most of the times a feasible mold to machine scheduling in less than one minute which is surprising for a scheduling problem that has a very large value $\mathcal{M}$ in constraint (9). Actually, the number of molds that may overlap in several machines is small compared to the ones that are fixed to only one machine, making easy for MOD to find a feasible scheduling. Notice that the computational time of MOD is equivalent for both H and E.

In Table 1 we can see more counterintuitive cases. For example, the last line instances present an instance where 13 molds are used twice on different machines and one mold is used on three different machines. Nevertheless, it is in this set where we find the two instances out of 100 for which MOD could not find a feasible mold to machine scheduling for the solution given by LSA. These instances needed a second iteration of the algorithm. We summarize the results obtained:

- The execution times of the second iteration where similar to the ones of the first iteration.
- After the second iteration, the decrement of the initial solution value is around 0.25%.
- Although the more convenient way of cutting off the infeasible solution of LSA is by including the cut (21) (this restriction only forbids the infeasible solution), we obtained the same results (same solution and same objective value) by forbidding the previous value of the objective function $\bar{z}$ i.e., adding to LSA $z \leq \bar{z} - 1$. Contrary to cut (21), this last option does not guarantee that we are leaving out an interesting feasible solution.

- Experimentally, we did not find two different solutions with the same objective value which validates the use of the simpler cut.

From this experimental section we conclude that our decomposition approach is efficient since for the company the solutions are close enough to the optimum. In practice, the production planning must be done only once a week therefore the computational time of our algorithm is reasonable. As mentioned before, the optimality gap in this study was set close to 0. For a practical use, a company could set the optimality gap to 1% to reduce the execution time of LSA.

6 Conclusions

We present in this study a real manufacturing process of pieces that are produced with molds which are mounted on machines. Setup times between jobs, dedicated parallel machines, dedicated molds and different production rates for each piece-mold pair are some of the main characteristics of the problem. Moreover, we do not make the common assumption of forcing a mold to be placed on a single machine. As a consequence, we deal with a more realistic description of the production process itself. The objective function is about maximizing the weighted cost of the produced pieces.

We first propose a new integer quadratically constrained linear programming that represents the PMMP problem. Since this model is difficult to tackle by itself, we present a decomposition approach based on two new MILPs: the first one determines the lot-size of each piece and the piece-mold-machine assignments, and the second one verifies that there is a feasible mold to machine scheduling along the planning period.

Experimental results show that indeed, most of the solutions yield to configurations where the molds visit more than one machine. Some instances have molds that visit up to three machines. Our exact-based methodology gives, in a reasonable amount of time, solutions of high quality for real size instances.

Acknowledgements: This research was partially supported by grant 101857 from the Mexican National Council for Science and Technology (CONACYT). Omar Ibarra-Rojas and Mario Saucedo-Espinosa wish to acknowledge graduate scholarships from CONACYT. We thank Edgar Possani for his valuable comments. We are very grateful to the anonymous reviewers for their kind comments and suggestions that improve the presentation of this work.

References

Boctor, F., J. Renaud, and J.-E. Rapp (2009). Sequencing and scheduling multi-mold injection molding machines. Working paper 2009–015, Université Laval, Québec, Canada.

Boyd, S. and L. Vandenberghe (2004). *Convex Optimization*. Cambridge, UK: Cambridge University Press.

Chen, B., Y. Ye, and J. Zhang (2006). Lot-sizing scheduling with batch setup times. *Journal of Scheduling 9*(3), 299–310.

Chen, J.-F. and T.-H. Wu (2006). Total tardiness minimization on unrelated parallel machine scheduling with auxiliary equipment constraints. *Omega 34*(1), 81 – 89.

Chen, Z.-L. and W. B. Powell (2003). Exact algorithms for scheduling multiple families of jobs on parallel machines. *Naval Research Logistics 50*(7), 823–840.

Dastidar, S. G. and R. Nagi (2005). Scheduling injection molding operations with multiple resource constraints and sequence dependent setup times and costs. *Computers & Operations Research 32*(11), 2987–3005.

Eren, T. (2007). A multicriteria scheduling with sequence-dependent setup times. *Applied Mathematical Sciences 1*(58), 2883–2894.

Eren, T. and E. Güner (2006). A bicriteria scheduling with sequence-dependent setup times. *Applied Mathematics and Computation 179*(1), 378–385.

Haase, K. and A. Kimms (2000). Lot sizing and scheduling with sequence-dependent setup costs and times and efficient rescheduling opportunities. *International Journal of Production Economics 66*(2), 159–169.

Ibarra-Rojas, O. J., Y. A. Rios-Solis, and O. L. Chacon (2010). Piece-mold-machine manufacturing planning. In B. Nag (Ed.), *Intelligent Systems in Operations: Models, Methods, and Applications in the Supply Chain*, pp. 105–117. Hershey: IGI Global.

Karimi, B., S. M. T. F. Ghomi, and J. M. Wilson (2003). The capacitated lot sizing problem: a review of models and algorithms. *Omega 31*(5), 365–378.

Li, S., G. Li, and S. Zhang (2005). Minimizing makespan with release times on identical parallel batching machines. *Discrete Applied Mathematics 148*(1), 127–134.

Lin, C. K. Y., C. L. Wong, and Y. C. Yeung (2002). Heuristic approaches for a scheduling problem in the plastic molding department of an audio company. *Journal of Heuristics 8*(5), 515–540.

Shim, S.-O. and Y.-D. Kim (2008). A branch and bound algorithm for an identical parallel machine scheduling problem with a job splitting property. *Computers & Operations Research 35*(3), 863–875.

Sourd, F. (2006). Dynasearch for the earliness-tardiness scheduling problem with release dates and setup constraints. *Operations Research Letters 34*(5), 591–598.

Stecco, G., J.-F. Cordeau, and E. Moretti (2008). A branch-and-cut algorithm for a production scheduling problem with sequence-dependent and time-dependent setup times. *Computers & Operations Research 35*(8), 2635–2655.

Tsai, C. and C. Tseng (2007). Unrelated parallel-machines scheduling with constrained resources and sequence-dependent setup time. In M. H. Elwany and A. B. Eltawil (Eds.), *Proceedings of the 37th International Conference on Computers and Industrial Engineering*, Alexandria, Egypt.

Vilím, P. (2006). Batch processing with sequence dependent setup times. In P. Van Hentenryck (Ed.), *Principles and Practice of Constraint Programming - CP 2002*, Volume 2470 of *Lecture Notes in Computer Science*, pp. 153–167. Berlin: Springer.

Weng, M. X., J. Lu, and H. Ren (2001). Unrelated parallel machine scheduling with setup consideration and a total weighted completion time objective. *International Journal of Production Economics 70*(3), 215–226.