Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 1, No. 3, pp. 447–464, September 2005

A HEURISTIC ALGORITHM FOR SCHEDULING THE STEELMAKING CONTINUOUS CASTING PROCESS

Adil Bellabdaoui, Antonio Fiordaliso and Jacques Teghem

Dedicated to Professor Toshihide Ibaraki in honor of his 65th birthday.

Abstract: This paper presents an algorithm for flow management in the context of steelmaking continuous casting. The system under study is based on a real world industrial application and covers the elaboration of crude steel from the oxygen converters to the refining and adding of chemicals in refining stands, to moulding and solidification in continuous casting. The scheduling is characterized by several constraints: job grouping, technological interdependence, no dead time inside the same job group and dynamic job processing time. The aim is to maximize productivity and to reduce the sojourn time of the product in the system. With this aim in view, we propose a heuristic based on the elimination of machines conflicts. The algorithm has been developed with the Matlab software. Some numerical results are presented and discussed.

Key words: *scheduling, heuristic, steelmaking*

Mathematics Subject Classification: *90B35, 90B90, 68W25*

1 Introduction

A central problem in industry is to implement efficient algorithms to improve production planning and scheduling for given resources over a short-term or long-term horizon. The scheduling aspect has attracted the Operations Research community's interest [1]. Particular attention has been paid to steel plants, recognized as one of the most difficult industrial scheduling problems because of their complicated manufacturing environment with various batches and continuous modes ([5], [6]).

A specific type of steel plants, namely the steelmaking continuous casting (SCC), is of prime interest. The literature contains various production planning and scheduling techniques developed for SCC. For example, Nuamo et al. [7] treat the problem on three levels: (1) sub-scheduling, which fulfils the scheduling of individual charge sets, (2) rough scheduling, which merges sub-schedules, and (3) optimal scheduling based on an inference engine, which eliminates machine conflicts. Chokshi et al. [3] demonstrate that the problem of achieving minimal stoppages at the caster in steelmaking can be viewed as a distributed, coordinated scheduling problem. De Schutter [4] develops hybrid techniques to design (sub-) optimal timing and sequencing schemes for a continuous steel foundry. His method makes use of linear programming, genetic algorithms, tabu search and heuristics. Tang et al. [8] formulate the SCC problem using a nonlinear programming model to fix machine conflicts. This model exploits the just-in-time (JIT) idea and introduces it into a linear programming model. Two major limitations of the study [8] are:

- The assignment of the charges to the converters is supposed to be known. An a priori "rough scheduling" has been done giving this assignment, which is thus data in the model [8]. The main extension in our model consists in integrating this assignment into the model.
- Moreover [8] does not take into account the possibility to slow down the charges of a sequence.

In [2], we present a generalization of this mathematical programming model [8], taking these two aspects into account.

In this paper, we suggest a heuristic algorithm which is able to solve this complex scheduling problem in steelmaking continuous casting production. The plan considered is based on the Belgian Arcelor Group SCC production system. The problem under study is described in the next section. In Section 3, the heuristic algorithm is presented in three phases. Finally, in Section 4 the method is analyzed with the Matlab software and some numerical results are discussed. Conclusions are drawn and future work is suggested in Section 5.

2 Description of the Problem

The layout of the SCC (steelmaking continuous casting) plant considered in this study is shown in Figure 1. The considered steelmaking process occurs in two oxygen converters (CV), two refining stands (RS) and two continuous casting units (CC). The steelmaking operation starts with a CV in which the crude steel is obtained after burning the unwanted elements (carbon and residues). The steel is then poured into a ladle (charge) of approximately 210 tons. The next stage consists in refining and adjusting the chemical composition of the steel with possible corrections of temperature in the RS. Finally, the liquid steel is poured continuously into a bottomless mould (CC). The moulded metal descends, guided by a set of rollers and continues to cool. For a more detailed description of the process, see http://www.arcelor.com/.

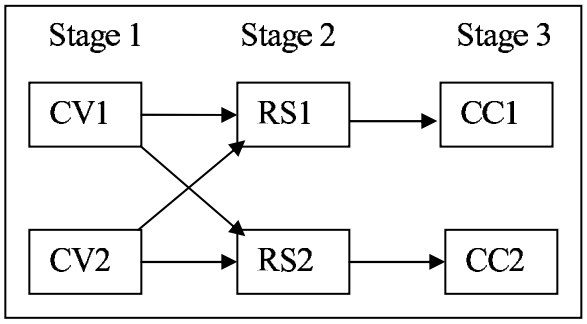

Figure 1: A schematic representation of the layout

The order book consists of two ordered sequences of steel containers (charges), each sequence being dedicated to a particular (RS, CC) pair. These ordered sequences are defined by the logistic department. Let n_1 and n_2 denote the number of charges in the sequence devoted to CC1 and CC2 respectively. The processing time in the CV is the same for each charge and each CV. The processing time in the RS is the same for each charge but depends on the RS. Each charge has a different processing time in the CC. Moreover, this variable processing time has a lower bound. We will use the following notations:

- $p^{(1)}$ the processing time of each charge in the CV,
- $p^{(2)}$ the processing time of each charge in the RS (with different values for RS1 and RS2),
- $p_i^{(3)}$ the variable processing time of charge i in the CC. The minimal value of $p_i^{(3)}$ is denoted as $\underline{p_i^{(3)}}$; this value is a known function of the weight, the width and the maximal speed of the flow of charge i into the CC.

The other data describing the problem are:

- the first date r_1 (resp. r_2) on which the CV1 (resp. CV2) is available,
- the first date t_1 (resp. t_2) on which the CC1 (resp. CC2) is available,
- the transfer times of a charge from stage 1 (CV) to stage 2 (RS) and from stage 2 to stage 3 (CC). They are denoted as t_{12} and t_{23} respectively.

In the construction of a schedule several constraints have to be considered.

- There is no overlap between two charges processed by the same machine.
- The continuity in the treatment of the charges in a single sequence at the CC stage. This means that idle time is prohibited between two successive charges. So it is sometimes necessary to slow down the flow in a CC for a subset of charges in order to increase the processing time $p_i^{(3)}$ of some charges i. As far as possible, it is better to distribute such slowdown homogeneously over several charges than to strongly decelerate a single charge. Indeed it is not easy to successively decelerate and accelerate the flow in the CC. Moreover, it is not recommended to do so because this could alter the quality of the product.
- Each charge has a limited sojourn time (the time between its leaving the RS and its starting in the CC (see Figure 2)). This constraint is due to the necessity of keeping a sufficient temperature before the charge is processed in the CC. This upper bound on the sojourn time is denoted as S.

The objective of the manager is to treat the two sequences as rapidly as possible, that is, to minimize the completion time of both sequences. This objective can be formulated mathematically as $\min \sum_{k=1}^{2}(s_k + \sum_{i=1}^{n_k} p_i^{(3)})$, where s_k $(\geq t_k)$ is the starting time of the sequence in CCk for $k = 1, 2$. It is equivalent to reducing, as far as the constraints allow it, the delay $s_k - t_k$ of the sequence k ($k = 1$ or 2 for sequences for CC1 and CC2 respectively) and the different slowdowns $p_i^{(3)} - \underline{p_i^{(3)}}$ of the charges $i = 1, \ldots, n_k$.

3 The Heuristic Approach

The heuristic algorithm consists of three phases. The first one is a simple initialization of the two sequences at stages 2 and 3, taking the availability dates of the two CC and the transfer time t_{23} into account. The sequences are scheduled at the earliest dates using the minimal processing times of the charges. Such schedule may eventually generate some overlapping at stage 2. The second phase will also treat each sequence separately at stages 2 and 3. The schedule will be modified in order to eliminate the possible conflicts at stage 2 and to respect the maximal sojourn constraint. During this phase, first some advances will be

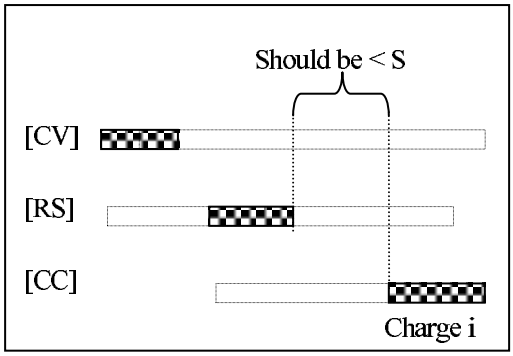

Figure 2: Limitation of the sojourn time of the charges

introduced for some charges at stage 2. If these advances are too large with respect to the maximal sojourn time constraint, it will be necessary to slow down some charges, i.e. to increase their processing times at stage 3. The third phase concerns the assignment of all the charges of both sequences to the two CV at stage 1. The charges are considered one by one, in chronological order from their starting time at stage 2. If the availability of the CV does not permit to assign a charge:

- either, if possible, all the sequences of this charge will be delayed,
- or, a new slowdown will be introduced for some charges belonging to this sequence.

In both cases, because the schedule at stages 2 and 3 has been modified, phase 3 must be repeated from the beginning. The heuristics stops when all the charges can be assigned to the CV.

The three phases of the heuristics will now be described in detail.

First phase:
It consists in an initialization of the schedule at stages 2 and 3, i.e. the RS and CC stages. This initialization is done independently for the two sequences since they use different machines. Each sequence is first initialized on the CC, starting at the availability date of the CC. The processing time of each charge is set to its minimal value. Let $x_i^{(3)}$ be the starting time of charge i at stage 3; we have iteratively:

$$x_1^{(3)} = t_k; \ (k = 1 \text{ or } 2, \text{ depending on the processed sequence}) \tag{1}$$

$$x_{i+1}^{(3)} = x_i^{(3)} + p_i^{(3)}; \ i = 1, \ldots, n_k - 1; \ k = 1, 2 \tag{2}$$

$$\text{with } p_i^{(3)} = \underline{p_i^{(3)}}. \tag{3}$$

Then, the starting time $x_i^{(2)}$ of charge i is computed at stage 2:

$$x_i^{(2)} = x_i^{(3)} - t_{23} - p^{(2)}, \ i = 1, \ldots, n_k; \ k = 1, 2. \tag{4}$$

Second phase:
The aim of this phase is twofold:

(a) eliminate the possible overlapping generated by the first phase at stage RS (see Figure 3)

(b) respect the maximal sojourn time S of the charges (see Figure 2).

This initialization phase is made separately for each sequence. Let us now explain how the heuristics fulfills objectives (a) and (b).

(a) To prevent possible overlapping, a so-called "necessary advance" $A_n(i)$ relative to charge i at stage RS is determined iteratively from the last charge of the sequence. These necessary advances are computed as follows: (see Figure 3)
$A_n(n_k) = 0$; ($k = 1$ or 2, depending on the processed sequence)

$$A_n(i) = \max(0, A_n(i+1) + x_i^{(3)} - x_{i+1}^{(3)} + p^{(2)});\ i = n_k - 1, \ldots, 1;\ k = 1, 2. \tag{5}$$

The starting times $x_i^{(2)}$ are updated as follows:

$$x_i^{(2)} \longleftarrow x_i^{(2)} - A_n(i);\ i = 1, \ldots, n_k;\ k = 1, 2. \tag{6}$$

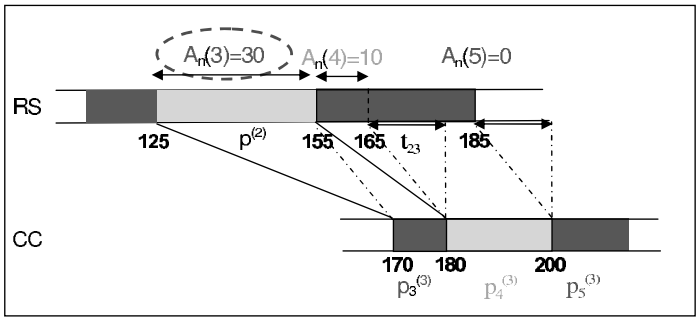

Figure 3: An example showing the use of necessary advances to avoid overlapping at stage RS (for instance: $n_1 = 5; p^{(2)} = 30; \underline{p_4^{(3)}} = 20; \underline{p_3^{(3)}} = 10; t_{23} = 15; S = 35; A^{max} = 20$)

(b) Despite modifications (6), the maximal sojourn constraint may not be satisfied for some charges. Indeed, this constraint implies a maximal necessary advance $A^{max} = S - t_{23}$. So iteratively, from $i = n_k - 1$ (with $k = 1$, 2) to $i = 1$, $A_n(i)$ is compared with A^{max}. If $A_n(i)$ is strictly greater that A^{max}, a slowdown on the CC is necessary. For a reason previously described in Section 2, such a slowdown must be distributed amongst charges $i, \ldots, n_k - 1$.

The charges to slow down are those that are responsible for the large value of $A_n(i)$. The charges to consider for a slowdown form a subset $I = \{l \mid i \leq l \leq L - 1\}$ where L is the smallest index such that $A_n(L) = 0$. This subset can also be written as: $I = \{r \mid A_n(r) \geq A_n(l)$ for all $l \in \{r + 1, \ldots, L\}\}$ (see Figure 4).

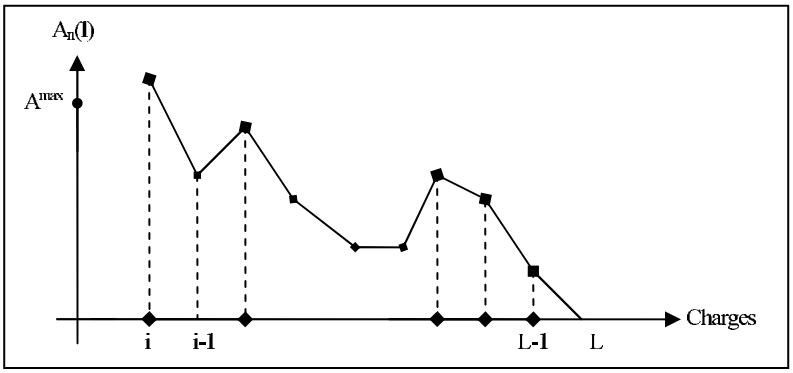

Figure 4: The charges to decrease in phase 2 when $A_n(i) > A^{max}$ (the symbol ♦ indicates an index belonging to I)

The slowdown Δ_r relative to charge $r \in I$ will be

$$\Delta_r = \frac{(A_n(i) - A^{max}) * (A_n(r) - A_n(s))}{A_n(i)},$$

where s represents the next charge after r in I or $s = L$ if r is the last charge in I. So, the processing times $p_r^{(3)}$ of charges $r \in I$ become:

$$p_r^{(3)} \longleftarrow p_r^{(3)} + \Delta_r; \ r \in I. \tag{7}$$

The starting times $x_i^{(3)}$, the necessary advances $A_n(i)$ and the starting times $x_i^{(2)}$ are updated by formulae (4), (5) and (6) respectively. The necessary advances can only decrease. In particular, the new value of $A_n(i)$ will be A^{max} because $\sum_{r \in I} \Delta_r = A_n(i) - A^{max}$ and by relation (5), $A_n(i) \longleftarrow A_n(i) - \sum_{r \in I} \Delta_r$, (see Figure 5).

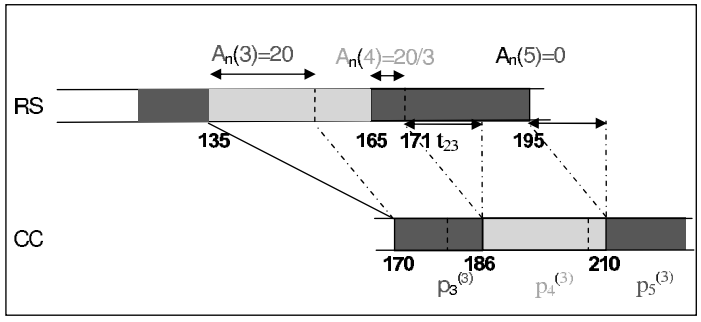

Figure 5: The example of Figure 3 after slowing down

This slowing down process is repeated until all the necessary advances are less than or equal to the maximal advance. At the end of this phase, a provisional schedule of the two sequences at stages 2 and 3 has been completed.

Third phase:
The aim of this phase is to schedule the charges on the CV (stage 1). Charges $(n_1 + n_2)$ are successively considered by increasing the value of $x_j^{(2)}$, $j = 1, \ldots, n_1 + n_2$. First, a provisional starting time $x_j^{(1)}$ is determined for the considered charge j. $x_j^{(1)}$ corresponds to the first availability date of one of the CV. At the beginning, these initial dates are r_1 and r_2. When some charges have already been assigned to the CV, these availability dates are obtained by taking the processing time $p^{(1)}$ of each charge at stage 1 into account.

The starting time $x_j^{(1)}$ implies a so-called "possible advance" $A_p(j)$ of charge j. This possible advance is determined by:

$$A_p(j) = (x_j^{(3)} - t_{23} - p^{(2)} - t_{12}) - (x_j^{(1)} + p^{(1)});\ j = 1, \ldots, n_k;\ k = 1, 2. \tag{8}$$

Let us now discuss the possible cases that may arise.

(α) In the case where $A_p(j) \geq A_n(j)$ (see Figure 6), the starting time $x_j^{(1)}$ can be kept at its value. A waiting time equal to $A_p(j) - A_n(j)$ is introduced before stage RS in the schedule. Next, charge $j+1$ is treated.

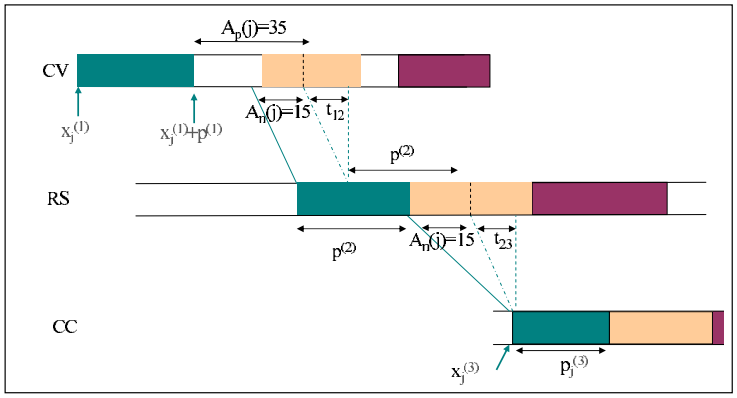

Figure 6: Case if $A_p(j) \geq A_n(j)$

(β) In the case where $A_n(j) > A_p(j)$, it is necessary to increase $A_p(j)$ to be able to use the starting time $x_j^{(1)}$. Two possible cases are considered, depending on whether charge j belongs to a currently processed sequence or not (we mean by a currently processed sequence one which has already started in CC on date $x_j^{(1)}$).

(β-1) If charge j belongs to a non currently processed sequence (the sequence of this charge has not started yet in CC on date $x_j^{(1)}$), the whole sequence will be delayed. The value of this delay is $A_n(j) - A_p(j)$. So for all the charges l of this sequence k (in particular for charge j), the starting times $x_l^{(3)}$ become

$$x_l^{(3)} \longleftarrow x_l^{(3)} + A_n(j) - A_p(j);\ l = 1, \ldots, n_k. \tag{9}$$

This uniform delay does not modify $A_n(l); l = 1, \ldots, n_k$ (see formula (5)) but $A_p(j)$ now becomes equal to $A_n(j)$ (see formula (8)), (see Figure 7). The information related to this

sequence, i.e. $x_l^{(2)}, x_l^{(3)} (l = 1, \ldots, n_k)$, is updated and we go back to the beginning of phase 3, i.e. with $j = 1$.

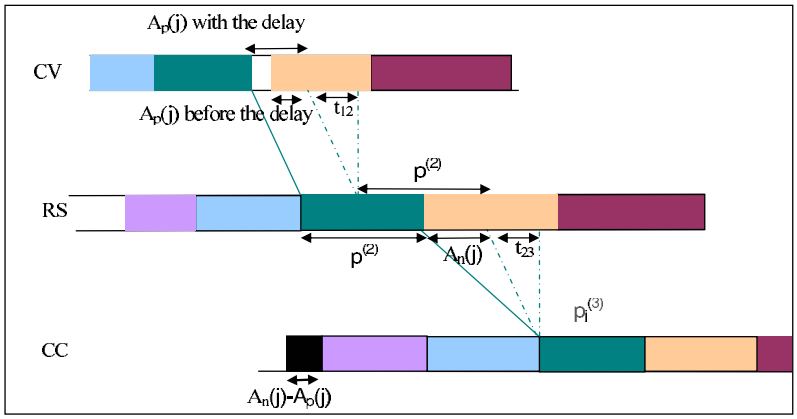

Figure 7: Case where $A_n(j) > A_p(j)$ and charge j belongs to a non currently processed sequence

(β-2) If charge j belongs to a currently processed sequence, the treatment performed at point (β-1) is no longer possible. So, a new slowdown will be introduced in sequence k for charge j. All the charges $l \in \{1, \ldots, j-1\}$ preceding charge j in this sequence k will be slowed down. The value of the delay is:

$$\Delta_l = \frac{(A_n(j) - A_p(j)) * p_l^{(3)}}{\sum_{l=1}^{j-1} p_l^{(3)}} \tag{10}$$

and thus

$$p_l^{(3)} \longleftarrow p_l^{(3)} + \Delta_l; \ l = 1, \ldots, j-1. \tag{11}$$

With such a slowdown, all the possible advances of the charges belonging to sequence k will increase (see formula (8)); in particular, the value of the increase on the possible advance $A_p(j)$ will be $\sum_{l=1}^{j-1} \Delta_l$, i.e. $A_n(j) - A_p(j)$, and thus $A_p(j)$ becomes equal to $A_n(j)$ (see Figure 8).

The necessary advances $A_n(l)$ with $l \geq j$ have not changed. The necessary advances $A_n(l)$ for $l < j$ will decrease or remain constant if it is equal to zero. All the information related to this sequence, i.e. $x_l^{(3)}$, $x_l^{(2)}$, $A_n(l)$ and $A_p(l)$, $l = 1, \ldots, n_k$, is updated and we go back to the beginning of phase 3, i.e. with $j = 1$. The heuristics stops when all the charges have been treated and assigned to the CV.

Note: In practice there can also exist an upper bound on the processing time $p_i^{(3)}$ due to a maximal decrease of the flow speed on the CC. If this upper bound is taken into account, the necessary slowdown of some charges (in phases 2 and 3 of the heuristics) can be impossible. At this time, the heuristics stops due to some contradictions between the constraints. For the presentation of the heuristics described in this paper, such a possibility of infeasibility has been avoided.

The flow chart for the heuristic method is given in Figure 9.

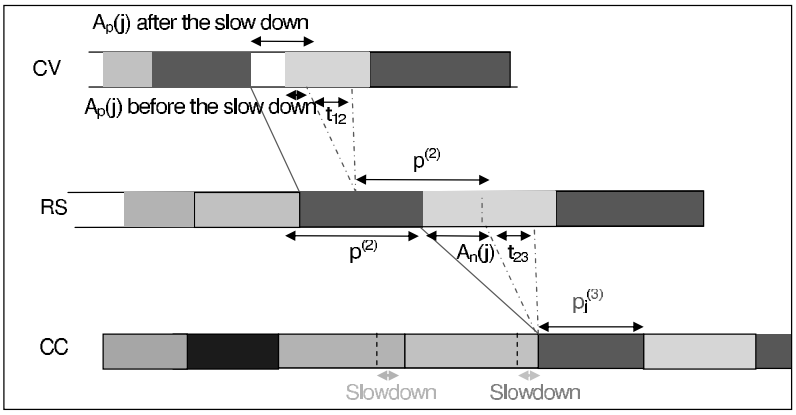

Figure 8: Case where $A_n(j) > A_p(j)$ and charge j belongs to a currently processed sequence

Table 1: Initial situation

r_1	r_2	t_1	t_2
0	24	117	90

4 Experiments

This section presents the results of some experiments aiming at validating the heuristics. The algorithm is implemented in Matlab and is evaluated with some test problems and one real data set. Section 4.1 presents a didactic example as an illustration; an experiment with a real data set is treated in Section 4.2 and compared with the solution provided by the company (ARCELOR Group). Section 4.3 presents some other numerical results.

4.1 Illustration

Table 1 describes the initial state of the system, i.e. the availability of machines CV and CC. Machine CC1 is in use at instant 0, i.e. sequence 1 is currently being processed at the initial instant; so the charges for this sequence are those not yet scheduled at instant 0. Table 2 presents the data of each sequence devoted to CC1 and to CC2. Table 3 gives the processing time of machines CV and RS, the transfer times between the stages, and the upper bound of sojourn time for these sequences.

The results of the heuristics are presented in Tables 4 and 5. The GANTT chart is shown in Figure 10 (to distinguish the charges of the two sequences, those of the second sequence

Table 2: Data of sequences devoted to CC1 (line 1) and to CC2 (line 2)

i	1	2	3	4	5	6	7	8	9
$p_i^{(3)}$ for $k=1$	13.02	19.06	19.06	20.97	19.41	21.35	18.57	20.46	23.53
$p_i^{(3)}$ for $k=2$	35.66	49.91	49.91	49.92	49.91				

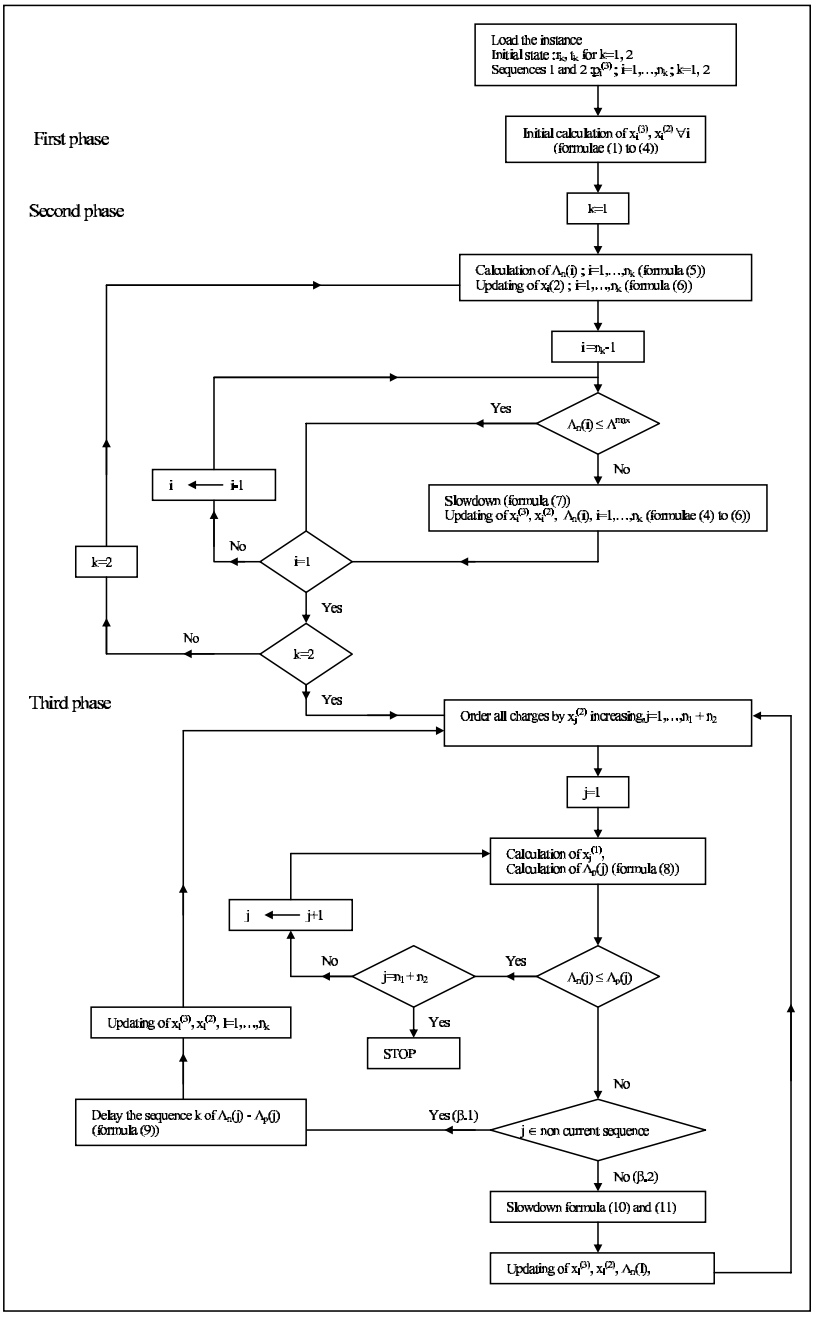

Figure 9: Flow chart for the heuristics

Table 3: Other system parameters

Processing times			Transfer times		Maximum Sojourn times
$p^{(1)}$	$p^{(2)}$ for $k=1$	$p^{(2)}$ for $k=2$	t_{12}	t_{23}	S
44	23	32	15	15	35

Table 4: Results of sequence to CC1

i	$x_i^{(1)}$	$x_i^{(2)}$	$x_i^{(3)}$	$p_i^{(3)}$
1	0	79	117	24.34
2	24	103.34	141.34	32.45
3	44	135.79	173.79	32.45
4	68	168.24	206.24	32.96
5	112	201.20	239.20	29.70
6	156	230.9	268.9	25.88
7	176	256.78	294.78	24.51
8	220	280	319.29	21.71
9	244	303	341	23.53

are denoted by a, b, c, d, e in this figure).

Table 6 presents some additional information. For each sequence, this table reports:

- the completion time: $(x_{n_1}^{(3)} + p_{n_1}^{(3)})$ and $(x_{n_2}^{(3)} + p_{n_2}^{(3)})$ respectively,
- the total slowdown of the charges computed at the CC: $\sum_{i=1}^{n_1}(p_i^{(3)} - \underline{p_i^{(3)}})$ and $\sum_{i=1}^{n_2}(p_i^{(3)} - \underline{p_i^{(3)}})$ respectively,
- the possible delay in starting the sequence computed at the CC: $(x_1^{(3)} - t_1)$ and $(x_1^{(3)} - t_2)$ respectively.

Note: As sequence 1 is currently processed at instant 0, we have necessarily $x_1^{(3)} = t_1$, i.e., the delay for this sequence is equal to zero.

4.2 Validation on an Industrial Case

The following case study is based on a real world industrial application. The process considered is the steelmaking continuous casting of an industrial company, namely the ARCELOR

Table 5: Results of sequence to CC2

i	$x_i^{(1)}$	$x_i^{(2)}$	$x_i^{(3)}$	$p_i^{(3)}$
a	88	187.51	234.51	35.66
b	132	223.17	270.17	49.91
c	200	273.08	320.08	49.91
d	264	323	370	49.92
e	288	372.92	419.92	49.91

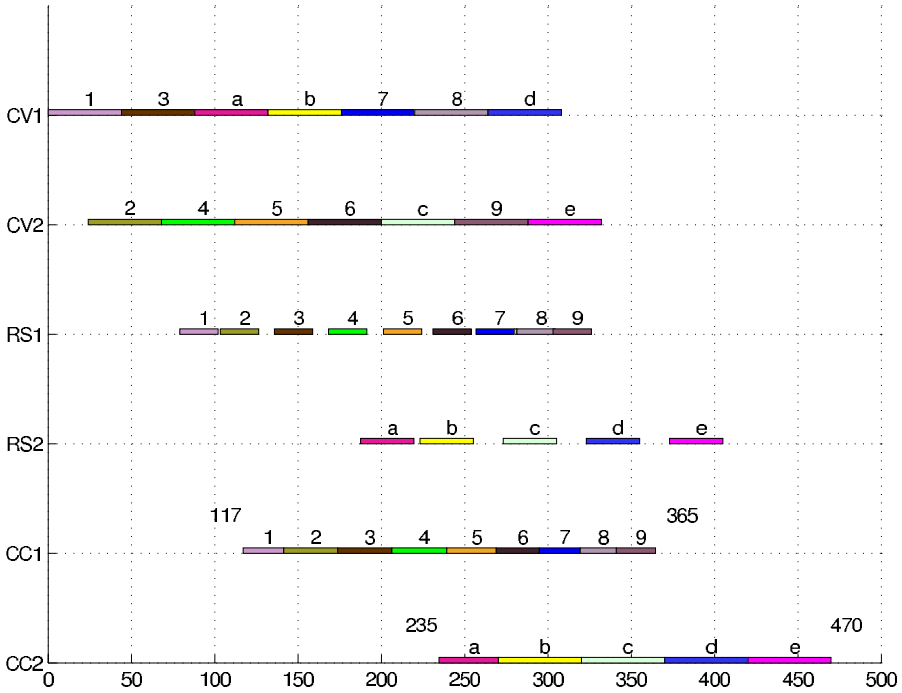

Figure 10: Gantt chart associated with the illustrative example

Table 6: Results obtained with our heuristics (example of Section 4.1)

Optimum	Running time for CC1			Running time for CC2		
	Completion time	Slowing down	Delay	Completion time	Slowing down	Delay
834.33	364.53	72.10	0	469.83	0	144.51

Table 7: Initial situation (validation example)

r_1	r_2	t_1	t_2
4	24	116	179

Table 8: Data of sequence devoted to CC1 (line 1) and to CC2 (line 2) (validation example)

i	1	2	3	4	5
$\underline{p_i^{(3)}}$ for $k=1$	23.94	33.52	33.52	33.52	35.16
$\underline{p_i^{(3)}}$ for $k=2$	37.96	53.14			

Group located in Belgium (Liege). The data used in this section have been provided by the process engineers of this company. The added value of this example is that it allows to compare our heuristic solution with the solution applied by the company on this data set (this solution is obtained in a very pragmatic way, based only on practical experience and an expert's report). The context is the following: there are 5 charges in a sequence to CC1 and 2 charges in a sequence to CC2. Both machines CC1 and CC2 are in use at instant 0. Tables 7–9 present data of the system. The schedule proposed by the company is presented in Table 10. The results of our heuristics are reported in Tables 11 and 12.

The experimental results reported in Table 13 show that our heuristics gives interesting results compared with the solution provided by the company. Indeed, compared with the solution provided by the company, the solution obtained with the heuristics introduces less slowing down for both sequences. The resulting completion times of the two sequences are thus improved by 6 and 11 units respectively.

4.3 Experimental Results

Real data sets are used in several experiments. For each set, the results for each sequence are presented with three indicators: completion time (or process), slowdown and delay (Table 14). In this table, the first two columns respectively indicate the number of charges in both sequences and which sequences are currently being processed at instant zero.

In one instance of the five real data sets (instance 33*14), the availability dates of the CV and the CC are given in Table 15; the other system parameters are the same as those described in Table 9.

The results of our heuristics are presented in Tables 16 and 17 in more detail. The GANTT chart is shown in Figure 11 (to distinguish the charges of the two sequences, those of the second sequence are denoted by a, b,..., n on this figure).

These results prove the ability to treat data of large dimension ($n_1 = 40, n_2 = 16$ are exceptional large sequences, relatively seldom in practice). Our algorithm provides the

Table 9: Other system parameters (validation example)

Processing times			Transfer times		Maximum Sojourn times
$p^{(1)}$	$p^{(2)}$ for $k=1$	$p^{(2)}$ for $k=2$	t_{12}	t_{23}	S
44	22	32	15	15	35

Table 10: Schedule proposed by the company (rounded off to the closest integer)

Seq	i	$x_i^{(1)}$	$x_i^{(2)}$	$x_i^{(3)}$	$p_i^{(3)}$
1	1	4	79	116	29
1	2	24	108	145	33
2	1	48	132	179	49
1	3	68	141	178	34
1	4	92	171	212	35
2	2	112	181	228	53
1	5	136	210	247	35

Table 11: Results obtained with our heuristics (sequence devoted to CC1)

i	$x_i^{(1)}$	$x_i^{(2)}$	$x_i^{(3)}$	$p_i^{(3)}$
1	4	79	116	24.21
2	24	103.21	140.21	33.90
3	68	137.11	174.11	33.89
4	112	171	208	33.52
5	136	204.52	241.52	35.16

Table 12: Results obtained with our heuristics (sequence devoted to CC2)

i	$x_i^{(1)}$	$x_i^{(2)}$	$x_i^{(3)}$	$p_i^{(3)}$
1	48	132	179	37.96
2	92	169.96	216.96	53.14

Table 13: Comparison between the solution obtained with our heuristics and the solution proposed by the company

		Solution of heuristics	Subjection company solution
Total completion time		546.78	563
Sequence to CC1	Completion time	276.68	282
	Delay	0	0
	Slowing down	1.02	7
Sequence to CC2	Completion time	270.10	281
	Delay	0	0
	Slowing down	0	11

Table 14: Computation results for some industrial examples

Problem structure		Running time for CC1			Running time for CC2		
$n_1 * n_2$	currently processed sequence	process	Slowing down	Delay	process	Slowing down	Delay
14*5	1, 2	549.48	117.04	0	523.99	8.7	0
15*5	1	566.42	43.65	0	542.06	0	93.87
15*9	2	632.19	0	60.7	656.16	8.1	0
33*14	1	1138.30	170.38	0	1141.40	0	145.58
40*16	1, 2	1361.10	257.20	0	1180.80	117.12	0

Table 15: Initial situation (instance 33*14)

r_1	r_2	t_1	t_2
4	24	150	170

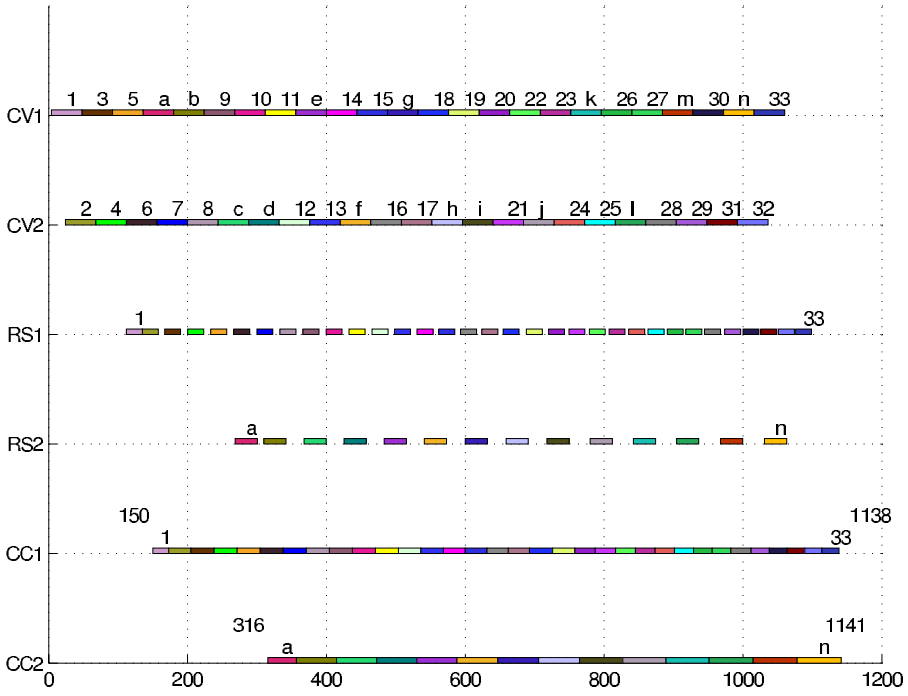

Figure 11: Gantt chart associated with industrial example 33*14

Table 16: Results of sequence to CC1 (industrial example 33*14)

i	$x_i^{(1)}$	$x_i^{(2)}$	$x_i^{(3)}$	$p_i^{(3)}$	$\underline{p_i^{(3)}}$
1	4	111.90	150.00	22.90	17.00
2	24	134.90	172.90	32.05	23.80
3	48	166.95	204.95	33.22	24.66
4	68	200.17	238.17	33.22	24.66
5	92	233.39	271.39	33.22	24.66
6	112	266.61	304.61	33.22	24.66
7	156	299.83	373.83	33.22	24.66
8	200	333.05	371.05	33.22	24.66
9	224	366.27	404.27	33.22	24.66
10	268	399.49	430.49	33.13	24.66
11	312	432.62	470.62	33.13	24.66
12	323	465.75	503.75	32.36	24.66
13	376	498.11	536.11	32.36	24.66
14	400	530.47	568.47	31.39	24.66
15	444	561.85	595.85	31.39	24.66
16	464	593.24	631.24	30.63	24.66
17	508	623.87	661.87	30.63	24.66
18	532	654.49	692.49	33.42	29.95
19	576	687.91	725.91	32.00	29.95
20	620	719.91	757.91	29.35	24.66
21	640	749.26	787.26	29.35	24.66
22	646	778.61	816.61	28.43	24.66
23	708	807.04	845.04	28.43	24.66
24	728	835.47	873.47	27.71	24.66
25	772	836.18	901.18	27.71	24.66
26	796	890.89	928.89	26.80	24.66
27	840	917.69	955.69	26.80	24.66
28	860	944.48	982.48	29.10	27.95
29	904	973.58	1011.58	26.39	25.00
30	928	999.97	1037.97	25.68	25.00
31	948	1025.65	1063.65	25.35	24.66
32	992	1051.00	1089.00	24.66	24.66
33	1016	1075.66	1113.66	24.66	24.66

Table 17: Results of sequence to CC2 (industrial example 33*14)

i	$x_i^{(1)}$	$x_i^{(2)}$	$x_i^{(3)}$	$p_i^{(3)}$	$\underline{p_i^{(3)}}$
a	136	268.58	315.58	41.28	41.28
b	180	309.86	356.86	57.80	57.80
c	244	367.66	414.66	57.80	57.80
d	288	425.46	472.46	57.80	57.80
e	356	483.26	530.26	57.80	57.80
f	420	541.06	588.06	58.83	58.83
g	488	599.89	646.89	58.83	58.83
h	552	658.72	705.72	58.83	58.83
i	596	717.55	764.55	62.24	62.24
j	684	779.80	826.80	62.24	62.24
k	752	842.04	883.04	62.24	62.24
l	816	904.29	951.29	63.36	63.36
m	884	967.64	1014.64	63.36	63.36
n	972	1031.00	1078.00	63.36	63.36

manager with an easy way to determine automatically a good schedule of the sequences. Our heuristics must be compared with the pragmatic way applied by the company and based on experts' experience. For a large dimension sequence, such an expert generally assigned the charges alternately on converters; such a solution introduced often unnecessary slowdowns and is thus far from the optimal schedule.

5 Conclusions

The paper presents a heuristic algorithm devoted to the steelmaking continuous casting (SCC) scheduling problem. The system under study is based on an industrial application from Arcelor Group in Liege (Belgium). The objective is to construct a planning and scheduling that maximizes productivity. With this in view, we propose a heuristic technique based on eliminating machine conflicts. The model has been implemented with the Matlab software and analyzed with some test problem instances and a real data set. Experimental results show that our heuristics gives encouraging results compared with the scheduling solution applied by the company.

In the future, further developments will be envisaged:

- Even if the company is currently treating the schedule of only two sequences, it would be interesting to model the schedule of several sequences on each CC simultaneously. Nevertheless, it is assumed that the sequences are ordered for each CC. Hence, it will be necessary to introduce a minimal intersequence period between two successive sequences on the same CC.

- Even if the converters (CV) constitute a bottleneck in the system, idle time may be necessary between two successive charges, especially between the last charge of a sequence and the first charge of the next sequence. Such extension will also be analyzed in the future.

References

[1] A.D. Baker, A survey of factory control algorithms which can be implemented in a multi-agent heterarchy: dispatching, scheduling, and pull, *Journal of Manufacturing Systems* 17 (1998) 297–320.

[2] A. Bellabdaoui, J. Teghem, A mixed-integer linear programming model for the continuous casting planning, *International Journal of Production Economics* (2005) to appear.

[3] N.N. Chokshi, J.B. Matson and D.C. McFarlane, Distributed co-ordination of steelmaking operations for reduced production stoppages, in *Proceedings of MCPL* 2000, France, July, 2000.

[4] B. De Schutter, Designing optimal timing and sequencing strategies for a continuous steel foundry, in *Proceedings of the European Control Conference 1999 (ECC'99)*, Karlsruhe, Germany, Aug.-Sept., 1999, Paper 160/BP–2.6.

[5] H.S. Lee, S.S. Murthy, S.W. Haider and D.V. Morse D, Primary production scheduling at steelmaking industries, *IBM J. Res. Develop. Vol.* 40 No. 2 March 1996.

[6] S. Moon, A.N. Hrymak, Scheduling of the batch annealing process - deterministic case, *Computers and Chemical Engineering* 23 (1999) 1193–1208.

[7] M. Nuamo and S.I. Morishita, Cooperative scheduling and its application to steelmaking processes, *IEEE Transactions on Industrial Electronics* 38 (1991) 150–155.

[8] L. Tang, J. Liu, A. Rong and Z. Yang, A mathematical programming model for scheduling steelmaking-continuous casting production, *European Journal of Operational Research* 120 (2000) 423–435.

Manuscript received 19 October 2004
revised 9 June 2005
accepted for publication 12 June 2005

Adil Bellabdaoui
Service de Mathématique et de Recherche Opérationnelle (MATHRO), Faculté Polytechnique de Mons, 9 rue de Houdain, 7000 Mons, Belgique
E-mail address: Adil.Bellabdaoui@fpms.ac.be

Antonio Fiordaliso
Service de Mathématique et de Recherche Opérationnelle (MATHRO), Faculté Polytechnique de Mons, 9 rue de Houdain, 7000 Mons, Belgique
E-mail address: Antonio.Fiordaliso@fpms.ac.be

Jacques Teghem
Service de Mathématique et de Recherche Opérationnelle (MATHRO), Faculté Polytechnique de Mons, 9 rue de Houdain, 7000 Mons, Belgique
E-mail address: Jacques.Teghem@fpms.ac.be