Computers and Chemical Engineering 164 (2022) 107864

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

Centralized and hierarchical scheduling frameworks for copper smelting process

Hussain Ahmed[a], Luis A. Ricardez-Sandoval[b], Matti Vilkko[a,*]

[a] Automation Technology and Mechanical Engineering, Tampere University, Tampere, 33720, Finland
[b] Department of Chemical Engineering, University of Waterloo, Waterloo, ON, N2L 3G1, Canada

ARTICLE INFO

Article history:
Received 10 December 2021
Revised 20 May 2022
Accepted 29 May 2022
Available online 2 June 2022

Keywords:
copper smelting
copper losses
scheduling
hierarchical framework
heuristics
coordination

ABSTRACT

Optimal scheduling of copper smelting process is an ongoing challenge due to conflicting objectives of the various process units and the inter-dependencies that exist among these units. To design a scheduling framework, two potential alternatives – centralized and hierarchical approaches – can address those inter-dependencies in this process. These approaches represent the two extremes and the choice depends on the accuracy, reliability, and complexity of the scheduling task. In this study, optimization-based centralized and hierarchical scheduling frameworks are developed to find an optimal schedule for the smelting process, considering the inter-dependencies among process units. We propose a practical and effective coordination scheme for the hierarchical framework that finds a near-optimal schedule with reasonable computational demands. Two case studies are presented to demonstrate that the proposed hierarchical framework is capable of finding a near plant-wide optimum for the copper smelting process and it can be used in similar plant-wide scheduling applications.

Nomenclature

Centralized framework

Sets:

PSC unit number n, $n \in \{1, 2, 3, \ldots, N\} \subset \mathbb{Z}^+$, N is total number of PSC units

PSC batch number b, $b \in \{\{1, 2, 3, \ldots, B\} \subset \mathbb{Z}^+$, B is total number of batches

Scheduling horizon h(min), $h \in \{1, 2, 3, \ldots, H\} \subset \mathbb{Z}^+$, H is the maximum time value of complete scheduling horizon

PSC operation $z_i \in Z = \{loading_i,\ slag-blow_i,\ slag-skim_i,\ copper-blow,\ batch-end\}$ where $i \in \{1, 2, 3, \ldots, I\} \subset \mathbb{Z}^+$, and I is total number of repeated operations

Variables:

FSF:

$feed^h$(kg/min)= concentrate feed rate at time h
$FSFProd^h$(kg/min)= FSF production rate at time h
$FSFMass^h$(kg)= mass of matte in the FSF at time h

PSC:

$$b_{z_i}^{n,b,h} = \begin{cases} 1 & \text{operation } z_i \text{ is processed on unit } n \text{ during batch } b \text{ at time } h \\ 0 & \text{otherwise} \end{cases}$$

$$s_{z_i}^{n,b,h} = \begin{cases} 1 & \text{operation } b_{z_i}^{n,b,h'} \text{ is completed at time } h' \quad \forall h \geq h' \\ 0 & \forall\ h < h' \end{cases}$$

$$idle^{n,b,h} = \begin{cases} 1 & b_{z_i}^{n,b,h} = 1 \text{ on unit } n \text{ during batch } b \text{ at time } h \\ 0 & \text{otherwise} \end{cases}$$

$PSCMass^{n,b,h}$(kg)= mass of matte in PSC unit n during batch b at time h.

$eleMass^{n,b,h}$(kg)= content of unwanted element ele in the PSC matte of unit n during batch b at time h. This ele can be iron (Fe) or sulfur (S).

$CuMass^{n,b,h}$(kg)= mass of the copper loss in PSC unit n during batch b at time h.

$CuRatio^{n,b,h}$(kg)= minimum copper loss scheme points in PSC unit n during batch b at time h. These point are calculated using the scheme presented in our previous work (Ahmed et al., 2021).

Parameters:

FSF:

mg(percent)= matte grade
$feedMin$(kg/min)= minimum feed rate of input concentrates
$feedMax$(kg/min)= maximum feed rate of input concentrates
$FSFLow$(kg)= FSF capacity limit

* Corresponding author.
E-mail addresses: hussain.ahmed@tuni.fi (H. Ahmed), laricard@uwaterloo.ca (L.A. Ricardez-Sandoval), matti.vilkko@tuni.fi (M. Vilkko).

https://doi.org/10.1016/j.compchemeng.2022.107864

$FSFMass^{h=0}$(kg)= FSF initial inventory level
$Grad1$= gradient of the matte grade versus FSF production
$Grad2$= gradient of matte grade versus feed rate
Y_{inter}(kg/min)= y-intercept of the matte grade versus FSF production and feed rate trajectories
PSC:
$CuLoss_{z_i}$(kg/min)= copper loss rate during operation z_i
$eleRate$(kg/min)= oxidation rate of element *ele*
Pos_{z_i}= position number of operation z_i in set Z
$NumLoad$= number of loading operations to single PSC unit, $NumLoad \in \mathbb{Z}^+$
$ProcMax_{z_i}$(min)= maximum processing duration of z_i
$ProcMin_{z_i}$(min)= minimum processing duration of z_i
$ProcFix_{z_i}$(min)= fixed processing duration of z_i
$MatteFix$(kg)= fixed amount of matte transferred from FSF to PSC unit
$PSCStart^n$(min)= starting time of PSC unit n
$PSCPrior^n$= priority of PSC unit n
Objective functions:
f_{FSF}(kg/min)= FSF objective function
f_{Cen}(kg)= centralized framework objective function
f_{PSC}(kg)= single-batch problem objective function

Hierarchical framework

Sets:
Batch horizon t(min), $t \in \{1, 2, 3, \ldots, T\} \subset \mathbb{Z}^+$ *and* $T << H$, T is the maximum time value of the single batch scheduling horizon

Variables:
single PSC batch:

$$b^t_{z_i} = \begin{cases} 1 & \text{operation } z_i \text{ is processed at time } t \\ 0 & \text{otherwise} \end{cases}$$

$$s^t_{z_i} = \begin{cases} 1 & \text{operation } b^{n,b,t'}_{z_i} \text{ is completed at time } t' \quad \forall t \geq t' \\ 0 & \forall\ t < t' \end{cases}$$

Parameters:
single PSC batch:

$$LoadBatch^{n,b,t}(\text{min}) = \begin{cases} t & \textit{loading} \text{ time } t \text{ of PSC unit } n \text{ during batch } b \\ 0 & \text{otherwise} \end{cases}$$

$$BlowBatch^{n,b,t}(\text{min}) = \begin{cases} t & \textit{slag} \text{ or } \textit{copper} \text{ blow time } t \text{ of PSC unit } n \text{ during batch } b \\ 0 & \text{otherwise} \end{cases}$$

$$LocalEnd^{n,b,t}(\text{min}) = \begin{cases} t & \textit{batch end} \text{ time } t \text{ of PSC unit } n \text{ during batch } b \\ 0 & \text{otherwise} \end{cases}$$

$$LoadDelay^t(\text{min}) = \begin{cases} t & \text{time } t \text{ where no } \textit{loading} \text{ can be made to a batch} \\ 0 & \text{otherwise} \end{cases}$$

$$BlowDelay^t(\text{min}) = \begin{cases} t & \text{time } t \text{ where no } \textit{slag} \text{ or } \textit{copper blow} \text{ can be made to a batch} \\ 0 & \text{otherwise} \end{cases}$$

Coordinator:
$BatchStart^{n,b}$(min)= starting time of batch b on PSC unit n
$BatchEnd^{n,b}$(min)= end time of batch b on PSC unit n

$$FSFNo^h = \begin{cases} 1 & \text{FSF capacity violates at time } h \\ 0 & \text{otherwise} \end{cases}$$

$$PSCLoad^{n,h} = \begin{cases} 1 & \textit{loading} \text{ is made to PSC unit } n \text{ at time } h \\ 0 & \text{otherwise} \end{cases}$$

$$PSCBlow^{n,h} = \begin{cases} 1 & \textit{copper} \text{ or } \textit{slag} \text{ blow is made to PSC unit } n \text{ at time } h \\ 0 & \text{otherwise} \end{cases}$$

$$LogisNo^h = \begin{cases} 1 & \text{logistical constraint is active at time } h \\ 0 & \text{otherwise} \end{cases}$$

$$LogisNo^h = \begin{cases} 1 & \text{flow constraint is active at time } h \\ 0 & \text{otherwise} \end{cases}$$

1. Introduction

The copper industry, which is composed of mining, extraction and processing sectors, has contributed much to the European economy (Institute, 2018; Group, 2019). Over the past couple of decades, this industry has struggled to neutralize the growing market demands considering that the high-quality concentrate deposits are running out (Group, 2019; Iiro Harjunkoski and Weidemann, 2005). Consequently, the utilization of low-quality concentrates, along with outdated scheduling and planning techniques, affect the copper industry production and key goals (FIMECC, 2014). Therefore, state-of-the-art scientific techniques are required to provide benefits to this industry in the long run (Iiro Harjunkoski and Weidemann, 2005).

In the copper industry, copper smelters are used to produce pure copper ($\approx$ 99.9 percent). This process is composed of various units through which an input concentrate is passed to oxidize unwanted elements such as iron and sulfur by following a scheduling recipe that is generally defined by the technical staff (Ahmed et al., 2021). However, designing a scheduling recipe in advance is challenging due to the unavailability of measured data, external disturbances, and inter-dependencies among the process units (Iiro Harjunkoski and Weidemann, 2005; Harjunkoski and Grossmann, 2001). Hence, on-site schedule generation is the most effective way to efficiently operate the smelter.

Two important units of the copper smelting process that contribute much to the oxidation of unwanted elements are flash smelting furnace (FSF) and Peirce-Smith converter (PSC), which are generally operated and monitored by separate process personnel. The operation of these units is subject to various types of constraints that arise due to variation in the quality of raw materials, logistics issues, capacity constraints, and operational inter-dependencies (Iiro Harjunkoski and Weidemann, 2005). The latter is of great concern in the smelting process because the operational inter-dependencies among process' units limit the performance of the overall smelting process. Therefore, these inter-dependencies need to be resolved in a sophisticated fashion to achieve optimal operation of the process (Harjunkoski et al., 2008).

In a copper smelting process, operating the FSF and PSC units independently moves the smelter from an optimal to a sub-optimal operational point (Harjunkoski et al., 2006; 2008; Iiro Harjunkoski and Weidemann, 2005; Ahmed et al., 2021). Furthermore, when these units are operated and scheduled independently, it is difficult for the process personnel to react promptly alone and resolve the inter-dependencies, which are produced by other smelter units. Therefore, it is highly unlikely that the technical staff will foresee the consequences that their operational decisions have on the overall process behavior (Iiro Harjunkoski and Weidemann, 2005; Harjunkoski et al., 2008). The losses associated with this kind of process operation are significant, so close coordination among process units is required to achieve better process performance (Ewaschuk et al., 2018).

To achieve optimal coordination between the process units, selecting an appropriate coordinating parameter is key. This selection requires a deep understanding of the scheduling application and the pros and cons associated with the selected parameter. An important parameter in the copper smelting process is the matte grade that influences the copper losses during the process operation (Ahmed et al., 2021). As minimization of the copper losses is an important goal in the copper smelting process, process personnel are interested in finding the optimal matte grade that can generate the minimum copper losses. Therefore, matte grade can be a potential coordinating parameter in the smelting process.

In the present study, discrete-time mixed integer linear programming (MILP) centralized framework and a hierarchical scheduling framework are proposed for a smelting process that consists of one FSF and multiple PSC units. This study is an extension of our previous work, which deals with the scheduling of a single PSC batch only (Ahmed et al., 2021). The objective of this work is to design frameworks that can produce schedules for a

multiple-PSC units and multiple-batch process, while maximizing the FSF throughput, satisfying the FSF capacity constraint, minimizing the copper losses during PSCs operation, and resolving the scheduling inter-dependencies among the process units.

In this work, if the matte grade is used as the coordinating parameter, it will move the frameworks' formulation from linear to non-linear. The objective of this work is not to find the optimal schedule through non-linear and intensive strategies. Therefore, to maintain the linearity, matte grade is not used as the coordinating parameter during the hierarchical framework formulation. This study focuses on the development of the practical and efficient linear optimization models that can be implemented in practice and still improve operations management for the smelting process. One practical way to coordinate information among a smelter's units is to use heuristics that are inspired from industrial practices. The advantage of heuristics is that they can provide close coordination among the smelter's units without losing the linearity of the framework. Therefore, a practical and efficient heuristic is proposed for the coordinator; thus enabling an efficient smelter operation and better overall coordination among units

The remainder of paper is organized as follows. In Section 2, the centralized and hierarchical frameworks are briefly discussed, and the scheduling literature is presented. Section 3 provides a summary of the copper smelting process operation and Section 4 describes the problem statement in detail. The core of this study is presented in Section 5, in which the centralized and hierarchical frameworks are formulated. It also provides a detailed description of the heuristics used by the coordinator. In particular, the motivation is to demonstrate that the coordinator can handle the inter-dependencies effectively and provides a near-optional solution. Section 6 describes two case studies to show that the hierarchical framework is suitable for large-scale complex scheduling problems. Section 7 presents the conclusion and offers an outlook for future work.

2. Theoretical background

Industrial processes with complex process dynamics and objectives, such as copper smelters, have been categorized as large-scale problems that attracted much attention in the 1970s and 1980s (Cheng et al., 2006). The interest in the optimal operation of large-scale systems has increased in the last few decades due to the benefits that can be obtained when process decisions are made based on the optimal process scheduling and operation management (Martí et al., 2013). Despite the availability of advanced systems and tools, large-scale scheduling problems are usually solved manually by experienced technical personnel. This can be explained by the fact that large-scale scheduling applications are sensitive to solution quality, choice of the heuristics that are adopted during the formulation of the scheduling framework, and the computational time required to find an acceptable solution (Roslóf et al., 2002).

When designing a large-scale scheduling framework, two common paradigms in copper smelters are centralized and hierarchical approaches, which have also been used successfully in various industrial applications (Harjunkoski and Grossmann, 2001; Cheng et al., 2006; 2007; 2004; Popa, 2014). The choice among these approaches depends on the accuracy of the solution, availability of computing power, and the computational costs. Beside these limitations, other dominant factors are their vulnerability to the single point of failure, scalability, and the ability to repair in case of a failure (Christodoulopoulos et al., 2009). These approaches are introduced in turn below.

The centralized approach is based on the idea that the information of the entire process is gathered centrally and then processed accordingly. This approach considers direct communication among the process units; hence, it leads to a large and complex

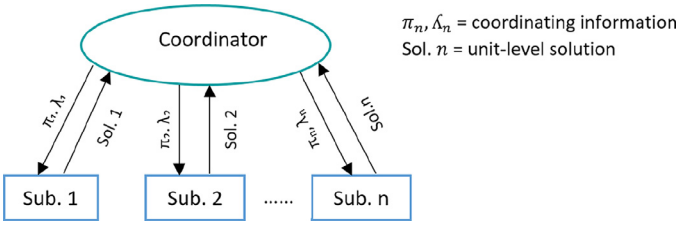

Fig. 1. Decomposition-coordination.

scheduling problem formulation (Shiquan et al., 2019). A fully centralized scheduling solution for a large-scale system is often undesirable and sometimes impractical due to the large demands of the computational resources; such formulation is only suitable for small and medium-sized scheduling problems (Cheng et al., 2007; Moroşan et al., 2010; Martí et al., 2013; Christodoulopoulos et al., 2009). Moreover, studies have shown that solving the entire process as a single scheduling problem can exhibit poor fault tolerance and can be difficult to tune and maintain (Cheng et al., 2007). In addition, when units are managed by distinct process personnel, especially from different organizations, sharing data might be undesirable due to data protection issues. From a safety perspective, the malfunctioning of a single process unit in the centralized approach might result in the collapse of the entire scheduling framework (Shiquan et al., 2019).

Several strategies have been proposed to overcome the shortcomings of the centralized approach (Cheng et al., 2007; Gupta and Maranas, 1999; Anderson and Papachristodoulou, 2012). To reduce the complexity of the scheduling problem, a decentralized approach is adopted where a large-scheduling problem is virtually decomposed into multiple unit-level scheduling problems, which are solved independently. A conventional decentralized scheme may not be able to provide the plant-wide optimum or feasibility since no communication mechanisms exist among process units (Cheng et al., 2007).

This problem is overcome using the hierarchical approach, which consists of an upper layer and a lower layer. The upper layer, which is usually referred to as the coordinator, enables communication among unit-level problems. This layer acts as a supervisory layer. The coordinator can be formulated mathematically as a separate optimization problem or it can be based on certain heuristics, which can be stochastically based and can be designed based on the industry practices or any other available technique depending on the scheduling application. The lower layer consists of the unit-level scheduling problems, similar to the decentralized approach. Every unit-level problem receives information from the coordinator, solves a scheduling problem and returns the required information back to the coordinator. A schematic diagram of the hierarchical approach is shown in Fig. 1.

The hierarchical approach has attracted significant interest because it provides an evolutionary path towards achieving the centralized scheduling and monitoring, without demanding huge computational resources. Unlike in the centralized approach, the unit-level scheduling problems do not share their operating conditions and the only information that is shared is that required by the coordinator for the global consensus. Therefore, data protection and single-point-of-failure issues are prevented in a more simplified way.

Scheduling solutions for the smelting process can be broadly divided based on the structure of the scheduling framework and time representation. From the structure perspective, they are categorized as centralized and hierarchical frameworks, while from the time representation viewpoint, they are divided into continuous-time and discrete-time frameworks. In the former, the time intervals are flexible, whereas in the latter the time is typically parti-

tioned into intervals of equal size (Ewaschuk et al., 2018; Floudas and Lin, 2004).

In (Ewaschuk et al., 2018), Christopher *et al.* proposed a continuous-time centralized framework for the nickel smelting plant. This framework considered two FSF and four PSC units to find an optimal schedule. The solution was also then applied in the real-time scenarios, which emerge from a nickel smelting plant. Harjunkoski *et al.* (Harjunkoski et al., 2006) presented a novel centralized scheduling formulation for the copper smelting process that is based on continuous-time representation. This approach is formulated using MILP techniques where the objective is to maximize the throughput explicitly and the solution is then applied in a copper smelting process.

Suominen et al. (2016) introduced a scheduling framework for the copper smelting process that consists of single FSF and three PSC units. This framework maximizes the smelting throughput and provides an optimal schedule by solving an optimization problem that is based on continuous-time MILP techniques. This formulation is applied to an industrially oriented case study to show the novelty of the framework. Pradenas et al. (2003) proposed another framework for the copper smelting process that maximizes the smelter production. The framework generates a schedule from various production cycles that have numerous operational, metallurgical, and environmental constraints. The framework is applied to a copper smelter to show the applicability of the framework.

In our previous work (Ahmed et al., 2021), we presented a framework that finds a schedule for a single-PSC unit single-batch problem. That framework made use of the discrete-time representation and is formulated using MILP techniques. The framework minimizes unwanted elements content in matte, copper losses, and unnecessary idle times during the PSC operation. We presented a case study as a simulation example to show the significance of that framework. The current study is an extension of that previous work.

3. Process description

A generic flow diagram of the copper smelting process is shown in Fig. 2. In this process, the concentrates are fed to the FSF with changeable feed rate after passing though feeder (Harjunkoski et al., 2008; Association, 2018). These concentrates contain a small percentage of copper (Cu), while a major portion of the concentrates are made of unwanted elements such as iron (Fe), sulfur (S), Nickel (Ni), and other minor elements, such as Bismuth (Bi) (Harjunkoski et al., 2008).

FSF operates continuously to produce copper-enriched matte. Oxygen react with the concentrates to produce molten slag, matte and gases (Suominen et al., 2016; Liu et al., 2014a; Davenport and Partelpoeg, 2015). Gases are transferred to the acid plant for the sulfuric acid production using the gas-transfer pipelines. The slag is removed repeatedly and used by the slag treatment unit for post-processing.

The FSF produces copper-enriched matte with the predefined matte grade, which depends on the oxygen and air flow rate to the FSF (Davenport and Partelpoeg, 2015). This matte is transferred to the PSC units for further processing using the crane, which is usually installed in the vicinity of the smelting process. In the PSC, oxygen is passed through the matte that oxidize the unwanted elements as slag and gases, while copper is left as the final product. The final product, generally referred to as blister copper, has a copper percentage of at least 98 percent (Ahmed et al., 2021; Harjunkoski et al., 2008). The slag is shifted to slag treatment unit, while gases are fed to the acid plant. For gas transfer, both the FSF and PSC units use the same gas pipelines. At the end of the batch, the blister copper is shifted to the anode furnace, followed by the casting unit and then to electrolysis, before the product is available for commercial use.

The PSC follows a predefined sequence of actions to produce a single batch of the blister copper. A schematic diagram of the PSC is shown in Fig. 3. The PSC process begins with the loading of matte. During the slag-making stage, the iron and sulfur in matte are oxidized to iron oxides and sulfur dioxide (SO_2) gas, respectively. The slag is skimmed away periodically during the slag skimming operation. After the last slag skimming, the copper-making stage begins and this stage lasts until the required product quality is achieved. A detailed description of the PSC operation is provided in our previous work (Ahmed et al., 2021).

During this process, PSC bath temperature affects the copper losses to the slag and the quality of the blister copper; therefore, it is maintained within limits (Harjunkoski et al., 2008; Ahmed et al., 2021). Furthermore, the reactions during the slag blows are highly exothermic; therefore, the process operator defines the maximum duration of the first and second slag blows for keeping the PSC bath temperature within its limits. During the third slag blow and copper blow, the temperature is maintained by adjusting the oxygen flow. During the slag blows, copper losses start increasing exponentially as the iron content in the matte approaches zero level (Tan, 2007). Therefore, for an acceptable process operation, the slag blows duration should be carefully controlled to keep those losses at a minimum (Ahmed et al., 2021).

4. Problem statement

In this study, we consider a copper smelting process, as shown in Fig. 4. While the actual smelting process is more complex in nature, this formulation still reflects the main aspects of an actual smelter.

Given: A smelting process that consists of one FSF and multiple PSC units. The FSF operates continuously to produce matte by receiving input concentrates with a variable feed rate. The matte level in the FSF must remain above a minimum value, which is set by the technical staff.

To transfer the matte from the FSF to the PSC units, a crane is installed in the vicinity of the smelter. All the PSC units produce the blister copper batches by following a predetermined sequence of actions. All the PSC units operate independently and without any maintenance break.

Determine: A production schedule where PSC units produce blister copper batches in a synchronized way with the required product quality, while respecting the associated production and scheduling constraints.

Goal: The objective is to formulate both the centralized and the hierarchical discrete-time frameworks to produce optimal schedules. The motivation behind formulating both frameworks is to compare their solutions in terms of quality and computational demands. Furthermore, the interest is to show that the centralized framework is suitable for scheduling applications with smaller problem sizes, while the hierarchical framework is suitable for large-scale scheduling applications. In addition, the centralized framework can act as a benchmark for the hierarchical framework. Furthermore, both frameworks should provide schedules with minimum batch time and must handle the inter-dependencies that arises from the operations in these units. In the smelting process, inter-dependencies are generated due to units' demands for a common scarce resource (Harjunkoski et al., 2008). In this work, inter-dependencies arise due to crane availability and gas-pipelines flow capacity.

Logistical constraints:

Logistical constraints arise due to crane availability. The crane is used for the matte transfer from the FSF to PSC units, slag transfer from PSC units to slag processing unit, and blister copper from PSC

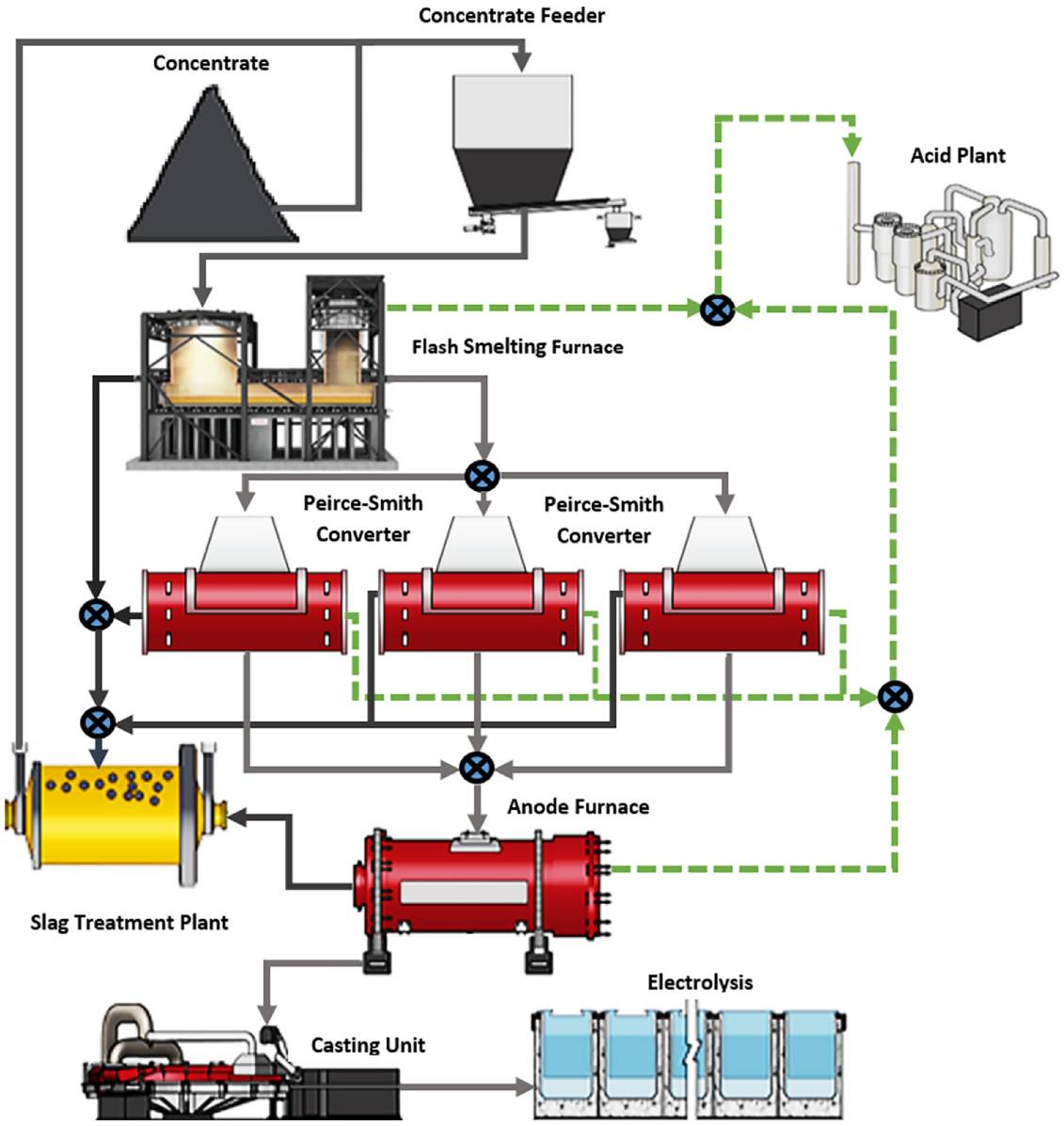

Fig. 2. Copper smelting process (Ahmed et al., 2021).

units to anode furnace. We assumed that the logistical constraints are generated only due to the matte transfer from FSF to PSC units and other units' operations do not contribute to the activation of these constraints.

The crane can transfer only one ladle of matte from the FSF to a PSC unit at any given time. Therefore, the crane operational availability is limited. Since the PSC units are operated separately, logistical constraints arise frequently when two or more PSC units request for the matte loading at the same time or when a PSC unit requests for the matte loading before its proposed starting time.

Flow constraints:

Flow constraints arise due to the limited capacity of gas pipelines. The FSF produces SO_2 gas continuously, while the PSC units produce this gas during the slag and copper blows. This gas is transferred to the acid plant using the gas pipelines. Considering the continuous FSF operation and stopping its operation results in serious consequences, the PSC units are scheduled accordingly to satisfy the gas pipelines' capacity constraints. We assumed that the flow constraints become active only if two or more PSC units are in slag or copper blow stage at the same time.

Assumptions and Limitations:

This study is subject to multiple assumptions and limitations. The FSF initial inventory value and input concentrates feed levels are selected such that they provide a feasible initial operation point for the proposed frameworks. The FSF capacity levels are generally subjected to the FSF type that is being considered. Since this study does not focus on any specific type of FSF, they are chosen arbitrarily. Furthermore, PSC units can be available at same or different times. If they are available at different times, each PSC unit is assigned with a unique priority number, which depends on its availability with reference to the current time. The earliest available PSC unit has the highest priority, while the latest available unit has the lowest priority. However, priority is assigned randomly if two or more PSC units are available at the same time. Additional assumptions are:

- FSF processes same type of input concentrates.
- FSF produces matte with the specific matte grade, which is defined beforehand by the process personnel. The FSF production depends on the concentrate feed rate and matte grade only. When the process begins, there is always an initial inventory available in the FSF.

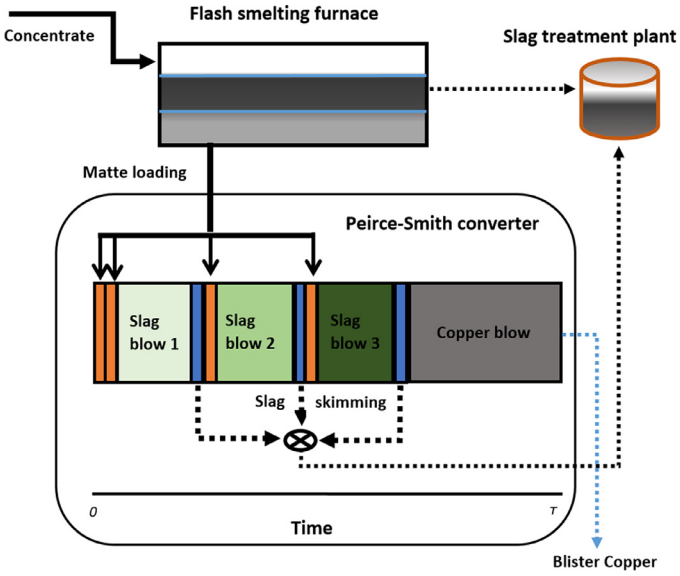

Fig. 3. PSC operation summary (adopted from Ahmed et al. (2021)).

- Due to PSC units' priority, the highest priority PSC unit completes its loading operations first, followed by the next highest priority PSC unit. The logistical constraints become active if a PSC unit requests for matte loading before the preceding high-priority PSC units.
- To satisfy flow constraints, there can be only one slag or copper blow operation at any given time.
- PSC units' initial availability is known in advance. It is assumed that all the previous schedules generated by the framework are feasible. Therefore, inter-dependencies between the previous and current schedule are not considered.
- All the PSC units are same in dimensions, and they follow the same sequence of operations for producing blister copper batches.
- The slag type does not change during the PSC operation.
- The slag treatment unit and acid plant have unlimited storage capacity; hence, they are not considered in the present formulations.
- Oxygen is supplied constantly without any interruption; hence, the oxidation rate of elements in the FSF and PSC remains constant. Therefore, oxygen has not effect on smelter units' performance.
- It is assumed that the temperature is kept in a feasible range by pre-selecting an appropriate constant oxygen enrichment during the slag blow stages; therefore, temperature is not a decision variable in this study (Ahmed et al., 2021).
- During each slag blow, a minimum amount of slag is produced. This is achieved by setting a minimum duration constraint for all the slag blow operations.
- None of the unit requires any maintenance during the process operation.

5. Mathematical formulation

This section describes the mathematical formulation of the centralized and hierarchical framework. The main characteristics that are common to both frameworks are as follows:

- The objective function for each unit remains the same.
- All capacity levels are deterministic and do not change over time.
- Each PSC unit is available only when the previous batch has been completed successfully.

5.1. Centralized framework

5.1.1. Flash smelting furnace

The FSF model is based on the law of mass conservation. The FSF matte production depends on the matte grade mg and the concentrate feed rate $feed^h$. Decreasing mg increases the FSF production $FSFProd^h$ as the FSF requires less time span for oxidizing the required amount of unwanted elements; hence, it increases PSC batch time (Liu et al., 2014b). Similarly, increasing the $feed^h$ increases the $FSFProd^h$ as concentrates are available at a faster rate for the matte production.

The FSF matte production is given in Eq. (1). During the FSF operation, the matte grade mg is selected beforehand and it remains unchanged, while the $feed^h$ needs to be maintained within its bounds. The accumulated mass of the FSF matte is given in Eq. (2). The FSF model must fulfil the minimum matte level constraint, which is added to the model using Eq. (3). The goal of the FSF is to utilize as much input concentrates as possible; hence, maximization of the feed rate is the primary objective, as shown in Eq. (4).

$$FSFProd^h = (Grad1 \times mg) + \left(Grad2 \times feed^h + Y_{sinter}\right)$$
$$\text{feedMin} \le feed^h \le feedMax$$
$$\forall\ Grad1, Grad2, Y_{inter} \in R^+ \tag{1}$$

$$FSFMass^h = FSFMass^{h=0} + FSFMass^{h-1} + FSFProd^h - MatteFix \times \sum_{n=1}^{N}\sum_{b=1}^{B}\sum_{i=1}^{I} b_{z_i}^{n,b,h} \quad \forall\ z_i \neq loading_i \tag{2}$$

$$FSFMass^h \ge FSFLow \tag{3}$$

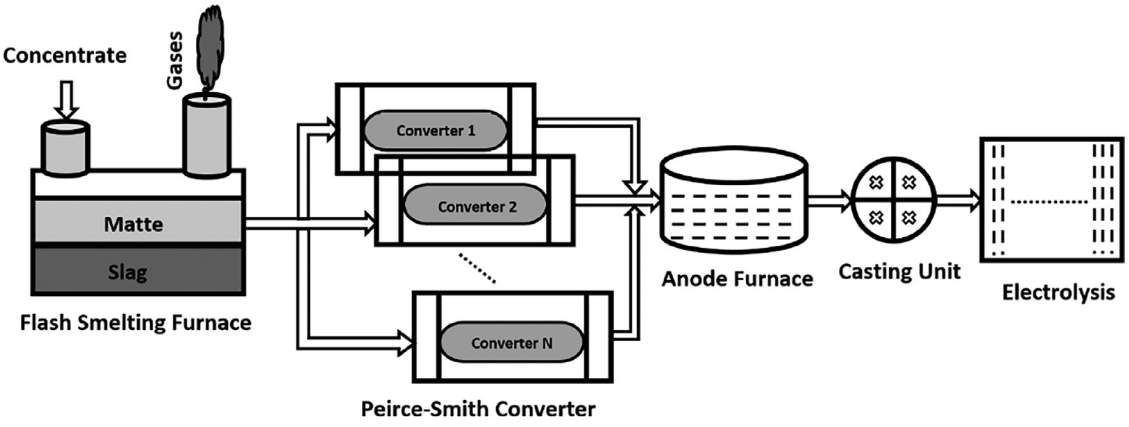

Fig. 4. Smelting process.

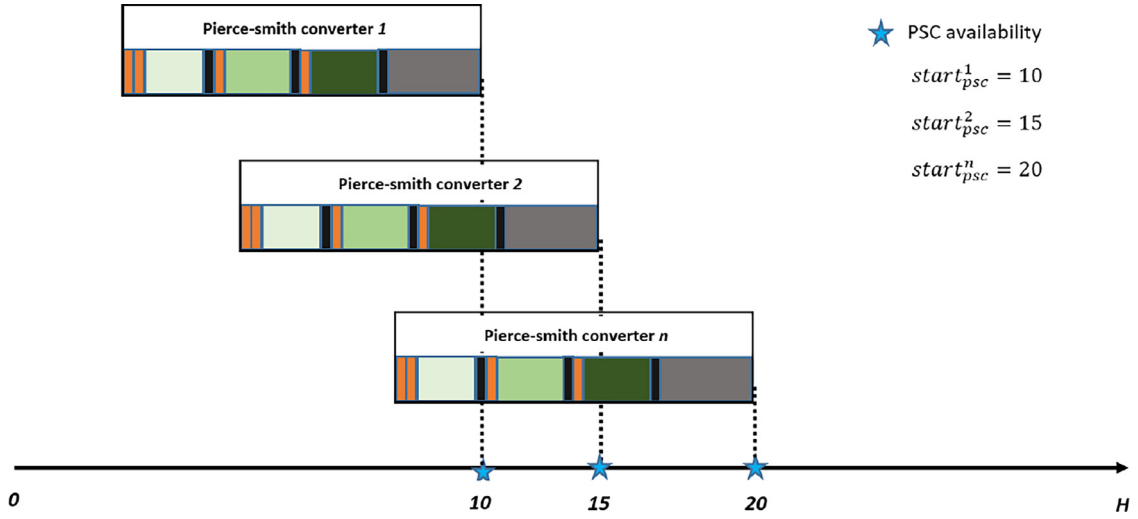

Fig. 5. PSC units availability.

$$\max_{feed} \quad f_{FSF} = \sum_{h=1}^{H} feed^h \tag{4}$$

5.1.2. Peirce-Smith converter

In our previous work, we developed a single-PSC unit single-batch formulation (Ahmed et al., 2021). In the present work, we extended our single-unit PSC formulation to multiple-PSC unit and multiple-batch formulation. This new formulation is used for developing the centralized scheduling framework. However, this multiple-PSC unit and multiple-batch formulation can be configured effortlessly to a single-PSC unit single-batch problem, which is used in the hierarchical approach.

In smelting process, PSC units are functioning in parallel to produce numerous blister copper batches, as shown in Fig. 5. The PSC units' availability is known beforehand using the parameter $PSCStart^n$, which contains the information about the starting time of a PSC unit. Based on the PSC unit's availability, a priority number is assigned to each PSC unit using Eq. (5). This priority number determines which PSC unit begins its operation first, second, and so forth.

$$PSCPrior^n = \begin{cases} Highest & \text{unit } n \text{ has the lowest } PSCStart^n \text{ value} \\ Highest-1 & \text{unit } n \text{ has the second lowest } PSCStart^n \text{ value} \\ \vdots & \\ Lowest & \text{unit } n \text{ has the highest } PSCStart^n \text{ value} \end{cases} \tag{5}$$

Each PSC unit can process only one operation on unit n at a single time, as given in Eq. (6). The processing time of operation remains fixed or its optimal value is estimated by the framework, as provided in Eq. (7).

$$\sum_{i=1}^{I} \sum_{z_i=loading_i}^{Z} b_{z_i}^{n,b,h} \le 1 \qquad \forall\ b \in B, h \in H \tag{6}$$

$$\sum_{h=1}^{H} b_{z_i}^{n,b,h} \begin{cases} = ProcFix_{z_i} & \forall z_i \ne slag-blow_i, copper-blow, b \in B, n \in N \\ \ge ProcMin_{z_i} & \forall z_i = slag-blow_i, copper-blow, b \in B, n \in N \\ \le ProcMax_{z_i} & \forall z_i = slag-blow_i, copper-blow, b \in B, n \in N \end{cases} \tag{7}$$

For scheduling operations of a PSC unit, a combination of two distinct binary variables is used. Variable $b_{z_i}^{n,b,h}$ represents the occurrence of an operation during batch b on unit n at time h. The other variable $s_{z_i}^{n,b,h}$ ensures that all the individual operations $b_{z_i}^{n,b,h}$ during each batch b are scheduled in the same manner as described in the set Z. Therefore, Eqs. (8)-(9) are used to ensure that any succeeding operation during batch b on unit n can happen only if the all the previous operations are successfully completed on the same batch b on unit n; otherwise, the succeeding operation is put on hold (Ahmed et al., 2021).

$$s_{z_i}^{n,b,h} = s_{z_i}^{n,b,h-1} + b_{z_i}^{n,b,h} \qquad \forall\ z_i \in Z \tag{8}$$

$$\begin{aligned} &\textstyle\sum_{i=1}^{I} \sum_{z_1=loading_1}^{z_i} \sum_{h=1}^{H} s_{z_i}^{n,b,h} \ge Pos_{z_i} \times b_{z_i'}^{n,b,h} \\ &\forall\ z_i' = \text{operation succeeding } z_i, b \in B, n \in N \end{aligned} \tag{9}$$

Mass of the matte and various elements contained in this matte are calculated using Eqs. (10)-(11). The amount of unwanted elements in matte depend on the matte grade value, which is represented by function $f(mg)$. Here, all the masses are represented by continuous variables. During slag blows, copper is lost to the slag and its accumulated quantity in slag can be estimated using Equation (12). In order to keep the copper losses at a minimum level, this formulation uses a simple but efficient scheme that minimizes the copper losses by finding the optimal duration of the slag blows. Details about this scheme (such as variables and symbols) can be found in (Ahmed et al., 2021).

$$\begin{aligned} PSCMass^{n,b,h} = PSCMass^{n,b,h-1} &+ \sum_{i=1}^{I} \sum_{h=1}^{H} MatteFix \times b_{loading_i}^{n,b,h} \\ &- \sum_{i=1}^{I} \sum_{h=1}^{H} (CuLoss_{z_i} + eleRate) \times b_{z_i}^{n,b,h} \\ &\forall\ z_i \ne loading_i, b \in B, n \in N \end{aligned} \tag{10}$$

$$\begin{aligned} eleMass^{n,b,h} = eleMass^{n,b,h-1} &\sum_{i=1}^{I} \sum_{h=1}^{H} [f(mg)] \times b_{loading_i}^{n,b,h} \\ &- \sum_{i=1}^{I} \sum_{h=1}^{H} (CuLoss_{z_i} + eleRate) \times b_{z_i}^{n,b,h} \\ &\forall\ z_i \ne loading_i, b \in B, n \in N \end{aligned} \tag{11}$$

$$\begin{aligned} CuMass^{n,b,h} = CuMass^{n,b,h-1} &+ \sum_{i=1}^{I} \sum_{h=1}^{H} CuLoss_{z_i} \times b_{z_i}^{n,b,h} \\ &\forall\ z_i = slag\ blow_i, b \in B, n \in N \end{aligned} \tag{12}$$

A PSC batch ends once the copper blow is completed. Upon successful completion, the variable $s^{n,b,h}_{batch-end}$ is set to 1, as shown in Eq. (13). Therefore, the next batch can begin its processing only if the current batch variable $s^{n,b,h}_{batch-end}$ is set to 1, as shown in Eq. (14).

$$s^{n,b,h}_{batch-end} + b^{n,b,h}_{copper-blow} \leq 1 \qquad \forall\ b \in B, n \in N, h \in H \tag{13}$$

$$s^{n,b,h}_{batch-end} \geq 1 \times b^{n,b+1,h}_{z_i} \qquad \forall\ z_i = loading_i, b \in B, n \in N, h \in H \tag{14}$$

Unnecessary idle times during the batch are minimized using a penalty term $idle^{n,b,h}$, which is calculated using Eq. (15). The objective function of the PSC units is shown in Eq. (16). The overall goal is to minimize the unwanted elements content in matte, copper losses in the slag and idle times. Here, $CuRatio^{n,b,h}$ represents the list of feasible points which are calculated by the proposed copper minimization scheme (Ahmed et al., 2021).

$$idle^{n,b,h} = \sum_{i=1}^{I} \sum_{z_i=loading_i}^{Z} \sum_{h=1}^{H} b^{n,b,h}_{z_i} \qquad \forall\ z_i \in Z, b \in B, n \in N \tag{15}$$

$$\min_{CuMass,eleMass,CuRatio,idle} f_{PSC} = \sum_{c=1}^{C} \sum_{b=1}^{B} \sum_{h=1}^{H} CuMass^{n,b,h} + eleMass^{n,b,h} + CuRatio^{n,b,h} + idle^{n,b,h} \tag{16}$$

5.1.3. Inter-dependencies

In this work, PSC units can be available at any time. If PSC units are available at the same time, priorities are assigned randomly to them; otherwise, priorities are assigned based on their availability.

The processing order of the batches depends on the priority of their PSC units. The batch belonging to the highest priority PSC unit begins its processing first and batches on the low-priority PSC units wait until all the loading operations to this high priority batch have been performed successfully. If a batch on a low-priority PSC unit violates the proposed loading principle, inter-dependencies are generated among the PSC units and the loading constraints become active.

To avoid the inter-dependencies among the PSC batches due to matte loading, Eq. (17) ensures that the loading operations to the PSC units are made in a synchronized manner based on their priority number.

Interdependence among the PSC batches is also generated due to flow constraints when two or more PSC batches are in the slag or copper blow stage together. To ensure that only one PSC batch remains in the slag or copper blow at any given time, Eq. (18) is used.

$$\sum_{i=1}^{I} \sum_{n=1}^{N} \sum_{h=1}^{H} s^{n,b,h}_{z_i} \geq NumLoad \times b^{n+1,b,h}_{z_i} \qquad \forall\ z_i = loading_i, b \in B, n \in N \tag{17}$$

$$\sum_{i=1}^{I} \sum_{n=1}^{N} b^{n,b,h}_{z_i} \leq 1 \qquad \forall\ z_i = slag\ blow_i,\ copper\ blow, b \in B,\ h \in H \tag{18}$$

The schematic diagram of the centralized framework is shown in Fig. 6. The objective function is the addition of local objective functions of the process units, as given in Eq. (19). This framework is initialized with the *mg*, $BatchStart^n$ and other necessary parameters, which are required for the process operation, and it terminates once an optimal schedule with the minimum objective function value has been found.

$$\min \quad f_{Cen} = f_{PSC} - f_{FSF} \tag{19}$$

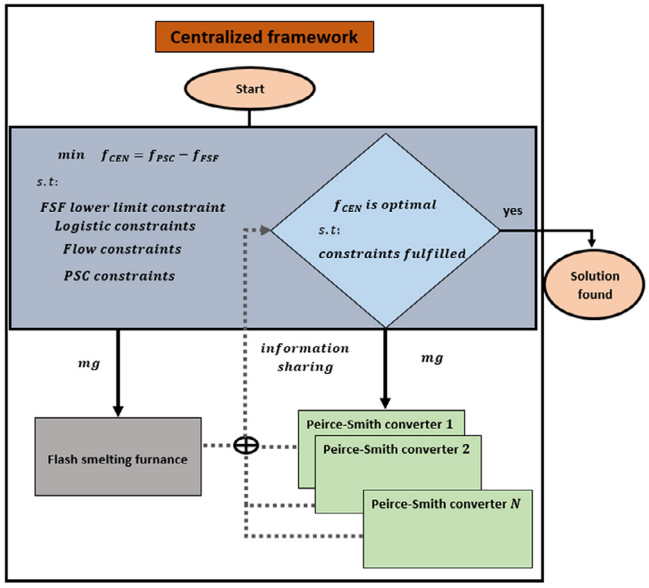

Fig. 6. Centralize scheduling framework.

The centralized framework solves the problem as a single large optimization problem; therefore, the schedule is expected to have minimum idle times, short schedule duration and a higher computation load requirement.

5.2. Hierarchical framework

The hierarchical framework execution is shown in Fig. 7. This scheme is based on the idea that no direct communication exists among the smelting units; therefore, all the information among the units is exchanged through the coordinator. Therefore, the FSF and PSC models are modified accordingly, to accommodate that coordinating information.

5.2.1. Flash smelting process

The mathematical formulation of the FSF model remains the same as described in Section 5.1.1. However, the FSF receives the time instances of the loading operations that have been made during the previous iteration using the parameter $PSCLoad^{n,h}$. This new information is added to the FSF model by replacing Eq. (2) with Eq. (20). Since the coordinator is responsible for the fulfilment of FSF capacity constraint, Eq. (3) is removed from the FSF model.

$$FSFMass^h = FSFMass^{h=0} + FSFMass^{h-1} + FSFProd^h - MatteFix \times \sum_{n=1}^{N} PSCLoad^{n,h} \qquad \forall\ h \in H \tag{20}$$

At each iteration, the FSF model estimates the mass production trajectory $FSFMass^h$ and returns it to the coordinator for further analysis.

5.2.2. Peirce-Smith converter

In the hierarchical framework, PSC units are operated and scheduled independently; therefore, the single-PSC unit single-batch formulation is used here. This unit-PSC formulation is configured from the multiple-unit multiple-batch formulation, which is presented in Section 5.1.2, by equating N and B equal to 1 ($N = 1, B = 1$) and replacing the scheduling horizon (h) with the single batch scheduling horizon (t). Consequently, the hierarchical framework solves for the maximum of $C \times B$ number of independent scheduling problems during each iteration. For simplicity, this single-PSC unit single-batch problem is referred to as a *batch*.

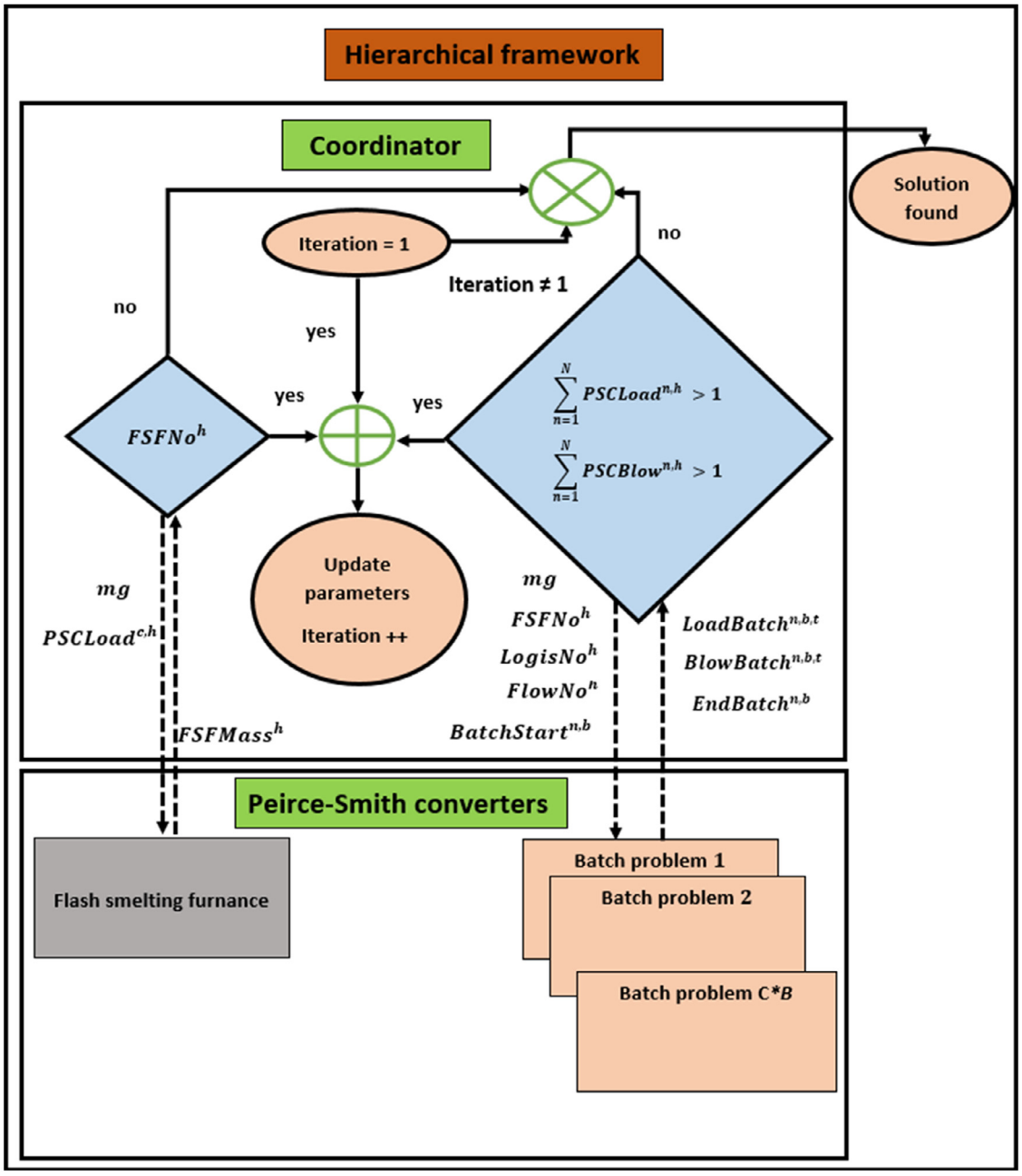

Fig. 7. Hierarchical scheduling framework.

To resolve the inter-dependencies and FSF capacity constraint, all batches receive four distinct pieces of information from the coordinator during each iteration: $BatchStart^{n,b}$, $FSFNo^h$, $LogisNo^h$ and $FlowNo^h$. The parameter $BatchStart^{n,b}$ contains the starting time of batches, while $FSFNo^h$ contains those time values where the FSF capacity constraint is active due to violation of its minimum capacity limit. Parameters $LogisNo^h$ and $FlowNo^h$ contain information about the inter-dependencies due to logistical and flow constraints. $FSFNo^h$, $LogisNo^h$ and $FlowNo^h$ have a value of 1 for all the violating time values.

During each iteration, every batch uses $BatchStart^{n,b}$ to retrieve the concerned and important information. Using $FSFNo^h$ and $LogisNo^h$, every batch retrieves time instants at which the FSF capacity or logistical constraints are active. Similarly, $FlowNo^h$ is used to extract the flow constraint violating time instances. All batches store those violating time instants in $LoadDelay^t$ and $BlowDelay^t$ by assigning a value of 1 against each violating time instant, as given in Eqs. (21) and (22).

The above new information is added to the single-PSC unit single-batch formulation by replacing Eq. (7) with Eqs. (23) and (24). With the addition of this new information, every batch is able to produce a new batch schedule with no *loading* or *blow* operations at the conflicting time instants; this enables the coordinator to resolve the FSF capacity and inter-dependencies at a batch level.

$$LoadDelay^t = \begin{cases} 1 & t = h - BatchStart^{n,b} \text{ and } \{FSFNo^h = 1 \text{ or } LogisNo^h = 1\} \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

$$BlowDelay^t = \begin{cases} 1 & t = h - BatchStart^{n,b} \text{ and } FlowNo^h = 1 \\ 0 & \text{otherwise} \end{cases} \tag{22}$$

$$\sum_{t=1}^{T} b_{z_i}^{t} = \begin{cases} ProcFixz_i & LoadDelay^t = 0 \quad \forall z_i = loading_i \\ 0 & LoadDelay^t = 1 \quad \forall z_i = loading_i \end{cases} \tag{23}$$

$$\sum_{t=1}^{T} b_{z_i}^{t} \begin{cases} = 0 & BlowDelay^t = 1 \quad \forall z_i = slag-blow_i,\ copper-blow \\ \geq ProcMax_{z_i} & BlowDelay^t = 0 \quad \forall z_i = slag-blow_i,\ copper-blow \\ \leq ProcMin_{z_i} & BlowDelay^t = 1 \quad \forall z_i = slag-blow_i,\ copper-blow \end{cases} \tag{24}$$

All batches solve their problems independently to find batch-level optimal schedules and store their local loading, slag blow and copper blows, and batch termination times in $LoadBatch^{n,b,t}$, $BlowBatch^{n,b,t}$ and $LocalEnd^{n,b,t}$, using Eqs. (25)-(27). Since batches may have different starting times, which may result in different termination times, the corresponding batch termination time is stored in $BatchEnd^{n,b}$ using Eq. (28). In the end, each batch shares all the information with the coordinator. Following that, all the batches move to a waiting stage to receive new information from the coordinator.

$$LoadBatch^{n,b,t} = \begin{cases} t & b_{z_i}^{n,b,t} = 1 \quad \forall z_i = loading_i,\ b \in B, n \in N \\ 0 & \text{otherwise} \end{cases} \tag{25}$$

$$BlowBatch^{n,b,t} = \begin{cases} t & \text{if } b_{z_i}^{n,b,t} = 1 \quad \forall z_i = slag-blow_i,\ copper-blow, \\ 0 & \text{otherwise} \\ & b \in B,\ n \in N \end{cases} \tag{26}$$

$$LocalEnd^{n,b,t} = \begin{cases} t & b_{z_i}^{n,b,t} = 1 \quad \forall z_i = batch-end,\ b \in B,\ n \in N \\ 0 & \text{otherwise} \end{cases} \tag{27}$$

$$BatchEnd^{n,b} = \begin{cases} LocalEnd^{n,b,t} & \forall b = 1, n \in N \\ LocalEnd^{n,b,t} + BatchStart^{n,b} & \forall b \neq 1, n \in N \end{cases} \tag{28}$$

5.2.3. Coordinator

The coordinator part of the hierarchical framework is based on heuristics. As the FSF and PSC units have different functionality and there is no interlinking constraint between these two units, heuristics-based coordination is a preferable way of solving the inter-dependencies between these units. This heuristic is divided into parts: the first part is based on the industrial practices in the copper industry, while the second part is inspired by the manual scheduling. The first part of the heuristic is used to solve the inter-dependencies among the units, whereas the second part is used to ensure that it converges to a near-optimal solution with minimum computational costs. In the hierarchical framework, the role of the coordinator is to coordinate between the FSF and PSC units, as well as between the PSC units, to resolve the operational inter-dependencies to achieve smooth operation of the smelting process. Therefore, the coordinator is responsible for:

- Optimal operation of the FSF and PSC units.
- Resolving the inter-dependencies.
- Find a near-optimal schedule without existence of inter-dependencies.

At the start of the hierarchical framework, the coordinator assigns priorities to the PSC units using Eq. (5). During each iteration, all batches solve separate scheduling problems, as discussed in Section 5.2.2. At the end of each iteration, the coordinator receives $LoadBatch^{n,b,t}$, $BlowBatch^{n,b,t}$ and $EndBatch^{n,b}$ from the batches. The coordinator arranges the associated batches of a PSC unit in ascending order of their batch termination times. Therefore, $LoadBatch^{n,b,t}$ and $BlowBatch^{n,b,t}$ are arranged on their corresponding PSC units using the $EndBatch^{n,b}$ as given in Eqs. (29)-(30), and they are stored in parameters $PSCLoad^{n,h}$ and $PSCBlow^{n,h}$.

$$PSCLoad^{n,h} = \begin{cases} 1 & h = \left(LoadBatch^{n,b,t} + EndBatch^{n,b-1}\right) \\ & \forall LoadBatch^{n,b,t} \neq 0, b \in B, n \in N \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

$$PSCBlow^{n,h} = \begin{cases} 1 & h = \left(BlowBatch^{n,b,t} + EndBatch^{n,b-1}\right) \\ & \forall BlowBatch^{n,b,t} \neq 0, b \in B, n \in N \\ 0 & \text{otherwise} \end{cases} \tag{30}$$

The coordinator shares this loading information $PSCLoad^{n,h}$ to the FSF. The FSF returns the $FSFMass^h$ to the coordinator and the coordinator retrieves the conflicting time instants from the $FSFMass^h$ using Eq. (31). The coordinator assigns a value of 1 against each violating time instant, if any exists.

$$FSFNo^h = \begin{cases} 1 & FSFMass^h < FSFMin \quad \forall h \in H \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

Using Eqs. (32) and (33), the coordinator identifies the conflicting time instants that are due to logistical and flow constraints. These conflicting time instances are stored in $LogisNo^h$ and $FlowNo^h$, respectively. Lastly, the coordinator updates the starting time of every batch $BatchStart^{n,b}$ using Eq. (34).

$$LogisNo^h = \begin{cases} 1 & \sum_{n=1}^{N} PSCLoad^{n,h} > 1 \quad \forall h \in H \\ 0 & \text{otherwise} \end{cases} \tag{32}$$

$$LogisNo^h = \begin{cases} 1 & \sum_{n=1}^{N} PSCBlow^{n,h} > 1 \quad \forall h \in H \\ 0 & \text{otherwise} \end{cases} \tag{33}$$

$$BatchStart^{n,b} = BatchEnd^{n,b-1} + 1 \quad \forall b \in B, n \in N \tag{34}$$

The coordinator shares the new $BatchStart^{n,b}$, $FSFNo^h$, $LogisNo^h$ and $FlowNo^h$ with all the batches. The batches are simulated accordingly and they return $LoadBatch^{n,b,t}$, $BlowBatch^{n,b,t}$ and $BatchEnd^{n,b}$ back to the coordinator. The coordinator investigates the FSF mass trajectory and the latest schedule. If FSF capacity constraint violation or inter-dependencies exist in the schedule, the coordinator updates the $BatchStart^{n,b}$, $FSFNo^h$, $LogisNo^h$ and $FlowNo^h$ using Eqs. (31)-(34) and a new iteration is performed. This exchange of information continues until a feasible schedule is found with no violation of the FSF capacity constraint and free of the inter-dependencies. When this happens, the coordinator returns the current schedule and the framework terminates.

The aforementioned heuristic is based on the sharing of the information between the coordinator and the lower-level optimization problems in a systematic manner, enabling it to converge to a near-optimal operational point. However, the convergence pace and its computational cost depends on a few factors, particularly the number of PSC units employed, number of batches per PSC unit, concentrate feed rate, initial FSF inventory level, PSC units' availability, and single-batch horizon value. Generally, these values remain unchanged during process operation. Therefore, in order to increase the speed of convergence, decrease the computational costs, and improve the quality of the solution, the second part of the heuristic becomes active; this is discussed next.

The purpose of the hierarchical framework is to provide a near-optimal solution by reducing the presence of unnecessary idle times and significant computational costs. If a feasible schedule is allowed to contain unnecessary idle times, fewer iterations are required to find a feasible schedule, which means the computational costs reduce. However, such practice can affect adversely the quality of the schedule. On the other hand, strong coordination among process units can improve the quality of the schedule by reducing unnecessary idle times, but also produces high computational costs.

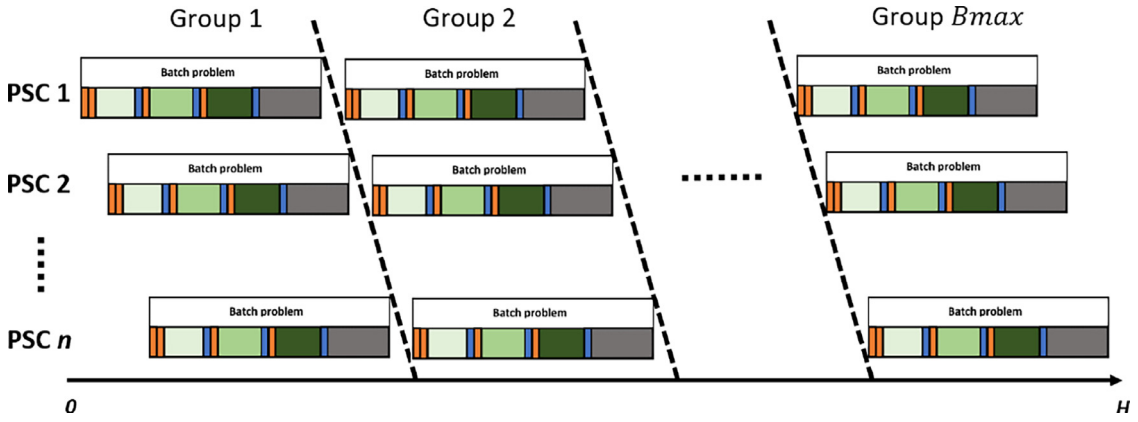

Fig. 8. Grouping of batches.

To ensure that the hierarchical framework always converges to a near-optimal point, the proposed heuristic takes advantage of the PSC unit's priorities. Here, all the batches are divided into separate groups, as shown in Fig. 8. Group 1 consists of the first batch on each PSC unit, while the last batch on each PSC unit is assigned to group *Bmax*. The allocation of batches to the groups is arranged such that, if the starting time of a batch in a group is higher than the maximum termination time of the batch in a group, it is not added to the same group, but to the next group if its batch starting time is smaller or equal to the maximum termination time of the batch in the next group, and so on. Here, *Bmax* is the maximum number of batches processed on a PSC unit. Considering this batch's grouping, the hierarchical framework resolves the inter-dependencies in ascending order of the group numbers. Hence, the batches of Group 1 are handled first, while the batches of group *Bmax* are addressed last.

To resolve inter-dependencies in a group, the hierarchical framework uses a periodic approach instead of resolving all the inter-dependencies simultaneously. This periodic approach refers to solving only some of the inter-dependencies during a single iteration; hence, the number of iterations to find a feasible schedule increases. However, this approach increases the quality of the solution by reducing the presence of unnecessary idle times. On the contrary, if all the inter-dependencies are solved simultaneously in a group, it can add unnecessary idle times to the schedule, which affects the quality of solution. In the worst-case scenario, when inter-dependencies are handled simultaneously in a group, the framework may start oscillating between two local-optimal points and the framework may terminate without a feasible solution.

For resolving inter-dependencies in a group using a periodic approach, the coordinator first identifies the conflicting batches and then their corresponding PSC units. The coordinator then always selects the lowest priority PSC unit from the set of conflicting PSC units and the batch on this unit is simulated only during the current iteration. This way, the operation of the batch on the lowest priority PSC unit is rescheduled and, consequently, the inter-dependencies are resolved between this batch and the batches on other high-priority PSC units. In the next iteration, it repeats the same procedure by only selecting and simulating the batch on the lowest priority PSC unit from the given conflicting PSC units. This process continues until all the inter-dependencies within a group are resolved successfully.

The benefit associated with the periodic approach is that the batches in a group are rescheduled according to the priority associated with their PSC units after resolving their inter-dependencies. For example, if inter-dependencies exist among all the batches of Group 1, the batch on PSC unit n is rescheduled first, considering that this unit has the least priority. In the next iteration, if inter-dependencies still exist between the batch of the PSC unit n and the other batches on high-priority PSC units in Group 1, the same batch is rescheduled again. However, if inter-dependencies exist in Group 1 and the batch of PSC unit n is not the source of these inter-dependencies' generation, the batch on PSC unit $n-1$ is rescheduled to resolve the inter-dependencies in Group 1.

After resolving all the inter-dependencies that may exist in a group, the framework searches for the existence of inter-group inter-dependencies before examining the following group. As groups are arranged in ascending order of their group numbers, inter-dependencies between the current and succeeding groups are resolved by rescheduling all the batches associated with the succeeding groups at once. After solving the inter-group inter-dependencies, the framework searches for the inter-dependencies among the batches of the following group. If inter-dependencies exist in this group, the framework repeats the same procedure, and this process continues until all the inter-dependencies among all the batches in all the groups are resolved successfully.

In a hierarchical framework, the FSF capacity limit violations and inter-dependencies among batch problems can occur simultaneously. Resolving all of them at the same time results in weak optimality of the solution due to the presence of unnecessary idle times. The quality of the solution can be improved by prioritizing these conflicts. For example, if the FSF capacity limit and logistical constraints become active at the same time, resolving the logistical constraints prior to the FSF one is more valuable because it can possibly resolve the FSF capacity constraints automatically or it can reduce the violating time span, thus minimizing the computational demands. Fig. 9 shows how conflicts are handled. The coordinator heuristic is summarized in Algorithm 1.

The hierarchical framework will produce a sub-optimal solution compared to the centralized framework in terms of idle times presence in the schedule and schedule duration. In contrast to centralized framework, the problem size of the hierarchical framework does not grow exponentially with the addition of new batch problems; therefore, the computational costs will be lower than the centralized framework. Hence, the schedule produced by this framework is expected to have extra idle times and higher schedule duration, but lower computational demands.

6. Case studies

In this section, we present two case studies to validate the proposed frameworks. In these case studies, the centralized and

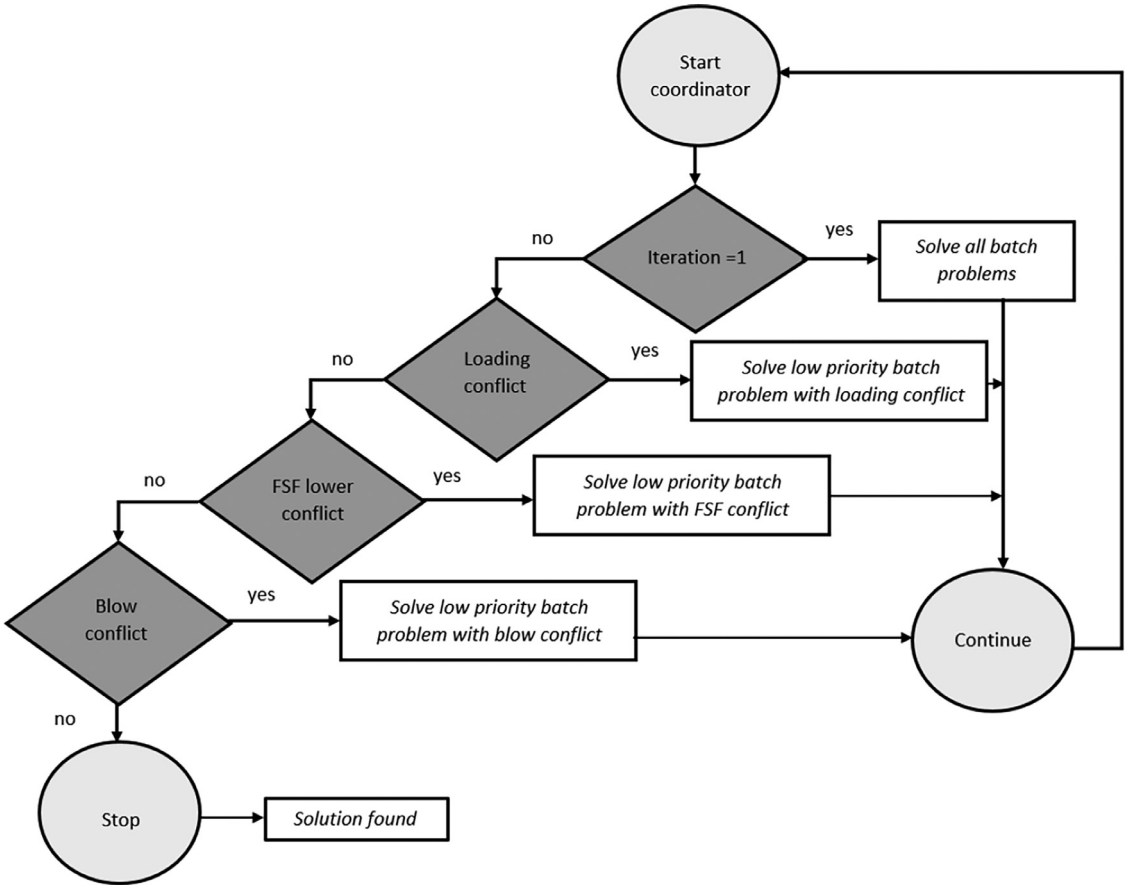

Fig. 9. Conflicts handling in the coordinator.

hierarchical frameworks are compared in terms of their solution quality and computational demands.

In both case studies, the PSC units' availability is known in advance from the process history. The process parameters are given in Table 1. Each PSC batch recipe consists of three loading operations, three slag blow operations, three slag skimming operations, and a single long copper blow operation; therefore, $I = 3$. Both frameworks are modelled in GAMS and solved with CPLEX 12.7 with the default optimally gap of 10 percent (Bussieck and Meeraus, 2004; IBM, 2017). The computations were performed using a 2.60 GHz IntelCoreTM i7 6700HQ processor with 32GB of RAM, running Windows 10 Enterprise 64-bit.

CPLEX uses the branch-and-cut algorithm to find an optimal solution for MIP problems. Centralized framework with few PSC units or batches per PSC unit have lower computational demands. However, increasing the number of PSC units or batches per PSC unit results in a larger problem formulation of the centralized frameworks; hence, the computational demands increase significantly. Larger problem formulation of the centralized framework implies that the framework has a higher number of discrete variables. Consequently, the number of nodes in the branch-and-cut algorithm increases and the computational demands become higher. In given case studies, an upper bound on the computational time is defined and the frameworks will terminate automatically if no feasible solution is found during this maximum CPU time. The value of this upper bound can be any positive integer number and is subject to the scheduling application and process personnel experience. As the objective of this study is to compare the centralized and hierarchical frameworks in terms of their solution quality and computational demands, a high value for this upper bound is chosen and is set to 120 minutes. In real-time scheduling applications, this value can be very small compared to the value chosen here.

In MIP problems, one way to reduce the complexity of the framework is to choose an educated initial condition. A warm start like this assists the branch-and-cut algorithm determine where to begin the search. In the best possible scenario, a good initialization can direct the algorithm to perform the search near the node where the optimal solution may exist.

In the centralized framework, no prior solution exists, so all the framework variables are initialized with the zero value. However, the batches are identical in the hierarchical framework; therefore, the solution computed by the previous batch can be used as a warm start for the next batch, as shown in Fig. 10.

6.1. Case study 1 - Base case

In this case study, the centralized and hierarchical frameworks are simulated for the process that consists of one FSF and two PSC units. The process initial values are given in Table 2. Here, each PSC unit produces only two batches; hence, both frameworks produce feasible schedule for four PSC batches. As both PSC units are available at the same time, PSC Unit 1 is assigned the highest priority and PSC Unit 2 has the lowest priority.

Using the centralized approach, the framework finds an optimal schedule considering one FSF and four PSC batches as a single problem, as shown in Fig. 11. Fig. 11 shows that the centralized framework can resolve the inter-dependencies among the PSC batches effectively by producing a shorter schedule with fewer idle times and this framework is capable of maintaining the FSF matte level within the defined limits. However, the size of the central-

Algorithm 1: FSF and PSC units scheduling based using hierarchical framework.

Data: $PSCStart^{n}$, $PSCPrior^{n}$, $FSFMass^{h=0}$, mg
Result: Feasible process schedule with no FSF capacity constraint violations and inter-dependencies presence
1: Initialization
$iteration$=1
$FSFNo^{h}$=0
$LogisNo^{h}$=0
$FlowNo^{h}$=0
$BatchStart^{n,b} = PSCStart^{n}$
$Stop = false$
2: **while** $Stop = false$ **do**
  if $iteration = 1$ **then**
    solve the *FSF* model, obtaining $FSFMass^{h}$
    coordinator computes $FSFNo^{h}$
    solve the *batch problems* using $FSFNo^{h}$, obtaining $PSCLoad^{n,h}$, $PSCBlow^{n,h}$ and $EndBatch^{n,b}$
  else
    solve the *FSF* model using $PSCLoad^{n,h}$, obtaining the $FSFMass^{h}$
    coordinator computes $FSFNo^{h}$, $LogisNo^{h}$ and $FlowNo^{h}$
    if $(\sum_{h=1}^{H} FSFNo^{h} = 0$ *and* $\sum_{h=1}^{H} LogisNo^{h} = 0$ *and* $\sum_{h=1}^{H} FlowNo^{h} = 0)$ **then**
      feasible schedule is found
      $Stop = true$
    else
      batch problems are solved using $PSCStart^{n,b}$, $FSFNo^{h}$, $LogisNo^{h}$ and $FlowNo^{h}$, obtaining $PSCLoad^{n,h}$, $PSCBlow^{n,h}$ and $EndBatch^{n,b}$
      iteration ++
      Goto step 2
    end
  end
end

Table 1
Framework parameters.

Parameter	Description	Value
mg	matte grade	68 (percent)
$feedMin$	minimum feed rate	5 (kg/min)
$feedMax$	maximum feed rate	100 (kg/min)
$FSFLow$	minimum FSF level	20 (kg)
$MatteFix$	matte transferred from FSF to PSC	20 (kg/min)
$ProcMax_{slag\ blow_1}$	slag blow 1 maximum length	50 (min)
$ProcMin_{slag\ blow_1}$	slag blow 1 minimum length	5 (min)
$ProcMax_{slag\ blow_2}$	slag blow 2 maximum length	60 (min)
$ProcMin_{slag\ blow_2}$	slag blow 2 minimum length	5 (min)
$ProcFix_{slag\ skimming_i}$	slag removal time	1 (min)
$ProcFix_{loading_i}$	loading time	1 (min)
r_{Fe}	iron oxidation rate	0.240 (kg/min)
r_S	sulfur oxidation rate	0.0330 (kg/min)
$CuLoss_{slagblow_1}$	Copper loss rate during slag blow 1	0.103 (kg/min)
$CuLoss_{slagblow_2}$	Copper loss rate during slag blow 2	0.182 (kg/min)
$CuLoss_{slagblow_3}$	Copper loss rate during slag blow 3	0.80 (kg/min)

Table 2
Framework parameters.

Parameter	Description	Value
$mass_{FSF}^{h=0}$	FSF initial mass	300 (kg)
$start_{psc}^{c}$	PSC starting time	{0,0} (min)
H	scheduling horizon	150 (min)
T	batch scheduling horizon	50 (min)

Table 3
Computational requirement.

Approach	CPU time (min)	Schedule horizon (min)
Centralized	74:38	136
Hierarchical	2:12	143

ized problem formulation is larger than the hierarchical framework; therefore, the computational demands will also be high. The CPU time requirement for the centralized and hierarchical frameworks is given in Table 3.

In the hierarchical framework, one FSF and four batch problems are solved independently, and a near-optimal schedule is found, as shown in Fig. 11. The coordinator assigns the first batch of PSC units to Group 1 and the second batch of the PSC units to Group 2. At each iteration, the coordinator searches for the existence of inter-dependencies starting from Group 1. If inter-dependencies exist among batches of Group 1, the first batch on PSC Unit 2 is rescheduled to resolve the inter-dependencies. However, if rescheduling of this batch does not resolve the inter-dependencies in Group 1, especially when they are generated due to the activation of FSF capacity constraint, the coordinator will keep rescheduling both the batches in Group 1 until the given inter-dependencies are resolved.

After resolving inter-dependencies in Group 1, the coordinator searches for the existence of the inter-dependencies between Group 1 and Group 2. If inter-dependencies exist, they are resolved prior to Group 2. Resolving inter-dependencies in Group 2 follows the same procedure. At the end of each iteration, the coordinator reviews the complete schedule and updates the coordinating information, if required; otherwise, the final schedule is returned.

The centralized framework requires more CPU time due to large size of the scheduling problem. However, the schedule computed by this framework is optimal because it contains the minimal unnecessary idle times. Therefore, the centralized framework solution can act as a benchmark. On the other hand, the CPU time requirement for the hierarchical framework is substantially less than the centralized framework; hence, this approach is likely to be attractive in many smelting processes. As all the unit-level problems in the hierarchical framework communicate through the coordinator, the possibility of the unnecessary idle times in the schedule cannot be ruled out. Those unnecessary idle times in the schedule increase the schedule horizon, thus decreasing the quality of the schedule. One way to reduce unnecessary idle times in the hierarchical framework is to use a higher penalty term for the idle time in the objective function of the batches. However, such action can increase the CPU time requirement of the framework.

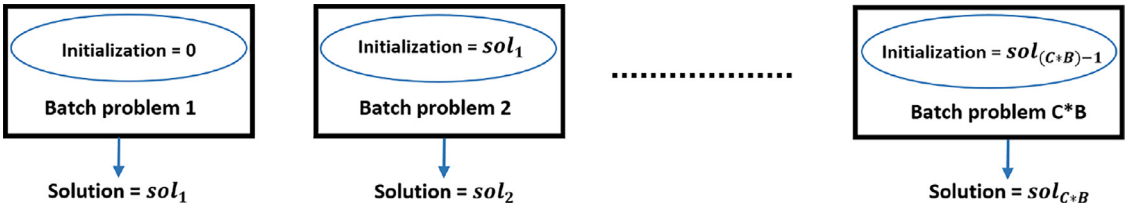

Fig. 10. Hierarchical framework initialization.

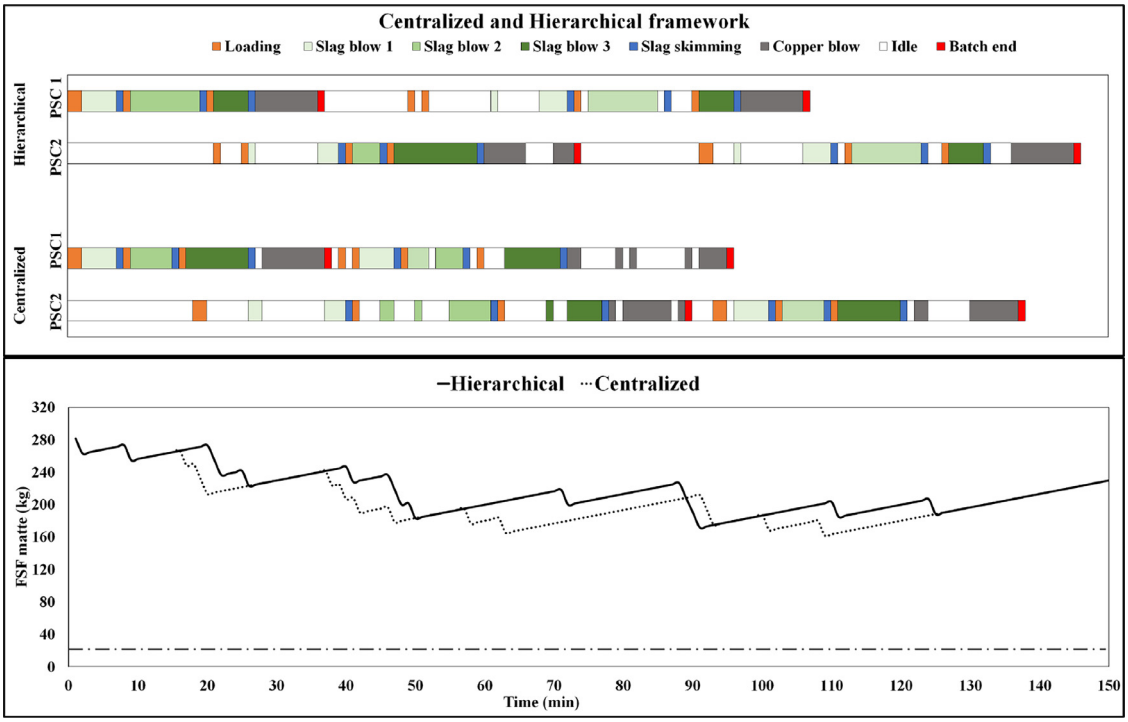

Fig. 11. Centralized and hierarchical schedule.

Table 4
Framework parameters.

Parameter	Description	Value (min)
H	scheduling horizon	600
T	batch scheduling horizon	120

Table 5
Computational requirement.

Approach	CPU time (min)	Schedule horizon (min)
Centralized	120:00	-
Hierarchical	8:50	570

6.2. Case study 2 - Higher number of PSC batches

In this case study, the smelting process consists of one FSF and three PSC units. Each PSC unit produces five batches; thus, a total number of 15 PSC batches are produced. For this case study, the scheduling and batch horizon values are modified, as given in Table 4. All PSC units are available at the same time; hence, the PSC Unit 1 has the highest priority while PSC Unit 3 has the lowest priority. Due to the large size of the problem, the centralized framework failed to find feasible schedule in the maximum CPU time. On the other hand, the hierarchical framework produces a near-optimal schedule with no inter-dependencies and FSF capacity constraints violations, as shown in Fig. 12.

In the hierarchical framework, one FSF and 15 batches are solved independently. At the beginning of the schedule, there is enough matte in the FSF available; hence, PSC units are the bottlenecks. However, after time 325 min, there is simply not enough matte available in the FSF to meet the PSC units' demand; therefore, the FSF is the bottleneck. Hence, matte *loading* operations to the PSC units are delayed and more idle times are added to the schedule by the coordinator due to FSF capacity constraint activation.

The hierarchical framework divides all the batches into five groups. Group 1 contains the first batch on each PSC unit, Group 2 contains the second batch on each PSC unit, and so forth. The framework first resolves the inter-dependencies among the batches of Group 1, if any exist. If no such inter-dependencies exist in Group 1, the framework searches for inter-dependencies between Groups 1 and 2, followed by the examination of Group 2 batches, and so on. This process continue until all the inter-dependencies among all the batches have been successfully resolved. Fig. 12 illustrates that the operations of the batches on the PSC Unit 3 are delayed with the maximum amount, while the operations of batches on PSC Unit 1 are delayed with the minimum amount as PSC Unit 3 has the least priority and PSC Unit 1 has the highest priority.

The computational demands for the centralized and hierarchical frameworks are shown in Table 5. The computational demands of the hierarchical framework are higher than those in Case 1 due to higher number of PSC batches and inter-dependencies, which are continuously produced by the FSF and PSC units. Consequently, the coordinator performs more iterations to find a feasible schedule; hence, the computational demands increase.

During the simulations, it has been observed that the computational costs depend on the FSF inventory level, matte grade selection, number of PSC units, and number of batches per PSC unit. Increasing the FSF inventory level minimizes the FSF capacity constraint activation, thus reducing the computational costs. On the other hand, the FSF matte production decreases with the increase in the matte grade value. Therefore, the computational costs can be reduced by increasing the FSF inventory level, decreasing the matte grade, reducing the number of PSC units or batches per unit.

For the above case studies, Table 6 summarizes the complexity of the frameworks in terms of their size. Increasing the number of PSC units or batches increases the size of the scheduling problem in a centralized framework, which increases the computational costs. In the hierarchical framework, adding a new batch to the framework implies adding a new scheduling problem to the list of batch problems. As all batch problems are solved indepen-

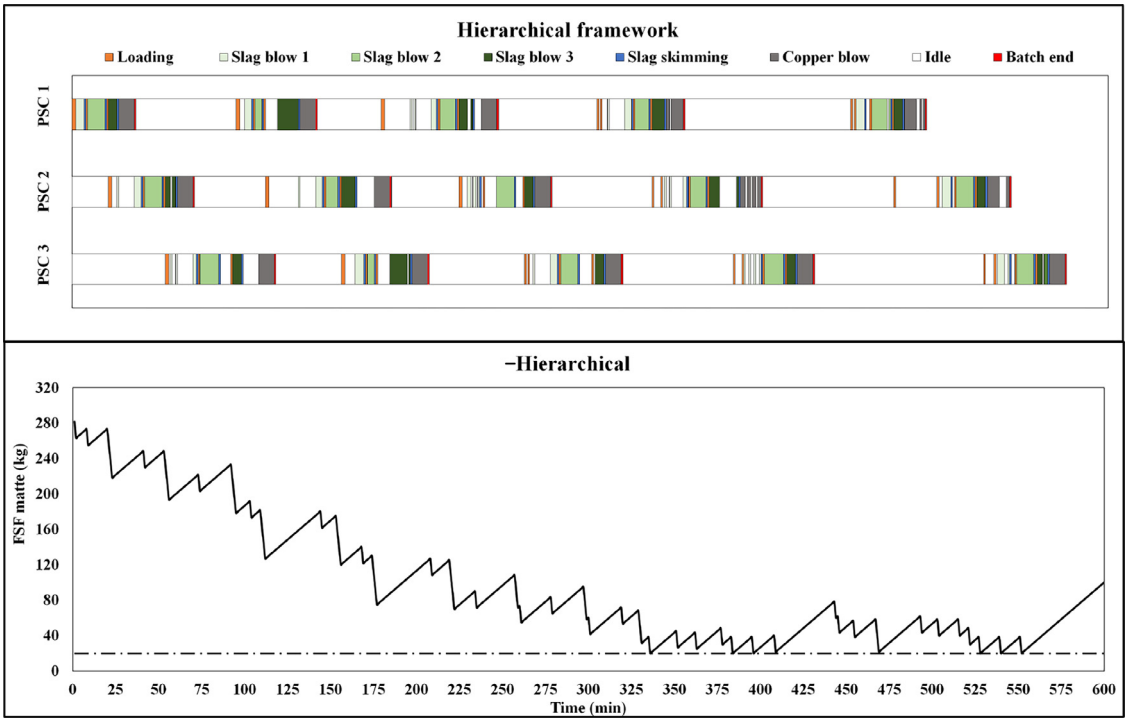

Fig. 12. Hierarchical schedule.

Table 6
Problem size.

Approach	Case	Continuous var.	Discrete var.	Idle times (s)	Optimality (percent)		Copper losses (kg)
					min	max	
centralized	1	17,251	7200	84	93.1	93.1	33.036
	2	117,000	274,951	-	-	-	-
hierarchical	1	8703	1650	103	94.7	99.2	32.844
	2	8703	1650	1042	90.7	99.2	93.581

dently, this new addition of the batch problem does not dramatically change the computational demands of the other individual batch problems. Consequently, the overall computational costs of the hierarchical framework are increased by a small amount.

In Case 1, the centralized framework produced a smaller number of idle times than the hierarchical framework. This is due to the structure of the framework. In the centralized framework, the units monitor each other by sharing all the information directly without any third-party arbitrator. Hence, the centralized framework provides a better solution by removing surplus idle times from the schedule. On the other hand, the hierarchical framework focuses on sharing the relevant information among the units using the coordinator; therefore, the schedule has a higher number of idle times than the centralized framework. Consequently, the schedule duration is higher and the schedule can be categorized as sub-optimal.

The copper losses in the frameworks depend on the quality of their solutions. Solutions with lower cost function values will produce reduced copper losses compared to solutions that have higher cost function value. As minimization of the copper losses is part of the batch problem's objective function, each batch problem finds a better trade-off between the slag blow operations' duration and copper losses in the slag; hence, the cost function value is lower and, consequently, the copper losses are maintained at a lower level (Ahmed et al., 2021). Therefore, the overall copper losses in the hierarchical framework are less than those in the centralized framework. On the other hand, the centralized framework is solved as a single large problem and the solution is returned with lower quality (93 percent) due to higher computational demands. As a result, the trade-off is not as good as it is in the hierarchical framework; hence, the copper losses are higher in the centralized framework.

Case 2 illustrates that the hierarchical framework efficiency depends only on the framework heuristics, which is used for the coordination. A good heuristic, such as that presented in Section 5.2.3, can provide a better trade-off between the solution quality and computational demands. An important factor that affects the hierarchical framework performance is the size of the batch scheduling horizon (T). Reducing this horizon reduces the batch solution time, thus reducing the overall computational demands. However, reducing the batch scheduling horizon excessively might result in infeasible solution for some batches, especially when FSF is the bottleneck due to its matte production, and the batches are waiting for matte loading for a longer time. Therefore, a suitable batch horizon should be selected considering the FSF production rate and initial inventory level.

The above case studies were solved within the default optimality gap in GAMS CPLEX (10 percent). In order to increase the quality of the solution and reduce copper losses, the optimality gap can be set to a lower number, such as 1 percent. However, such action will increase the computational costs, especially for the centralized framework. For an actual smelting process, the tolerance can be relaxed considering the machine power and time tolerance required to find a feasible solution in acceptable computational times.

7. Conclusion

This study has presented novel discrete-time centralized and hierarchical frameworks for the copper smelting process. These frameworks have potential value for technical staff to forecast the effects of their decisions on the complete smelting process. The proposed frameworks include modelling of the FSF unit, PSC units and resolving the unit's operational inter-dependencies. The key features of both frameworks include continuous operation of the FSF while maintaining its capacity levels, minimizing copper losses and idle times during the schedule, and resolving the operational inter-dependencies among the process units. The centralized framework cannot be utilized in many real-time smelting processes due to high computational demands, while the hierarchical framework provides a near-optimal solution by using promising heuristics. Furthermore, only deterministic case studies are presented in this study. Future work will include a sensitivity analysis of the hierarchical framework and finding other potential heuristics for the coordinator; for example, variation in the matte grade if matte grade is used as a decision variable. Furthermore, the optimality gap (10 percent) will be reduced in the future, and solutions with the reduced optimality gap will be analyzed in term of the copper losses and computational demands.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Hussain Ahmed reports was provided by EU Framework Programme for Research and Innovation Research Infrastructures and E-Infrastructures.

CRediT authorship contribution statement

Hussain Ahmed: Conceptualization, Methodology, Software, Validation, Writing – original draft, Writing – review & editing, Visualization. **Luis A. Ricardez-Sandoval:** Methodology, Writing – review & editing, Visualization. **Matti Vilkko:** Conceptualization, Methodology, Validation, Writing – review & editing, Visualization, Supervision, Project administration, Funding acquisition.

References

Ahmed, H., Ricardez-Sandoval, L., Vilkko, M., 2021. Optimal scheduling of the peirce–smith converter in the copper smelting process. Processes 9 (11). doi:10.3390/pr9112004.

Anderson, J., Papachristodoulou, A., 2012. A decomposition technique for nonlinear dynamical system analysis. IEEE Trans Automat Contr 57 (6), 1516–1521. doi:10.1109/TAC.2011.2175058.

Copper Development Association, 2018. https://copperalliance.org.uk/knowledge-base/education/education-resources/copper-mining-extraction-sulfide-ores/, Accessed: November 17, 2020.

Bussieck, M.R., Meeraus, A., 2004. General Algebraic Modeling System (GAMS). Springer US, Boston, MA, pp. 137–157.

Cheng, R., Forbes, J.F., Yip, W.S., 2006. Coordinated decentralized mpc for plant-wide control of a pulp mill benchmark problem. IFAC Proceedings Volumes 39 (2), 971–976. doi:10.3182/20060402-4-BR-2902.00971. 6th IFAC Symposium on Advanced Control of Chemical Processes

Cheng, R., Forbes, J.F., Yip, W.S., 2007. Price-driven coordination method for solving plant-wide MPC problems. J Process Control 17 (5), 429–438. doi:10.1016/j.jprocont.2006.04.003. Selected papers presented at the 2005 IFAC World Congress

Cheng, R., Fraser Forbes, J., San Yip, W., 2004. Dantzig-wolfe decomposition and large-scale constrained MPC problems. IFAC Proceedings Volumes 37 (9), 953–958. doi:10.1016/S1474-6670(17)31931-6. 7th IFAC Symposium on Dynamics and Control of Process Systems 2004 (DYCOPS -7), Cambridge, USA, 5–7 July, 2004

Christodoulopoulos, K., Sourlas, V., Mpakolas, I., Varvarigos, E., 2009. A comparison of centralized and distributed meta-scheduling architectures for computation and communication tasks in grid networks. Comput Commun 32 (7), 1172–1184. doi:10.1016/j.comcom.2009.03.004.

Davenport, W.G., Partelpoeg, E.H., 2015. Flash Smelting: Analysis, Control and Optimization, 2nd Elsevier Science. https://books.google.fi/books?id=cMcgBQAAQBAJ

Ewaschuk, C.M., Swartz, C.L.E., Zhang, Y., 2018. An optimization framework for scheduling of converter aisle operation in a nickel smelting plant. Computers & Chemical Engineering 119, 195–214. doi:10.1016/j.compchemeng.2018.08.024.

FIMECC, 2014. Energy and lifecycle-efficient metal processes – ELEMET. https://www.dimecc.com/wp-content/uploads/2019/05/FIMECC_FINAL_REPORT_5_ELEMET.pdf, Accessed: 24 September, 2021.

Floudas, C.A., Lin, X., 2004. Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. Computers & Chemical Engineering 28 (11), 2109–2129. doi:10.1016/j.compchemeng.2004.05.002.

Group, I. C. S., 2019. The world copper factbook 2019. https://static1.squarespace.com/static/5dee7d0867a5b420e84ed63b/t/5ea41d868315084d6db62095/1587813771457/2019_10_29_ICSG_Factbook_2019.pdf, Accessed: 24 September, 2021.

Gupta, A., Maranas, C.D., 1999. A hierarchical lagrangean relaxation procedure for solving midterm planning problems. Industrial & Engineering Chemistry Research 38 (5), 1937–1947. doi:10.1021/ie980782t.

Harjunkoski, I., Borchers, H.W., Fahl, M., 2006. Simultaneous scheduling and optimization of a copper plant. In: Marquardt, W., Pantelides, C. (Eds.), 16th European Symposium on Computer Aided Process Engineering and 9th International Symposium on Process Systems Engineering. In: Computer Aided Chemical Engineering, Vol. 21. Elsevier, pp. 1197–1202. doi:10.1016/S1570-7946(06)80209-9.

Harjunkoski, I., Fahl, M., Borchers, H.W., 2008. Scheduling and Optimization of a Copper Production Process. John Wiley & Sons, Ltd, pp. 93–109.

Harjunkoski, I., Grossmann, I.E., 2001. A decomposition approach for the scheduling of a steel plant production. Computers & Chemical Engineering 25 (11), 1647–1660. doi:10.1016/S0098-1354(01)00729-3.

IBM, 2017. V12.7 IBM ILOG CPLEX Optimization Studio CPLEX User's Manual. International Business Machines Corporation. https://www.ibm.com/support/knowledgecenter/SSSA5P_12.8.0/ilog.odms.studio.help/pdf/usrcplex.pdf.

Iiro Harjunkoski, M. Z., Beykirch, G., Weidemann, H. J., 2005. The process copper, copper plant scheduling and optimization. https://library.e.abb.com/public/dd346e3ae11a73b2c12570c800527bfd/51-54%204M576_ENG72dpi.pdf, Accessed: 24 September, 2021.

Institute, E. C., 2018. Part of the copper alliance. https://copperalliance.eu/, Accessed: 24 September, 2021.

Liu, J.-H., Gui, W.-h., Xie, Y.-F., Yang, C.-H., et al., 2014. Dynamic modeling of copper flash smelting process at a smelter in China. Appl Math Model 38 (7), 2206–2213. doi:10.1016/j.apm.2013.10.035.

Liu, J.-H., Gui, W.-h., Xie, Y.-F., Yang, C.-H., et al., 2014. Dynamic modeling of copper flash smelting process at a smelter in china. Appl Math Model 38 (7), 2206–2213. doi:10.1016/j.apm.2013.10.035.

Martí, R., Sarabia, D., Navia, D., de Prada, C., 2013. A method to coordinate decentralized NMPC controllers in oxygen distribution networks. Computers & Chemical Engineering 59, 122–137. doi:10.1016/j.compchemeng.2013.05.023. Selected papers from ESCAPE-22 (European Symposium on Computer Aided Process Engineering - 22), 17–20 June 2012, London, UK

Moroşan, P., Bourdais, R., Dumur, D., Buisson, J., 2010. Distributed model predictive control based on benders' decomposition applied to multisource multizone building temperature regulation. In: 49th IEEE Conference on Decision and Control (CDC), pp. 3914–3919. doi:10.1109/CDC.2010.5717092.

Popa, C., 2014. Application of plantwide control strategy to the catalytic cracking process. Procedia Eng 69, 1469–1474. doi:10.1016/j.proeng.2014.03.143. 24th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2013

Pradenas, L., Fernandez, R., Parada, V., Caballero, C., Zuniga, J., 2003. A solution to the copper smelter scheduling problem. Pyrometallurgy of Copper 4 (11), 351–357. doi:10.1016/j.compchemeng.2004.05.002. The Hermann Schwarze Symposium on Copper Pyrometallurgy

Roslóf, J., Harjunkoski, I., Westerlund, T., Isaksson, J., 2002. Solving a large-scale industrial scheduling problem using MILP combined with a heuristic procedure. Eur J Oper Res 138 (1), 29–42. doi:10.1016/S0377-2217(01)00140-0.

Shiquan, Z., Maxim, A., Liu, S., Keyser, R., Ionescu, C., 2019. Distributed model predictive control of steam/water loop in large scale ships. Processes 7, 442. doi:10.3390/pr7070442.

Suominen, O., Mórsky, V., Ritala, R., Vilkko, M., 2016. Framework for optimization and scheduling of a copper production plant. In: Kravanja, Z., Bogataj, M. (Eds.), 26th European Symposium on Computer Aided Process Engineering. In: Computer Aided Chemical Engineering, Vol. 38. Elsevier, pp. 1243–1248. doi:10.1016/B978-0-444-63428-3.50212-5.

Tan, P., 2007. Applications of thermodynamic modeling in copper converting operations. International Journal of Materials Research - INT J MATER RES 98, 995–1003. doi:10.3139/146.101548.