A Metaheuristic Approach for Scheduling Steelmaking and Casting Plants

Davide Armellini[1], Paolo Borzone[1], Sara Ceschia[2], Luca Di Gaspero[2], Andrea Schaerf[2]

[1] Danieli Automation S.p.A.
via Bonaldo Stringher 4, I-33042 Buttrio (UD), Italy / Piazza Borgo Pila 39, I-16129 Genoa, Italy
d.armellini@dca.it, p.borzone@dca.it

[2] DPIA – Università degli Studi di Udine
via delle Scienze 206, I-33100 Udine, Italy
sara.ceschia@uniud.it, luca.digaspero@uniud.it, schaerf@uniud.it

1 Introduction

The steelmaking process is one of the most complex industrial production operations due to the presence of many technological (physical, chemical, mechanical, . . .) and business constraints. The operation of a steelmaking plant is quite costly, therefore the throughput maximization by means of an optimized scheduling is of crucial importance to ensure productivity and competitiveness.

The steelmaking process comprises several stages, outlined in Figure 1. At first the iron scrap is melted in an *Electric Arc Furnace* (EAF), then the liquid metal is poured in a *ladle* that will be used to contain the steel in all the following processing steps. The next step is the *Ladle Furnace* (LF), in which some additives are added for obtaining the desired chemical composition of the product. Afterwards, the metal undergoes the *Vacuum Degassification* (VD) process, which reduces the gas content in order to avoid explosions. Once these steps in the common production line are over, the melted steel goes to a *Caster*. The destination of the job could either be a *Continuous Casting* (CC) or an *Ingot Casting* (IC) machine, which forge the metal into long (i.e., slabs or billets) or short (ingots) semifinished products, respectively. These products, then, could be subject to further independent processing (e.g., rolling).

The main feature of this initial part of the steelmaking process is the physical impossibility to buffer jobs between different processing steps because of the cooling of the liquid metal in case of waiting. As a consequence, the jobs should be scheduled in a *just-in-time* fashion.

Besides the main processing machines themselves, the process needs many additional resources. In particular, the *ladles* that contain the melted metal are reusable resources and so are the *tundishes*, used for pouring the liquid metal for continuous casting, or the *wagons* containing the molds for ingot casting. The ingot wagons wait on a *cooling line* and they are then processed by a final machine for *stripping* them from the molds.

In addition to the constraints arising from the specific process model, the production undergoes a

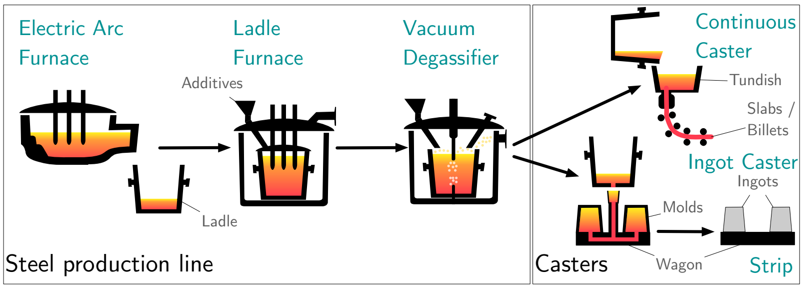

Figure 1: The steelmaking and casting production process.

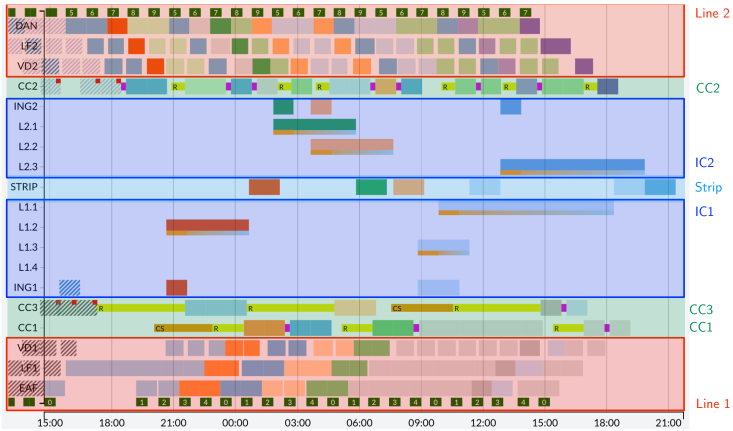

Figure 2: A daily schedule of the steelmaking plant.

number of further processing constraints on the job sequences. In particular, some additive materials used to reach the desired chemical composition of the final steel grade, can contain chemical elements considered as *polluters* of the ladle for other steel grades. Therefore, for those critical products it is mandatory to ensure a *cleaning* (or, in some cases, a *double cleaning*), that is using the ladle for a non-pollutant production before processing the critical steel grade. Also, although the continuous caster can be used uninterruptedly for the same alloy, thus leading to no-setup times, for quality reasons it is not possible to produce more than a given number of items for the same type of production. Other kind of constraints are related to some business rules. For example, some productions have *appointments* with some post-processing phase (e.g., hot rolling in a rolling mill) in order to fulfill delivery deadlines.

The coordination of all these resources and the technological and business constraints, makes the scheduling problem a particularly difficult one.

2 Problem formulation

In this work we deal with a specific real-world version of the problem arising in a mid-sized steel making plant located in north-eastern Italy. The requirements were specified by Danieli Automation, part of the Danieli group, one of the world-leading suppliers of equipment and plants to the metal industry.

The plant comprises two steel production lines followed by three CC machines, and two IC machines as casters. The Ingot Casters have 4 and 3 cooling lines, respectively, and they share a single stripping machine. Figure 2 shows a schedule obtained by our system for a typical working day. The two main production lines (highlighted in red) are at the top and the bottom of the Gantt chart whereas the ingot casters and all the related resources (highlighted in blue) are represented in the central part. The three continuous casters are highlighted in green.

A simpler version of the problem, together with a MILP model for its solution, was presented in [2]. For brevity, we refer the reader to that paper for a detailed description and we present the main differences with respect to that model.

In order to better meet the real specification of the problem and to ehnance the scheduling flexibility, we make the following modifications with respect to the model of [2]:

- we consider the possibility that a job switches from one line to another also before the casting machines (at a price of a longer moving time);

- since the plant runs for 24h a day/6 days a week, we have to schedule the production for the next time-horizon (typically a single day) by considering the *border data*, i.e., the last jobs scheduled at the end on the previous scheduling stage;
- instead of using the makespan as the objective function for the problem, we use a *throughput* measure that considers (and maximises) the number of jobs completed within the scheduling time-horizon.

Regarding the last point, we consider as input a set of available jobs larger than the ones that will be scheduled within the given horizon, and the solver undertakes also the task of selecting the ones that are processed (and the ones that are postponed).

3 Problem model and solution method

The problem has been tackled by means of a family of local search metaheuristics implemented using the EasyLocal++ framework [1]. The problem has been modeled as follows:

- The *search space* is an indirect representation of the schedule by means of the sequence (i.e., the permutation) of all the available jobs. Each job has also a set of additional data stating on which machine it has been assigned for each specific processing step. The mapping between the job order and the actual execution times of each job is obtained by applying a chronological, right-justification greedy algorithm. All the jobs whose execution time exceed the time-horizon are part of the *tail* of the solution, which consists of the jobs considered as unscheduled.
- The *neighborhood* relation is defined as the set of swaps of the processing order of two jobs and, possibly, the reassignment of the jobs to different machines (for some processing step). The neighborhood is refined by removing moves that are clearly uneffective, such as for example moves that swap jobs that are both in the tail of the state.
- The *cost function* takes into account the difference between the upper bound of scheduled jobs and the number of jobs actually scheduled in the solution. In addition, it considers the violations of some hard constraints as the number of times that a ladle is not clean as required.

Regarding the cost function, the upper bound of the number of scheduled jobs is obtained considering only the processing time of all machines, disregarding setups and waiting times.

As metaheuristic procedure to guide the search, we have implemented *Hill Climbing*, *Simulated Annealing*, and *Tabu Search*.

4 Current work

This is an ongoing research and we are currently working of the infrastructure to extract the input data and upload the solution automatically from the actual control monitor of the plant.

At the same time, we are testing and tuning our solver on a set of real-world instances coming from the actual production in the plant. For the near future, we plan to compare the results of the proposed metaheuristics among themselves and against the actual schedules produced by the human operators.

References

[1] Luca Di Gaspero and Andrea Schaerf. EasyLocal++: An object-oriented framework for flexible design of local search algorithms. *Software — Practice & Experience*, 33(8):733–765, July 2003.

[2] Maria Pia Fanti, Giuliana Rotunno, Gabriella Stecco, Walter Ukovich, and Stefano Mininel. An integrated system for production scheduling in steelmaking and casting plants. *IEEE Transactions on Automation Science and Engineering*, 13(2):1112–1128, April 2016.