

# Constraint Programming Approach to Steelmaking-making Process Scheduling

## Tieke Li

Management School of University of Science and Technology, Beijing. Xueyuan Road 30,  
Haidian District, Beijing, P. R. China  
Phone: +86-10-62333733, Fax: +86-10-62333582,  
Email Address: Tieke@manage.ustb.edu.cn

## Yan Li

Management School of University of Science and Technology, Beijing. Xueyuan Road 30,  
Haidian District, Beijing, P. R. China  
Phone: +86-10-62390146,  
Email Address: Stanleyleeustb@hotmail.com

## Qun Zhang

Management School of University of Science and Technology, Beijing. Xueyuan Road 30,  
Haidian District, Beijing, P. R. China  
Phone: +86-10-62332744, Fax: +86-10-62333582,  
Email Address: Zq@manage.ustb.edu.cn

## Xuedong Gao

Management School of University of Science and Technology, Beijing. Xueyuan Road 30,  
Haidian District, Beijing, P. R. China  
Phone: +86-10-62332269, Fax: +86-10-62333582,  
Email Address: Gaoxuedong@manage.ustb.edu.cn

## Shufen Dai

Management School of University of Science and Technology, Beijing. Xueyuan Road 30,  
Haidian District, Beijing, P. R. China  
Phone: +86-10-62333967, Fax: +86-10-62333582,  
Email Address: Daisf@manage.ustb.edu.cn

## ABSTRACT

*This paper presents a constraint programming (CP) approach to optimal steelmaking process scheduling with constraints of processing time, limited waiting time between adjacent stages, serial batching, sequence independent setup time, release/due time, and with the objective of minimizing maximal total waiting time between adjacent charges in the same casts. The model and search strategies are proposed. Numerical experiments with the steel making process show that CP approach, under appropriate formulation and search strategies, can not only describe the problem exactly, but also can solve the problem more effectively and efficiently compared with classical exact algorithms and heuristic rules.*

Key words: steelmaking process; hybrid flow shop; batch scheduling; constraint programming; search strategies

## INTRODUCTION

The steel-making process (SP) consists of three stages: steel-making, refining and continuous casting. Each stage further includes parallel identical units of electric arc furnace (EAF), refining furnace (RF), and continuous caster (CC) respectively. Since the process is very extensive in investment and energy consumption and runs in a continuous high-temperature material flow with complicated technological processes, effective scheduling for the process is vital, especially in today's highly competitive global steel market.

The SP scheduling problem is to decide unit assignment and charge sequence based on established cast plan and SP constraints to achieve some optimization objectives. Unlike general production scheduling in discrete industry, SP scheduling has to meet special requirements of steel production process. In SP, the products being processed are

handled at high temperature and converted from liquid (molten steel) into solid (drawn billets). There are extremely strict requirements on material continuity and flow time (including processing time on various devices and transportation and waiting time between operations). Particularly, the optimal scheduling should meet the requirements of no wait between processing stages and no breaking in casting process.

Traditional approach to this problem consists mainly of mathematical programming(Liu & Li, 2002; Tang *et al.*, 2002) and expert system(Stohl & Spopek, 1993; Sun *et al.*, 1998) methods. Because most scheduling problems are NP-hard, it is often difficult to solve them optimally, and combining heuristic methods to get near optimal solutions is often necessary. However, heuristic methods are problem specific with their optimality gap hard to estimate. The expert system can be used to attack difficult problems, offering approximate solution to various combinatorial optimization and industrial application problems, being more generic than heuristic methods. But the implementation is usually complicated for a number of parameters being considered.

This paper presents a Constraint Programming (CP) approach to this problem (Marriott & Peter, 1998). CP as an appealing technology in a variety of combinatorial search problems has grown steadily in the last two decades. Some of the current exciting research on CP include languages for modelling and solving problems, finite domain constraint solving, programming with finite domain constraints, non-linear constraints, modeling, debugging, soft constraints, constraint query languages, concurrent and distributed systems, and comparison/combination of various techniques. The main application domains in which CP technology has been applied and has shown to be competitive both in terms of modeling flexibility and solution efficiency are assignment problem, personnel assignment, network management, scheduling problems, transport problems, controlling electro-mechanical systems, constraint-based spreadsheets, constraint databases, interactive problem solving, constraint graphics/graphical interfaces, overconstrained problems, and other application domains (Rossi, 2000; Wallace, 2002).

Perhaps the most successful application for CP are scheduling problems(Baptiste *et al.*, 2001; Glaisner & Richard, 1997; Le Pape & Baptiste, 1997, 1998). By comparing the way scheduling problems are tackled by using imperative languages (C), generic CP languages (CHIP), and specific constraint programming tools (CP over sets), a study has shown that the CP approach is superior in many respects: development time, nodes visited in the search tree, number of generated feasible solutions, even efficiency(Bachelu *et al.*, 1997). According to this study, the main reason for the success of CP on these problems is the use of constraint propagation, which helps to reduce both the development and the execution time when the solving algorithms are not precisely known. It has also been realized that, for some classes of scheduling problems, there is no one technique which is better than all the others, but rather a combination of several techniques is the best approach. Thus hybrid algorithms, employing both CP and other techniques, have recently been developed. An example is the combination of CP and Integer Programming, which has been used to tackle multiple-hoist scheduling problems(Rodosek & Wallace, 1998).

Compared with Rodosek & Wallace (1998), we solve a different problem with specific optimization objective and constraints. Furthermore, they proposed CLP&MIP solver to solve several classes of hoist scheduling problems which have never previously been solved to optimality; we propose only CP solver to solve steelmaking process scheduling problem to get near optimal solutions. In addition, they implement the integration of CLP with MIP by using the ECLiPSe constraint logic programming platform and the XPRESS-MP mathematical programming package; we use ILOG Solver embedded in ILOG OPL Studio. For comparison of the two constraint programming languages, the readers are referred to Fernandez & Hill (2000).

The rest of the paper is organized as follows. Section 2 describes the SP problem. Section 3 presents a CP model and corresponding search strategies. Section 4 gives the computational results of some SP instances. Finally, section 5 makes some conclusions.

## PROBLEM DESCRIPTION

Notations:

Sets and indexes:

$I$  set of charges.  $i \in I$  is a charge.  $i_k \in I$  is the charge processing at stage  $k$ ;

$K$  set of processing stages, including steelmaking, refining and continuous casting.  $k \in K$  is a processing stage;

- $M$  set of units including EAF, LF and CC.  $m \in M$  is a unit;
- $M_k$  set of units at stage  $k$ ;
- $G$  set of casts.  $n \in G$  is each cast;
- $G_n$  set of charges belonging to cast  $n$ .

Parameters:

- $r_i$  release time of charge  $i \in I$ ;
- $d_i$  due time of charge  $i \in I$ ;
- $p_{ik}$  processing time of charge  $i \in I$  at stage  $k$ ;
- $s$  setup time between two consecutive casts  $u, v \in G$  on the same unit  $m \in M_3$  at the batch processing stage 3;
- $\alpha$  allowed maximal waiting time of charges between two consecutive stages.

Variables:

- $x_{ik}$  unit selected by charge  $i \in I$  at stage  $k \in K$ ;
- $y_{ik}$  starting time of charge  $i \in I$  at stage  $k \in K$ ;
- $z_{ik}$  ending time of charge  $i \in I$  at stage  $k \in K$ ;
- $\beta_{\max}$  maximal breaking time between two consecutive charges at stage 3.

An instance of the problem is described as follows. There are totally 12 charges processing at 3 stages sequentially. The stage set is  $K = \{1, 2, 3\}$  and the third stage, continuous casting, is the batch processing stage. Each stage consists of some identical units. Charges at stage 3 are grouped into 3 or 4 casts according to the established cast plan which also determines the sequence of charges in each cast. Charges belonging to the same cast should be processed in the same unit, and breaking time should be as small as possible. Sequence independent setup time  $s$  is needed between tow consecutive casts in the same continuous caster. Processing should occurred during the time window between release time and due time. Every charge can begin the next stage processing only after the preceded stage finished, and the waiting time between two consecutive stage should no more than an allowed value  $\alpha$ . One unit can process only one charge at any time. The optimization objective is to minimize maximal total breaking time, which is defined as the maximum of total breaking time in each cast.

### CP MODEL

Based on the descriptions above, the CP model is formulated as follows:

$$\min \beta_{\max} \tag{1}$$

Subject to:

$$\beta_{\max} = \max_{\forall n \in G, i, j \in G_n, i < j} (z_{j3} - y_{i3} - \sum_{h \in G_n} p_{h3}) \tag{2}$$

$$y_{i1} \geq r_i \quad \forall i \in I \tag{3}$$

$$z_{i3} \leq d_i \quad \forall i \in I \tag{4}$$

$$y_{ik} + p_{ik} = z_{ik} \quad \forall i \in I \quad \forall k \in K \tag{5}$$

$$z_{ik} \leq y_{i,k+1} \quad \forall i \in I \quad \forall k \in K, k < 3 \tag{6}$$

$$x_{ik} = m \iff i_k \text{ requires unary resource } m \tag{7}$$

$$\forall i \in I \quad \forall k \in K, m \in M$$

$$x_{i3} = x_{j3} \quad \forall n \in G, i, j \in G_n, i \neq j, m \in M_3, M_3 \subseteq M \tag{8}$$

$$z_{i3} \leq y_{j3} \quad \forall u, v \in G, u \prec v, i \in G_u, j \in G_v \tag{9}$$

$$x_{i3} = x_{j3} \implies z_{i3} + s \leq y_{j3} \quad \forall u, v \in G, u \prec v, i \in G_u, j \in G_v \tag{10}$$

$$\max_{\forall i \in I, k \in K, k < 3} (y_{i,k+1} - z_{ik}) \leq \alpha \quad (11)$$

$$x_{ik} \in M_k \quad \forall i \in I, k \in K, M_k \subseteq M \quad (12)$$

$$y_{ik} \geq 0 \quad \forall i \in I, k \in K \quad (13)$$

$$z_{ik} \geq 0 \quad \forall i \in I, k \in K \quad (14)$$

(1) means that the optimization objective is to minimize the maximal breaking time. (2) defines the optimization objective. (3) and (4) require that the processing time of each charge falls into the time window of the release and due time. (5) defines the relationship of starting time, processing time and finishing time. (6) ensures that every charge can be processed at the next stage only after processing at the precede stage finished. (7) defines  $x_{ik} = m$ , which means that charge  $i$  needs to be processed in unit  $m$  at stage  $k$ . For charges in the same cast, (8) requires that they should be processed in the same CC; (9) requires that they should be processed according to the sequence established by the cast plan. For two consecutive casts, (10) requires that setup time should be considered when processing in the same CC. (11) specifies that the waiting time between two consecutive processing stage should no more than allowed value  $\alpha$ . (12-14) defines the variable domains.

### SEARCH STRATEGY

We use ILog OPL Studio 3.7 to solve the model with incomplete search procedure and Depth-Bounded Discrepancy search strategy. OPL supports a number of instructions to support incomplete search procedures. Instruction “firstSolution( $n$ )” returns only the first  $n$  solutions to a goal. Instructions “failLimit” and “timeLimit” limit the number of failures and the CPU time in seconds (a float) allowed when solving a goal respectively. We adopt the instruction “timeLimit” of the form

`search {timeLimit(0.3) rankLocal;}`

to tell OPL to spend at most 0.3 seconds in rankLocal.

OPL offers a number of predefined search procedures for selection: Depth First Search (DFS), Slice-Based Search (SBS), Depth-Bounded Discrepancy Search (DDS)(Walsh, 1997), Best First Search (BFS) and Inter-leaved Depth First Search (IDFS)(Meseguer, 1997).

DFS is a blind search with low efficiency. Slice-Based Search (SBS) is a search strategy that assumes the existence of a good heuristic. Its basic intuition is that the heuristic, when it fails, probably would have found a solution if it had made a small number of different decisions during the search. The choices where the search procedure does not follow the heuristic are called *discrepancies*. As a consequence, SBS systematically explores the search tree by increasing the number of allowed discrepancies. SBS has been shown to be effective for job-shop scheduling problems and, more generally, for all problems where a good heuristic is available. Depth-bounded discrepancy search (DDS) is a variant of SBS where the discrepancies that occur high in the search tree are favored. Best-first search is a well-known strategy for optimization problems that consists of choosing the node with the best value of (the relaxation of) an expression which is, typically, the objective function. Interleaved depth-first search (IDFS) is a search procedure that simulates a parallel depth-first search exploration of a search space on a sequential machine. We adopt SBS as our search strategy just because experiments show that SBS is the most efficient.

For SP scheduling problem SBS and DDS are more efficient, this has been proved by our experiments, too.

### NUMERICAL TESTS

To test the performance of the method and to study the characteristics of the solutions, a computational experiment has been carried out on randomly generated problem instances, which were designed to reflect practical situations in iron and steel industries.

A number of 50 problem instances are used in the experiment. The number of charges to be scheduled is set to be 12 for each instance. Since the minimum scheduling time unit in the iron and steel plant is in an exact number of minutes, minute is taken as the basic time unit. The planning horizon is set to be 480 minutes since this study intends to solve the scheduling problem for an eight hour shift. The processing times is randomly generated from a uniform distribution [30, 50]. In order to reduce experimental cases, it is assumed that every stage has the same number of

units and that every charge has the same release\due time. However, our method can deal with practical problems including different numbers of units for different stages and different release\due time of charges. The number of casts is set to vary at two levels: 3 and 4. The number of units at each stage is set to vary at three levels: 3, 4 and 5. The release time, the due time and the allowed maximal waiting time between processing stages are set to be 0, 480mins, and 15mins respectively. The setup times are randomly generated from a uniform distribution [50, 60].

$G_n$  is generated by randomly partitioning the list 1,2,...,12 into  $|G|$  casts.

The model was solved by ILog OPL Studio 3.7, and the experiment was carried out on an Intel® Celeron® CPU 2.20GHz PC. Table 1 gives the descriptive statistics of the computational results.

Some observations can be made from Table 1:

- 1) High efficiency of solving process with the maximal computation time of 52.19 seconds, which is acceptable;
- 2) Good solution quality. There are 30% problems with the optimization value 0, 52% problems no more than 10 minutes, 78% no more than 20 minutes, and 98% no more than 30 minutes. Only one objective value of problem 11 is 33 minutes;
- 3) All the constraints are satisfied explicitly.

For problem 11, the only one with the maximal computation time of 52.19s, the optimization objectives are shown in Figure 3 while the maximal waiting times adjusted. Therefore we can expect improvements of the objective value by adjusting the maximal waiting time allowed by the process requirements.

	Maximal Completion Time(min)	Maximal Waiting Time(min)	Maximal Breaking Time(min)	Computing Time (s)
Average	476.16	14.18	10.54	11.7
Standard Error	0.75	0.36	1.43	1.25
Median	478	15	10	9.77
Standard Deviation	5.32	2.55	10.09	8.82
Minimum	458	0	0	1.08
Maximum	480	15	33	52.19

Table 1: Statistics of the computation results.

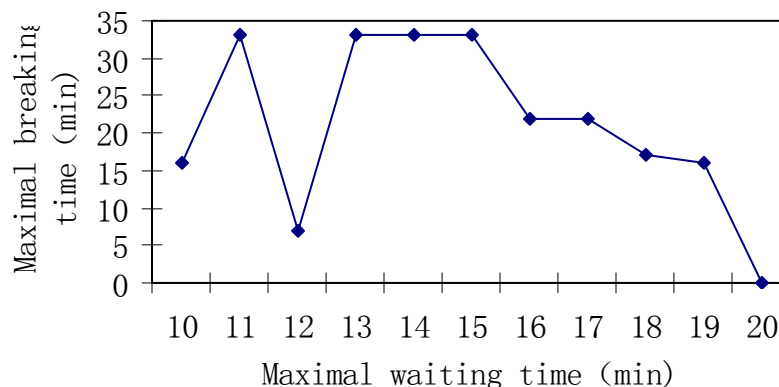


Figure 1. Objective values of problem 11 varying with the maximal waiting times.

The next is an instance from practical application(Liu & Li, 2002) with the problem specific data as follows:  
 Facility number: CF6, CF7, CF8, RF3, RF4, RF5, CC1, CC2.; Setup time=55mins; Number of casts=2;  
 Set of casts: {[1, 2, 3, 4, 5, 6],[7, 8, 9, 10, 11]}.

Stage	Charges										
	1	2	3	4	5	6	7	8	9	10	11
1	50	50	50	50	50	50	45	45	45	45	45
2	50	50	50	45	45	45	40	40	40	35	35
3	35	35	35	30	30	30	30	30	30	30	30

**Table 2: Processing time of 11 Practical Charges.**

For this instance, the computation time is 0.156s; the makespan is 305min; the maximal waiting time is 10min; the maximal total breaking time is 0. The Gantt chart of the optimal scheduling is shown in Figure 2 with numbers in bars representing charges.

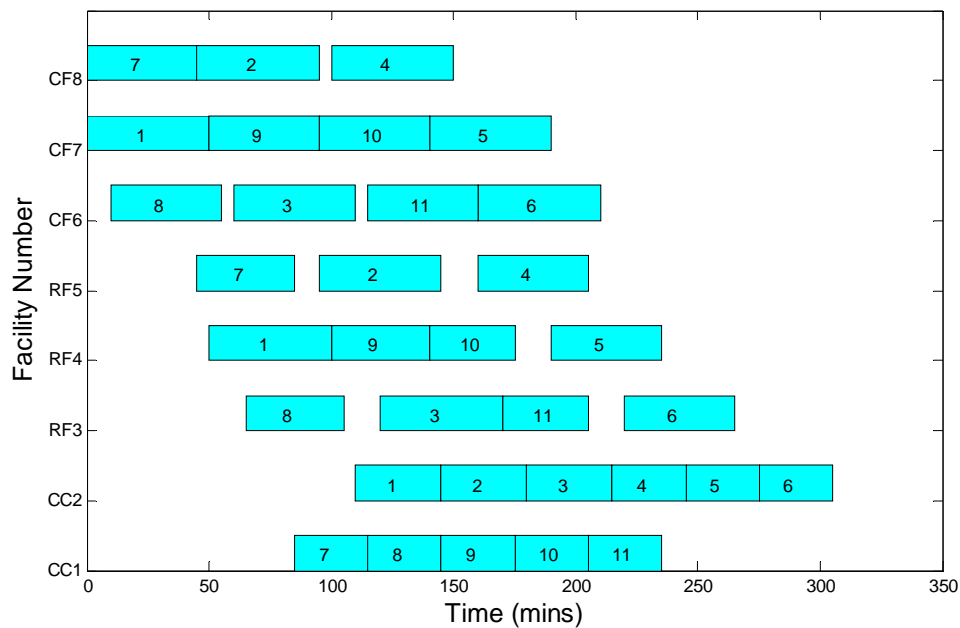


Figure 2: Gantt chart of the practical instance.

### CONCLUSIONS

This paper presents a CP approach to optimal steelmaking process scheduling on identical parallel units with constraints of processing time, limited waiting time between adjacent stages, serial batching, sequence independent setup time, release\due date and with the objective of

minimizing maximal total waiting times between adjacent charges among the same cast. The model and solving strategies are proposed. Numerical experiments with the steel making process show that CP approach, under appropriate solving strategies, can not only describe and formulize the problem exactly, but also can attack the

problem more effectively and more efficiently compared with classical exact algorithms and heuristic rules. The results are of immediate commercial value, too.

CP method is confronted with the problem of solving difficulty, too. How to solve problems efficiently is the key for the success of CP approach. One basic idea is to simplify the model based on thorough understanding of the problem characteristics. However, this is viable only within the limits of precise description of the problem. For further increasing solving efficiency it is usually necessary to devise appropriate search strategies. Sometimes we have to sacrifice the optimality, trying to find near optimal solutions by limiting failure times or solving time, or by limiting the number of feasible solutions, etc.

We will study in the future the combination of CP and other mathematical programming approaches to optimal scheduling for steelmaking process.

## REFERENCES

- Bachelu, A., Baptiste, P., Varnier, C., Boucher, E., & Legeard, B. (1997). *Multi-criteria comparison between algorithmic, constraint logic and specific constraint programming on a real scheduling problem*. Paper presented at the Proceedings of PACT97 - 3rd International Conference on the Practical Application of Constraint Technology, 23-25 April 1997, London, UK.
- Baptiste, P., Claude Le, P., & Nuijten, W. (2001). *Constraint-based scheduling: Applying constraint programming to scheduling problems*. Boston: Kluwer Academic Publishers.
- Fernandez, A. J., & Hill, P. M. (2000). Comparative study of eight constraint programming languages over the boolean and finite domains. *Constraints*, 5(3), 275.
- Glaisner, F., & Richard, L. M. (1997). *Forward c. A refinery scheduling system*. Paper presented at the Proceedings of PACT97 - 3rd International Conference on the Practical Application of Constraint Technology, 23-25 April 1997, London, UK.
- Le Pape, C., & Baptiste, P. (1997). *A constraint programming library for preemptive and non-preemptive scheduling*. Paper presented at the - Proceedings of PACT97 - 3rd International Conference on the Practical Application of Constraint Technology, 23-25 April 1997, London, UK.
- Le Pape, C., & Baptiste, P. (1998). Resource constraints for preemptive job-shop scheduling. *Constraints*, 3(4), 263.
- Liu, G., & Li, T. (2002). A steelmaking-continuous casting production scheduling model and its heuristic algorithm. *Systems Engineering*, 20(6), 44.
- Marriott, K., & Peter, J. S. (1998). *Programming with constraints: An introduction*. Cambridge, Mass.: MIT Press.
- Meseguer, P. (1997, 1997). *Interleaved depth-first search*. Paper presented at the Proceedings of 15th International Joint Conference on Artificial Intelligence. IJCAI 97, 23-29 Aug. 1997, Nagoya, Japan.
- Rodosek, R., & Wallace, M. (1998). *A generic model and hybrid algorithm for hoist scheduling problems*. Paper presented at the Principles and Practice of Constraint Programming - CP98. 4th International Conference, CP98. Proceedings, 26-30 Oct. 1998, Pisa, Italy.
- Rossi, F. (2000). *Constraint (logic) programming: A survey on research and applications*. Paper presented at the New Trends in Constraints Joint ERCIM/Compulog Net Workshop, 25-27 Oct. 1999, Paphos, Cyprus.
- Stohl, K., & Spopek, W. (1993). *Vai-schedex: A hybrid expert system for co-operative production scheduling in steel plant*. Paper presented at the International Conference on Computerized Production Control in Steel Plant, Korea.
- Sun, F., Tang, L., & Zheng, B. (1998). The steelmaking-continuous casting production scheduling based on expert system. *Metallurgical Industry Automation*, 31.
- Tang, L., Luh, P. B., Liu, J., & Fang, L. (2002). Steel-making process scheduling using lagrangian relaxation. *International Journal of Production Research*, 40(1), 55.

Wallace, M. (2002). Constraint logic programming. In *Computational logic: Logic programming and beyond. Essays in honour of robert a. Kowalski part 1 (lecture notes in artificial intelligence vol.2407)* (pp. 512): Springer-Verlag.

Walsh, T. (1997). *Depth-bounded discrepancy search*. Paper presented at the Proceedings of 15th International Joint Conference on Artificial Intelligence. IJCAI 97, 23-29 Aug. 1997, Nagoya, Japan.

This research is supported by National Science Foundation of China (70371057)