UNIVERSIDADE DE SÃO PAULO

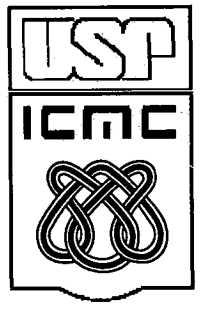
Instituto de Ciências Matemáticas e de Computação

Lot-Sizing and Furnace Scheduling in Small Market-Driven Foundries

Silvio A. de Araújo
Marcos N. Arenales
Alistair R. Clark

Nº 68

NOTAS



São Carlos - SP

UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação
ISSN 0103-2577

Lot-Sizing and Furnace Scheduling in Small Market-Driven Foundries

Silvio A. de Araújo
Marcos N. Arenales
Alistair R. Clark

Nº 68

NOTAS

Série Computação

São Carlos – SP
Fev./2003

Araujo, S. A., Arenales, M. N. and Clark, A. R., Lot-sizing and furnace scheduling in small market-driven foundries, Technical Report 68, ICMC-USP, 2003.

Lot-Sizing and Furnace Scheduling in Small Market-Driven Foundries

Silvio A. de Araujo
Universidade Estadual de Maringá
Departamento de Informática
87020-900, Maringá-PR, Brazil.
Email: Silvio@din.uem.br

Marcos N. Arenales
Universidade de São Paulo - Campus de São Carlos
Instituto de Ciências Matemáticas e de Computação
Caixa Postal 668, 13560-970 São Carlos SP, Brazil.
Email: Arenales@icmc.usp.br

Alistair R. Clark
Faculty of Computing, Engineering and Mathematical Sciences
University of the West of England
Bristol, BS16 1QY, England.
Email: Alistair.Clark@uwe.ac.uk
Fax +44 (0) 117 328 2724

Abstract: This paper researches a lot-sizing and scheduling problem encountered in small market-driven foundries. There are two related decision levels: (1) the furnace scheduling of metal alloy production, and (2) moulding machine planning which specifies the type and size of production lots. A mixed integer programming (MIP) model is proposed for this lot-sizing and scheduling problem and a solution method developed based on local search methods: simple local search, diminishing neighbourhood search and simulated annealing. The production planning is carried out using a rolling horizon schedule where only immediate-term schedules are implemented. The results of computational tests are analysed and compared with those obtained using a commercial MIP software solvers.

Key words: *lot-sizing and scheduling, meta-heuristics, mixed integer programming.*

Resumo: Neste artigo é investigado um problema de dimensionamento de lotes e programação da produção encontrado em fundições de mercado. Existem dois níveis de decisão relacionados: (1) a programação dos fornos, ou seja, a programação das ligas metálicas e (2) o planejamento do setor de moldes, o qual especifica os lotes de produção. Um modelo de programação inteira misto é proposto e um método de solução desenvolvido, o qual é baseado em buscas locais: busca local simples, redução de vizinhança e simulated annealing. O planejamento da produção é feito usando a estratégia de horizonte rolante, em que somente as decisões imediatas são implementadas. Testes computacionais foram realizados com o objetivo de comparar o desempenho das buscas locais com um pacote de otimização.

Palavras-chaves: *Dimensionamento de lotes e programação da produção, meta heurísticas, programação inteira mista.*

1. Introduction

Foundries are common in every region of Brazil, producing many types of item, ranging from simple ones for domestic use to sophisticated parts for the automobile and machine tool industries. According to the Brazilian Foundry Association [1], the sector is growing rapidly, having produced over 9,000 tonnes a day in 2003 and currently directly employs about 50,000 people.

Foundries can be classified as either captive or market-driven. The former are part of large companies, such as car manufacturers, who totally absorb the foundry production, composed of large quantities of a small number of parts with relatively stable demand. On the other hand, market-driven foundries are generally small or medium sizes companies that nevertheless produce a huge range of items with vastly varying demand. While captive foundries use a small number of different metal alloys, market-driven foundries need to work with a wide variety of alloys due to the diversity of their clients. This paper focuses on the problem of planning and scheduling production in such small market-driven foundries.

Research literature reviews of production planning & scheduling (PPS) problems include [2-10]. However, there is little published research on foundry PPS, and even less concerned with market-driven foundries. [11] studied the PPS problem in small and medium-sized foundries, while [12] researched PPS in a large captive foundry. The papers by [13] and [14] are also related to large foundries, but deal specifically with the problem of job sequencing on machines. Most research papers on the issue are concerned with production scheduling in large steel plants. By their nature, such plants do not have moulding sections since they produce steel sheets, of varying sizes and type, in rolling machines with cylinders that are configured in line the sheet specifications. [15] published a review of research on PPS in steel production, while a variety of papers have studies practical problems found in large steel plants, including [16-23].

In this paper, we propose a PPS optimization model that differs from conventional lot-sizing models in that, although it is possible to form lots and produce end-items in advance of demand, only a single one alloy can be produced in each period (furnace cycle) and cannot be stocked. This situation defines a very particular 2-level scheduling problem.

An initial attempt to optimally solve the resulting mixed integer programming (MIP) model using advanced optimisation software was not successful. To try to overcome this, a rolling horizon solution strategy was developed, but still the software was unable to cope. The relax-and-fix heuristic approach [24] was then applied on a rolling horizon basis, giving good results. Relax-and-fix involves the solution of a sequence of partially-relaxed MIPs, each one with just a reduced set of binary variables whose number is small enough to obtain optimal solutions. As the horizon rolls forward in time, each set of binary variables is permanently fixed at their solution values. For comparison, three methods involving neighbourhood search on the reduced set of binary variables

were developed, namely, simple local search, diminishing neighbourhood search and simulated annealing. Computational tests comparing all the methods are presented and analysed.

2. Problem Definition

A key process in a foundry is the transformation of iron ore and scrap metal into alloys with specified levels of carbon, silicon, zinc, etc, that determine its physio-chemical properties such as brittleness, resistance to corrosion, etc. The alloy, still in a liquid state, is then poured into moulds, normally made of sand and resin, and cools to produce the final item. These two processes, alloy production and item moulding, must be jointly scheduled.

In small foundries, a single furnace is usually operating at any time, so that only one alloy can be produced in each period. This is different from medium-sized or large foundries where several furnaces can be in operation simultaneously, enabling the production of several alloys in the same period [12]. Furthermore, in small foundries, the preparation of sand moulds is a manual process which, although carried out beforehand, does not have to be scheduled since the production bottleneck is in fact the furnace. This contrasts with automated foundries, where multiple moulding machines of varying capacity and efficiency have to be scheduled. In this case, the bottleneck could be either mould preparation or the furnaces.

Figure 1 illustrates the main activities in a market-driven foundry. Clients randomly and spontaneously submit orders specifying the item type, quantity and alloy. The Production Planning department negotiates a delivery date with the client, often agreeing unachievable dates that result in delivery delays and possible loss of future business from the client. Thus the minimisation of delivery delays is one of the principal concerns of the foundry company.

Each day, the Production Planning department specifies which alloys and items should be produced and informs the Furnace and Moulding sections of its decisions. The scheduling takes into account the order due in the following days as well as delayed orders. Some orders are given priority, depending on the importance of the client or the pressure exerted.

In each furnace load, an alloy is produced and then poured into previously prepared moulds. Excessive alloys changeovers are undesirable and so setups are important, but their nature is not, however, sequence-dependent. After the cast item has cooled, it is deburred and then made ready for delivery.

3. Mathematical Model

We now propose a mixed integer programme (MIP) to model the problem described above. As mentioned previously, the production bottleneck is the furnace, since all moulds are prepared beforehand in line with demand. The following notation is used:

Indices: $k = 1, ..., K$ alloys; $i = 1, ..., N$ items; $t = 1, ..., T$ periods.

Data:

Cap	Capacity of a single furnace loading
L	Number of furnace loadings per day
ρ_i	Physical weight of item i
d_{it}	Demand for item i in period t
$S(k)$	Set of items i that use alloy k (each item uses one and just one alloy. Thus $\{1, ..., N\} = S(1) \cup ... \cup S(k)$ and $S(h) \cap S(j)$ is empty for all $h \neq j$
h_{it}^-	Backorder penalty for delaying delivery of a unit of item i in period t
h_{it}^+	Inventory penalty for carrying a unit of item i in period t
s_k	Setup penalty for alloy k
st_k	Setup time for alloy k (i.e., the loss of capacity due to a setup for alloy k)

Note: Although demand is daily, the planning periods are defined by furnace loads of which there are L each day. Thus if $t \neq L, 2L, 3L, \ldots$ then $d_{it} = 0$, $h_{it}^- = 0$ and $h_{it}^+ = 0$. These data parameters can be positive only when the subscript t corresponds to the last period of each day i.e., if $t = L, 2L, 3L, \ldots$ then $d_{it} \geq 0$, $h_{it}^- \geq 0$ and $h_{it}^+ \geq 0$.

Variables:

x_{it}	Quantity (lot-size) of item i to be produced in period t
I_{it}^+	Inventory of item i at the end of period t
I_{it}^-	Backlog of item i at the end of period t
y_t^k	Binary variable, = 1 indicates that the furnace is configured for the production of alloy k in period t, otherwise = 0
z_t^k	Binary variable, = 1 if there is a setup of alloy k in period t, otherwise = 0. Thus $z_t^k = 0$ if $y_{t-1}^k \geq y_t^k$ and $z_t^k = 1$ if $y_{t-1}^k < y_t^k$. It is relaxed to be continuous for reasons explained below.

The mathematical model is formulated as:

$$\text{Minimize} \quad \sum_{i=1}^{N}\sum_{t=1}^{T}(h_{it}^{-}I_{it}^{-} + h_{it}^{+}I_{it}^{+}) + \sum_{k=1}^{K}\sum_{t=1}^{T}(s_k z_t^k) \tag{1}$$

subject to:

$$I_{i,t-1}^{+} - I_{i,t-1}^{-} + x_{it} - I_{it}^{+} + I_{it}^{-} = d_{it} \qquad i = 1, ..., N \quad t = 1, ..., T \tag{2}$$

$$\sum_{i \in S(k)} \rho_i x_{it} + st_k z_t^k \leq Cap\, y_t^k \qquad k = 1, ..., K \quad t = 1, ..., T \tag{3}$$

$$z_t^k \geq y_t^k - y_{t-1}^k \qquad k = 1, ..., K \quad t = 1, ..., T \tag{4}$$

$$\sum_{k=1}^{K} y_t^k \leq 1 \qquad t = 1, ..., T \tag{5}$$

$$y_t^k \in \{0,1\} \text{ with } y_0^k = 0 \qquad k = 1, ..., K \quad t = 1, ..., T \tag{6}$$

$$z_t^k \geq 0 \qquad k = 1, ..., K \quad t = 1, ..., T \tag{7}$$

$$x_{it} \geq 0 \text{ integer} \qquad i = 1, ..., N \quad t = 1, ..., T \tag{8}$$

$$I_{it}^{+} \text{ and } I_{it}^{-} \geq 0 \qquad i = 1, ..., N \quad t = 0, ..., T \tag{9}$$

The first part of the objective function (1) minimizes a weighted sum of inventory and backorders penalties for each period. The second part minimizes the setup penalties, i.e., the alloy changes in each period. Thus the objective function seeks a weighted balance between conflicting objectives: stocks, backorders and setups. The human scheduler can use his knowledge and experience by varying the values for h_{it}^{-}, h_{it}^{+} and s_k to explore alternative production schedules. However, multi-objective optimisation theory [25] suggests that there will almost certainly not be a one-to-one correspondence between the (h,s)-parameter values and possible schedules.

Constraints (2) balance inventories, backorders, demands and production of items for every item in each period. Constraints (3) not only keep production within the furnace capacity, but also ensure that only items of the same alloy are produced in a particular furnace loading. As y_t^k and y_{t-1}^k are both binary variables, constraints (4) force the continuous variable z_t^k to be at least 1 if there is a changeover to alloy k and, along with constraints (7), to be at least 0 if any other case. The minimising of the objective function leads the z_t^k variables to assume just 0 or 1 values in any solution which is continuously optimal, even if not integer optimal for the x_{it} and y_t^k variables (for example, at nodes during a branch-and-bound search). Constraints (5) and (6) ensure that there is only a single furnace loading in each period.

In captive foundries, lot sizes tend to be large since there are just a few types of item (generally components of standard products) which have a large stable demand, so that the integrality of x_{it} can

usually be relaxed. However, in contrast, the integrality condition is necessary for small market-driven foundries with their many small orders. Furthermore, an item cannot be partially made in one period and then completed in another.

Constraints (9) measure inventory I_{it}^{+} and backlogs I_{it}^{-} as non-negative variables, but note that in a continuously optimal solution, I_{it}^{+} and I_{it}^{-} will not both be strictly positive, for a given pair (i,t), due to their positive coefficients in the objective function.

Model (1)-(9) shares some similarities with the lot sizing and sequencing models in [8]. Sequencing decisions are implicitly determined by the furnace configuration variables y_t^k, but differently from [8] where an item's setup authorises the production of only that item, in our model the setup of an alloy makes possible the production of any item that use the alloy. Thus the *small bucket* concept in [8] is indirectly present, allowing the production of a particular set of items per period rather than a single item, resulting in a joint scheduling model for material processing (y_t^k) and lot sizing (x_{it}) as in [26].

4. Solution Method

Depending on the number of items and periods, lot-sizing MIP models are often very large in practice so that even heavy-duty solvers such as Cplex 7.1 [27], are unable to identify provably-optimal solutions in acceptable computational time. Trying to solve model (1)-(9) with realistic data using MIP solver Cplex 7.1 on a Pentium III 500 MHz with 512 MB of RAM, the default branch-and-cut (B&C) search ran out of memory without converging to optimality. The best solution found (incumbent) was poor compared to the lower bound.

However, it is not generally worthwhile investing a lot of computing time in the search for an exact optimal solution, given that input data is often imprecise in small foundries and in manufacturing generally. A more useful outcome might be a quickly-obtained solution of good quality. Backorder penalties are usually subjective estimates and the order book is changeable, being updated daily, so a theoretically optimal solution to model (1)-(9) will almost surely not be the best in practice. As a result, the exact model can be relaxed to a smaller faster one that includes integer variables only for the planning periods up to the time of the next rolling horizon application of the model, as in [28]. Only the production decisions relating to the immediate periods are actually implemented, after which the horizon of length T is rolled forward and the model applied once more with updated order and inventory information. In this way, updated orders are easily taken into account.

4.1 Rolling Horizon Method

Typical of the small foundries encountered in the course of this research, suppose that each furnace load lasts about 2 hours and that a working day can fit in 10 loads. In this section, we consider the immediate periods as those in day 1, i.e., the first $L = 10$ furnace loadings.

To apply the rolling horizon method, the model (1)-(9) is modified as follows. Let l_t be maximum permitted number of furnace loadings in period t. Redefine y_t^k as the (integer) number of furnace loadings of alloy k in period t, and replace constraint (5) by:

$$\sum_{k=1}^{K} y_t^k \leq l_t \qquad t = 1, ..., T \qquad (5^*)$$

In this way, the time periods t can have flexible lengths depending on the value of l_t ranging from a single furnace loading ($l_t = 1$, i.e., 2 hours) up to one working day ($l_t = 10$). The proposed rolling horizon method sets $l_t = 1$ for $t = 1,\ldots, 10$ (i.e., detailed day 1 scheduling), and $l_{11} = l_{12} = l_{13} = l_{14} = 10$ to aggregate several furnaces loadings in days 2 to 5 (periods $t = 11$ to 14). The resulting model has a horizon of 5 days (sufficient in a small foundry) using only $T = 14$ periods, as shown in Figure 2.

For the first day, the method determines a detailed schedule considering short periods where each loading is planned separately. The next four days are planned considering long periods so that excess demand in these periods can be produced earlier in time if necessary and possible. Thus the z_t^k variables relate to only the first day of scheduling. The decisions concerning the first L periods are implemented before advancing the horizon by L periods and applying the model again. To reduce problem size and computing time, the integer x_{it} variables are relaxed for periods $t = L+1,..., T$. The y_t^k variables for periods $t = L+1,..., T$ could also be relaxed, but are kept integer to improve future capacity evaluation. Thus constraints (6) to (8) can be replaced by constraints (6*), (6**), (7*), (8*) and (8**) below.

Model (1)-(9) is replaced by the following model, denominated **RH**:

Model RH:

Minimize $\sum_{i=1}^{N}\sum_{t=1}^{T}(h_{it}^{-}I_{it}^{-}+h_{it}^{+}I_{it}^{+})+\sum_{k=1}^{K}\sum_{t=1}^{L}(s_k z_t^k)$ (1*)

subject to:

$I_{i,t-1}^{+} - I_{i,t-1}^{-} + x_{it} - I_{it}^{+} + I_{it}^{-} = d_{it}$ $i = 1, ..., N$ $t = 1, ..., T$ (2)

$\sum_{i\in S(k)} \rho_i x_{it} + st_k z_t^k \le Cap\, y_t^k$ $k = 1, ..., K$ $t = 1, ..., L$ (3*)

$\sum_{i\in S(k)} \rho_i x_{it} \le Cap\, y_t^k$ $k = 1, ..., K$ $t = L+1, ..., L$ (3**)

$z_t^k \ge y_t^k - y_{t-1}^k$ $k = 1, ..., K$ $t = 1, ..., L$ (4*)

$\sum_{k=1}^{K} y_t^k \le l_t$ $t = 1, ..., T$ (5*)

$y_t^k \in \{0,1\}$ $k = 1, ..., K$ $t = 1, ..., L$ (6*)

$y_t^k \ge 0$ and integer $k = 1, ..., K$ $t = L+1, ..., T$ (6**)

$z_t^k \ge 0$ $k = 1, ..., K$ $t = 1, ..., L$ (7*)

$x_{it} \ge 0$ and integer $i = 1, ..., N$ $t = 1, ..., L$ (8*)

$x_{it} \ge 0$ $i = 1, ..., N$ $t = L+1, ..., T$ (8**)

I_{it}^{+} and $I_{it}^{-} \ge 0$ $i = 1, ..., N$ $t = 1, ..., T$ (9)

In model **RH**, as in model (1)-(9), the demand for items with due date on the first day is allocated to the last period of the first day, i.e., to period $t = L = 10$. Demand for items with due date on the second day is assigned to period $L+1 = 11$, on the third day to period $L+2 = 12$, and so on.

Model **RH** maintains the similarity to the General Lot-Sizing and Scheduling Problem (GLSP) in [8]. Day 1 can be considered a large time bucket divided into 10 small buckets in each of which only one alloy can be produced, while days 2 to 5 are, temporarily, indivisible large buckets with multiple alloy production.

Model **RH** model is much smaller than the model (1)-(9), but still not small enough to be solved optimally with realistic data using MIP solver Cplex 7.1 within acceptable computing time. However it is possible to solve the model in two phases using the *relax-and-fix* method [24]. This involves the solution of a series of partially relaxed MIPs, each with a small enough number of integer variables to be quickly and optimally solved by Cplex. As the series progresses, each set of integer variables is permanently fixed at their solution values. The procedure is similar to a depth-

first identification of an initial integer solution for a MIP model in a branch and bound search. Its big advantage is speed. The basic relax-and-fix procedure, denominated **RF**, is as follows:

Basic Relax-and-Fix Method (RF):

1. Maintain the first day's y_t^k ($t = 1,...,L$) variables as binary (the 10 furnace loadings that are the most important decisions in our rolling horizon method) and relax all the other integer variables. The resulting model is submitted to branch-&-cut search to try to find an integer optimal solution within a specified computing time limit.
2. The first day's y_t^k ($t = 1,...,L$) variables are then fixed at their binary values from the solution in step 1. The y_t^k ($t = L+1,...,T$) variables and x_{it} ($t = 1,...,L$) variables are specified as integer. This results in a model that can be optimally solved with Cplex in a few seconds. The MIP is easy to solve since, once a binary variable y_t^k is fixed to 1, then by constraints (3*), $x_{it} = 0$ for all $i \notin S(k)$, i.e., there is no production of items that do not use alloy k in period t, thus eliminating many integer variables.

A pure rolling horizon strategy would apply the **RF** method, implement the first day's variables, and then a new rolled horizon T would be considered. Note that a solution to model **RH** is not a solution to model (1)-(9), as only the first day's 10 loads are scheduled, while the other days are planned only approximately. However, the application of model RH 5 times starting consecutively at periods 1, 2, ..., 5, with an always-shortening horizon (T = 5, 4, 3, 2, 1), will produce a feasible overall solution to model (1)-(9), enabling a comparison of results.

4.2 Local Search (LS)

The basic **RF** method is dependent on a solver such as Cplex to solve two MIP problems. The first problem is to solve (and then fix) just the first day's binary variables. The second problem, in step 2, is to try to find an optimal solution for variables y_t^k ($t=L+1,..., T$) and x_{it} ($t=1,..., L$).

To solve the MIP for the first problem, a *local search method* [29-31] is used to find good values for day 1's binary y_t^k (t=1,...,L) variables. Starting, for example, with a random solution and fixing these variables, all the other integer variables are relaxed and the resulting linear programming model is solved. In the next local search iteration, the day 1 binary variables are modified so that a neighbouring solution is identified and the linear program is solved again. Depending on certain criteria, the neighbouring solution may be adopted in which case it becomes the current solution and the local search proceeds to the next iteration. The best solution

encountered as the search progresses is recorded. When the stop criterion of the local search is satisfied, the y_t^k ($t=1,\ldots, L$) variables are fixed at the best solution found.

In order to implement the local search method (Algorithm 1), it is necessary to define a series of parameters. In this paper, the solution representation consists of an array with L=10 positions, each of which represents the alloy used in one of the L furnace loadings in the first day. For example, the solution represented by the array (2, 2, 20, 1, 4, 4, 8, 2, 10, 3) means that the first two furnace loadings are with alloy type 2, the third loading with alloy type 20, and so on.

Three different types of construction heuristic were tested to obtain an initial solution: (1) a randomly greedy heuristic that chooses alloys k that have more associated items $|S(k)|$ with a larger probability; (2) running the MIP Cplex solver for a few minutes to obtain a initial solution; (3) a random allocation heuristic, i.e., each period is randomly assigned an alloy. The search is stopped after 1000 iterations, sufficient to obtain a good solution in acceptable computing time.

A neighbouring solution is selected by changing an alloy in a position to another alloy. The alloy to be changed is chosen either (with 90% probability) randomly, or (with 10% probability) using a greedy heuristic with a proportionally larger probability if the alloy has fewer associated items. Likewise, the new alloy is also chosen either randomly (90%) or using a greedy heuristic (10%) with a proportionally larger probability if it has more associated items. This selection mechanism lightly biases the selection towards more widely-used alloys, and was adopted after initial testing showed its positive impact.

Algorithm 1 (Local Search Procedure)

1. Select a starting solution $\{ y_t^k \mid t = 1,\ldots,L\}$.
2. Relax the x_{it} ($t = 1, \ldots, T$) and y_t^k ($t = L+1,\ldots,T$) integer variables.
3. Record the solution of the resulting linear programme as the current solution.
4. At each of IT iterations, do:
   (a) Generate a neighbouring solution: select a period t ($1 \le t \le L$) and choose its new alloy k, as described above.
   (b) Fix the values of the first day's y_t^k variables and solve the resulting relaxed linear programme.
   (c) Move to the neighbouring solution if it is better than the current one.
5. Fix the values of the y_t^k ($t =1,\ldots,L$) variables in the best solution found in step 4. Restore the x_{it} ($t = 1,\ldots,L$) and the y_t^k ($t = L+1,\ldots,T$) to be integer variables, and solve the resulting small MIP to obtain an optimal integer solution.

Many researchers have proposed ways to improve local search performance, including [29, 32-36]. In this paper, we used two strategies: Diminishing Neighbourhood search method and Simulated Annealing, with the same basic parameters as the local search described above.

4.3 Diminishing Neighbourhood (DN) Search

This method adapts local search, beginning with a large neighbourhood to encourage diversity, and then gradually diminishing its size so as to increasingly intensify the search. Too small a neighbourhood could cause premature convergence and increase the risk of stagnation at a local optimum, while too large neighbourhood one would lead to random meandering and an inefficient search. The search starts with the largest possible neighbourhood, i.e., all L alloy positions in a solution can change, and ends up with a neighbourhood where just one position is changed, i.e., as in the local search in section 4.2. The same stopping criterion of 1000 iterations was applied.

[37] used a similar method for lot-sizing on a drinks canning line. [38] also proposed, in a different manner, the use of varying-sized neighbourhoods.

4.4 Simulated Annealing (SA)

Simulated Annealing is an extension of local search that tries to avoid stagnation at a local optimum by permitting worsening movements whose probability of occurring is a function of a gradually cooling search "temperature" and the amount by which the new solution is worse than the current one.

The best results were obtained with the following parameters after initial computational tests:

- **Initial Temperature T_i**: The formula based on the initial solution by [34] was used:

$$T_i = \frac{\mu \times \text{Value of the Initial Solution}}{-\log(\theta)}$$

  where $\mu = 0.6$ and $\theta = 0.9$ indicate that at the start of the search a solution that is 60% worse than the current one has a 90% probability of acceptance.

- **Temperature Reduction Factor**: This was set at 0.95. Furthermore, a second temperature reduction process was used consisting of smooth decreases each time a worse solution is accepted, in line with the following empirical formula:

$$T_{i+1} = T_i - \left(0.1 \times T_i \times \frac{\Delta ofv}{ofv(S)} \right)$$

  where ofv(S) is the objective function value of the previous solution, and Δofv is the amount by which it worsened. Note that the worse the accepted solution, the greater the

reduction in temperature, thus penalising the acceptance of worse solutions through temperature reductions.

- **Number of iterations to reach equilibrium at a given temperature**: 50 iterations were allowed, but 10 iterations where the solution is accepted, even when worse.

- **Worse Solution Acceptance Function:** The conventional function was used:

$$p(\Delta C) = e^{-\left(\frac{\Delta ofv}{T}\right)}$$

where T is the temperature and Δofv as defined above.

5. Computational Experiments

In this section, we describe first how the test data was generated, then analyse the test results to compare the three neighbourhood search methods and the basic relax-and-fix method, and finally consider computing times.

5.1 Data Generation

Consider following parameter definitions:

α_i the number of days by which item i is already delayed at the beginning of the schedule, i.e., at $t = 0$.

β_i the production priority level for delayed item i.

ρ_i physical weight of item i (previously defined in section 3)

A value of zero for α_i means that the item's due date is day 1. Suppose that an item i is already delayed by $\alpha_i > 0$. Then at the end of periods $t = L, L+1,\ldots,T$ the item's delay will be $\alpha_i+t-L+1$. Its backorder penalty h_{it}^- is calculated as $\rho_i(\alpha_i+t-L+1)\beta_i$ so as to increasingly penalise any further delay in its production. However, if item i is not delayed, ie, $\alpha_i \leq 0$, then it can be produced up to period $L+|\alpha_i|$ without being delayed. Thus, for $t = L+|\alpha_i|$, a positive value $I_{it}^- > 0$ means a day of delay and so h_{it}^- is also calculated by as $\rho_i(\alpha_i+t-L+1)\beta_i$ for this period onwards.

Furthermore, if item i is not delayed at the beginning of the schedule, then the variable I_{it}^+ can be positive (i.e., item i can be produced before its due date) and its inventory penalty h_{it}^+ is defined as proportional to its weight ρ_i. In this case, to force I_{it}^- to be zero (given that there is no delay), the backorder penalty h_{it}^- is set to be a very large number G. Similarly, in the case of a delay, its inventory penalty h_{it}^+ is also set to G to force I_{it}^+ to be zero.

In summary:

if $\alpha_i \geq 0$ [i.e., item i is already delayed at the start, or the due date is day 1] then

for $t=L,...,T$, let $h_{it}^- = \rho_i(\alpha_i+t-L+1)\beta_i$ and $h_{it}^+ = G$;

else [i.e., ($\alpha_i < 0$, meaning that item i will be delayed after period $L+|\alpha_i|$]

for $t=L,...,L+|\alpha_i|-1$, let $h_{it}^- = G$ and $h_{it}^+ = \rho_i$;

for $t= L+|\alpha_i|,...,T$, let $h_{it}^- = \rho_i(\alpha_i+t-L+1)\beta_i$ and $h_{it}^+ = G$.

If an already delayed item will also be in demand within the planning horizon, then it will be considered within the model as two distinct items in order to have two different penalty backorder values. Although this could considerable increase the size of the model, in practice it will do so only a little at most, as it is rare in small market-driven foundries to have the same item demanded on different days within the one-week planning horizon. The two distinct items use the same alloy and have the same physical weight, of course, and so will not logically distort furnace scheduling via expression (3*). The use of a classical model that associates setups with items (rather than alloys) would have resulted in additional setups if such demand-splitting had been applied.

It might be tempting to explicitly prohibit backorders in the model, but as backorders are frequently unavoidable, this would result in infeasible problems and be unrealistically rigid. It is far better to include the possibility of backorders in the model and let the human scheduler manage them, calibrating the production priority parameters β_i. for individual items. Moreover, backorders help the scheduler to evaluate due dates. For example, if d_{iL} =30, then a model solution where I_{iL}^- = 10, $I_{i,L+1}^-$ = 5, $I_{i,L+2}^-$ = 0 flags that the demand for item i will be fully met only after 2 days of delay. In this case, the client could be alerted and, if necessary, the scheduler could increase the value of parameter β_i for that item or renegotiate the item's due-date.

The uniform sampling intervals used to randomly generate the test data were based on real cases encountered in small foundries, as shown in Table 1. Though not all-encompassing, the values are sufficiently typical to be confident that the test results will provide generally applicable conclusions. The furnace capacity base parameter C was generated as follows: first calculate the resources needed to exactly produce total item demand over the planning horizon (in this case 5 days, i.e., 50 furnace loads); then add the total setup time needed if the furnaces were setup just once for each alloy; finally divide the number of furnace loads, i.e., 50. Hence:

$$C = \frac{\sum_{i=1}^{N}\sum_{t=1}^{T} d_{it}\rho_i + \sum_{k=1}^{K} st_k}{50}$$

Thus in Table 1 four different levels of the tightness of Furnace Capacity *Cap* are shown (i) Very Loose capacity: $Cap = C / 0.6$; (ii) Moderately Loose: $Cap = C / 0.8$; (iii) Moderately Tight: $Cap = C / 1.0$; (iv) Very Tight: $Cap = C / 1.2$.

The parameters (N,K) pairs, s_k and *Cap* were varied in a 3-factor experimental design. Each factorial combination was generated 10 times, using a different random seed each time, resulting in a total of $3 \times 2 \times 4 \times 10 = 240$ test problems to obtain the computational results. Each generated test problem was solved using the three neighbourhood search methods (LS, DN and SA) and the basic relax-and-fix method (RF) to obtain a complete 5-day schedule.

5.2 Solution Quality

As commented previously, it was not viable to solve model (1)-(9) to plan all furnace loadings at once over the whole 5-day horizon. Even after running for 10 hours, the Cplex incumbent solution was on average 22% worst than the basic RF solution.

For small problems ($N = 10$, $K = 2$) using the basic RF method, Cplex easily found the optimal solution for all of the MIPs it solved in each of the 5 days, but less than all solutions for medium problems, and much less than all solutions for large problems. Table 2 shows the percentage of test problems where the basic RF method found the MIP optimal solution within 3, 6, and 12 minutes respectively for the small, medium and large problems. The basic RF method in section 4.1 used the MIP incumbent solution founded within these times limits.

Table 3 shows the mean solution variation of the heuristic *LS*, *DN* and *SA* methods compared to the basic RF method, calculated as:

$$\text{Variation} = \frac{(\text{Heuristic Solution} - RF \text{ Solution})}{RF \text{ Solution}} \times 100\%$$

Note that the three heuristic methods generally performed nearly as well as the basic RF method, the best being SA, then DN, then LS. Some simple modifications, such as those in DN, help LS to avoid stagnation at a local optimum, thought not totally. LS converges rapidly to a local optimum while DN and SA take many more iterations to achieve their best solution.

For small and medium problems, the LS method had the worst mean performance (4.36% and 3.38%), followed by DN (1.93% and 2.72%), then SA (1.02% and 1.48%). For large problems, the mean performance of LS improved to 0.90% respectively, and in fact was better than DN (1.31%), though worse than SA (0.56%). This difference in relative performance might be explained by two reasons. Firstly, for problems of all sizes, both LS and SA will follow the same search trajectory, but when either reaches a local optimum, LS has no way to jump out of it and so the search stagnates there, whereas SA can go to a worse solution to get away from the local optimum. The DN search follows a different path than LS and SA, and if it gets stuck at local optimum, this tends to be when its neighbourhood is near its minimum size towards the end of the search. Table 3

suggests that DN search is advantageous for small and medium problems, but not for large ones. Secondly, recall from Table 2 that the larger the problem, the fewer MIPs Cplex 7.1 is able to optimally, thus weakening the basic RF method. This means that the LS, DN and SA methods are not being compared against proven optimal MIPs solutions, but against possibly-suboptimal incumbent solutions obtainable within branch-&-cut search time limits.

Table 2 also shows that tightness of capacity does not much affect the basic RF method, with the possible exception of medium problems. However, Table 3 indicates that, regardless of problem size, there is generally less variation (i.e., better performance) of the LS, DN and SA methods relative to RF when capacity is tight and/or when setup penalties are small.

The computing time spent by the LS, DN and SA methods to solve each MIP was about 1, 3 and 5 minutes respectively for small, medium and large problems. These are viable times for practical problems and faster than the time spent by the basic RF method (3, 6 and 12 minutes respectively). The solution of the final MIP in step 4 (algorithm 1) of the LS, DN and SA methods was limited to a maximum of 5 minutes of computing time. In practice, the MIP was usually solved by Cplex in less than 10 seconds, even for large problems, almost certainly because the binary y_t^k ($t=1,...,L$), variables had been fixed, leaving only the non-zero integer x_{it} values to be optimised.

6 Conclusions

This paper modelled as a mixed integer linear programme (MIP) the production planning and scheduling in small market-driven foundries of the kind found in many countries. However, it is not possible to optimally solve the overall model within many hours of computing time, even using an advanced heavy-duty MIP solver. In order to efficiently solve the model approximately, a rolling horizon approach and associated relax-and-fix (RF) procedure was developed. Four solution methods were proposed using a basic RF approach and three variants of neighbourhood search. The computational results showed that all four solution methods are practicably fast and can be used operationally in small market-driven foundries. With respect to the basic RF methods on the one hand and the three neighbourhood search on the other hand, there is a light variation trade-off between solution quality and running time, especially in the presence of loose capacity and large set up penalties.

References

1. ABIFA - Associação Brasileira de Fundição. *Relatório anual do setor de fundição.* http://www.abifa.org.br, 2003.
2. Billington PJ, McClain JO, Thomas LJ. Mathematical programming approaches to capacity mrp systems: Review formulation and problem reduction. *Management Science.* 1983; **29** (10): 1126-1141.

3. Bahl HC, Ritzman LP, Gupta JND. Determining lot sizes and resource requirements: A review. *Operations Research.* 1987; **35**: 329-345.
4. Maes J, van Wassenhove LN. Multi-item single-level capacitated dynamic lot-sizing heuristics: A general review. *Journal of Operational Research Society.* 1988; **39** (11): 991-1004.
5. Goyal SK, Gunasekaran A. Multi-stage production-inventory systems. *European Journal of Operational Research.* 1990; **46**: 1-20.
6. Potts CN, Van Wassenhove LN. Integrating scheduling with batching and lot-sizing: A review of algorithms and complexity. *Journal of Operational Research Society.* 1992; **43** (5): 395-406.
7. Kuik R Salomom M, Van Wassenhose LN. Batching decisions: Structure and models. *European Journal of Operational Research.* 1994; **75**: 243-263.
8. Drexl A, Kimms. A Lot sizing and scheduling - survey and extensions. *European Journal of Operational Research.* 1997; **99**: 221-235.
9. Potts CN, Kovalyov MY. Scheduling with batching: A review. *European Journal of Operational Research.* 2000; **120**: 228-249.
10. Karimia B, Fatemi Ghomia SMT, Wilson JM. The capacitated lot sizing problem: A review of models and algorithms. *Omega.* 2003; **31**: 365-378.
11. Santos-Meza E, Santos MO, Arenales MN. Lot-sizing problem in an automated foundry. *European Journal of Operational Research.* 2002; **139** (2): 490-500.
12. Araujo SA, Arenales MN. Planejamento e programação da produção numa fundição cativa automatizada de grande porte. *Investigação Operacional.* 2004; in press.
13. Sounderpandian J, Balashanmugam B. Multiproduct multifacility scheduling using the transportation model: A case study. *Production and Inventory Management Journal.* 1991; **32** (4): 69-73.
14. Gravel M, Price WL, Gagné C. Schedulings jobs in an alcan aluminium foundry using a genetic algorithm *International Journal of Production Research.* 2000; **38** (13): 3031-3041.
15. Tang L, Liu J, Rong A, Yang Z. A review of planning and scheduling systems and methods for integrated steel production. *European Journal of Operational Research.* 2001; **133**: 1-20.
16. Petersen CM, Sorensen KL, Vidal RVV. Inter-process syncronization in the steel production. *International Journal of Production Research.* 1992; **30** (6): 1415-1425.
17. Hamada K, Baba T, Sato K, Yufu M. Hybridizing a genetic algorithm with rule-based reasoning for production planning. *IEEE Expert.* 1995; **10**: 60-67.
18. Bowers MR, Kaplan LA, Hooker TL. A two-phase model for planning the production of aluminum ingots. *European Journal of Operational Research.* 1995; **81**: 105-114.

19. Hendry LC, Fok KK, Shek KW. A cutting stock and scheduling problem in the copper industry. *Journal of Operational Research Society.* 1996; **47** (1): 38-47.
20. Lee HS, Murthy SS, Haider SW, Morse DV. Primary production scheduling at steelmaking industries. *IBM Journal of Research and Development.* 1996; **40** (2): 231-252.
21. Lopes L, Carter MW, Gendreau M. The hot strip mill production scheduling problem: A tabu search approach. *European Journal of Operational Research.* 1998; **106**: 317-335.
22. Tang L, Liu J, Rong A, Yang Z. A mathematical programming model for scheduling steelmaking-continuous casting production. *European Journal of Operational Research,* 2000; **120**: 423-435.
23. Tang L, Liu J, Rong A, Yang Z. A multiple traveling salesman problem model for hot rolling scheduling in Shanghai Baoshan iron & steel complex. *European Journal of Operational Research.* 2000; **124**: 267-282.
24. Wolsey LA. *Integer programming.* New York: Wiley, 1988.
25. Miettinen KM. *Nonlinear multiobjective optimization.* Kluwer, 1998.
26. Araujo SA, Arenales MN, Clark AR. Joint rolling-horizon scheduling of materials processing and lot-sizing with sequence-dependent setups. *Congreso Latino-Americano de Investigacion Operacional (CLAIO).* Havana, 2004.
27. ILOG Cplex 7.1 User's manual. ILOG SA BP 85 9 Rue de Verdun 94253. Gentilly, France. http://www.ilog.com, 2001.
28. Clark AR, Clark SJ. Rolling-horizon lot-sizing when setup times are sequence-dependent. *International Journal of Production Research.* 2000; **38** (10): 2287-2307.
29. Reeves CR. *Modern heuristic techniques for combinatorial problems.* Blackwell, 1993.
30. Pirlot M. General Local Search Methods. *European Journal of Operational Research.* 1996; **92**: 493-511.
31. Aarts EHL, Lenstra JK. *Local search in combinatorial optimization.* Chichester: Wiley, 1997.
32. Goldberg D E. *Genetic algorithms in search optimization and machine learning.* Massachusetts: Addison Wesley, 1989.
33. Laguna M. Tabu search tutorial. *II Escuela de Verano Latino-Americana de Investigation Operativa.* Rio de Janeiro, 1995.
34. Diaz A, Glover F, Ghaziri HM, González JL, Laguna M, Moscato P, Tseng FT. *Optimización heurística y redes neuronales.* Spain: Editorial Paraninfo, 1996.
35. Gen M, Cheng R. *Genetic algorithms & engineering Design.* Wiley, 1997.
36. Glover F, Laguna M. *Tabu search.* Norwell, Massachusetts: Kluwer, 1997.
37. Clark AR. Hybrid heuristics for planning lot sizes and setups. *Computers and Industrial Engineering.* 2003; **45** (4): 545-562.

38. Hansen P, Mladenović N. Variable neighbourhood search: principle and applications. *European Journal of Operational Research.* 2001; **130**: 449-467.

Figure and table captions

Figure 1. Main activities in the foundry

Figure 2: Number of furnace loads planned using a rolling horizon strategy.

Table 1: Parameters used for generation of uniformly-distributed test data.

Table 2: Percentage of test problems in which Cplex found an optimal solution for all the *RF* MIPs

Table 3: Mean solution variation (%) of the LS/DN/SA heuristics compared to the basic RF method.

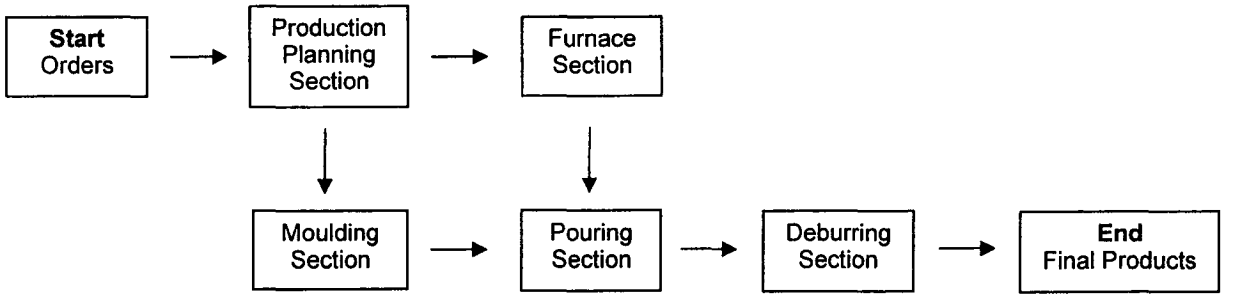


Figure 1. Main activities in the foundry

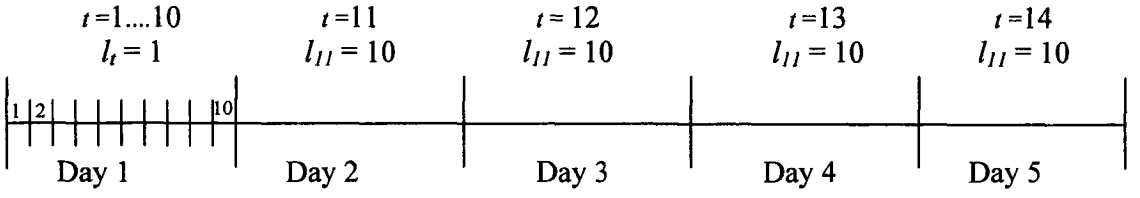


Figure 2: Number of furnace loads planned using a rolling horizon strategy.

Parameters	**Values**
Number of Items and Alloys: (N, K) pairs	Small Problem: (10, 2) Medium Problem: (50, 10) Large Problem: (100, 20)
Number of Days:	5
Demand: d_{it}	[10, 60]
Days of Delay: α_i	[-10, 10]
Priority Level: β_i	1
Physical Weight of Item: ρ_i	[1, 30]
Setup Time of Alloy: st_k	[5, 10]
Setup Penalty of Alloy: s_k	Low: $5 \times st_k$ High: $50 \times st_k$
Tightness of Furnace Capacity: *Cap*	C / 0.6, C / 0.8, C / 1.0, C / 1.2

Table 1: Parameters used for generation of uniformly-distributed test data.

Problem Size	Small: (N, K) = (10, 2)				Medium: (N, K) = (50, 10)				Large: (N, K) = (10, 20)			
Setup Penalty s_k	C/0.6	C/0.8	C/1.0	C/1.2	C/0.6	C/0.8	C/1.0	C/1.2	C/0.6	C/0.8	C/1.0	C/1.2
$5 \times st_k$	100	100	100	100	62	66	68	80	44	42	36	56
$50 \times st_k$	100	100	100	100	68	68	86	90	44	42	32	44

Table 2: Percentage of test problems in which Cplex found an optimal solution for all the RF MIPs

Method:		LS			DN			SA		
Setup Penalty Factor s_k		$5 \times st_k$	$50 \times st_k$	Mean	$5 \times st_k$	$50 \times st_k$	Mean	$5 \times st_k$	$50 \times st_k$	Mean
Prob. Size	Capacity									
Small: (N, K) = (10, 2)	C/0,6	4.67	15.36	10.02	3.50	7.72	5.61	0.86	4.56	2.71
	C/0,8	2.85	5.56	4.21	0.70	2.37	1.53	0.52	1.68	1.10
	C/1,0	1.35	1.86	1.61	0.00	0.63	0.32	0.21	0.32	0.26
	C/1,2	1.46	1.74	1.60	0.17	0.36	0.26	0.00	0.00	0.00
	Mean	2.58	6.13	**4.36**	1.09	2.77	**1.93**	0.40	1.74	**1.02**
Medium: (N, K) = (50, 10)	C/0,6	1.88	9.64	5.76	1.91	7.69	4.80	1.27	2.75	2.01
	C/0,8	1.25	5.21	3.23	1.11	4.50	2.80	0.62	2.41	1.51
	C/1,0	0.97	4.23	2.60	0.74	2.97	1.86	0.58	0.77	0.67
	C/1,2	1.13	2.73	1.93	0.89	1.95	1.42	0.88	2.58	1.73
	Mean	1.31	2.54	**3.38**	1.16	4.28	**2.72**	0.84	2.13	**1.48**
Large: (N, K) = (100, 20)	C/0,6	0.18	2.93	1.56	1.68	2.93	2.30	-0.01	1.41	0.70
	C/0,8	0.19	2.05	1.12	0.33	2.27	1.30	0.19	1.08	0.63
	C/1,0	-0.15	0.96	0.40	0.05	1.28	0.67	-0.15	0.96	0.40
	C/1,2	0.02	1.01	0.52	0.38	1.60	0.99	0.00	1.01	0.51
	Mean	0.06	1.74	**0.90**	0.61	2.02	**1.31**	0.01	1.11	**0.56**
Overall Mean (%)		1.32	4.44	**2.88**	0.95	3.02	**1.99**	0.41	1.63	**1.02**

Table 3: Mean solution variation (%) of the LS/DN/SA heuristics compared to the basic RF method.

NOTAS DO ICMC

SÉRIE COMPUTAÇÃO

067/2003 ARAUJO, S.A.; ARENALES, M.N. – Dimensionamento de lotes e programação do forno numa fundição automatizada de porte médio.

066/2002 VALERIO NETTO, A.; OLIVEIRA, M.C.F. – Industrial application trends and market perspectives for virtual reality and visual simulation.

065/2002 VALERIO NETTO, A.; OLIVEIRA, M.C.F. – Desenvolvimento de um protótipo de um torno CNC utilizando realidade virtual.

064/2002 MARQUES, F.P.; ARENALES, M.N. - O problema da mochila compartimentada e aplicações.

063/2001 TOMÉ, M F; MANGIAVACHI, N; CUMINATO, J A; CASTELO, A – A marker-and-cell technique for simulating unsteady viscoelastic free surface flows.

062/2001 VARGAS, A J C; NONATO, L G. – β-conexão: uma família de objetos tridimensionais reconstruídos a partir de seções planares.

061/2001 OLIVEIRA JR., O N; MARTINS, R T; RINO, L H M ; NUNES, M G V – O uso de interlínguas para comunicação via internet: o projeto UNL/Brasil.

060/2001 SILVA, E Q ; MOREIRA, D A – Use of software agents to the management of distance education courses over the internet.

059/2001 OLIVEIRA, M C F; LEVKOWITZ, H – Visual data exploration and mining: a survey.

058/2001 SOARES, M D; FORTES, R P M; MOREIRA, D A – Version–web : a tool for helping web pages version control.